INTERACTIVE GRID COMPUTING: ADAPTING HIGH LEVEL ARCHITECTURE-BASED APPLICATIONS TO THE GRID

KATARZYNA RYCERZ[1,2], MARIAN BUBAK[1,2], MACIEJ MALAWSKI[1,2] AND PETER SLOOT[3]

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

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

[3] *Faculty of Sciences, Section Computational Science, University of Amsterdam, Kruislaan 403, 1098 SJ Amsterdam, The Netherlands*
sloot@science.uva.nl

(Received 26 July 2004)

Abstract: In this paper we present the design of a Grid HLA Management System (GHMS) supporting execution of High Level Architecture-based interactive applications in a Grid environment, while at the same time allowing for running HLA legacy codes to preserve backward compatibility with the HLA standard. In particular, we describe the following system components: an *HLA-Speaking Service* for multiple federates that interface the HLA application to the system, a *Monitoring Service* integrated with the OCM-G monitoring system and HLA-based *Benchmark Services* informing the *Broker Service* of what can be expected from the application's behavior.

Keywords: interactive simulation, Grid Computing, HLA, OGSA

1. Introduction

Interactive Grid Computing is a very interesting aspect of modern computer science. The definition of interactivity is very broad and a possible Grid architecture to support such applications can be found in [1]. In this paper we focus on applications composed of distributed elements, where one or more elements are interaction components that interface with a human and thus their behavior cannot be easily predicted. Interaction components are used to steer the simulation components of the application in near-real time. We assume that communication between the components is performed using one of the most common standards for distributed interactive applications, *viz.* the High Level Architecture (HLA) [2]. The conceptual architecture of

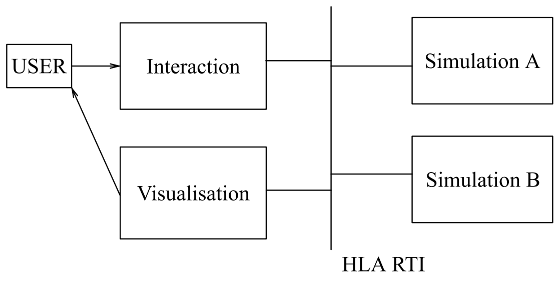

Figure 1. The conceptual architecture of a distributed interactive application

such an application is shown in Figure 1. For example, we refer the reader to the ISS-Conductor [3] system that is a piece of middleware built over HLA RTI for interactive applications such as the CrossGrid medical application [4].

This kind of applications usually requires extensive computing resources. The Grid [5, 6] offers an opportunity of better and more convenient usage of distributed resources that were previously inaccessible. However, the existing distributed simulation community standards for large-scale simulations are not yet suitable for direct usage on the Grid.

HLA [2] supports development of simulations (or federations in the HLA terminology) from distributed components (or federates). It provides the simulation developer with many useful features such as time and data management required by time- or event-driven distributed simulations. However, the HLA standard was developed assuming a certain quality of service in the underlying environment of simulation execution. It is based on the assumption that the environment it executes on is faultless. By definition, a Grid environment is a set of shared resources that cannot be strictly dedicated to near-real time applications and therefore the behavior of such applications can vary according to circumstances. Moreover, the Grid is unreliable and requires fault tolerance mechanisms.

This paper is organized as follows: in Section 2 we provide a brief overview of HLA, in Section 3 we outline related work, and in Section 4 we present the overall architecture of the system. In the next three sections we describe its essential components. The service that interfaces HLA application to the system is presented in Section 5. The service that monitors the execution of the HLA application supporting the *Broker Service* when the application is performing badly and needs to be migrated is described in Section 6. Last but not least, the service that measures the potential behavior of HLA applications on the Grid and supporting the broker in its decisions on where to migrate the application is described in Section 7. We conclude the paper with Section 8.

