CROSSGRID – TOOLS AND SERVICES FOR INTERACTIVE GRID APPLICATIONS

MARIAN BUBAK[1,2], MACIEJ MALAWSKI[1],
KATARZYNA RYCERZ[1] AND MICHAŁ TURAŁA[2,3]

[1] *Institute of Computer Science, AGH University of Science and Technology,*
Al. Mickiewicza 30, 30-059 Cracow, Poland
{bubak, malawski, kzajac}@uci.agh.edu.pl

[2] *Academic Computer Centre CYFRONET,*
Nawojki 11, 30-950 Cracow, Poland

[3] *Institute of Nuclear Physics, Polish Academy of Sciences,*
Radzikowskiego 152, 31-342 Cracow, Poland
Michal.Turala@cern.ch

(Received 11 August 2004)

Abstract: The CrossGrid Project aims to develop new Grid services and a tool environment for a representative set of interactive compute- and data-intensive applications. This paper presents the architecture of the software and a description of the functionality of the main components. The new tools and grid services are based on the Globus Toolkit and the EU DataGrid middleware. CrossGrid develops and operates a large international testbed, where the developed applications, services and tools are deployed and run.

Keywords: Grid, architecture, services, interactive applications, Globus Toolkit, OGSA

1. Introduction

The subject of research and development in CrossGrid [1] are new Grid services and tools for interactive compute- and data-intensive applications like the biomedical simulation and visualization for vascular surgical procedures, the flooding crisis team decision support system, distributed data analysis in high energy physics and air pollution combined with weather forecasting. One of the main objectives of the Project is to propose and develop a unified approach to running interactive distributed applications on the Grid, and therefore the following issues are of key importance:

- porting applications to a grid environment;
- development of user interaction services for interactive start-up of applications, online output control, parameter study and runtime steering;
- development of an advanced user interface that anables easy plugin of applications and tools;

- elaboration of a tool for on-line, interactive performance analysis combined with on-line monitoring of grid applications;
- development of a scheduler for distributed interative applications;
- benchmarks and performance prediction for interactive applications;
- optimization of data access for various storage systems.

The architecture of the CrossGrid software was first defined at the very begining of the Project as the result of detailed analysis of application requirements [2] and was recently presented in its original form in overview [3]. During the progress of the Project the architecture was refined [4]. The usage of Globus Toolkit 2.x was decided at the beginning of the Project for stability reasons and because of a close collaboration with the DataGrid project [5]. The tools and services of CrossGrid are complementary to those of DataGrid, GridLab and GrADS [6].

2. Requirements of interactive applications

The biomedical application [7] is a distributed application consisting of three base components: simulation, interaction and visualization. The interaction component allows the user to change simulation parameters in near-real time. Therefore, support for remote steering of simulations run on the Grid is required.

Another type of interactive applications is represented by the flood protection system [8]. An interactive Grid system fulfilling the needs of this application should allow experts to prepare cascades of meteorological, hydrological and hydraulic simulations based on the assumption that each preceding step of the cascade produces the input for the next simulation. After each of the steps is completed, the expert should be allowed to decide whether there is a need for the next simulation step in the cascade to be performed. The main requirement for interaction is the need to support on-line control of cascade execution.

High energy physics and air pollution modeling applications cover various areas from distributed neural network training for HEP [9], weather prediction providing input to air pollution [10] or sea wave model simulations, and data mining using self-organizing maps [11]. The applications require support for on-line output monitoring of their results in order to help operators decide about further job execution (*i.e.* interrupting the execution or letting it finish). The jobs are mainly MPICH-G distributed jobs, making on-line output monitoring even more difficult.

The main requirements of interactive applications can be classified as follows:

- **interactive start-up** – a possibility to start it at a predictable time point;
- **on-line output control** – an ability to control applications' output on-line and to allow the user to decide about canceling it according to the output results;
- **runtime steering** – similar to the above, with an additional requirement of the ability to change the simulation's execution parameters while it is running;
- **quality control** – once the application is controlled by the user, near-real time response is desired and, therefore, we must assure a certain quality of service.

