European Research Network on Foundations, Software Infrastructures and Applications
for large scale distributed, GRID and Peer-to-Peer Technologies

A Network of Excellence funded by the European Commission

Reliable Orchestration of distributed MPI-Applications in a UNICORE-based Grid with MetaMPICH and MetaScheduling

Boris Bierbaum, Carsten Clauss
{boris, carsten}@lfbs.RWTH-Aachen.DE
Chair for Operating Systems, RWTH Aachen University
D-52056 Aachen, Germany

Thomas Eickermann, Lidia Kirtchakova
{th.eickermann, l.kirtchakova}@fz-juelich.de
Central Institute for Applied Mathematics, Research Centre Jülich
D-52425 Jülich, Germany

Arnold Krechel, Stephan Springstubbe, Oliver Wäldrich, Wolfgang Ziegler
{Arnold.Krechel, Stephan.Springstubbe, Oliver.Waeldrich, Wolfgang.Ziegler}@scai.fhg.de
Fraunhofer Institute SCAI
D-53754 Sankt Augustin

CoreGRID Technical Report
Number TR-0052
August 19, 2006

Institute on Resource Management and Scheduling

CoreGRID - Network of Excellence
URL: http://www.coregrid.net

CoreGRID is a Network of Excellence funded by the European Commission under the Sixth Framework Programme

Project no. FP6-004265

Reliable Orchestration of distributed MPI-Applications in a UNICORE-based Grid with MetaMPICH and MetaScheduling

Boris Bierbaum, Carsten Clauss
{boris, carsten}@lfbs.RWTH-Aachen.DE
Chair for Operating Systems, RWTH Aachen University
D-52056 Aachen, Germany

Thomas Eickermann, Lidia Kirtchakova
{th.eickermann, l.kirtchakova}@fz-juelich.de
Central Institute for Applied Mathematics, Research Centre Jülich
D-52425 Jülich, Germany

Arnold Krechel, Stephan Springstubbe, Oliver Wäldrich, Wolfgang Ziegler
{Arnold.Krechel, Stephan.Springstubbe, Oliver.Waeldrich, Wolfgang.Ziegler}@scai.fhg.de
Fraunhofer Institute SCAI
D-53754 Sankt Augustin

CoreGRID TR-0052
August 19, 2006

Abstract

Running large MPI-applications with resource demands exceeding the local site's cluster capacity could be distributed across a number of clusters in a Grid instead, to satisfy the demand. However, there are a number of drawbacks limiting the applicability of this approach: communication paths between compute nodes of different clusters usually provide lower bandwidth and higher latency than the cluster internal ones, MPI libraries use dedicated I/O-nodes for inter-cluster communication which become a bottleneck, missing tools for co-ordinating the availability of the different clusters across different administrative domains is another issue. To make the Grid approach efficient several prerequisites must be in place: an implementation of MPI providing high-performance communication mechanisms across the borders of clusters, a network connection with high bandwidth and low latency dedicated to the application, compute nodes made available to the application exclusively, and finally a Grid middleware glueing together everything. In this paper we present work recently completed in the VIOLA project: MetaMPICH, user controlled QoS of clusters and interconnecting network, a MetaScheduling Service and the UNICORE integration.

Keywords: MetaMPICH, Grid, Co-allocation, UNICORE, Network QoS

1 Introduction

1.1 The VIOLA project

The work presented here is carried out in the context of VIOLA [16] (Vertically Integrated Optical testbed for Large Applications), a co-operative project with a consortium of 12 partners from German research labs, universities and

This research work is carried out under the FP6 Network of Excellence CoreGRID funded by the European Commission (Contract IST-2002-004265).

industry, lead by DFN, the German NREN. VIOLA is funded by the German federal ministry of education and research BMBF. The project has set up an optical testbed in North-Rhine-Westfalia with an extension to Bavaria. Main objectives are evaluation and testing of advanced networking equipment and technologies in a close-to-production environment and development of software for user-driven dynamical provisioning of network bandwidth and quality-of-service. A set of initially four applications with high communication demands has been selected to provide real-life requirements and to stress-test the network. Three of them are performing distributed simulations with MPI-based codes on the currently five Linux- and Solaris-based clusters in the testbed.