2. HLA as a support for distributed interactive simulations

HLA Runtime Infrastructure (RTI) federations [2] are distributed systems that consist of multiple processes (federates) communicating across computer nodes. HLA provides application developers with an RTI that acts as a tuple space. In order to send messages, the applications that are plugged into the RTI have to publish well-defined objects in that space. The applications that receive messages have to subscribe to those objects. Each time an object is to be sent, the application must call a method that updates the attribute values of this object; RTI then notifies the subscribed applications that the object has been updated. This approach allows for easy plugging in of late-arriving components into a running system. HLA also supports time synchronization (*i.e.* for time-driven or event-driven simulations) and data distribution management between distributed components of the interactive application. Federates are controlled by the RTI Executive (RTIexec) process.

In HLA there is no mechanism for migrating federates according to the dynamic changes of host loads or failures. This means that HLA does not guarantee fault tolerance in an unreliable Grid environment.

3. Related work

The modeling and simulation community has realized that there is no widely accepted standard that can handle both DOD Modeling and Simulation standards like HLA [2] and Web/IT standards [7]. It has been noticed that the current solutions in HLA implementations, such as DoD HLA-RTI, do not perform efficiently in wide area networks and traditional approaches to high-performance RTI implementations assume relatively static configurations of federates. This has been the motivation behind the XMSF project [8], aiming to develop a common standard for Web and DOD Modeling and Simulation. As a prototype, a web-enabled RTI has been developed, being basically a Web service wrapper over an existing RTI functionality. It serves as an example to create more general XMSF profiles. These profiles are planned as documents describing a set of protocols, data and metadata standards used for the application domain, and a detailed description of applying protocols and data standards to implement the architecture. Unfortunately, the final specification of XMSF profiles has yet to be defined.

In parallel, research is in progress within the Commercial Off-The-Shelf Simulation (COTS) Package Interoperability Forum (HLA-CSPIF) [9]. As there is no real standard use pattern for the High Level Architecture within the context of this area, the ultimate goal of the Forum is to create standards that will facilitate interoperation of COTS simulation packages.

There is also research related to supporting management of HLA-distributed simulations. Cai *et al.* [10] have implemented a load management system for running large-scale HLA simulations in a Grid environment based on Globus Toolkit 2 and have presented a framework in which the modeler can design parallel and distributed simulations with no knowledge of HLA/RTI [11]. In [12] they have presented a protocol that supports efficient checkpointing and federate migration for dynamic load balancing.

Further efforts towards a scalable and fault-tolerant framework for HLA-based applications are being made by the Swedish Defence Research Agency (FOI), where a Decentralized Resource Management System (DRMS) [13] is under development. DRMS is a JXTA-based peer-to-peer system for the execution of HLA federations in a decentralized environment.

In [14], the authors have proposed a fault-tolerant resource-sharing system (FT-RSS) for HLA-based simulations. The framework and its components allow individual configurations of FT mechanisms to be included in federates and federations.

In the design and implementation of a fault tolerant federation, the developers are supported by an FT configuration tool. During the execution of a distributed HLA simulation using an FT-RSS, the manager and client components of the FT-RSS are responsible for the enforcement of FT mechanisms. Another approach has also been elaborated that uses the HLA concept to build a scalable peer-to-peer infrastructure [15], proposed by FederationGrid (FedGrid), built over the standard High Level Architecture and based on the concept of hierarchical HLA federations. It implements a scalable Grid supporting real-time collaborative applications.

The works presented above are related to the issue considered in this paper, but none of them have presented an environment designed for running interactive applications on the Grid based on the Grid Services approach. Only the approach presented in [10, 11] bears a significant resemblance to our proposal, but it introduces the concept of an additional layer between HLA and the actual applications in order to facilitate execution management. Although it allows for developing efficient migration protocols [12], it is insufficient for porting an HLA legacy code to the Grid. In contrast, the approach adopted in our project allows not only to build HLA-based applications that can be executed on the Grid, but also to easily adapt the HLA legacy code.