The proposed solutions for these interactivity scenario requiremnents are presented in Section 6.

3. The CrossGrid architecture

The CrossGrid architecture consists of a set of self-contained subsystems which can be divided into layers that contain applications, software development tools and Grid services. The components and layers are shown in Figure 1. The application subsystem represents all the applications introduced in Section 2.

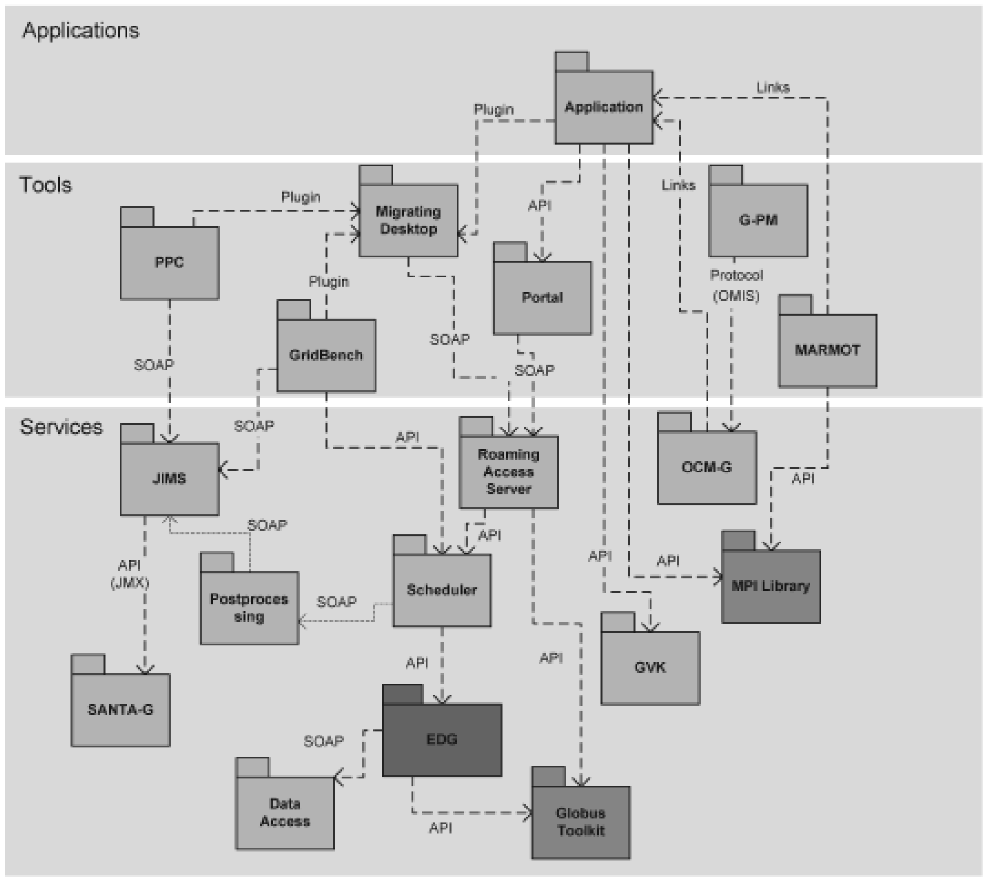

Figure 1. Overview of the CrossGrid architecture

Since the development of interactive applications also requires new developer tools, the second layer contains the following tools: PPC (performance prediction), GridBench (benchmarking on the Grid), G-PM (Grid performance measurement) and MARMOT (MPI verification). There are also the Migrating Desktop and the Portal that are the user interfaces to CrossGrid. An overview of these tools is given in Section 4.

The Grid services layer consists of the following subsystems: a Roaming Access Server (RAS), a Scheduler, OCM-G application monitoring, SANTA-G and JIMS infrastructure monitoring, postprocessing, a Data Access subsystem and GVK (Grid Visualization Kernel). An overview of these services is presented in Section 5.

