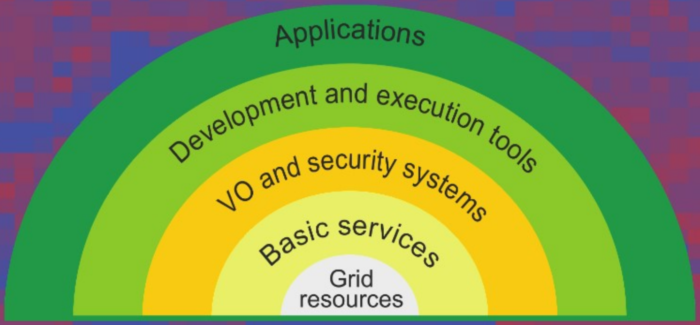
State-of-the-Art Survey

LNCS 7136

Marian Bubak
Tomasz Szepieniec
Kazimierz Wiatr (Eds.)

Building a National Distributed e-Infrastructure – PL-Grid

Scientific and Technical Achievements



Springer

Marian Bubak Tomasz Szepieniec
Kazimierz Wiatr (Eds.)

Building a National Distributed e-Infrastructure – PL-Grid

Scientific and Technical Achievements

Springer

Volume Editors

Marian Bubak
AGH University of Science and Technology
Department of Computer Science and ACC Cyfronet AGH
30-950 Kraków, Poland
E-mail: bubak@agh.edu.pl

Tomasz Szepieniec
AGH University of Science and Technology
ACC Cyfronet AGH
30-950 Kraków, Poland
E-mail: t.szepieniec@cyfronet.pl

Kazimierz Wiatr
AGH University of Science and Technology
ACC Cyfronet AGH and Department of Electronics
30-950 Kraków, Poland
E-mail: k.wiatr@cyfronet.pl

ISSN 0302-9743 e-ISSN 1611-3349
ISBN 978-3-642-28266-9 e-ISBN 978-3-642-28267-6
DOI 10.1007/978-3-642-28267-6
Springer Heidelberg Dordrecht London New York

Library of Congress Control Number: 2012930730

CR Subject Classification (1998): C.2, H.4, I.2, D.2, H.3, F.1

LNCS Sublibrary: SL 3 – Information Systems and Application, incl. Internet/Web and HCI

Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

Preface

The main objective of the PL-Grid project was to provide the Polish scientific community with a grid-based e-infrastructure enabling research in various domains of e-Science. Our infrastructure supports scientific investigations by integrating experimental data and results of advanced computer simulations carried out by geographically distributed research teams.

PL-Grid not only extends the amount of computational resources provided to the Polish scientific community by 215 Tflops of computing power and 2500 TB of storage space, but – more importantly – facilitates effective use of these resources by providing innovative grid services and end-user tools, as well as continuous technical support. Extensive collaboration between project developers and domain scientists has led to the release of new domain-specific services and applications.

Another important achievement of PL-Grid was the introduction of a professional separation of responsibilities upon which the infrastructure is based. This organizational schema enables better user support and ensures consistent quality of service, in line with European Commission policies which actively encourage development and integration of distributed computing infrastructures under the EGI InSPIRE and PRACE umbrella projects.

This book describes the experience and the scientific results obtained by the project partners as well as the outcome of research and development activities carried out within the project.

The first chapter provides an overview of motivation, current status and future development within the PL-Grid e-infrastructure. The second chapter presents the way in which three main grid middleware suites (gLite, UNICORE and QosCosGrid) are integrated in the PL-Grid platform and extended with additional support services. The third chapter describes the experience and current status of service operations. This is followed by three chapters which provide details on the middleware implemented in PL-Grid. The next four chapters are dedicated to essential security mechanisms, including the tools and security solutions implemented in our e-infrastructure. We also present chapters devoted to new methods of infrastructure monitoring, storage provisioning and implementation of service level agreements. Subsequently, seven consecutive chapters provide an in-depth description of user environments: the Eclipse Parallel Tools Platform integrated with QosCosGrid, the Migrating Desktop, Science Gateways based on the Vine Toolkit, the GridSpace Experiment Platform, and the InSilicoLab environment. A very important contribution to the book comes in the form of five chapters prepared by end-user teams. They describe how the PL-Grid middleware and infrastructure are applied to solving such scientific problems as grain morphology analysis, seeking common structural motifs in protein families, chemistry computations, molecular dynamics models, LHC computing, and

Monte Carlo simulations for the Cherenkov Telescope Array. Finally, the last two chapters describe activities which are of great importance in any emerging computing infrastructure – namely, training and dissemination. The book is supplemented with a glossary of terms.

We hope that it will serve as an important intellectual resource for researchers, developers and system administrators working on their own grid infrastructures and promote collaboration and exchange of ideas in the process of constructing a common European e-Infrastructure.

We owe our thanks to all authors and reviewers for their diligent work which ensures the high quality of this book. We would like to express our gratitude to Jacek Kitowski, Director of PL-Grid, for his personal involvement. We are also indebted to Zofia Mosurska, Robert Pająk, Milena Zając, Piotr Nowakowski, Magdalena Szopa, and Joanna Kocot for their enthusiastic editorial work, as well as to numerous colleagues from ACC Cyfronet AGH for their help. Finally, we wish to thank Springer for the fruitful collaboration during the preparation of the book.

We invite you to visit the PL-Grid website (`http://www.plgrid.pl`) which carries up-to-date information regarding our e-Infrastructure.

This book, like all PL-Grid project activities, was co-funded by the European Regional Development Fund as part of the Innovative Economy Program.

December 2011

Marian Bubak
Tomasz Szepieniec
Kazimierz Wiatr

INNOVATIVE ECONOMY
NATIONAL COHESION STRATEGY

PL GRID

EUROPEAN UNION
EUROPEAN REGIONAL
DEVELOPMENT FUND

Book Preparation

This book was prepared with the help of the members of the PL-Grid project.

Editors' Support Team

Help in editing the book:	Z. Mosurska, R. Pająk and M. Zając
English proofreading:	J. Kocot, P. Nowakowski and M. Szopa

Reviewers

P. Arłukowicz
P. Bała
T. Bartyński
K. Benedyczak
M. Borcz
B. Bosak
M. Bubak
E. Ciepiela
Ł. Dutka
A. Eilmes
M. Filocha
Ł. Flis
G. Frankowski
W. Funika
T. Gubała
D. Harężlak
T. Jadczyk
M. Jarząb
M. Kasztelnik
J. Kocot
P. Kopta
J. Kosiński
M. Krakowian
B. Kryza
K. Kurowski
M. Malawski
M. Mamoński
J. Meizner
N. Meyer
D. Nikołow
K. Nowiński
Ł. Olejnik
A. Oziębło
B. Palak
T. Piontek
M. Radecki
I. Roterman
K. Rycerz
R. Słota
M. Sterzel
B. Szurgot
T. Szymocha
M. Teodorczyk
M. Tomanek
M. Turała
M. Tykierko
R. Tylman
M. Uchroński
W. Waga
W. Wiślicki
M. Witek
P. Wolniewcz
S. Zieliński

Table of Contents

List of Contributions

PL-Grid: Foundations and Perspectives of National Computing Infrastructure

Jacek Kitowski[1,2], Michał Turała[2], Kazimierz Wiatr[2], and Łukasz Dutka[2]

[1] AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Department of Computer Science, al. Mickiewicza 30, 30-059 Kraków, Poland
[2] AGH University of Science and Technology, ACC Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków, Poland

Abstract. The Polish Grid Initiative commenced in 2009 as part of the PL-Grid project funded within the framework of the Innovative Economy Operational Programme. The main objective is to provision a persistent heterogeneous computational platform for the Polish scientific community with a unified interface, enabling easy access to the distributed large-scale computing infrastructure. The project establishes a countrywide computing platform which supports scientific research through integration of experimental data and results of advanced computer simulations carried out by geographically distributed computer infrastructure and teams. The solutions applied in setting up this e-infrastructure facilitate integration with other, similar platforms around the world.

In this chapter the foundations of the PL-Grid architecture, a brief history of the project and its most important objectives are presented.

Keywords: Grid, e-Infrastructure, PL-Grid Consortium.

1 Introduction

Over the past several years we have witnessed rapid increases in the dependence of scientific research on access to large-scale computing systems. Theoretical sciences rely on computing to evaluate hypotheses in many areas, including mathematics, economics, astrophysics, chemistry, climatology, physics, etc. Computerized simulations enable researchers to test their ideas and draw initial conclusions before moving on to the much more expensive and time-consuming experimentation phases. Moreover, experimental and natural sciences, including climatology, meteorology, high energy physics, and astrophysics generate enormous amounts of data which needs to be processed on several levels and distributed to research teams all over the world.

Many of the largest research initiatives are driven by international institutions such as CERN (European Organization for Nuclear Research), EMBL (European Molecular Biology Laboratory) or ITER (International Thermonuclear Experimental Reactor) which can share their results with geographically distributed research teams, increasing their demand on computational power and storage resources.

Fortunately, at the same time we have observed unsurpassed growth in the computational power and storage capabilities of high-end computing solutions, ranging from simple clusters through cluster platforms with advanced interconnects to custom-built supercomputers with peak performance in the PFlop range (10^{15} floating-point operations per second). In order to efficiently harness such computational power special software solutions are necessary. Research scientists need to be able to actually use the resources in a transparent manner, without having to deal with the issues of scalability, distribution, data access and security.

Currently the most popular paradigm for provisioning distributed computing resources for research is grid computing – a term coined in 1999 [1,2]. The initial idea was to develop a global computational infrastructure which would be as readily accessible as the power grid and whose users would form Virtual Organizations (VO) based not on their geographical location but on their respective research domains. Over the last decade, several such infrastructures have been developed all over the world – most notably the LHC Computing Grid [3], TeraGrid [4], EGEE [5] and DEISA [6]. Most grid deployments are based on one of the following frameworks: Globus [7], UNICORE [8], Condor [9] or ARC [10], all of which evolved in parallel and provide comprehensive sets of components and features, allowing users to run their computational jobs on federated computing resources. The emergence of grids facilitates integration and collaboration of research teams in the US and Europe, at least in the scope of large computing centers.

Unfortunately, although major Polish HPCs have indeed participated in grid development projects such as CrossGrid [11] or EGEE, Polish scientists often had to rely solely on their local computing resources or personally apply for access to external infrastructures (for instance through EGEE channels). This strongly discouraged the Polish research community from conducting interdisciplinary computational research with other institutes in Poland and produced a situation where different research teams were affiliated with different European grid domains and developed software for different grid systems (e.g. gLite [12] or UNICORE).