4. The design of the Grid HLA Management System

4.1. Requirements of interactive applications

We have focused our research on "human in the loop" simulations, which a user can **steer while they are running** by providing input data online. This approach is suitable for the kind of applications where the **parameter space is not known before runtime** and therefore parameter study approaches cannot be applied. Three main requirements of such applications may be defined as follows:

1. Distributed simulations require specific functionalities, such as time management for **event-driven or time-driven simulations** and data distribution management for convenient data exchange between distributed components of the simulation. This issue can be solved by a system supporting interaction at the application level.
2. Distributed interactive simulations belong to the field of High Performance Computing (HPC) in that they require a certain level of quality in the execution environment, which allows for **near-real time communication** between **distributed** components.
3. Usually, these applications consists of modules with different functionalities and require **access to distributed resources** which additionally turns them into High Throughput Computing (HTC) applications.

To fulfill requirements specific for interactive simulations, such as time and data management services, we have decided to use the existing solutions for distributed and interactive simulation systems. We have chosen the High Level Architecture (HLA) [2] as it is an IEEE standard well recognized in the area of distributed simulations and it offers all the functionalities necessary for simulation developers. HLA fulfills the first requirement mentioned above, but it does not include any management support for simulation execution in its underlying environment and assumes a certain quality of service. It is therefore only a partial solution of the problem.

One of the most important concepts that have strongly influenced the scientific distributed computing area is Grid computing, which has emerged as an important and rapidly expanding approach to distributed systems in the past few years [16]. Ian Foster [5] has presented a checkpoint list defining a Grid as a system that coordinates resources that are not subject to centralized control, but where coordination should be done by means of standard, open, general-purpose protocols and should deliver non-trivial quality of service. This approach creates a potential opportunity for better and more convenient usage of distributed resources that were previously inaccessible, and can fulfill the requirements of distributed simulations. However, current Grid environments lack the ability to fully support near-real-time applications. When no network reservation mechanisms are present, it becomes difficult to achieve a certain quality of service while assuring near-real-time responses to the human in a processing loop. Although various QoS mechanisms are technically possible, they are often absent from Grid environments and require support from network devices as well as additional effort to set them up, which can limit access to some of the resources for which it is difficult or impossible (for various reasons).

It is often very difficult to predict the performance of a task in a shared Grid environment. If no reservation or prioritization mechanisms are available, HPC applications require support for dynamic adaptation to the changes in the execution environment and thus call for a system that would control their execution. Unfortunately, the Globus Resource Allocation Manager (GRAM) [17], which is the Grid interface to local management systems, does not support all of the features of the underlying batch system. A variety of useful features, such as the ability to checkpoint, are missing. Condor-G [18], used by the DataGrid resource broker [19], has the same disadvantages as GRAM. The same problems arise in the current OGSI-enabled Globus Toolkit 3 [6], which actually exposes GRAM functionality as a service.

In earlier research, the **Grid HLA Management System (GHMS)** has been presented as a continuation of a system proposed in [20] to **support efficient execution of HLA-based interactive simulations in the Grid** environment in spite of the issues presented above. **GHMS fills the gap between interactive simulations and Grid Infrastructure** as shown in Figure 2.

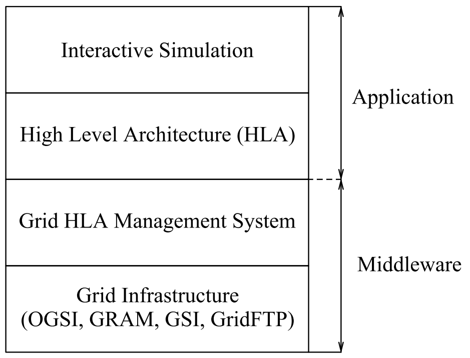
Interactive Simulation
High Level Architecture (HLA)
Grid HLA Management System
Grid Infrastructure (OGSI, GRAM, GSI, GridFTP)
Application
Middleware

Figure 2. The Grid HLA Management System in its context

GHMS is placed between HLA and Grid layers. HLA supports distributed interactive simulations on the application level, supplying the developer with the necessary services related to time and data management. The Grid infrastructure is the middleware that provides access to the available resources, like computational power, storage systems, *etc.* GHMS prepares the Grid environment to be efficiently used by interactive simulations.