In addition to the CrossGrid components, we also distinguish the most important external subsystems used, namely the Globus Toolkit [12], the EU DataGrid [5] and MPI libraries. All the subsystems of CrossGrid are developed as independent software components to facilitate the distribution of work between partners, to allow code reusability and exploitation of the produced software in the future. However, there are many connections between these subsystems that make CrossGrid an integrated software environment for Grid application developers and users. These connections are depicted in Figure 1 with a UML dependencies diagram. Arrows run from subsystems that use other CrossGrid modules to those being directly used by them.

Figure 1 also shows the types of interfaces. There are Web service interfaces (SOAP), APIs, Protocols, Plugins and direct library links as interface types.

More detail on the dependencies and the interfaces is included in Sections 4 and 5.

4. The tool environment

Development of Grid applications is a difficult task because of its unpredictable and changing environment and complex structure. Interactivity results in new challenges. The tools developed in CrossGrid aim to provide new solutions for Grid application developers.

MARMOT. MPI is the standard for many parallel applications ported to the Grid environment. The MARMOT [13] MPI verification tool enables not only strict correctness of program code compliance with the MPI standard, but also helps to locate deadlocks and other anomalous situations in the running program. MARMOT is a library that is linked to the MPI application in addition to the existing MPI library and allows a detailed analysis of this application at runtime.

GridBench. The GridBench tool [14] is an important step towards definition and implementation of a standard set of Grid benchmarks. Such benchmarks are important both for site administrators testing resources and for application developers and users who require an overview of performance of their applications on the Grid without actually running them. The GridBench provides reference data for high-level analysis of applications and system parameters for performance prediction.

The Performance Prediction Tool (PPC). The PPC performance prediction tool [15] is tackling the difficult problem of predicting performance in the Grid environment. Through the analysis of application kernels it is possible to derive certain analytical models and use them later for predictions. This task is automated by PPC which allows choosing the available application kernels and site configuration parameters. The output is presented in the form of charts. Predictions are based on monitoring data from NWS or JIMS monitoring systems.

The Performance Measurement Tool (G-PM). To the best of our knowledge, no other online performance analysis tool can currently match G-PM [16]. Using the OCM-G application monitoring system [17], G-PM can display the behaviour of Grid applications in various modes. The displayed measurements can be toggled during the runtime and new metrics can be defined by the user and composed using the newly-defined Performance Metrics Specification Language (PMSL) [18]. This makes G-PM a powerful and flexible tool for advanced performance analysis of applications running in the Grid environment. G-PM consists of three components: a performance measurement component (PMC), a component for high-level analysis (HLAC) and a user interface and visualization component (UIVC). The PMC component provides the functionality for basic performance measurements of both Grid applications and the Grid environment. The results of these basic measurements can be directly visualized by the visualization component. In addition, they can serve as an input to the high-level analysis component and the performance prediction component. The HLAC component supports application- and problem-specific analysis of the performance data. On the one hand, it allows to measure application-specific performance metrics. On the other hand, it provides a specification language which allows combining and correlating different performance measurements in order to derive higher-level metrics. The objective of this approach is to provide more meaningful data to application developers. The UIVC component allows one to request performance measurements and displays the resulting performance information in various graphical ways. The interface to G-PM is provided as a GUI (UIVC). It is an X11 application running on Linux. The input to HLAC is a file containing measurements defined in the Performance Measurement Specification Language (PMSL) [18]. The G-PM tool uses the OCM-G application monitoring service. It connects using the OMIS [19] interface, which is a text-based protocol using a direct TCP connection.