Eventually, in 2007, the major computing centers in Poland (see Fig. 1) – specifically the Academic Computer Centre Cyfronet AGH (ACC Cyfronet AGH) in Kraków, the Poznań Supercomputing and Networking Center (PSNC), the Interdisciplinary Centre for Mathematical and Computational Modeling (ICM) in Warsaw, the Gdańsk Academic Computer Centre (CI TASK) and the Wrocław Centre for Networking and Supercomputing (WCNS) – came together to form the PL-Grid Consortium [13] with the stated goal of "providing the Polish scientific community with an IT platform based on grid computer clusters, enabling e-science research in various fields".

The foundation of the consortium was a long-term agreement expressing the will to integrate national distributed computing resources in order to provide the Polish research community with on-demand access to a sustainable computing platform and participate in international grid initiatives and e-science activities under the umbrella of a federated national infrastructure.

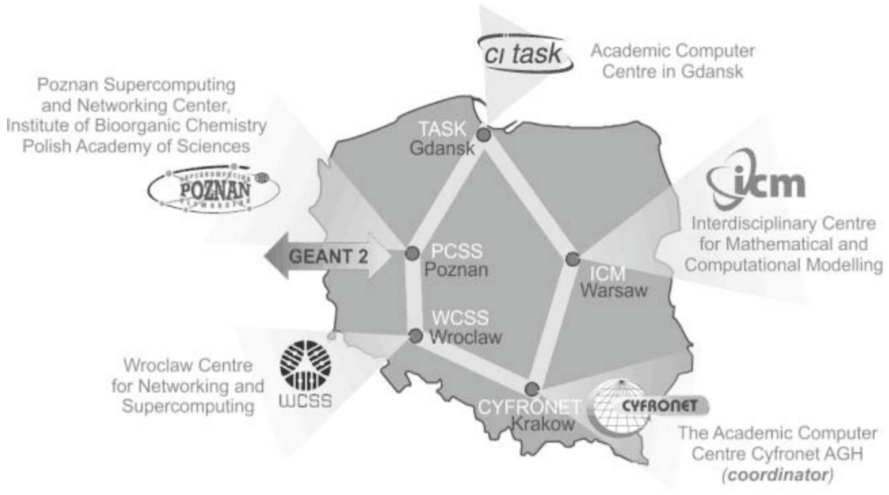

ci task
Academic Computer Centre in Gdansk
Poznan Supercomputing and Networking Center, Institute of Bioorganic Chemistry Polish Academy of Sciences
POZNAN
TASK Gdansk
icm
Interdisciplinary Centre for Mathematical and Computational Modelling
GEANT 2
PCSS Poznan
ICM Warsaw
WCSS Wroclaw
Wroclaw Centre for Networking and Supercomputing
WCSS
CYFRONET Krakow
CYFRONET
The Academic Computer Centre Cyfronet AGH (coordinator)

Fig. 1. PL-Grid Consortium

The PL-Grid Consortium prepared a project (called PL-Grid) which obtained funding from the European Regional Development Fund as part of the Innovative Economy Program [14] and commenced in March 2009. The total project budget was 21M€ and project completion was scheduled for March 2012, with the aggregated offer for the community consisting of 215 TFlops of computational power and 2.5 PB of storage resources. The overall goal of this project was to deploy a persistent national large-scale distributed computing infrastructure for Polish scientists, allowing them to conduct interdisciplinary research on a national scale, and giving them transparent access to international grid resources via affiliated international grid infrastructures.

As the project enters its final stage, the PL-Grid Consortium carries on its activities in the area of grids and clouds for scientific community and has prepared a follow-up project called PL-Grid Plus, funded in October 2011 by the same institution with a total budget of 18M€. This project will last until the end of 2014.

In addition, consortium partners have been participating in other national and European projects extending the infrastructure and available middleware while constantly enhancing the maturity and functionality of the resources provided to Polish scientific teams.

2 Objectives

The main objectives of the project were drawn from the experience of all consortium partners based on their activities in international grid initiatives as well as the requirements expressed by users of local computing resources. The list of expected properties of the Polish grid infrastructure included the following:

– Multiscale,
– Multidisciplinary,
– Heterogeneity,
– Unification of access,
– Capability of international cooperation,
– SLA support and computational grant tools,
– Sustainability,
– Training and user support.

Multiscale. The PL-Grid Consortium believes that the national grid platform must be easy enough for users not to discourage them even if they run relatively simple computations. Basing on experience from projects which were tailored for large-scale experiments and suffered from the lack of user variety, our intention from the start was to make the infrastructure as user friendly as possible, with full support even for the smallest, entry-level computations. Moreover, in practice, the overhead the users must "pay" when migrating their applications to the grid sometimes is a barrier, as most experiments start from small-scale test cases and then evolve into more advanced scenarios. The PL-Grid infrastructure simplifies running the smallest jobs using direct access to PBS systems installed at particular locations, while encouraging the users to progressively migrate their applications to scalable grid job implementations in order to get the full benefit of the infrastructure.

Multidisciplinary. One of the fundamental aims of PL-Grid was to build a computational platform for multiple scientific domains. By analysing the structure and heterogeneity of Polish scientific communities we discovered that the platform could not be bound to, or otherwise favor, any particular scientific domain, but must be flexible enough to support different communities with various software requirements, different collaboration practices and varying security needs. This challenge led to the concept of domain grids, as planned in PL-Grid Plus, which allow creation and grouping of scientists and resources based on their specializations (see Fig. 2).

Heterogeneity. The multidisciplinary aim described above determines another very important challenge which PL-Grid must face – infrastructural heterogeneity. Provisioning a flexible infrastructure, with support for various scientific domains, requires the project to maintain a set of differing computing architectures and storage solutions. Thus, while the majority of PL-Grid resources are based on x86 computing clusters (mainly due to their universality) we also integrate additional, more specialized platforms and systems based on such technologies as GPGPUs and software enhanced clusters by Versatile SMP (vSMP). Although integration of such systems with the wider grid infrastructure requires substantial effort, we believe it will benefit many users who deal with atypical problems – which can be solved more efficiently using custom hardware.

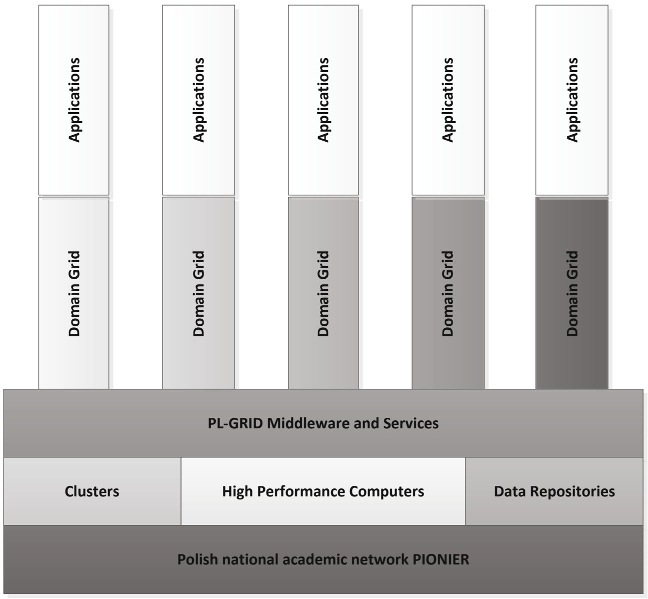


Fig. 2. Layered view of the PL-Grid infrastructure

Unification of Access. Due to the fact that many of our users have already used some grid infrastructure in the past and are accustomed to different types of interactions with the grid, another major challenge was to make the migration of their applications to the national grid platform as natural as possible. In practice, this resulted in the need to support two main grid middleware stacks, i.e. gLite and UNICORE. Furthermore, a special portal has been developed, enabling users to acquire credentials with which they can access both grid middleware components.

SLA Support and Computational Grant Tools. Another important requirement from the management perspective is ensuring that each of the computing centers which contribute resources to the PL-Grid infrastructure complies with a certain Service Level Agreement (SLA) in terms of the amount of resources and their availability. On the other hand, the resources centers must be able to monitor how their resources are used and that only authorized users obtain access and run jobs on the infrastructure, respecting their assigned quotas.

Training and User Support. In spite of substantial effort put into making the PL-Grid infrastructure as user-friendly as possible, it is still necessary to provide training for current and future system users. We need to not only show how to access and use the system, but also how to exploit the full potential of the hardware and software capabilities provided by our infrastructure in order to allow the users to create and run scalable and sustainable in-silico experiments.

Moreover, even advanced and experienced users sometimes require expert support on emerging technical on administrative issues, and thus PL-Grid provides persistent helpdesk support through the user portal.

Availability and Sustainability. In order to support the entire Polish scientific community with a distributed computing infrastructure it is crucial that the service is provided 24/7/365, with minimal downtime. Although most of the infrastructure is distributed (thus minimizing the likelihood of total system failure), we nevertheless do our best to ensure that the system remains at its full potential all the time. Hence, the average infrastructure availability is 95%-98%, depending on the period. Another issue here is that the development and deployment of an infrastructure of such a scale cannot be justified just for duration of a single project and PL-Grid Consortium partners have made long-term commitments to support the hardware infrastructure, software and users, as well as to promote continuous evolution by extending the hardware and software made available to the Polish scientific community.

Capability for International Cooperation. Last but not least, the PL-Grid infrastructure aims to be a key player in the international grid community, giving our users access to a much more extensive set of varied computational resources. The goal is to integrate our infrastructure with such initiatives as EGI and other European grids, PRACE and, where possible, US infrastructures.

3 Infrastructure

At the moment PL-Grid provides 230 TFlops of computational infrastructure and 2500 TB of storage to the Polish scientific community. Over the coming 3 years the consortium plans to extend these resources to 700 TFlops and 6000 TB of storage.

Currently the total number of cores is about 25 000 and the number of jobs ran per quarter is constantly growing, having reached $3 * 10^6$ in Q2 2011 (see Fig. 3). The number of registered users is approximately 800 (see Fig. 4).

To fulfill the aims of the project and the expectations of consortium members regarding availability, datacenters were redesigned to the multi-megawatt scale. Each server room is equipped with highly efficient megawatt-scale air conditioning systems and sustainable power supplies backed by large Uninterrupted Power Systems and diesel power generators, giving the possibility to operate the computational infrastructure for many hours despite any mains power failures.

The described infrastructure consists of:

– Intel-based Xeon clusters with 4- or 6-core processors and Infiniband connectivity,
– AMD-based clusters with 8- or 12-core processors and Infiniband connectivity,
– vSMP integrated machines with extensive memory,
– GPGPU accelerated hosts running NVidia Tesla cards.