4.2. System overwiev

In previous papers [20, 21] we have presented an overall concept of a system architecture that supports running distributed interactive applications based on the HLA standard in a Grid environment. In [20] we have focused on a prototype of the *HLA-Speaking* service that interfaces the system with the actual code of one federate. Management includes saving and restoring fedarate state from a checkpoint file.

In [21] we have proposed HLA-based benchmarks as possible support for choosing resources for HLA-based applications on the Grid. In this paper we present an advanced architecture including a redesigned *HLA-speaking service* that can manage all federates from an application that are executed at its site. We also describe in more detail the system component responsible for monitoring federates in order to help the *Broker Service* decide when an application starts to perform badly and needs to change its location.

Our current experimental architecture is based on the OGSA/OGSI standard, since it has been in the mainstream of the Grid effort. Although new concepts such as WSRF have appeared [22], we expect that the architecture described here, based on Grid and Web Services is immune to such changes in standards. The main idea behind WSRF is to converge the concepts of Grid and Web Services.

The components of the architecture can be categorized as follows:

4.2.1. Services global per application

The Broker Service responsible for setting up and controlling all federates within an application basing on information from the external *Registry Service*, as well as information from services described below, an application monitoring components that include the *Main Service Manager* responsible for gathering information from *Service Managers* residing on each site and presenting it to the *Performance Decision Service* which triggers the *Broker Service* to perform the action of migration.

4.2.2. Services that should reside at each Grid site

The HLA Management Services that interface and manage actual HLA installations at the site. This group includes the *HLA-Speaking Service* that represents HLA installations at a particular Grid site and the *RTIexec service* that interfaces with the RTIexec process. A more detailed description of the advanced *HLA-Speaking Service* that can manage multiple federates running at its site can be found in Section 5.

The Broker Support Services that aim at providing the *Broker Service* with information necessary for decisions about the setup and migration of application components. The *Broker Service* should take into account information from *Infrastructure Monitoring Services* present in the Grid environment that provide it with host load information within a single Grid site. This research is part of the Cross-Grid project [23], which also involves infrastructure monitoring tools [24]. In [21] we have described *Benchmark Services* as another possible source of information about HLA-based application behavior.

The Migration Support Services that provide direct support for fault tolerant and effective performance of HLA-based applications. This group includes *Local Monitors* and the *Migration Service*. Once the *Broker Service* decides where to migrate, the actual action is done by the *Migration Service* that asks the *HLA Speaking Service* to checkpoint states of federates that are running on that site. Next, the *Migration Service* starts the *HLA Speaking Service* on the remote site it wants to migrate to, transfers the code and checkpoint files and restarts the federates. A technical description of the HLA migration process can be found in the previous paper [20].

5. The HLA-Speaking Service as an interface to federates

The *HLA-Speaking Service* is a service residing at a Grid site that interfaces HLA federates executing at that site. The communication scheme is depicted in Figure 3.