The Migrating Desktop (MD). The Migrating Desktop together with its back-end, the Roaming Access Server (RAS) [20], provides an integrated user front-end to the Grid environment. It is a Java-based GUI (which can be run as an applet) with many features. It can store user profiles and has tools for job submission (Job Wizard), as well as status monitoring and file transfer (Grid Explorer and Grid Commander). It is extensible by means of application and tool plugins, so that it can easily handle application-specific parameters and output (possibly graphical or dynamically updated). It can also provide interactive connections to the application using the VNC protocol [21]. The various CrossGrid applications and tools are integrated with the Migrating Desktop by means of plugins that are designed to act similarly to those used in popular browsers. Application plugins are used to input application-specific input parameters. This enables a portal framework that is independent of application types, simple to extend by adding new applications. Tool plugins are used to modify job parameters according to the needs of specific tools. Each plugin is delivered as a set of Java archive files (available at a network location) from which the Migrating Desktop loads classes dynamically using the Java reflection mechanism [20]. Additionally, it is possible to use MD for interactive application to launch a VNC Server on a RAS machine. A VNC client is integrated with MD. Advanced graphical visualisation tools can thus be used to interact with the application.

The Portal. the Portal, in comparison with the Migrating Desktop, is the lightwight client to CrossGrid resources. It requires only Web browser and offers the simpler functionality of job submission and monitoring. The portal uses the interface of the Roaming Access Server (RAS) through Web services (using SOAP over HTTP).

5. Grid services

New Grid services developed within the Project include the Roaming Access Server, scheduling support for MPI and interactive applications on the Grid, monitoring systems (OCM-G for applications and SANTA-G with JIMS for infrastructure), and optimization of data access.

The Roaming Access Server (RAS). The Roaming Access Server (RAS) [20] is a server for the Migrating Desktop. It stores user profiles, so that users can have the same working environment regardless of where they connect to the Grid. It serves the requests from the Migrating Desktop or the Portal and forwards them to appropriate Grid services (scheduling, data access, *etc.*). It can also act as a VNC server for running interactive applications. This component serves as a gateway for accessing the Grid. RAS exposes its interface as a Web Service. It uses Scheduler API for job submission and GridFTP API for data transfer. It provides interface to MD and the Portal by Web services and uses the interface of the Scheduler via a Java EDG Job Submission Service API and Globus GridFTP via a Java CoG API.

The Scheduler. The CrossGrid scheduling system [22] extends the EDG resource broker through support for MPICH-G2 applications running at multiple sites. The Scheduler is integrated with the DataGrid Job Submission Service (JSS). It improves the following EDG JSS components: Workload Manager, Matchmaker, Resource Broker, Job Adapter and Job Controller [22]. The Scheduler exposes its interface as an EDG JSS Java API, which is used by RAS. This interface allows for job submission and retrieval of its status and output. The Scheduler provides an interface to RAS by a Java EDG JSS API.

Postprocessing. The main objectives of postprocessing are to gather monitoring data from the Grid, such as cluster load and data transfers between clusters, to build a central monitoring service in order to analyze the above data and provide it in the format suitable for the scheduler, which is to be the main task's data consumer, and, finally, to use the prediction built for the first prototype to forecast the collected Grid parameters.

Application Monitoring (OCM-G). The application monitoring system, OCM-G [17], is a unique on-line monitoring system for Grid applications. While other systems rely on application traces, in the on-line system monitoring requests and response events are generated dynamically and can be toggled at runtime. This imposes much less overhead on the application and can thus provide more accurate measurements, *e.g.* for performance analysis tools such as G-PM using the OMIS protocol. A user can launch OCM-G from the Migrating Desktop by means of a tool plugin.

JIMS. The infrastructure monitoring system, JIMS (JMX-based Infrastructure Monitoring System) [23], provides information that cannot be acquired from standard Grid monitoring systems. JIMS can yield information on all cluster nodes and routers or any SNMP devices. It provides infrastructure monitoring data using the Java JMX Technology. It exposes its interface as a SOAP/HTTP Web Service interface, which is going to be used by GridBenchmarks, the Performance Prediction Tool and in Postprocessing. JIMS uses SANTA-G via a JMX interface [24].

SANTA-G. SANTA-G can provide detailed data on network packets and publishes its data into DataGrid R-GMA [25] by using the CanonicalProducer API. The Viewer module provides a Java Swing GUI that presents a graphical interface to users. It allows users to collect data from the R-GMA, by using the Consumer API, and to graphically view packets stored in the log files.