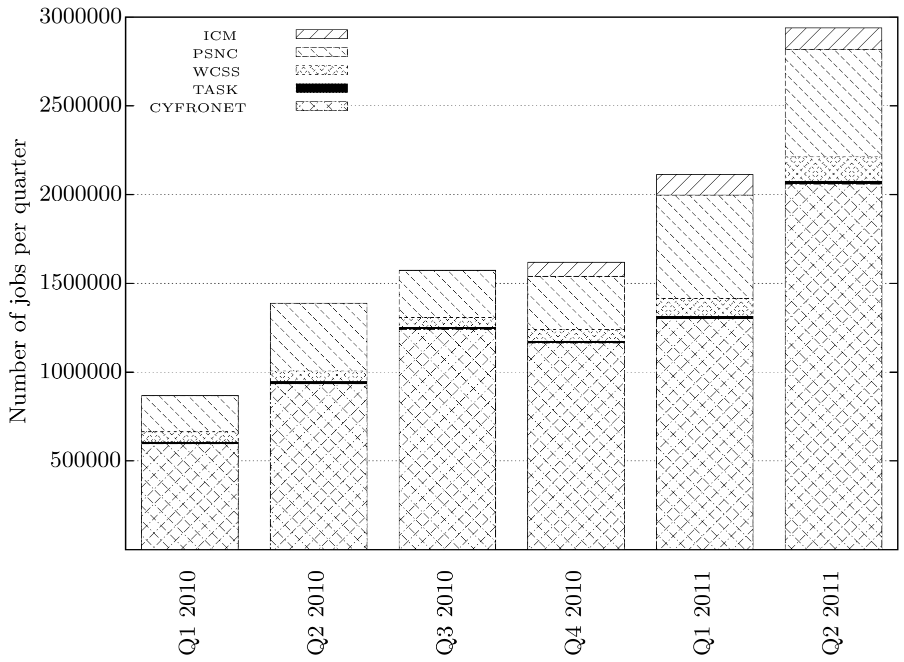


Fig. 3. Number of jobs executed in PL-Grid infrastructure per quarter

In addition to the computational infrastructure we provide large-scale storage using fiber channel connectivity and (in most cases) the Lustre distributed file system.

4 Technological Advancements

The requirements of the project and its end users, i.e. the Polish scientific community, have resulted in the overall layered architecture of the PL-Grid platform, which is presented in Fig. 5. In addition to deploying the basic hardware infrastructure and standard grid middleware stacks, we have designed several new features and components with the aim of improving user experience and extending the scope of possible applications which can be developed by the research communities.

4.1 Middleware Plurality

One of the main technological challenges was to provide unified access to grid infrastructures managed by both gLite and UNICORE middleware stacks. Although this issue has been addressed (to some extent) by other projects in the past, we could not find a solution which would satisfy all our needs and thus we decided to extend the QosCosGrid [23] system for scheduling jobs across

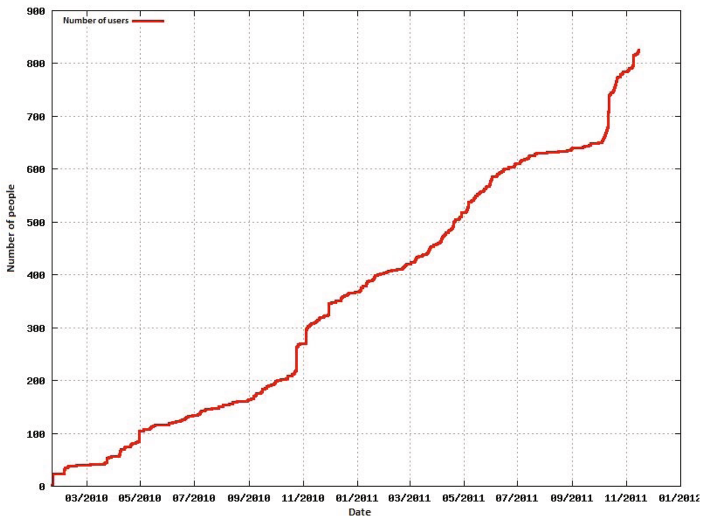


Fig. 4. Number of registered PL-Grid users over time

grid middleware domains. The need to support multiple middleware suites is dictated by users' applications. Some users have to use gLite because they collaborate with international partners who already use this middleware. On the other hand, UNICORE users tend to prefer this system due to the ability to schedule computational jobs in a workflow-like manner. Moreover, QCG (developed in PL-Grid) offers easy-to-use mapping, execution and monitoring capabilities for a variety of applications, such as parameter sweeps, workflows, MPI or MPI-OpenMP hybrids.

4.2 SLA and Grants

In order to enforce the expected quality of service of our platform and allow users to describe their hardware and software requirements, PL-Grid provides a collaborative SLA negotiation tool called Bazaar, which simplifies the process of establishing a compromise between user expectations regarding the infrastructure and the actual capability which can be assigned to handle a particular user request. Bazaar is compatible with our computational grant model and takes into account information about user quotas and current infrastructure status, as well as the history of resource provisioning and consumption, enabling system managers to evaluate how different QoS parameters are respected by providers and users.

4.3 New User Interfaces

PL-Grid provides computational power for users representing various scientific domains. In many of these users have very specific ways of conducting their experimentation, which, in some specific cases, involves demands for new

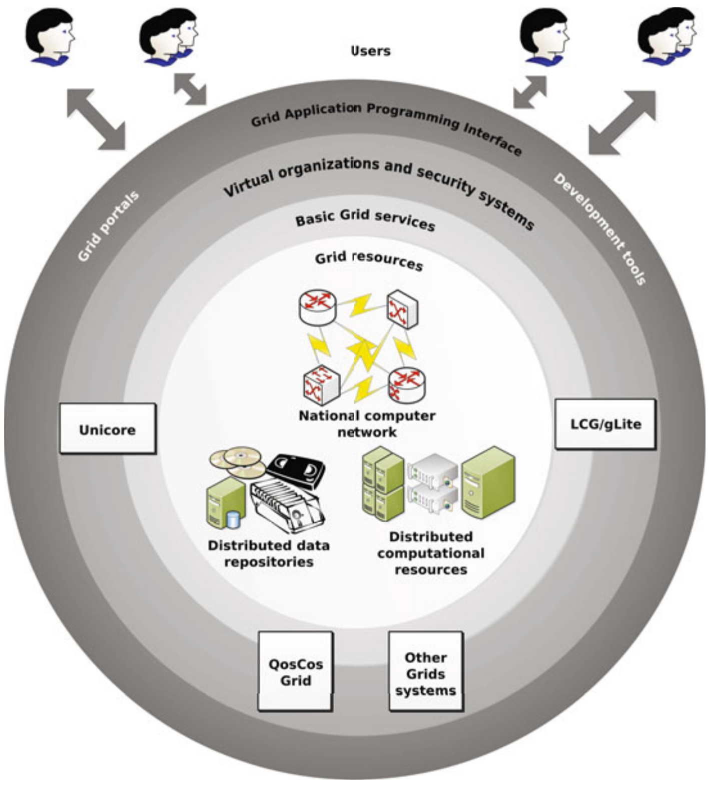

Users
Grid Application Programming Interface
Virtual organizations and security systems
Basic Grid services
Grid resources
Grid portals
Development tools
National computer network
Unicore
LCG/gLite
Distributed data repositories
Distributed computational resources
QosCos Grid
Other Grids systems

Fig. 5. Structure of the PL-Grid Computing Platform

user interfaces. One of the interesting outgrowths of PL-Grid users' requests is the GridSpace2 Experiment Workbench [18,19]. It is a novel virtual laboratory framework which enables researchers to conduct virtual experiments on grid-based resources. A web portal has been developed for this tool and serves as the main access point to all the mechanisms of the virtual laboratory from any workstation equipped with a Web browser. The portal encapsulates the so-called Experiment Workbench layer which provides tools to create experiments, collaborate, communicate, share resources and run large-scale computations in a user-friendly manner.

Many users deem command-line grid interfaces as too unwieldy and obfuscated. Thus, a graphical user interface, called the Migrating Desktop [20,21], has evolved in PL-Grid. This advanced GUI, similar to a window-based operating system, hides the complexity of the grid middleware and makes access to grid resources easy and transparent. It allows the user to run applications and tools, manage data and store personal settings independently of the location or terminal type. The Migrating Desktop offers a flexible personalized roaming work environment, scalability and portability, a rich set of tools, a single sign-on mechanism and support for multiple grid infrastructures.

4.4 Security

The dynamically evolving scene of security threats requires the project to provide its system administrators with targeted security management tools. One such tool is ACARM-ng [22] – an extensible plugin-based alert correlation framework. It consists of abstractions for correlation, reporting, reaction, gathering data from multiple sources and data storage. Real-time reporting is supported, which means that alerts can be reported while still being correlated. A web-based administrative interface presents the gathered and correlated systemwide data in a consistent way.

Another tool developed for PL-Grid security management is SARA (System for Automated Reporting and Administration). The system combines information on vulnerabilities stored (with the help of the standards mentioned above) in the National Vulnerability Database. It makes management of well-known vulnerabilities easier and more efficient.

5 International Computational Arenas

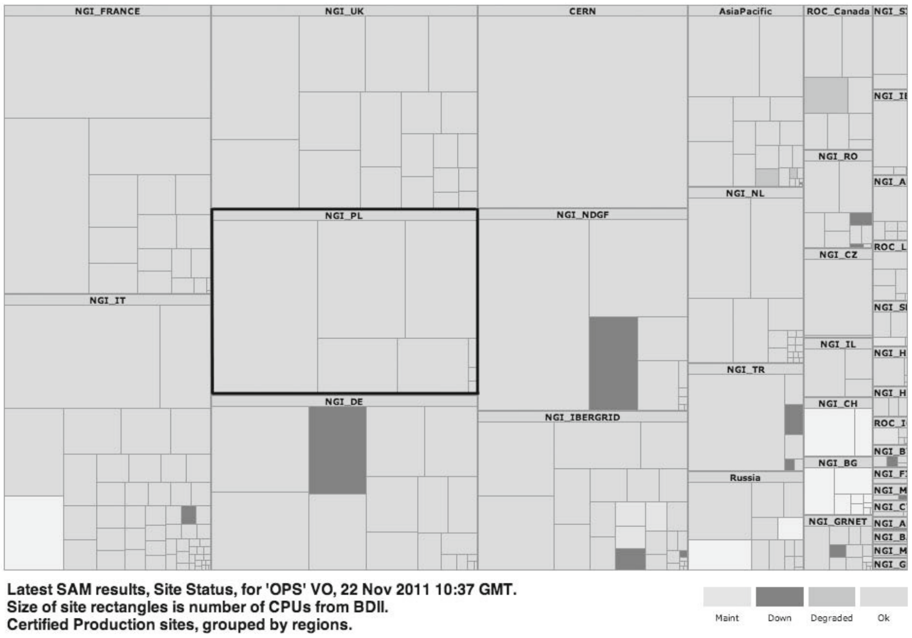
Collaboration with existing and future high-performance computing initiatives is a very important goal for the PL-Grid Consortium, as it allows the Polish scientific community to collaborate and conduct research in truly international teams and gain access to special hardware and software resources available abroad.