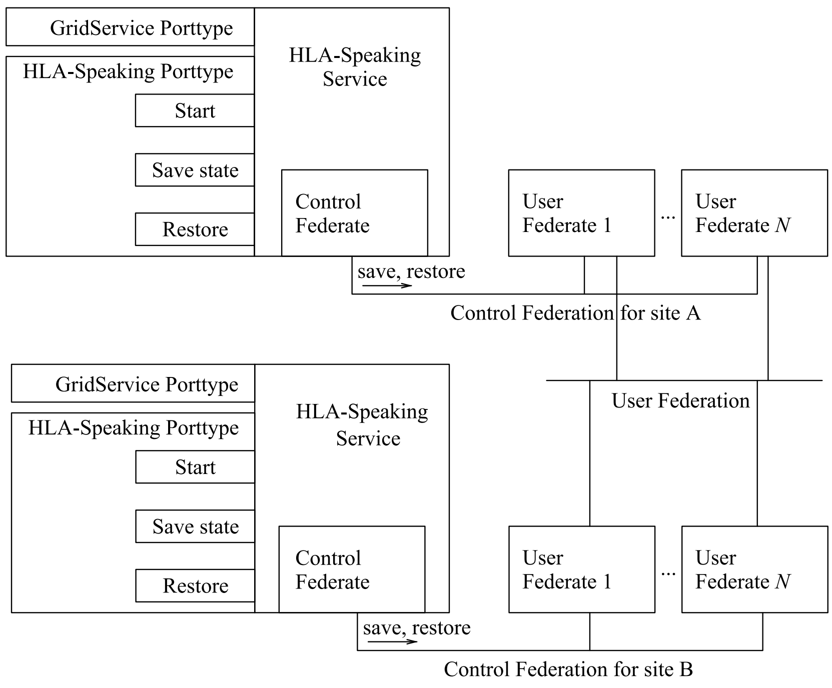

Figure 3. HLA-Speaking Service for multiple federates

The *HLA-Speaking Service* communicates with a control federates via control federation that is created solely for that purpose and it is invisible to the users. Apart from that, application federates communicate with each other by joining federations specific for their applications. *HLA-Speaking Services* communicate with other components of the system using the Grid Services interface. The *Broker Service* can invoke on the *HLA-Speaking service* any operations that start application federates at its site, save the state of selected federates and restore them from a checkpoint file. The *HLA-Speaking Service* is designed to cooperate with GRAM in order to start the application processes at its site. Save and restore requests are sent via the control HLA federation.

6. When to migrate? HLA Application Monitoring Services

For monitoring the application in order to extract performance information, we use the OCM-G monitoring system [25] developed in the CrossGrid project [23]. OCM-G is an autonomous, distributed, decentralized system which exposes monitoring services via a standardized OMIS interface [26].

OCM-G consists of two types of components: per-host Local Monitors and per-site Service Managers. Additionally, there is a Main Service Manager which distributes data to and collects it from site-SMs. The components are shown in Figure 4.

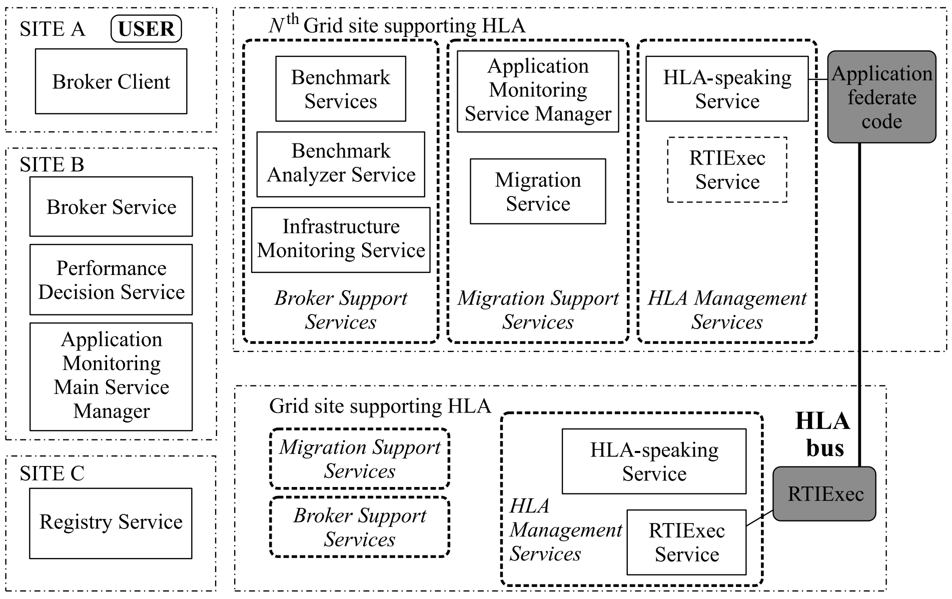

Figure 4. The Grid HLA Management System

By using OCM-G we can gather all the information required by the *Decision Performance Service* that communicates with the *Main Service Manager* using the OMIS [26] protocol. The system can monitor the invocation of all HLA services: federation management, data management, data distribution management, time management and ownership management. The system allows for gathering data required for performance measurement such as the amount of data sent, time of sending data and process identifiers. This allows the *Decision Performance Measurement Service* to discover which federates participate in communication within the HLA tuple space and to decide if their location on the Grid is sufficient for achieving the performance needed by the user in a processing loop.