Optimization of Data Access. The data access optimization uses a unique component-expert subsystem to select optimal components to handle various file types and storage systems (*e.g.* tape libraries). The access time estimator extends the functionality of the Reptor replica manager developed in DataGrid by providing access time for various storage systems. All these components are integrated into one package. The data access optimization task also copes with storage heterogeneity. The proposed Unified Data Access Layer (UDAL) [26] simplifies access to Grid storage and makes it more efficient.

The Grid Visualization Kernel. The Grid Visualisation Kernel [27] is a service that provides a visualization engine running in the Grid environment. It distributes the visualization pipeline and uses various optimization techniques to improve the performance of the system. It is used for visualization in medical applications.

6. User interactive services

Interactive start-up. One of the main requirements for the application to be interactive is to provide a possibility to start it in the predictable future. This is done by the scheduler component MPICH-G2 Application Launcher [22] that provides support for executing MPI applications on a distributed set of Grid nodes. The Laucher has been built on top of Condor-G [28]. Globus DUROC library [12] was used to guarantee that all the subjobs are synchronized at the beginning of its execution.

On-line output control. Once the application is started, it is desirable to control its output on-line and to allow the user to decide about canceling it according to the output results. This can be achieved with streaming techniques presented in [20]. After job submission, the streams (in, out and error) of a job running on the Grid have to be sent to a VNC Server machine to allow a user to see the output on-line. This is done via a Console Agent (CA) that is a process launched together with the user job. The CA opens a connection towards a Job Shadow (JS) running on the VNC server machine. The Job Shadow receives the job streams from the CA and sends them to the user via VNC.

Runtime steering. Runtime steering is similar to on-line output control with the additional requirement of the ability to change the simulation execution parameters while it is running. This requirement can be fulfilled by using the High Level Architecture (HLA) [29]. The CrossGrid medical application uses the agent system [30] built over HLA that supports development and execution of interactive distributed components (namely simulation, interaction and visualization). Also, there is an experimental attempt to port HLA-based applications to the Grid. The experimental system based on the OGSA [31] concept has been partly implemented and the results can be found in [32], where we have proposed and partially implemented an OGSA-based system that supports running a distributed interactive application based on the HLA standard in a Grid environment.

Quality control. Once the application is controlled by the user, near-real time response is desired. Therefore, we must assure a certain quality of service. Although a CPU reservation mechanism is assumed, there remains the issue of network performance, which can be resolved by the diffserv [33] (or other network QoS) mechanism and is under consideration. Another issue appears when the network changes its state. This can be partly solved by migration. An experimental system supporting migration of HLA-based simulations has been implemented and the results are described in [32]. It assumes that if the network stays busy for some time and the responses take longer, it is better to spend some time on migration to another site, which may provide better response times. It is also less annoying to wait once for migration than to wait for each response.

7. Prototype and testbed

The Project has entered a mature phase with the deployment of software prototypes. The main objective of the testbed is now to support them in an adequate and stable Grid framework. The main activity is oriented towards integration, including deployment and feedback collection. The CrossGrid testbed [34, 1] is now composed of 18 sites in 9 countries supporting three testbed environments. The current basic middleware version is LCG-2. Most sites offer storage capacities below 60GB. The hardware type mostly ranges from Pentium III to Pentium Xeon based systems, with RAM memories between 256MB and 2GB. Many sites offer dual CPU systems. The operating system at most sites is RedHat 7.3.

8. Conclusions and future work

The paper comprises an in-depth analysis of the current state of the CrossGrid architecture following the integration workshop which took place in Dagstuhl (July 2004). It follows from the analysis that the main requirements of interctive applications are successfully designed and implemented and the architecture has been solidified.

From the very beginning, CrossGrid has been evaluating [35] the possibility of migrating to the new OGSA Globus Toolkit and the necessity of gearing the scientific work towards the service-oriented architecture [31, 36]. Web Services and XML are extensively used in many CrossGrid components, hence the possible future migration to OGSA should be easy. In the Project, there is also ongoing research in the area of Grid Services or Service-Oriented Architectures, which means that effort invested in the development of applications, tools and services within CrossGrid could be used in the future Grid or industrial environments, based on the currently emerging standards.