The clusters are attached to the testbed with multiple Gigabit-Ethernet adapters, in most cases one adapter in each node of the cluster. The clusters are interconnected over the testbed via 10 Gigabit-Ethernet. Given this complex and bandwidth-rich environment, it is obvious that a scalabe high-performance MPI-implementation with wide-area support is a prerequisite for efficient use of the testbed. Also, Grid middleware is required to orchestrate the various resources and to provide reliable, secure and seamless access to them. For the former, the RWTH has extended their Metacomputing MPI-implementation MetaMPICH [11], for the latter we have integrated a MetaScheduling Service into the Grid system UNICORE [14]. In VIOLA the MetaScheduling Service does not orchestrate Web Services as the applications are not wrapped in services and the orchestration is made for a synchronous start. However, as described in Section 6 future versions of the MetaScheduling service will also support workflows - or choreography - of two or more Web Services.

1.2 Related work

Besides MetaMPICH, there are other MPI implementations enabling the coupling of compute resources over wide-area networks, most notably PACX-MPI [3], MPICH/Madeleine [2], and MPICH-G2 [9]. The features that differentiate MetaMPICH from some or all of these approaches are the startup mechanism using a single configuration file, the choice between two different methods to couple clusters (*routers* and *multidevice*), and the fact that it is not tied to a specific grid system.

There are also a number of approaches for co-allocation of resources like KOALA, CSF, GridWay or products like MP Synergy or Moab. However, most of them are not providing advance reservation and neither reliable SLAs nor co-allocation of compute resources with the interconnecting network guaranteeing a user-requested QoS.

In the UNICORE Plus project [6] a proof-of-concept implementation of a MetaScheduling Service based on a proprietary negotiation protocol has been implemented. It supports PACX-MPI as MPI library, the the CCS (Computing Center Software) [13] as reservation system for advance reservation of compute resources. Our system uses some of the ideas of the UNICORE Plus development (e.g. making the functionality accessible via a UNICORE client plugin), but is besides that a completely independent design and implementation.

Related projects based on optical Grid testbeds are e.g. the Japanese g-lambda project or the Polish CLUSTERIX project. These projects use different middleware and have a different focus, but co-operation has been launched to exchange developments made and to work on interoperability.

1.3 Remainder of the Paper

The remainder of the paper is organised as follows. In Section 2 we present the MetaMPICH library developed at RWTH. The VIOLA MetaScheduling environment is described in Section 3, followed by the description of the MetaMPICH integration in both the MetaScheduling Service and the UNICORE system in Section 4. Experiences made are discussed along a use case we present in Section 5. An overview about further developments for the MetaScheduling environment and the MetaMPICH library in Section 6 concludes the paper.

2 MetaMPICH

Based on MPICH1 [8], MetaMPICH was originally developed to couple MPPs from different vendors in the Gigabit Testbed West project [7]. Since those systems internally had very fast networks, but only dedicated I/O-nodes for external communication, a router-based communication architecture was chosen, as depicted in the left part of Figure 1. We call each of those coupled systems a *meta host* in the context of meta computing.

In a second stage, MetaMPICH was extended to support the emerging class of PC-based cluster systems with high-performance interconnects like SCI [17]. MetaMPICH has been optimised for coupling such clusters, as published in