The architecture of our HLA monitoring system [27] is as follows. Between the HLA Applications and the RTIambassador we insert an additional class, which looks for applications similar to the original RTIambassador. This class has to communicate with the original RTI (it has to be invisible to the application) and to send data about it to the OCM-G Monitoring service.

The OCM-G architecture allows us to control the amount and frequency of monitored data sent from the monitored application to the *Decision Performance Measurement Service.* It is scalable and suitable for running in a Grid environment.

7. Where to migrate? Infrastructure Monitoring and Benchmark Services for HLA applications

Network performance is important for the operation of applications with a human in the processing loop. One of the possible solutions to this problem is to use a prediction system such as the Network Weather Service (NWS) [28] or network monitoring systems like JIMS [24].

We have decided to develop our own benchmark suite to experimentally infer the type of behavior that can be expected from the Grid environment for HLA-based interactive applications. Details can be found in [21]. We have focused on the scenario present in the CrossGrid biomedical application [23]. This application consists of three components: interaction, simulation and visualization. While simulation is performed, a user can change its parameters using the interaction module (sending small messages from interaction to simulation) and see the results through visualization (simulation sends large updates to visualization). According to this scenario, our benchmark consists of two components: an integrated Interaction-Visualization Component (IVC) and a Simulation Component (SC). IVC sends small messages, which are human interactive requests to change selected SC parameters. SC responses are large messages that present the altered state of the simulation to the user. Certainly, this scenario is suitable only for some classes of applications. In the future, we are going to extend our set of benchmarks.

The HLA benchmark federation consists of two federates, acting as IVC and SC respectively. IVC sends the SC requests in which it asks for certain amounts of data and measures the time between sending the request and receiving a response. Messages are sent through updates of HLA data objects [2]. The benchmark is managed within the Grid Services framework by the *Benchmark_IVC Service* and the *Benchmark_SC Service*, that start, control and measure performance of IVC and SC, respectively.

Apart from the *Benchmark Services*, our architecture contains a *Benchmark Analyzer Service.* The goal of this service is to gather statistics based on *Benchmark Services'* results.

8. Summary and future work

This paper has presented our approach to Interactive Simulations on the Grid. As runtime support for interactive simulations, we have chosen the HLA standard [2]. To efficiently execute HLA-based applications on the Grid we have proposed a Grid HLA Management System. We have focused on important parts of the system, namely the *HLA-Speaking Service* for multiple federates that interface the HLA application to the system, the *Monitoring Service* integrated with the OCM-G monitoring system and HLA-based *Benchmark Services* informing the *Broker Service* what can be expected from the application's behavior. The parts of the system were tested with a mock application and are planned to be used in medical application developed in CrossGrid [4]. In the future, we plan to develop the following:

The Broker Service. Performing adequate brokering decisions in spite of the highly-changeable nature of the network and host environments is a non-trivial task, although benchmark results can provide valuable support. We plan to develop a *Broker Service* architecture and the associated algorithms. For that purpose, it is also necessary to have a service that will analyze benchmark results according to the specific needs of the broker.

A Performance Decision Service. There is a clear need for a service that will analyze the performance of an application basing on data gathered by the monitoring system to supply the *Broker Service* with information on whether and when migration is needed.

Infrastructure Monitoring. Apart from benchmark services we also plan to analyze the use of existing infrastructure monitoring [24] and prediction services [28] for our purposes.

Acknowledgements

The authors wish to thank Mr. Piotr Nowakowski for his additional remarks. This research has been partly funded by the European Commission IST-2001-32243 Project "CrossGrid".