In March 2010 an international organization called the European Grid Initiative, with headquarters in Amsterdam, was launched with the goal to oversee the operation and development of the pan-European grid infrastructure. About 40 European National Grid Initiatives (NGIs), three European Research International Organizations (EIROs) and several grid organizations from outside of Europe supported this initiative. In this context the PL-Grid platform is one of the NGI members.

As of today EGI.eu integrates and supervises more than 200 thousand logical CPUs (cores) and about 100 PB of disk and 80 PB of tape storage; more than 13 thousand users are organized in more than 180 VOs, out of which about 30 are active – about 1 million jobs are executed daily, mainly related to particle physics but also representing the domains of archeology, astronomy, astrophysics, civil protection, computational chemistry, earth sciences, finance, fusion, geophysics, life sciences, multimedia, material sciences, etc.

PL-Grid is one of the key players in the EGI infrastructure, currently ranked 4th (see Fig. 6). PL-Grid was actually the first NGI in the EGI initiative and the PL-Grid Consortium has devised many procedures for migrating from an EGEE ROC to National Grid Initiative status, thus paving the way for other countries. PL-Grid operates a regional Service Availability Monitoring system with the ability to dispatch Regional Technical Support teams to assist service administrators in investigating problems. In addition, we are responsible for our grid configuration management, i.e. certification of new sites in Poland, adding services and downtimes. Recently we have contributed to the EGI UNICORE

Integration Task Force, helping NGIs willing to contribute their UNICORE resources to EGI. PL-Grid is strongly involved in the EGI security team called CSIRT (Computer Security Incident Response Team). Our responsibility is to coordinate security incidents with all Polish sites, including fixes for vulnerabilities reported by EGI as well as those found by our own staff.



Latest SAM results, Site Status, for 'OPS' VO, 22 Nov 2011 10:37 GMT.
Size of site rectangles is number of CPUs from BDII.
Certified Production sites, grouped by regions.

Fig. 6. PL-Grid site named *NGI_PL* size in comparison to others NGIs registered in EGI. Size of the boxes represents the computational power. The PL-GRID box has been highlighted.

Additionaly, several activities related to middleware development have been undertaken in cooperation with other international partners. For instance, the advanced reservation and co-allocation of distributed computing resources provided by QosCosGrid drew attention from other supercomputing centres from USA and Europe. Consequently, the QCG community has grown and today includes users from INRIA, the University of Dortmund, the Leibniz Supercomputing Centre, SARA, the Queensland University and the Louisiana State University.

6 Future Work

PL-Grid is a stable and sustainable distributed computing platform operating on a national level. However, the increasing user requirements and expectations force us to keep provisioning the best possible technological solutions. Given the current performance level of computing hardware and the observed rate of progress, the high-performance computing community should reach the exascale performance range in the next 5-7 years. Furthermore, current trends in distributed computing, especially in business scenarios, favor simpler and more scalable infrastructural models such as cloud computing. Thus, the designers and developers of the PL-Grid infrastructure are actively participating in several projects focusing on technological advancement in both hardware and software areas. The most interesting solutions will be integrated with our infrastructure in the future.

Over the next three years, we will focus however mainly on supporting and building true distributed user communities called domain grids. This approach is motivated by the need to bring together scientists from all over Poland and attract them to our computing infrastructure, encourage collaboration and sharing of results and improve the impact of research performed in Poland as well as on the international stage. We wish to involve single users and entire user communities, both existing and emerging, in the process of refining the infrastructure in order to meet their needs. Since there is no universal solution which solves all the domain-specific problems we have decided to invite community representatives and involve them in the process of making the grid easier and more efficient to use.

Near-term technological refinements will include forays into the cloud computing world, making our infrastructure more flexible from the system administration point of view while still providing all the grid features usually lacking in cloud platforms.

7 Conclusions

The PL-Grid Polish National Grid Initiative was initiated almost 5 years ago. Over time it has received support from several projects, which extended its hardware and software capabilities. At the moment the infrastructure is both powerful and mature. The quality, availability and versatility of tools offered to users increases its impact, as evidenced by the constant growth in the number of registered participants (see Fig. 4).

All of the objectives stated in the design phase already have been fulfilled to some extent. We provide a flexible infrastructure, supporting users from all domains of science, including high energy physics, chemistry and linguistics.

The availability indicators of the infrastructure, especially when taking into account its complexity, remain very high. At present users can run their experiments on different platforms, including x86-based clusters, vSMP with up to 6TB of memory and GPGPU-accelerated clusters.

The extensive effort of more than 100 people working on practical applications of modern computational technologies has made the Polish NGI a key player in the European grid community, and we believe that multi-level heterogeneity is the key to the project's unquestionable success.

References

1. Foster, I., Kesselman, C. (eds.): The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
2. Foster, I., Kesselman, C.: The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann (2004)
3. LCG Project, http://lcg.web.cern.ch/lcg/
4. Bollen, J., Fox, G., Singhal, P.R.: How and where the TeraGrid supercomputing infrastructure benefits science. J. Informetrics 5, 114–121 (2011)
5. EGEE Project, http://www.eu-egee.org/
6. Gentzsch, W.: DEISA, the Distributed European Infrastructure for Supercomputing Applications. In: Gentzsch, W., Grandinetti, L., Joubert, G.R. (eds.), vol. 18, pp. 141–156. IOS Press (2008)
7. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) NPC 2005. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005), doi:10.1007/11577188_2
8. Streit, A., Bala, P., Beck-Ratzka, A., Benedyczak, K., Bergmann, S., Breu, R., Daivandy, J.M., Demuth, B., Eifer, A., Giesler, A., Hagemeier, B., Holl, S., Huber, V., Lamla, N., Mallmann, D., Memon, A.S., Memon, M.S., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Schlauch, T., Schreiber, A., Soddemann, T., Ziegler, W.: Unicore 6 – Recent and Future Advancements. Annales des Telecommunications 65, 757–762 (2010)
9. Thain, D., Tannenbaum, T., Livny, M.: Distributed computing in practice: the Condor experience. Concurrency – Practice and Experience 17(2-4), 323–356 (2005)
10. Eerola, P., Ekelöf, T., Ellert, M., Grønager, M., Hansen, J.R., Haug, S., Kleist, J., Konstantinov, A., Kónya, B., Ould-Saada, F., Smirnova, O., Szalai, F., Wäänänen, A.: Roadmap for the ARC Grid Middleware. In: Kågström, B., Elmroth, E., Dongarra, J., Waśniewski, J. (eds.) PARA 2006. LNCS, vol. 4699, pp. 471–479. Springer, Heidelberg (2007)
11. Bubak, M., Malawski, M., Zając, K.: The CrossGrid Architecture: Applications, Tools, and Grid Services. In: Fernández Rivera, F., Bubak, M., Gómez Tato, A., Doallo, R. (eds.) Across Grids 2003. LNCS, vol. 2970, pp. 309–316. Springer, Heidelberg (2004)
12. Laure, E., Grandi, C., Fisher, S., Frohner, A., Kunszt, P., Krenek, A., Mulmo, O., Pacini, F., Prelz, F., White, J., Barroso, M., Buncic, P., Byrom, R., Cornwall, L., Craig, M., Di Meglio, A., Djaoui, A., Giacomini, F., Hahkala, J., Hemmer, F., Hicks, S., Edlund, A., Maraschini, A., Middleton, R., Sgaravatto, M., Steenbakkers, M., Walk, J., Wilsonm, A.: Programming the Grid with gLite. Computational Methods in Science and Technology (2006)
13. PL-Grid Project, http://www.plgrid.pl/
14. Innovative Economy Operational Programme, http://www.poig.gov.pl/english/
15. International Organization of Standardization (ISO), Information technology – Service Management – Part 1: Specification (ISO/IEC 20000-1:2005)
16. EGI Consortium, http://www.egi.eu/
17. Szepieniec, T., et al.: On Importance of Service Level Management in Grids. Accepted for Workshop: Managing and Delivering Grid Services 2011 (MDGS 2011) (2011)

18. Bubak, M., Malawski, M., Gubała, T., Kasztelnik, M., Nowakowski, P., Harężlak, D., Bartyński, T., Kocot, J., Ciepiela, E., Funika, W., Król, D., Baliś, B., Assel, M., Tirado Ramos, A.: Virtual Laboratory for Collaborative Applications. In: Cannataro, M. (ed.) Handbook of Research on Computational GridTechnologies for Life Sciences, Biomedicine and Healthcare, Information Science Reference, ch. XXVII. IGI Global (2009) ISBN: 978-1-60566-374-6
19. GridSpace2 Experiment Workbench, https://gs2.cyfronet.pl/
20. Owsiak, M., Palak, B., Plociennik, M.: Graphical Framework for Grid Interactive and Parallel Applications. Computing and Informatics, 223–232 (2008)
21. Kupczyk, M., Lichwala, R., Meyer, N., Palak, B., Plociennik, M., Wolniewicz, P.: Applications on demand as the exploitation of the Migrating Desktop. Future Generation Comp. Syst., 37–44 (2005)
22. ACARM-ng, http://www.acarm.wcss.wroc.pl
23. QosCosGrid Middleware, http://www.qoscosgrid.org/

Integrating Various Grid Middleware Components and User Services into a Single Platform

Marcin Radecki, Tadeusz Szymocha, Daniel Harężlak,
Maciej Pawlik, Jakub Andrzejewski, Wojciech Ziajka, and Marcin Szelc

AGH University of Science and Technology, ACC Cyfronet AGH,
ul. Nawojki 11, 30-950 Kraków, Poland
m.radecki@cyfronet.pl
http://www.cyfronet.pl

Abstract. The PL-Grid project is constructing a grid computing infrastructure supporting mature middleware suites such as gLite and UNICORE, as well as new ones like QosCosGrid. It also delivers various services, e.g. graphical tools and databases. Users need to be able to access all services and resources in a uniform way, if possible – with a single set of credentials. PL-Grid integrates the middleware and services accessible to users by delivering a coherent platform upon which to interact with the services. The required integration of procedures and tools is described in this chapter. The resulting platform provides the scientists conducting research with a wide variety of powerful tools and resources, enabling them to maximize their scientific potential.