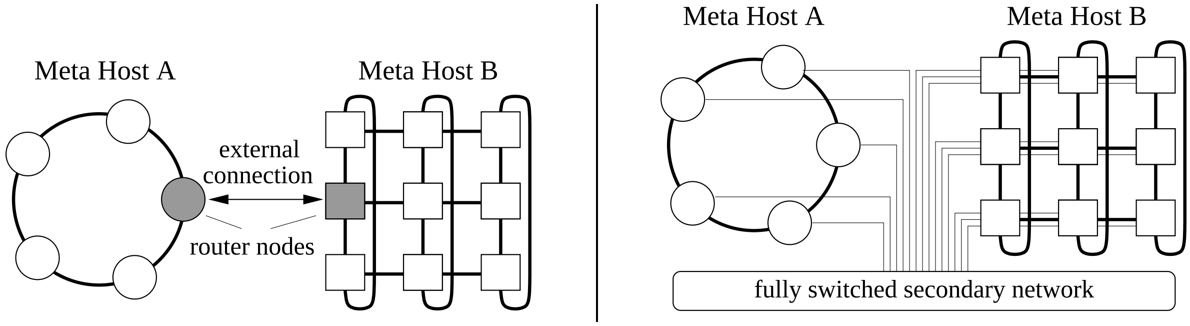

Figure 1: Communication Architectures of MetaMPICH

[11]. One advantage of the router approach was the possibility to couple an arbitrary number of cluster nodes via fast, dedicated external connections to achieve higher scalability and higher bisectional bandwidth between the systems.

However, when coupling clusters via the high-bandwidth VIOLA network, the communication performance between clusters is limited by the speed with which the I/O-nodes can handle the traffic. Since all compute nodes within the VIOLA project are connected to the VIOLA WAN, it becomes preferable to let the application processes communicate directly with each other. To enable this, we implemented a new architecture for MetaMPICH.

The result is shown in the right part of Figure 1. It is called the *multidevice* architecture, because it enables the usage of multiple MPICH communication drivers (called *devices*) side-by-side. Note that every node of system A can send data directly to every node of system B and vice versa. That way, this approach allows to run large applications that benefit from the dedicated internal cluster networks and from the connecting high-performance optical network at the same time. Nevertheless, in order not to lose the flexibility of a router driven communication, which is the only choice in some environments, MetaMPICH also supports setups combining router-based and multidevice coupling.

The needs of the VIOLA project also led to several other improvements of MetaMPICH: Support for Myrinet was added by integrating code from the MPICH-GM distribution. The device for TCP/IP communication, `ch_usock`, was made *instantiable* to be able to use it for cluster-internal communication as well as for coupling clusters at the same time. The syntax of the *meta configuration file* [12], with which a coupled system is configured, was extended to support several new requirements, e.g. automated startup of server processes for remote parallel I/O.

3 VIOLA MetaScheduling Environment

The MetaScheduling Service (MSS) has beed developed to ensure that all resources necessary for executing the distributed applications are available. The MSS receives the information on resources needed for an application from the UNICORE client via an agreement proposal [1] containing the specification of resources and QoS. The MSS then starts the negotiation process with the local Resource Management Systems (RMS) of these resources, where the compute resources are managed by the local scheduling systems and the network resources by the ARGON (Allocation and Reservation in Grid-enabled Optic Networks) system. Due to the heterogenous nature of the employed RMS we used a set of adapters to suport the MSS during the negotiation process by providing a stable interface to the different RMS (see Fig. 2). The negotiation process consits of four main phases:

1. querying the local RMS for free slots to execute the application within a preview period
2. determining a common time slot
3. if such a time slot exists, perform a reservation request of this slot on behalf of the user;
   otherwise restart the query with a later start time of the preview period
4. check whether the reservation was made for the correct time slot on all systems, if yes, we are done;
   otherwise restart the query with a later start time of the preview period.

The successful negotiation and reservation is sent back as agreement to the UNICORE client which then continues processing the job as usual. Once the job starts at the negotiated starting time the MSS collects the IP addresses of the compute nodes finally allocated by the local RMS. The IP addresses are used to generate the meta-configuration file as described in Section 4 below and are communicated to the network RMS ARGON which in turn is then able to manage the end-to-end connections with the requested QoS.