Acknowledgements

The authors wish to thank the CrossGrid Collaboration for their contribution and Mr. Piotr Nowakowski for his comments. This research is partly funded by the European Commission IST-2001-32243 Project "CrossGrid" and the Polish Committee for Scientific Reaearch SPUBM 112/E-356/SPB/5.PR UE/DZ224/2002-2004.

References

[1] CrossGrid Project Homepage, http://www.crossgrid.org/
[2] Bubak M, Malawski M and Zając K 2002 *Recent Advances in Parallel Virtual Machine and Message Passing Interface, Proc. 9th European PVM/MPI Users' Group Meeting*, Linz, Austria, LNCS **2474** 16

[3] Berman F, Fox G and Hey T (Eds.) 2003 *Grid Computing. Making the Global Infrastructure a Reality*, Wiley, pp. 873–874

[4] Bubak M, Malawski M and Zając K 2003 *Proc. Computational Science – ICCS 2003, Int. Conf.*, Melbourne, Australia and St. Petersburg, Russia, LNCS **2657** (Part I) 207

[5] DataGrid Project Homepage, http://www.eu-datagrid.org/

[6] Gerndt M (Ed.) 2004 *Performance Tools for the Grid: State of the Art and Future*, APART White Paper, Shaker Verlag, ISBN 3-8322-2413-0

[7] Sloot P M A, Tirado-Ramos A, Hoekstra A G and Bubak M 2004 *Proc. 2nd Int. Workshop on Biomedical Computations on the Grid (BioGrid'04)* in conjunction with *4th IEEE/ACM Int. Symposium on Cluster Computing and the Grid (CCGrid2004)*, Chicago, Illinois, USA (accepted)

[8] Hluchy L, Tran V, Simo B, Habala O, Astalos J and Gatial E 2004 *Proc. 2nd European Across Grids Conf.*, Nicosia, Cyprus (to appear)

[9] Ponce O, Cuevas J, Fuentes A, Marco J, Marco R, Martinez-Rivero F, Menendez R and Rodriguez D 2002 *Recent Advances in Parallel Virtual Machine and Message Passing Interface, Proc. 9th European PVM/MPI Users' Group Meeting*, Linz, Austria, LNCS **2474** 33

[10] Mourino J C, Martin M J, Gonzalez P, Boullon M, Cabaleiro J C, Pena T F, Rivera F F and Doallo R 2004 *Proc. 1st European Across Grids Conf.*, Santiago de Compostela, Spain, LNCS **2970** 155

[11] Luengo F, Cofino A and Gutierrez J M 2003 *Proc. 1st European Across Grids Conf.*, Santiago de Compostela, Spain, LNCS **2970** 163

[12] Globus Project Homepage, http://www.globus.org/

[13] Krammer B, Mueller M and Resch M 2004 *Proc. Computational Science, ICCS 2004, Int. Conf.*, Cracow, Poland, LNCS **3038** (Part III) 464

[14] Tsouloupas G and Dikaiakos M 2003 *Proc. 4th Int. Workshop on Grid Computing (Grid2003)*, Phoenix, Arizona, IEEE Computer Society, pp. 60–67

[15] Blanco V, Gonzalez P, Cabaleiro J C, Heras D B, Pena T F, Pombo J J and Rivera F F 2003 *Future Generation Computer Systems* **19** (5) 721

[16] Bubak M, Funika W and Wismüller R 2002 *Recent Advances in Parallel Virtual Machine and Message Passing Interface, Proc. 9th European PVM/MPI Users' Group Meeting*, Linz, Austria, LNCS **2474** 50

[17] Baliś B, Bubak M, Funika W, Szepieniec T and Wismüller R 2003 *Proc. Computational Science – ICCS 2003, Int. Conf.*, Melbourne, Australia and St. Petersburg, Russia, LNCS **2657** (Part I) 214