Keywords: PL-Grid tools, single platform, user portal.

1 Introduction

The PL-Grid project aims at providing computing infrastructure to members of the Polish scientific community. Since the infrastructure is geographically distributed, with each site governed by its own rules regarding resource allocation, flexibility is a crucial demand. Managing such an infrastructure is a challenge and demands well-defined operational procedures [1] as well as integrated tools to support them. Providing a uniform computing infrastructure requires integrating services available at various sites. The integration process is non-trivial as a variety of middleware suites and other software components need to be taken into account. As an example, PL-Grid deploys (on the production infrastructure) its most mature software products, i.e. QosCosGrid middleware [2], GridSpace services [3], the Migrating Desktop and gEclipse tools [4]. Each of these services may enforce its own access policies, but all of them need to be delivered and presented to users in a uniform way.

In order to achieve this goal, a common access platform integrating various tools has been created. It enables users to access services with a single set of credentials. Moreover, the platform allows users to easily obtain these credentials and manage them. Tools for user access management, accounting and infrastructure monitoring have to be flexible in order to efficiently deal with various middleware packages. Adding a new service to the existing infrastructure is a sensitive task and there is a need to develop clear procedures for introducing new services in a way which preserves the consistency and uninterrupted operation of the infrastructure. Proper management of the PL-Grid infrastructure requires not only mature services, tools and procedures but also interoperation of these elements with external components derived from European grid projects of which PL-Grid represents the Polish part. Such interoperation is particularly necessary in the field of service availability monitoring, trouble ticket handling and resource usage accounting. The integrated environment through which PL-Grid users may access and use various different services in a unified way, will hereafter be called the Single Platform.

2 PL-Grid Infrastructure Overview

While defining the PL-Grid infrastructure service catalogue, a number of components were identified as essential to create the foundation for higher-level user services. These components and their features are listed below:

– compute and storage facilities (including user access management, user training, computational grants)
– user database (storing user details, account setups and roles)
– service availability monitoring (providing service status information)
– helpdesk system (for handling user requests)

Prior to the establishment of the PL-Grid project, each participating computer center operated its own system for handling some/all of the tasks mentioned above in support of local users. In the PL-Grid project it was decided that information concerning users should be *shared by all participating centers* and that all computing and storage resources available at a site should constitute *a consistent pool* and be delivered to users in a uniform way. Realization of this goal led to integration of various grid middleware packages, tools and user services into a single platform. Additionally, the single platform provided by the PL-Grid project needed to be integrated with internationally accepted tools and grid user databases.

Consolidation of the above features led to the following structure of the PL-Grid platform: **User Portal** – user and credential management, service access management, **Bazaar** tool – resource allocation, **Blackboard** – training activities, **Nagios** – service availability monitoring, **Helpdesk** – handling user requests, consistent **Compute and Storage** facility. All these components were integrated with each other, enabling access to key features with a single set of user credentials (username and password or personal certificate).

The User Portal contains features for both regular users and project staff. These include a common user and staff database as well as the ability to manage user and staff credentials, roles, and services. A key concept is the idea of the *Access Service*, which is the only way to access Compute and Storage facilities. All resources at a site are available via a standard batch interface. This interface is used by *Access Services* – which, in PL-Grid, include gLite [5], UNICORE [6] and the recently developed QosCosGrid [2], as well as direct job submission (in batch mode). Through *Access Services* users can also interface a specific pool of resources such as GPGPU processors and Symmetric Multiprocessing (SMP) nodes.

In order to apply for the *Access Service* each user is first authenticated and authorized. Initial authentication takes place during the registration process which results in granting the user a unique login and password. With these credentials users are able to generate personal X.509 certificates which, in PL-Grid, are equivalent to a login-password pair. Both the login-password pair and the certificate are recognized by almost all PL-Grid tools and services (with the exception of those grid services which have been designed to only accept certificates).

User authorization is based on roles defined in the User Portal and *Access Services* that have been granted to the user. These basic elements are, however, not enough to actually use the resources. In PL-Grid we have introduced the notion of *computational grants*. A grant consists of a Service Level Agreement (SLA) specifying the quantity of resources offered to users and conditions of their use. Computational grants are handled by the Bazaar Resource Allocation Manager, which enables rational planning of resource usage by site administrators and encourages users to plan their resource consumption. At the same time resource providers – although consolidated within the infrastructure – maintain their right to enforce custom resource management policies. Therefore, each PL-Grid computational grant includes a specification of resources at sites where users have their *Access Services* activated. In the proposed model we have assumed that resources defined by an SLA can be used with any *Access Service* the user has activated and that the resources can be used freely with any of the available middleware packages.

Service availability is an important part of the production system. Therefore, all services should be monitored and emerging incidents should be solved as soon as possible. For this purpose a monitoring system has been put in place. The system is integrated with the user database. Incidents are reported to the helpdesk system by operating teams.

The helpdesk system was designed to serve as a central place to track issues concerning the PL-Grid infrastructure. It provides users with the ability to handle problems with such tasks as account registration, applying for roles, accessing services or grants etc. It also enables project staff to report and track solutions of incidents occurring within the infrastructure.

The following part of this paper focuses on integration of the previously mentioned components and describes the tools that support this goal. We also present how the integration is beneficial to PL-Grid users and staff.

3 Processes Integrated in User Portal

The main reason for introducing the single-platform paradigm was to make it easier for users to access resources at each phase of interaction with the infrastructure. This has been achieved through uniformization and centralization of the service management process within in the User Portal.

In order to meet the stated goal, project services were analyzed in order to identify common stages of usage. These stages include: obtaining user credentials, requesting access to services, actual service usage, activity monitoring and – finally – accounting. The following subsections describe the features implemented in the User Portal.

3.1 User Management

User management in the PL-Grid Infrastructure is handled by the User Portal. Upon receiving an account the user obtains access to authorization methods which are valid throughout the whole infrastructure. For this purpose a user database has been created which allows for user authentication and keeps track of user roles.

Registration Process: Users, Staff Members, Trainees. Three modes of registration are supported by the Portal:

1. Infrastructure Users – regular users whose aim is to use resources,
2. PL-Grid Staff – employees who require accounts for infrastructure development and maintenance,
3. Trainees – participants of training events.

Each class of users comes with its own procedure for obtaining an account of a given type. Typical steps include affiliation confirmation, supervisor review etc. Once an account is activated the user may apply for access to tools and services adequate for the given account type.

Roles: Authorization and Access Control. In order to control authorization within each class of users more precisely and efficiently, user roles have been introduced. For example the *Access Service Administrator* role allows its holders to manage site resources as well as user access to these resources.

User Teams. In order to facilitate user cooperation when using the PL-Grid infrastructure, we have introduced *user teams.* A global team can be created by any user of the infrastructure. As a result, it is possible for team members e.g. to share resources, share common files on a computing cluster or collectively apply for access to resources.

User team membership is structured hierarchically: there are regular team members and team managers. Team managers are entitled to manage other users' membership status through dedicated management tools. This feature allows managers to take care of the formal issues whereas the remaining members can devote their time to actually using the infrastructure in order to achieve the team's scientific goal.

User Authentication and Authorization. From the perspective of the infrastructure the most important pieces of information about users are their identities and the services they are entitled to. A unique and unalterable login constitutes the main user ID which – along with a password – provides a basic authentication method. One user can only use a single ID within the infrastructure.

Grid tools and services frequently consume the so-called "grid credentials" which are personal certificates issued by a trusted Certification Authority (CA) – for instance, the Polish GRID CA [7].

Obtaining a personal certificate from a CA supported by EGI (namely those affiliated with EuGridPMA [9]) is often troublesome as it requires the user to personally visit a Regional Authority office. In order to streamline the certificates acquisition process a new online CA, called PL-Grid Simple CA, has been created [8]. It requires certificate applicants to be registered users or trainees (which implies verification at an earlier stage, i.e. during the registration process) who have the right to use the infrastructure. Managing credentials through the User Portal is possible as users will have already been authenticated with the portal during the login process. However, the easily-obtainable Simple CA certificates are only accepted within the PL-Grid infrastructure.

Users can manage their Simple CA personal certificates in the User Portal – from generating a certificate through downloading it to (possibly) canceling it and receiving a new one. Registration of personal certificate (Polish Grid CA or Simple CA) in the User Portal provides users with the ability to be automatically recognized by all services present in the infrastructure, such as UNICORE and gLite. It also provides access to other convenient features such as automatic Portal login.

Certificate generation tools (SimpeCA only) also support training certificates which can be issued only upon a trainer's request and are valid only for the duration of the training course. Training courses are carried out using the production infrastructure due to the bulk nature of training jobs and the need for high-quality services. Credentials for trainees become an issue as such people usually do not meet the conditions for obtaining regular PL-Grid accounts. The ability to automatically generate limited-access certificates for all trainees has largely streamlined the training process.

3.2 Access to Tools and Services

Service Model. When a user receives a PL-Grid account, they also obtain access to support tools such as the User Portal and Helpdesk. Other services, especially the infrastructure *Access Services*, require the user to apply for them separately and receive clearance from the appropriate staff member (usually a service administrator).

In order to unify this process for a variety of services a "standard access service model" has been developed. It refers to a unified process of applying for a service and providing user support. The process is flexible enough to cover all of the services provided within the project. It can also reflect the individual needs of service providers. Various services may have differing requirements, such as requiring a certificate or prior access to another service, or some additional information related to the expected use of resources. Various phases of a service activation process (as well the whole process) may be automated, depending on the service administrator's wishes. It is also possible to activate a group of services with a single process so that the user does not have to apply for each service individually. All these possibilities have been left open in order to allow service administrators to decide for themselves how the activation process is to be conducted for the services they provide.

Unified Service Portfolio. All *Access Services* are listed in the User Portal in a uniform way: each record contains the service name, a link to the relevant documentation and the current level of access for the user. New services have to be delivered in a preformatted way, which includes all the above information. As a result, users receive a uniform list of available services.

Data Distribution Services. Integration of various services in a single platform implies close interaction of many components. Such interaction is necessary for exposing some of the features of the services within a unified platform. Individual components have to maintain close cooperation for the whole process to be consistent.

Component interaction can follow two different paradigms. The first involves parallelizing actions which belong to the central system and those performed by the service. In such a model performing an action in the central system triggers an effect in one or more services. The central system is capable of "pushing" actions to external services. The alternative solution implies that data needs to be made available to an external service upon request. In contrast with the previous model, this system is based on "pulling" data (for instance, user account information) from the central system by services whenever required.