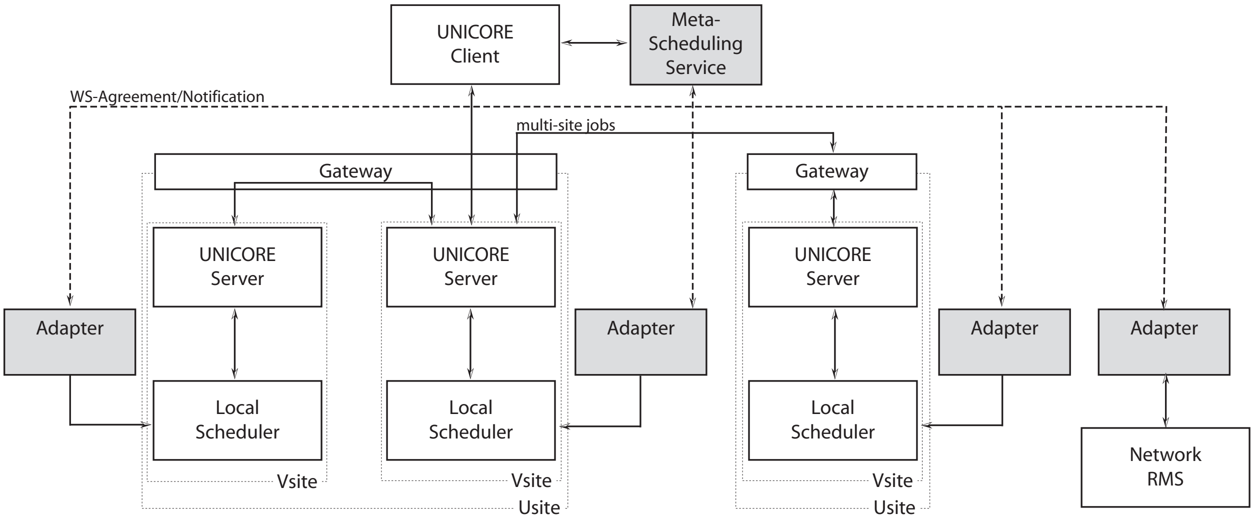

Figure 2: The meta-scheduling architecture

4 Grid middleware integration

4.1 Integration into the MetaScheduling Service

In this section we describe how the MSS supports the preparation of the runtime environment for MetaMPICH applications. As mentioned in Section 3 an important functionality of the MSS is the coordinated allocation of multiple resources (e.g. compute and network) at different sites. Compute reservations mainly consist of the number of required nodes at a compute site, the duration, an executable and the start time for the reservation. Network reservations specify a network service, which consists of a set of point-to-point connections, the bandwidth for each connection, the duration and a start time of the service. Further on, each connection is specified by two connection endpoints (source and destination), where each endpoint is associated with one compute site in the VIOLA network, respectively with the router that represents this compute site in the network.

The process of negotiating the execution time for a MetaMPICH application allocating the required resources was already described in Section 3. After submission of all reservations the MSS continuously monitors the partial reservations. When all reservations entered the state running (active), the MSS queries the compute RMS in order to determine which nodes (IP addresses of the nodes) where finally assigned to a reservation. This information is collected and aggregated by the MSS, and is then published as runtime configuration to all subsystems.

Publishing this runtime configuration to the network RMS ARGON comprises the completion of the reservation data in terms of which compute nodes (list of IP addresses) belong to each connection endpoint defined in the reservation. This address data is used at the network layer to create access control lists (ACL) at the routers in order to enable the compute nodes belonging to a MetaMPICH job to communicate with each other using the QoS level specified in the network reservation. Therefore the ARGON system implements a bind functionality that allows completing reservation data of existing reservations at runtime.

The runtime configuration data is used in a different way for compute resources. Here an XML configuration file is created on each cluster, which contains the nodes (a list of IP addresses) that belong to a reservation for every site. This configuration file then is used together with the job description submitted by UNICORE to generate a MetaMPICH configuration during the startup process of the application, as further described in Section 4.2.

4.2 UNICORE integration

As the underlying Grid software, UNICORE is responsible for several tasks during the lifetime of a MetaMPICH application:

- providing a user interface for specification of the Meta-Job,
- interacting with the MetaScheduling Service to allocate the requested resources,
- management of the job: start and monitor the sub-jobs on the individual clusters, retrieve and present the job output.

The first and second task are performed by the UNICORE user-client by means of a Metacomputing-plugin, developed specifically for this purpose. The third is one of the core UNICORE server responsibilities. Managing MetaMPICH jobs did not require changes of this server, but just some user-level wrapper scripts for starting the MPI-application.