References

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

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

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

[4] 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, CD-ROM, IEEE Catalog #04EX836C

[5] Foster I 2002 *What is the Grid? A Three Point Checklist*, GRIDToday, http://www.globus.org/

[6] Foster I, Kesselman C, Nick J and Tuecke S 2002 *The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration*, Open Grid Service Infrastructure WG Global Grid Forum

[7] Web Services, http://www.w3.org/2002/ws/

[8] XMSF Project Homepage, http://www.movesinstitute.org/xmsf/xmsf.html

[9] HLA CSPIF Homepage, http://www.cspif.com/

[10] Cai W, Turner S J and Zhao H 2002 *Proc. 6th IEEE Int. Workshop on Distributed Simulation and Real-Time Applications*, IEEE Computer Society, Fort Worth, Texas, pp. 7–14

[11] Yuan Z, Cai W and Low M Y H 2003 *Proc. 7th IEEE Int. Symposium on Distributed Simulation and Real Time Applications (DS-RT 2003)*, IEEE Computer Society, Delft, The Netherlands, pp. 20–28

[12] Yuan Z, Cai W, Low M Y H and Turner S J 2004 *Proc. 4th Int. Conf. on Computational Science*, Cracow, Poland, LNCS **3038** 856

[13] Eklöf M, Sparf M and Moradi F 2003 *Proc. European Simulation Interoperability Workshop*, organized by the Simulation Interoperability Standardization Organization, Stockholm, http://www.sisostds.org/

[14] Lütchi J and Grossman S 2004 *Proc. 4th Int. Conf. on Computational Science*, Cracow, Poland, LNCS **3038** 865

[15] Vuong S, Cai X, Li J, Pramanik S, Suttles D and Chen R 2004 *Proc. 4th Int. Conf. on Computational Science*, Cracow, Poland, LNCS **3038** 889
[16] Berman F, Fox G and Hey T (Eds.) 2003 *Grid Computing. Making the Global Infrastructure a Reality*, Wiley
[17] Globus Project Homepage, http://www.globus.org/
[18] Thain D, Tannenbaum T and Livny M 2003 *Grid Computing: Making the Global Infrastructure a Reality* (Berman F, Hey A J G, Fox G, Eds.), John Wiley, pp. 299–335, http://www.cs.wisc.edu/condor/publications.html
[19] DataGrid Project Homepage, http://www.datagrid.com/
[20] Zając K, Bubak M, Malawski M and Sloot P 2003 *Proc. 7th IEEE Int. Symposium on Distributed Simulation and Real Time Applications (DS-RT 2003)*, IEEE Computer Society, Delft, The Netherlands, pp. 4–11
[21] Rycerz K, Bubak M, Malawski M and Sloot P 2004 *Computational Science, ICCS 2004, 4th Int. Conf.*, Cracow, Poland, LNCS **3038** (Part III) 848
[22] WS-Resource Framework Project Homepage, http://www.globus.org/wsrf
[23] Crossgrid project Homepage, http://www.crossgrid.org/
[24] Balos K, Bizon L, Rozenau M and Zielinski K 2004 *Proc. Cracow'03 Grid Workshop*, Cracow, Poland, ACC Cyfronet UMM, pp. 245–253
[25] Baliś B, Bubak M, Funika W, Szczepieniec T and Wismüller R 2003 *Proc. Computational Science – ICCS Int. Conf.*, Melbourne, Australia and St. Petersburg, Russia, LNCS **2657** (Part I) 214
[26] Ludwig T, Wismüller R, Sunderam V and Bode A 1997 *OMIS – On-line Monitoring Interface Specification*, Version 2.0, Technische Universität, München, Germany, TUM-I9733, SFB-Bericht Nr. 342/22/97 A, http://wwwbode.in.tum.de/˜omis/
[27] Rycerz K, Balis B, Szymacha R, Bubak M and Sloot P 2004 *Proc. Int. Symposium on Distributed Simulation and Real Time Applications (DS-RT 2004)*, Budapest, Hungary (in print)
[28] Gaidioz B, Wolski R and Tourancheau B 2000 *Proc. 9th IEEE High-performance Distributed Computing Conf.*, Pittsburgh, Pennsylvania, pp. 147–154