A hybrid of both paradigms has been used for data handling. Specifically, a middle layer separating the central system and the services has been developed. This layer consists of a special service called LDAP, aggregating all user data, which is then distributed among services. The model works in such a way that, at first, the central system pushes data to LDAP (first paradigm) which then makes it available to client systems so that they can pull data from LDAP (second paradigm) at any given time. Data propagation is depicted in Fig. 1.

The User Portal acts as the authorizing system for user data. In some cases it may apply identity and authorization delegation mechanisms (SSO).

Replicators as Service Manipulation Tools. The model of data handling described above implies the need for a flexible means of communication between the centralized system and the services. In order to solve this problem, a set of so-called *replicators* has been introduced. The main concept is that each service can be linked with a replicator which propagates information from the central platform so that the service state for a given user is unified throughout the infrastructure.

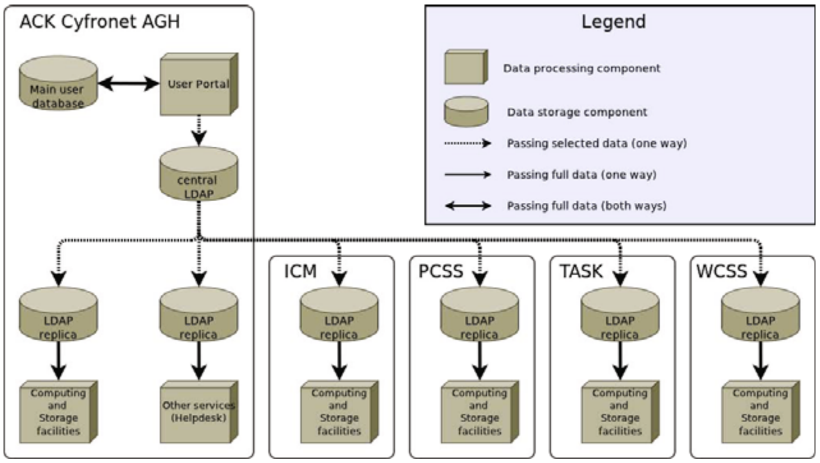

ACK Cyfronet AGH
Main user database
User Portal
central LDAP
LDAP replica
LDAP replica
Computing and Storage facilities
Other services (Helpdesk)
ICM
PCSS
TASK
WCSS
LDAP replica
Computing and Storage facilities
Legend
Data processing component
Data storage component
Passing selected data (one way)
Passing full data (one way)
Passing full data (both ways)

Fig. 1. User data flow in the PL-Grid infrastructure

In this way every step in the service lifecycle (initiated in the central system) triggers an event which is then passed to replicators. Replicators are modules that include an interface enabling them to receive events. Each replicator receives every event signal sent by the system. When a replicator recognizes the signal as related to the service the replicator is responsible for, it performs an appropriate action. Such actions most commonly refer to some changes in the service user database.

The user data propagation method described above is used – for instance – in the management process of middleware packages such as gLite and UNICORE. These services use appropriate servers called the Virtual Organization Membership Service (VOMS) for gLite and the UNICORE Virtual Organization Systems (UVOS) for UNICORE, which contain user databases. However, there is also a group of services which only use LDAP as a source of user data. These services require a single database and do not need the data to be propagated further as the LDAP database includes all the necessary attributes that govern access to services.

External Tools. The diversity of scientific domains in which grid computations are conducted has resulted in a considerable number of tools developed to facilitate such research. For the PL-Grid Portal to be ready to integrate and deliver tools to users a dedicated mechanism had to be implemented. The main requirements from the tools were to be able to retrieve basic information about the current user and to have a well-documented instruction set on how to embed a given tool in the portal. Since our portal is based on the Portlet Specification [10], which is a well-established standard for web integration, the path towards preparing a tool to be embeddable in a portlet container is straightforward. The user data sharing requirement was also handled by using a common standard – namely, REST services [11]. XML was applied as a standard information exchange format. This approach assumes that a tool features a front-end in the form of a web page, able to invoke a remote REST service. The second requirement is, however, optional: if it isn't satisfied the tool may still be embedded but it will not retrieve any information about the user currently logged in. The deployment process does not require the Portal to be taken offline for maintenance as new tools can be added in the course of normal operation. As part of the protocol, a designated user needs to be given the role of the new tool's administrator and a new web space needs to be created. Within this web space the administrator is able to arrange the tool's portlets and take care of maintaining them. The channel through which user information is transferred from the portal to the tool is established either through a local API or a remote secure connection.

3.3 Use of Services

Various PL-Grid services may share the same resources – computations from various middleware packages are all performed on the same pool of computing nodes no matter which middleware package is used. In order to make the use of resources more easily predictable and manageable for users and resource administrators, we have introduced ways of measuring resource use and granting access to specific amounts of resources.

PL-Grid Computational Grants. Having registered with the PL-Grid Portal and received access to appropriate roles and authorizations the user may begin using the infrastructure. The use of resources is governed by the PL-Grid *computational grants* which define the amounts of resources and carry other requirements that need to be satisfied in order to perform computations (such as additional software or slot reservations).

In PL-Grid grants are given only to user teams. This requirement stems from the nature of scientific research where (usually) several people collaboratively pursue a specific goal. An exception to this rule comes in the form of personal grants, given to single users for testing the infrastructure or conducting minor computations without the need for regular grants.

The lifecycle of a grant begins in the PL-Grid Portal where the user (on behalf of the team) fills out a form where they define what scientific purpose the grant is going to serve. The application process also includes definition of parameters in which the users wish to receive the resources. Subsequently, the application is reviewed so that resources can be allocated in required amounts, or in amounts that suit the resource providers. If the suggested parameters are not accepted by the user, there is room for negotiation so that, in the end, a solution that suits both users and resource providers can be reached. Once the negotiations reach are over, at the time specified in the grant, the grant becomes active and all team members can use the resources. After the grant expires it needs to be subjected to accounting. This is the final phase of the grant process and it will be described later on.

Service Availability Monitoring. In an ideal world all services provided in the infrastructure should be available at all times. However, in real situations service downtime may occur – this is typically related to maintenance or unexpected incidents, both resulting in service unavailability. In light of this fact we have integrated a monitoring system which monitors the availability of all services provided within the PL-Grid Infrastructure. The monitoring system is based on Nagios – a popular software package used for IT infrastructure monitoring. In order to introduce the monitoring service and integrate it with the platform it is necessary to establish ways and mechanisms of monitoring.

In the PL-Grid infrastructure all services are monitored – this includes supported middleware suites, services developed by the project as well as scientific applications (e.g. Gaussian, Fluent). Introducing any new software into the infrastructure requires developers to deliver a functionality probe for this software.

Service unavailability happens rarely enough that, in the service model provided to users, the topic of availability has been omitted. This simplification of the service model is meant to reduce its complexity; however, if necessary, users can always check the current state of service availability via the monitoring service interface.

We define service availability in terms of basic requirements which may be established differently for different service types. These requirements usually refer to interface accessibility, user authorization features, delivering the declared functionality, response time etc.

The monitoring system can control even the most complex services. If services are dependent on one another, the system can recognize the root of the dependency tree, providing a convenient starting point for troubleshooting and maintenance.

Grant Metric Monitoring. The Resource Usage Monitoring system (shortened to MWZ following its Polish name) provides information regarding the amount of resources (in hours spent by a process on a single core) used by PL-Grid users as part of their grants. Such information facilitates grant accounting, both globally and separately for each PL-Grid resource center. Additionally, the system provides information on the current use of computing clusters, including the number of jobs being executed as well as those waiting in the queue. Delivering this information helps administrators manage their resources efficiently.

The MWZ system enables integration of accounting data from all grid middleware packages supported by the PL-Grid Infrastructure. Data is collected through a universal protocol (in the XML format) and then uploaded to a database. Such records can then be processed so that the outcome contains aggregated information regarding the use of resources within computational grants from all participating resource centers. Information regarding the use of resources is presented in the user's account in the User Portal.

Grant Accounting. From the resource providers' point of view it is important that the PL-Grid Infrastructure is used according to its declared purpose. In the context of computational grants the purpose is specified in more detail whenever a grant research subject is described by the user. This subject, as well as the user's declarations regarding the amount of resources required to conduct the research, are later reviewed by PL-Grid representatives and taken into account by resource providers during the resource allocation process.

Declarations filed by the user during this initial phase are later assessed from two points of view: scientific merit and the use of resources. The bases on reports which users have to periodically prepare during the activity covered by the grant and after its termination. The purpose of such reports is to compare the declared grant objectives with actual research results. The use of resources is controlled through MWZ-generated analysis reports and information on the declared resources.

Irregularities may result in grant suspension, i.e. revocation of the right to use resources associated with a given grant. Typical issues involve e.g. lack of proper progress reports or exceeding the declared amount of resources. In the future a dedicated subsystem will be implemented to track the actual use of resources in relation to the declared use and block access to resources whenever the former exceeds the latter.

4 Processes Integrated in Helpdesk – Infrastructure Support

PL-Grid infrastructure support is realized by the Helpdesk System. It has been designed to manage and track service requests regarding incidents, problems, features etc. Tickets can be submitted online, on a round-the-clock basis. Requests are then handled by groups of specialized experts during business hours. Experts are grouped into teams supporting particular components or resource types.

Our primary aim in developing this system was to provide a single point of contact for user requests regarding any matter, as recommended by the Information Technology Infrastructure Library (ITIL) [12]. The Helpdesk can therefore handle requests from infrastructure users as well as operations staff.

Once registered, a request is directed to an appropriate suitable support unit by the 1st Line Support Team which plays the role of request administrator. This team is responsible for routing all tickets (independently of middleware), contacting users, closing requests as well as ensuring all requests are properly handled by support teams. Adding a new service or middleware package to the infrastructure always requires creating (or choosing an existing) support team which will be held responsible for handling support calls related to this new element of the infrastructure. After a new team is created or an existing one is given the responsibility, team members are introduced to their new duties and information on the new team is passed to the 1st Line Support Team.

Access rights to the Helpdesk System must be provided both for regular users and staff. This is realized uniformly, based on the PL-Grid user database (through LDAP), as well as according to the Grid Operation Center Database used to identify resource center administrators. The PL-Grid Helpdesk is also integrated with the EGI Global Grid User Support (GGUS). Our Helpdesk System supports mutual information exchange and ticket synchronization with GGUS.

5 Training

Training courses are an especially important element of the infrastructure popularization process in the academic society. Such courses can be conducted in a twofold way: either in a traditional fashion where the physical presence of trainees and trainers is required at a particular place and time, or online, at any place and time suitable to trainees.