The Metacomputing-plugin is an extension of UNICORE's graphical user-client, that lets the user specify the MetaMPICH job in a convenient way: In a main panel, the user specifies the duration and favored start time of the job and selects the clusters, on which the individual sub-jobs shall run. In a communication matrix, the number of MetaMPICH router-pairs (which defaults to 0) and the required bandwidth between each pair of clusters can be specified. Then for each sub-job, the user enters the executable, the number of MPI-tasks and various other optional configuration parameters in a separate form.

The job description entered via the plugin is sent along with the job encoded in XML. At job startup time, when the actually allocated nodes are known, the MetaMPICH configuration-file is created on each of the participating clusters, based on the XML job description and the IP-addresses provided by the MetaScheduling system.

The startup of the MetaMPICH application is also different from the standard way, where a single execution of 'mpirun' will start the sub-jobs on all clusters via ssh. In the UNICORE integrated version, each MetaMPICH sub-job is represented as a UNICORE sub-job and is started individually by the local scheduling system of the cluster. The advantages are that no ssh-logins between the clusters are required and that the sub-jobs can be monitored individually by UNICORE.

5 Use case: Distributed Algebraic Multi Grid solver

Solving huge, linear, sparse systems of equations is an important subtask in many simulation codes, e.g. of computational fluid mechanics, structural mechanics or semiconductor device simulation [15], [4]. Typically, the efficiency of a such simulation code is restricted by the efficiency of the linear solver used. Algebraic multigrid methods provide a well established, state-of-the-art solver technology for wide classes of applications. They are optimal since they turn out to be numerically scalable : the time for solving a problem in a certain class grows only linearly with the problem size. The AMG solver technology has been made available in the VIOLA Grid for all simulation codes where the

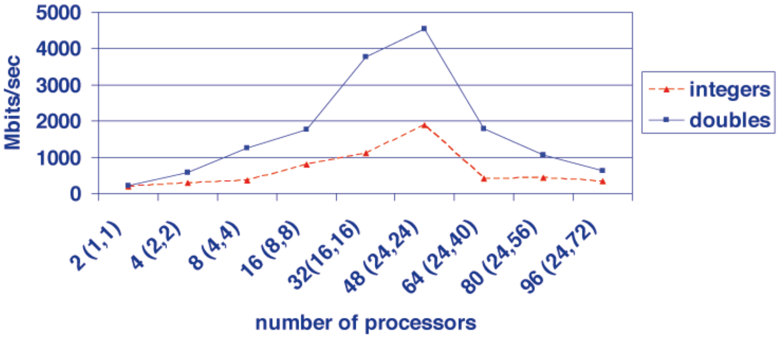

Figure 3: Throughput for data redistribution on two clusters. The throughput for integers is lower than for the doubles as more messages of smaller length have to be sent for the integer data.

solver of a linear system is a numerical bottleneck. We have shown that the VIOLA Grid is suitable for these industrial simulation codes.

Especially, when considering problem sizes which are expected to become relevant in the nearest future. The parallelization of algebraic multigrid methods requires various communication patterns and therefore is a real challenge for the network and the communication software. When starting the solver process, huge amounts of data describing the problem to be solved have to be distributed to the computing MPI processes. The most important factor here is the transfer rate of the network. After having distributed the problem data, a hierarchy of coarse and fine grids has to be calculated. In this setup phase the network latencies become more important. The reason is an increased number of messages which are at the same time of much smaller length than in the redistribution phase. For the same reason the network latencies become the most important in the solution phase.

An existing parallel, MPI-based algebraic multigrid code has been ported to the VIOLA Grid using MetaMPICH. It could be demonstrated that the throughput of the network can be exploited for the program phase dealing with the redistribution of the problem data, provided that enough processors are used. In addition, the timings show that in the Compute Grid VIOLA the redistribution of a sparse matrix problem is always an option: typically, the time of the redistribution of a larger problem is strictly less than the time for solving it.