[18] Wismüller R, Bubak M, Funika W, Arodź T and Kurdziel M 2004 *Proc. AxGrids: 2nd European Across Grids Conf.*, Nicosia, Cyprus, LNCS (in print)

[19] On-line Monitoring Interface Specification, version 2.0, http://wwwbode.cs.tum.edu/∼omis/

[20] Kupczyk M, Lichwala R, Meyer N, Palak B, Plociennik M, Stroiski M and Wolniewicz P 2004 *Proc. Computational Science, ICCS 2004, Int. Conf.*, Cracow, Poland, LNCS **3036** (Part I) 91

[21] VNC, http://www.realvnc.com/what.html

[22] Heymann E, Senar M A, Fernández A and Salt J 2004 *Proc. 2nd European Across Grids Conf.*, Nicosia, Cyprus (to appear)

[23] Ławniczek B, Majka G, Słowikowski P, Zieliński K and Zieliński S 2003 *Electronic Notes in Theoretical Computer Science* **82** (6)

[24] JMX Technology, http://java.sun.com/products/JavaManagement/

[25] Cooke A, Gray A, Ma L, Nutt W, Magowan J, Oevers M, Taylor P, Byrom R, Field L, Hicks S, Leake J, Soni M, Wilson A, Cordenonsi R, Cornwall L, Djaoui A, Fisher S, Podhorszki N, Coghlan B, Kenny S and O'Callaghan D 2003 *Proc. 11th Int. Conf. on Cooperative Information Systems*, Catania, Sicily, Italy, LNCS **2888** 462

[26] Stockinger K, Stockinger H, Dutka Ł, Słota R, Nikolow D and Kitowski J 2003 *Proc. 4th Int. Workshop on Grid Computing (Grid2003)*, Phoenix, Arizona, IEEE Computer Society Press, pp. 149–157

[27] Tirado-Ramos A, Ragas H, Shamonin D, Rosmanith H and Kranzmueller D 2004 *Proc. 2nd European Across Grids Conf.*, Nicosia, Cyprus (in print)

[28] Condor-G, http://www.cs.wisc.edu/condor/condorg

[29] HLA Specification, http://www.sisostds.org/stdsdev/hla/

[30] Zhao Z, van Albada G D, Tirado-Ramos A, Zając K and Sloot P M A 2003 *Proc. Computational Science, ICCS 2003, Int. Conf.*, Melbourne, Australia and St. Petersburg, Russia, LNCS **2657** (Part I) 679

[31] Foster I, Kesselman C, Nick J M and Tuecke S 2002 *The Physiology of the Grid. An Open Grid Services Architecture for Distributed Systems Integration*, http://www.globus.org/

[32] Zając K, Bubak M, Malawski M and Sloot P 2003 *Proc. 7th IEEE Int. Symposium on Distributed Simulation and Real Time Applications*, Delft, The Netherlands, IEEE Publishing, pp. 4–11

[33] Diffserv Mechanism Homepage, http://diffserv.sourceforge.net/

[34] Gomes J, David M, Martins J, Bernardo L, Marco J, Marco R, Rodriguez R, Salt J, Gonzalez S, Sanchez J, Fuentes A, Hardt M, Garcia A, Nyczyk P, Ozieblo A, Wolniewicz P, Bluj M, Nawrocki K, Padee A, Wislicki W, Fernandez C, Fontan J, Gomez A, Lopez I, Cotro Y, Floros E, Tsouloupas G, Xing W, Dikaiakos M, Astalos J, Coghlan B, Heymann E, Senar M, Merino G, Kanellopoulos C and van Albada G D 2003 *Proc. 1st European Across Grids Conf.*, Santiago de Compostela, Spain, LNCS **2970** 67

[35] Malawski M, Bubak M and Zając K 2002 *Proc. Cracow'02 Grid Workshop*, Cracow, Poland, ACC Cyfronet AGH, pp. 140–147

[36] The WS-Resource Framework, http://www.globus.org/wsrf/