Both methods require trainee authentication as well as access to resources. The PL-Grid infrastructure has been prepared to fulfill such requirements. The main advantages of using the the User Portal in the process of preparation and execution of training sessions include performing user registration via the Portal as well as issuing personal certificates. Moreover, each trainee can receive access to the integrated Blackboard [13] platform which contains helpful training materials. Other benefits related to training courses include access to a convenient User Interface. This allows users to familiarize themselves with a large scope of practical information regarding the infrastructure, which can later influence their decision to potentially continue cooperation with the PL-Grid project and request production-level access. Once a training course has concluded, trainees are easily able to upgrade their accounts to regular Infrastructure User accounts provided that they meet the formal requirements.

6 Summary

In this paper we have presented our approach to integrating different types of services (i.e. grid middleware packages, graphical tools and other services) in one platform. The main task was to identify key aspects related to user interaction with infrastructure service and to develop flexible and generic procedures governing such interaction. This goal can be achieved with support from advanced portal frameworks. The core part of our system, namely the PL-Grid User Portal, is based on Liferay software. A key aspect of service integration is the concept of Access Services and the agreement on a common interface for computing resources. The resulting platform provides precise definitions of new services while respecting the principles of service consistency and stability throughout the entire infrastructure. It also facilitates user interaction, enabling users to perform their work online (including obtaining credentials). Maintaining such a platform is a challenge. There are many dependencies upon external services and the core of the system must be prepared for failures of some of these services, reacting gracefully to any emerging problems.

References

1. Radecki, M., Szepieniec, T., Szymocha, T., Szopa, M., Krakowian, M.: Towards professional service operations in grids. In: Bubak, M. (ed.) PL-Grid 2011. LNCS, vol. 7136, pp. 27–39. Springer, Heidelberg (2012)

2. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)
3. Bubak, M., Baliś, B., Bartyński, T., Ciepiela, E., Funika, W., Gubała, T., Harężlak, D., Kasztelnik, M., Kocot, J., Malawski, M., Meizner, J., Nowakowski, P., Rycerz, K.: The Capabilities of the GridSpace2 Experiment Workbench. In: Proceedings Cracow Grid Workshop (2009)
4. Palak, B., Wolniewicz, P., Płóciennik, M., Owsiak, M., Żok, T.: User-Friendly Frameworks for Accessing Computational Resources. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 191–204. Springer, Heidelberg (2012)
5. gLite – Lightweight Middleware for Grid Computing, http://glite.cern.ch/
6. UNICORE – Uniform Interface to Computing Resources, http://www.unicore.eu
7. Polish Grid Certification Authority, http://www.man.poznan.pl/plgrid-ca
8. PL-Grid Simple CA, http://plgrid-sca.wcss.wroc.pl
9. European Policy Management Authority for Grid Authentication (EUGridPMA) (2011), http://www.eugridpma.org
10. JSR-000286 Portlet Specification 2.0, http://jcp.org/aboutJava/communityprocess/final/jsr286/index.html
11. Fielding, R., Taylor, R.: Principled Design of the Modern Web Architecture. ACM Trans. Internet Technol. 2(2), 115–150 (2002)
12. van der Veen, A., van Bon, J.: Foundations of IT Service Management Based on ITIL V3. Van Haren Publishing (2007) ISBN: 9789087530570
13. Blackboard platform, http://www.blackboard.com/ or https://blackboard.cyfronet.pl/webapps/login/

Towards Professional Service Operations in Grids

Marcin Radecki, Tomasz Szepieniec, Tadeusz Szymocha, Magda Szopa, and Małgorzata Krakowian

AGH University of Science and Technology, ACC Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków, Poland
m.radecki@cyfronet.pl
http://www.cyfronet.pl

Abstract. Management of large distributed IT infrastructures requires a Service Operations framework covering day-to-day maintenance. In this chapter we describe the Service Operations model developed in the PL-Grid project. First, we sketch the processes that needed to be covered by Service Operations. Subsequently, we present the management solutions worked out in PL-Grid and describe how they were influenced by the Information Technology Infrastructure Library (ITIL). We also show how to adapt ITIL rules to manage geographically and administratively distributed infrastructures, namely those providing grid resources to the academic community. We claim that adapting a certain Service Operations model is a prerequisite of developing a mature infrastructure, ensuring user satisfaction.

Keywords: grid infrastructure, IT management, ITIL, Service Operations, PL-Grid.

1 Introduction

The PL-Grid project [1] aims at building and providing computing infrastructure in order to support Polish scientific research. The underlying resources are contributed by five of the largest Polish computer centers – partners of the PL-Grid project – and by other academic institutions willing to share their facilities on the Grid. The PL-Grid infrastructure is part of the European Grid, coordinated under the auspices of the EGI.eu foundation [2]. PL-Grid continually extends the infrastructure by adding new services while maintaining compatibility with the European Grid.

A grid infrastructure is, at its core, geographically and administratively distributed and therefore its smooth operation requires coordination of many teams' efforts. Cooperation is crucial for achieving the common goal, which is defined as end-user satisfaction regarding the computational and storage resources they use.

Computational and storage resources are presented to the researchers through a set of services, allowing them to manage their computations and data. Some scientists are satisfied with the way they use the infrastructure. However, there are some research teams which find it essential to precisely plan their computations and receive guarantees as to how and when the requested resources will be provided.

One of the main success measures for any IT service is its end-user availability. Since availability is a serious challenge in large, geographically distributed infrastructures, the common approach to improving availability is to deploy services in multiple instances. It should be noted that the grid service stack is very rich and that grid services are often sophisticated and frequently depend on one another. For these reasons it is impossible to avoid faults and incidents completely. Rather, any faults and incidents must be efficiently managed. From the user's point of view service unavailability is often worse than its nonexistence as it means that users are unable to perform activities which they have planned and invested effort in preparing.

The term "Service Operations", as described in this paper, is understood as a total set of actions related to service management, starting from the design of the service, through its implementation, deployment and maintenance up to its termination. The greatest challenge for PL-Grid Service Operations is organizing the infrastructure in a way which supports guarantees for users regarding the delivery of services and a certain level of service quality. Therefore, we see the need to introduce support for Service Level Management (SLM) [3] which is understood as the ability to provide resources on an agreed-upon service level. On the one hand, the user specifies the resources and their expected quality while on the other the service provider negotiates, accepts and endeavors to provide the desired quality. This process is highly laborious not only in the sense of providing appropriate user services, but also in the sense of introducing ways to measure and control infrastructure parameters. Supporting SLMs is, however, extremely beneficial for users as it brings certain guarantees regarding resource delivery. It is beneficial not only for the users but also from the infrastructure point of view as having information about user needs makes it possible to plan the use of resources.

In this paper we describe the experience and achievements gathered by the PL-Grid Operations Center in the field of Service Operations in our infrastructure. Firstly, we present the motivation behind improving the level of service quality, including guarantees regarding the available amounts of resources. Secondly, we present what others have done in the field of introducing service level standards in IT infrastructures, especially in distributed environments such as grids. The core part of the paper is devoted to presenting how ITIL – the standard we decided would suit our infrastructure best – has been introduced in PL-Grid. Finally, we show what remains to be done and what we are currently working on in the field of introducing service level standards in the PL-Grid Infrastructure.

2 Motivation

Due to its inherent complexity, different functional areas of the Infrastructure are handled by various teams. One of them is the PL-Grid Operations Center team whose objective is to keep the production infrastructure available to users. The team consists of several people with various roles: Operators, Technical Support, Tool Developers and Maintainers. External relations are kept with Trainers who organize training sessions and online courses for users, with Programmers who produce and prepare software for deployment in the infrastructure and with software Testers. Finally, there is a large group of regular Infrastructure Users. The number of teams shows that smooth cooperation and reaching common goals is not possible without defining and documenting the process.

It is easier to contribute resources to users on a *best effort* basis than to give guarantees that services will remain on a particular level. In order to provide services that will satisfy the users' computing requirements it must be possible to provide users with appropriate guarantees. We believe that teams must approach Service Operations in a way that coordinates the work and thus creates space for introducing higher standards of service provisioning. Before giving guarantees it is important to define proper ways to measure the level of provided resources.

Commercial entities which deliver IT services on the basis of a business model adopt various service management standards [4]. Without them, such entities would not be able to provide services on a level which, from the users' point of view, would be worth paying for. PL-Grid tries to reach a level similar to that represented by commercial entities while providing resources free of charge to the academic society. Therefore, we have found it extremely useful to draw on others' experience and to adapt some solutions present in those standards. We thus chose to rely the on the Information Technology Infrastructure Library (ITIL) [5] which appears to be the most mature and functionally complete solution in the area of service operations.

3 Related Work

In this section we will present how the problem of introducing a Service Operations framework into an IT infrastructure has been approached by others. We will focus mainly on papers referring to distributed IT infrastructures (such as grids) and infrastructures serving the academic community.

To the best of our knowledge, there are relatively few scientific papers regarding IT Service Management (ITSM). Works based on ITIL referring to Service Operations and IT maintenance centers for campus environments can be found in papers by S.H.C Wan [6] and Z. Yang [7]. Although this experience contributes to IT management, campus management essentially differs from grid management as campuses typically do not include separate administrative domains. Perhaps the most interesting outline of challenges inherent in adopting SLM to the Grid is presented by T. Shaaf et al. in [8].

On the other hand, there are papers about ITIL and its applications in business, written from a more general point of view. For example, B. Barafort et al. describe the use of both ISO 15504 and ITIL in Small and Medium Enterprises [9]. Nevertheless, companies do not seem to willingly share their ITSM experience as they are afraid it will disclose their internal strategies to the public.

Therefore papers related to ITSM in private companies are limited in number and often vague. One such paper was written by A. Hochstein et al., and shares some general management experience from several companies [10].

There are also some papers related to Service Operations in grid infrastructures, which, however, do not refer to any existing management standards and instead mainly focus on technical solutions. For example, T. Fieseler and W. Gurich describe technical solutions achieved in a German grid infrastructure D-Grid [11]. On the other hand, J. Coles [12], shares the experience of building a European grid infrastructure within a series of Enabling Grid for E-science and collaborating projects [13]. However, none of these papers directly expresses the need for working out a Service Operations framework for grid infrastructures based on a preexisting standard. Moreover, the experience of the authors of these papers shows that these projects implement Service Operations on a basic level where services are delivered only on a *best effort* basis, leaving users with no guarantees that services will be any better than *as is*. Our ambition is achieving a higher level of service quality, which, in our opinion, cannot be reached without adopting a specific Service Operations standard.