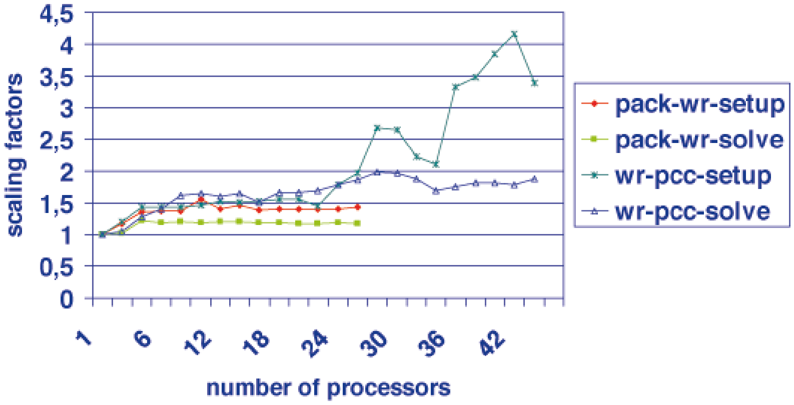

Figure 4: Scaling test using two clusters.

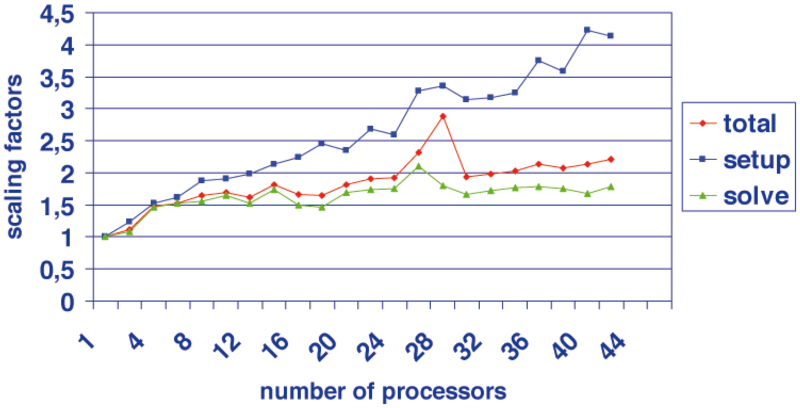

Figure 5: Scaling test using three clusters.

Fig. 4 and 5 show the results of scaling tests for AMG when connecting two and three PC-Clusters respectively using MetaMPICH. The PC-Clusters are up to 80 km apart. For a scaling test the problem size per processor is kept fixed. Scaling factors are defined by dividing the time of the larger problem on the larger number of processors by the time used for the smaller problem on one processor. Therefore, scaling factors of one are ideal, whereas scaling factors of n imply a restriction to a factor of 1/n of the maximum efficiencies that can be anticipated for the larger problem. The scaling factors (Fig. 5) for the total time and the solving time are very satisfactory. The scaling factors for the setup however indicate that restriction of communication in the setup will have to be investigated further [10]. The total scaling factors are not bogged down by the scaling factors of the setup as the time for the solution phase is dominating

6 Future Perspectives

Methods improving the interoperability of MetaMPICH are currently in development, namely a new device that can use other MPI implementations for communication inside a meta host and support for the Interoperable MPI (IMPI) standard. To improve application performance, the implementation of optimized collective communication operations for wide-area networks and support for MPI process topologies are planned.

The results of the VIOLA project wrt orchestration of services including support for user-driven dynamical provisioning of network bandwidth and quality-of-service will be adopted and made available on a European level in the EU funded PHOSPHORUS project. While the current version of the VIOLA Grid testbed expects the user to describe the resource demands of his application using the UNICORE client and do a pre-selection of resources satisfying this demand, we are working in several other projects to have applications providing this information. E.g. together with the Swiss EPFL an interface for a resource Broker responsible for generating a candidate set of potentially suitable resources to run an application has been defined and is currently implemented [5].

Furthermore, the MetaScheduling Service will be made available for GT4 or gLite based Grid environments in the near future. It will then support reliable orchestration and reservation across Grids based on different middleware.

7 Acknowledgements

Some of the work reported in this paper is funded by the German Federal Ministry of Education and Research through the VIOLA project under grant #01AK605L. This paper also includes work carried out jointly within the CoreGRID Network of Excellence funded by the European Commission’s IST programme under grant #004265.

References

[1] A. Andrieux and K. Czajkowski and A. Dan and K. Keahey and H. Ludwig and T. Nakata and J. Pruyne and J. Rofrano and S. Tuecke, and M. Xu. WS-Agreement - Web Services Agreement Specification, 2006. 19 April 2006 <https://forge.gridforum.org/projects/graap-wg/document/WS-AgreementSpecificationDraft.doc/en/31>.

[2] O. Aumage and G. Mercier. MPICH/MADIII: a Cluster of Clusters Enabled MPI Implementation. In *Proceedings of the 3rd IEEE/ACM International Symposium on Cluster Computing and the Grid (CCGrid 2003)*, pages 26–36, Tokyo, Japan, 2003.

[3] T. Beisel, E. Gabriel, M. Resch, and R. Keller. Distributed Computing in a Heterogeneous Computing Environment. In *Recent Advances in PVM and MPI – Lecture Notes in Computer Science*, pages 180–187, 1998.

[4] T. Clees. AMG Strategies for PDE Systems with Applications in Industrial Semiconductor Simulation., 2005. Fraunhofer Series n Information and Communication Technology, vol. 6. Fraunhofer SCAI, Sankt Augustin, Germany.

[5] K. Cristiano, R. Gruber, V. Keller, P. Kuonnen, S. Maffioletti, N. Nellari, M.-C. Sawley, M. Spada, T.-M. Tran, O. Wäldrich, Ph. Wieder, and W. Ziegler. Integration of ISS into the VIOLA Meta-scheduling Environment. In *Proc. of the 2nd CoreGRID Integration Workshop*, volume 4 of *CoreGRID Series*, pages 47–54. Springer, 2006. To appear.

[6] D.Erwin (eds). UNICOREi Plus Final Report. Technical report, Research Center Jülich, Germany, 2003. ISBN 3-00-011592-7.

[7] T. Eickermann, R. Völpel, and P. Wunderling. Gigabit Testbed West – Final Report. Technical report, Research Center Jülich, Germany, 2000.

[8] W. Gropp, E. Lusk, and A. Skjellum. A High-Performance, Portable Implementation of the MPI Message Passing Interface Standard. *Parallel Computing*, 22(6):789–828, 1996.

[9] N. Karonis, B. Toonen, and I. Foster. MPICH-G2: A Grid-Enabled Implementation of the Message Passing Interface. *Journal of Parallel and Distributed Computing (JPDC)*, 63(5):551–563, 2003.

[10] A. Krechel and K. Stüben. Parallel algebraic multigrid based on subdomain blocking. *Parallel Computing*, 27:1009–1031, 2001.

[11] M. Pöppe, S. Schuch, and T. Bemmerl. A Message Passing Interface Library for Inhomogeneous Coupled Clusters. In *Proc. of CAC Workshop at IPDPS'03*, 2003.

[12] M. Pöppe, S. Schuch, R. Finocchiaro, C. Clauss, and J. Worringen. MP-MPICH User Documentation and Technical Notes., 2005. Aachen: Lehrstuhl für Betriebs-systeme, RWTH Aachen.

[13] F. Ramme, T. Romke, and K. Kremer. A Distributed Computing Center Software for the Efficient Use of Parallel Computer Systems. In *Proc. of HPCN Europe, Int. Conf. on High-Performance Computing and Networking 1994*, volume 797 of *Lecture Notes in Computer Science*, pages 129–136. Springer, 1994.

[14] A. Streit, D. Erwin, Th. Lippert, D. Mallmann, R. Menday, M. Rambadt, M. Riedel, M. Romberg, B. Schuller, and Ph. Wieder. UNICORE - From Project Results to Production Grids. In *L. Gandinetti (Edt.), Grid Computing: The new Frontiers of High Performance Processing, , Advances in Parallel Computing 14, Elsevier*, 2005.

[15] K. Stüben. A Review of algebraic multigrid. *Comp. Appl. Math.*, 128:281–309, 2001.

[16] VIOLA – Vertically Integrated Optical Testbed for Large Application in DFN, 2006. 29 March 2006 <http://www.viola-testbed.de/>.

[17] J. Worringen. SCI-MPICH: The Second Generation. In *Proceedings of SCI-Europe 2000 (Conference Stream of Euro-Par 2000)*, pages 11–20, Munich, Germany, 2000.