4 Landscape of ITIL Processes in PL-Grid

ITIL [5] contains a number of documents describing methods (best management practices) to achieve high quality in the field of IT Service Management. It was first established in the late 1980s by a British government agency called CCTA. Since 1989 it has been registered as a trademark of the British Office of Government Commerce (OGC) which has published a series of books on ITIL. It focuses on processes which lead to customer satisfaction rather than on technologies, and thus helps reduce costs. For this reason it has been recognized as universal and widely adopted by various IT organizations not just in the UK and Europe but also worldwide. The main benefit of adapting ITIL is that it focuses on customer-oriented management and thus makes processes more efficient and effective.

ITIL is constructed in such a way that it contains rules regarding process organization; however, the most difficult task in applying ITIL in a particular company is finding the proper way to implement these rules inside a specific organizational environment. In subsequent chapters we will describe these ITIL processes have been identified as crucial for the purposes of PL-Grid.

In this chapter we present an overview of processes defined by ITIL that either have been adopted within PL-Grid or have, in some way, inspired our Project. Detailed descriptions of process implementation are also presented.

4.1 Service Level Management

Service Level Management (SLM) is an ITIL process whose aim is to make sure the services delivered to users reach a certain pre-defined level. Introducing such requirements entails keeping the services on a level better than that produced by *best effort*. When SLM is introduced users who apply for access may also specify requirements regarding the service level.

In PL-Grid SLM is realized through the concept of computational grants. A grant allows users to express their needs and expectations while simultaneously enabling service providers to answer and negotiate the users' calls for resources. When applying for a grant the user specifies the details of a Service Level Agreement, an agreement which includes information about parameters and location of resources they are interested in. Each grant is divided into two parts: formal and technical.

Applying for a grant begins in the *User Portal* where the user specifies their research topic and predicted scientific outcome of the grant. In parallel, the user specifies his/her demand for resources in the Grid Resource Bazaar system [14]. This declaration, containing both formal and technical descriptions, is then reviewed by a PL-Grid representative who grades the proposal depending on the strength of the predicted results in relation to the specified need for resources. Once the grant has been reviewed the process of negotiations between resource providers and the user begins. Again, this takes place in the Grid Resource Bazaar and is coordinated by a PL-Grid Operator. Negotiations end with signing SLAs – agreements between providers and the user concerning the resources they have applied for. Once negotiations have finished the grant is activated and information about SLAs is added to PL-Grid LDAP which is a source of grant information for other tools. After the grant becomes active the user has to file grant activity reports every 6 months; upon grant termination a final report has to be submitted.

Introducing grants enabled us to solve the resource allocation problem: resource providers can now plan and manage the use of resources while users can receive guarantees that resources will become available to them according to SLAs. This change has raised the level of services offered by PL-Grid – instead of being an *as is* infrastructure, it is now based on agreements which guarantee resource delivery according to specified parameters.

Additionally, Service Level Management in PL-Grid supports another ITIL process, called Capacity Management. Its aim is to enable resource occupancy planning (via Capacity Plans) which includes ensuring that the resources delivered to users match their requirements. By introducing grants in PL-Grid it is possible for the administrators to observe user plans and thus predict future demand for resources.

4.2 Request Fulfillment

Request Fulfillment (also called Service Request) is an ITIL term which refers to the process of fulfilling users' requests that are not results of any sudden and unexpected failures (incidents) but rather refer to some additional user needs. This process includes realizing standard requests that are inexpensive to fulfill, such as changing account parameters etc.

In PL-Grid users can file *service requests* whenever they feel there is something they need regarding the PL-Grid Infrastructure. Some of the typical and simple to realize service requests have been implemented in the *User Portal* with specific forms that need to be filled out by users. These requests include: registering as a new user, obtaining X.509 certificates from Simple CA, obtaining test accounts for project staff.

Other requests can be handled by the Helpdesk. If analysis indicates that a given request would incur significant costs, it may be handled under Change Management, which involves detailed planning – including the circumstances, cost and risk calculations that need to be performed before implementation can be decided upon.

4.3 Change Management

The Change Management process includes preparing and coordinating changes regarding the infrastructure. Changes need to be carried out carefully as they are prone to affect user work. In particular, all dependencies and consequences need to be analyzed.

In PL-Grid change proposals are based on user requests (submitted by regular Infrastructure Users or PL-Grid staff). Each change is analyzed and coordinated by the Change Manager who "translates" the user's request into the necessary modifications in the infrastructure. Changes are consulted with site representatives and the security board before making a deployment plan.

Release Management is a related process, which, in PL-Grid, relates to preparing and releasing new versions of software packages. Releases in PL-Grid may refer to scientific applications, project-produced software, grid middleware or updates to tests within the monitoring system. For example, scientific applications provided by PL-Grid are supported by their creators whereas new versions can be installed by site administrators based on user requests. If a new version of a middleware package needs to be deployed, the Operations Center first recommends that it be deployed at a pilot site and then throughout the PL-Grid computing infrastructure. Core services, which are indispensable for proper functioning of the infrastructure, operate in multiple instances so that while changes are being introduced, at least one of the instances remains available to users.

4.4 Access Management

Access Management is the part of the ITIL Service Operations area which focuses on the rules and methods for granting access to resources or services. In addition to resource access, it also covers managing user permissions within the infrastructure and removing them at appropriate time. Access Management is an especially important area of management because it deals directly with customers and thus it directly influences the way the company is perceived by the market.

This section specifically discusses access to resources, provided to users through a mechanism called *access services*. In some contexts it may also refer to access available to PL-Grid employees which allows them to manage users and services. On the other hand, access to physical resources and low-level software is reserved for site administrators and is not treated as part of Access Management.

A requirement specific to our infrastructure is that every person who obtains access to services must either be affiliated with a scientific organization or have a supervisor within such an organization. Additionally, they may only use such access to conduct research within a specific organization. Therefore PL-Grid has introduced its own methods for Access Management, which include user verification and focus on scientific affiliation.

Our second distinctive requirement is the ability for a group of users (and even for a single user) to apply for a given amount of resources. This process, called *Resource Allocation*, is also coordinated in the project. The PL-Grid resource allocation process is therefore different from EGI where access is negotiated by the Virtual Organization (VO) manager who grants access to VO resources. Applying for resources in the European Grid relies on direct contracts between users and VOs on the one hand, and between VOs and resource providers on the other.

In order to manage user access, PL-Grid has developed an online tool called the *User Portal*. The primary use of the *User Portal* is to manage access to resources offered within PL-Grid. During registration the applicant's academic record is verified by the Information Processing Center [15] which maintains a database of information concerning the Polish scientific community. Subsequently, we make sure the applicant is employed by an academic organization or that they have a supervisor who fulfills that condition. When affiliation with the employer or supervisor expires and is not replaced by a different affiliation, the relevant account is blocked.

Access to resources and services is realized through a mechanism of user roles and rights. For example, an applicant first receives the Infrastructure User role and then they apply for services available for this role, such as *access services*. We believe this access model is adequate for real-world services and the way they are provided. Elements of the User Portal which serve the goal of granting resource access to authorized users include: user roles, user credentials, access services and computational grants [16].

Trainee access is an especially interesting topic as it includes access to the production infrastructure but only in a limited scope and for a limited period of time. A Trainee is granted all the accesses required by the training session (a special short-lived personal certificate, access to the production infrastructure services), but only for a period defined by the training schedule.

PL-Grid supports Certification Authorities approved by the European Policy Management Authority for Grid Authentication [17] including Polish Grid CA [18]. However, in order to make certificate management easier for users, we have introduced the PL-Grid Simple CA [19] which issues online certificates valid only within the PL-Grid infrastructure. Users can obtain and manage these certificates through the User Portal.

4.5 Event Management

Event Management is an ITIL term describing processes which handle event monitoring within the infrastructure in order to ensure proper operation of the system as well as escalating any irregularity reports. This process requires a definition of what is considered *an event*, how such events are monitored and how they need to be handled.

As in any geographically distributed and complex infrastructure, PL-Grid Service Operations aim at identifying and removing faults before they can stand in the way of user work. In order to identify appearing faults a monitoring system has been introduced.

Events that are monitored by the system relate to faults in the features available to users. Symptoms that may result in service unavailability are monitored as well – this applies e.g. to host memory load of over 99% or upcoming host certificate expiration date. Early detection of such problems allows us to follow the principle of not only reacting to dangers, but mitigating them in advance.

If a fault ends up affecting end users, such users may also trigger the corresponding handling procedures by filing a report with the Helpdesk. This kind of situation happens when an element of the system has not yet become subject to monitoring. Consequently, user reports help us improve the monitoring system.

The monitoring system in PL-Grid is based on the Nagios software [20] and was designed as an extension of a similar system present in EGI. Owing to this approach we remain compliant with EGI and are able to extend its achievements towards our specific needs.

It is crucial for Service Operations that, when a new service is introduced, it must come with appropriate probes for the monitoring system. The procedure of introducing new services in PL-Grid requires functional probes as well as accepted means of handling events. Our procedures for adding new probes include validation on a testing instance followed by deployment on the production instance of the monitoring system.

An example of introducing a new services in PL-Grid is integration of UNICORE middleware. It has been handled in cooperation with EGI and a specific set of probes for UNICORE monitoring has been prepared and deployed [21]. PL-Grid extends monitoring to services that are not monitored by EGI. The most important of these services is the PL-Grid User Interface (UI). Monitoring the UI introduces specific needs: it requires access to a shell account within the service so that its features can be monitored exclusively from the inside.

4.6 Incident Management

Incident Management is an ITIL-defined process focusing on restoring services to their proper state as early as possible after an incident. It manages incidents, defined as sudden and unexpected events which include unavailability of a service. Efficient Incident Management requires immediate reaction, accurate diagnosis and precise reparation. Therefore, it is an especially important part of infrastructure maintenance and has been the subject of special attention in PL-Grid.

An incident can be reported in two ways: either by the monitoring system or by an infrastructure user. The former results in alarms displayed by the Operations Dashboard [22]. A persistent alarm may trigger a ticket in the PL-Grid

Helpdesk. Users can report incidents by directly submitting tickets to the same Helpdesk system.

We use the same rules for incident handling and other issues reported by users in their tickets. The team which replies to tickets is organized in *support units.* The scope of each support unit may be limited to a specific site (as in the case of site administrators) or it may be distributed among several centers (e.g. a collaboration of application experts representing various sites).

Escalation Levels. Helpdesk handling rules include *functional* and *hierarchical escalation* processes described below.

Functional. Several teams covering key elements of the infrastructure have been set up in the Helpdesk system. Each team is assigned to one of three support lines. 1^{st} Line Support is responsible for quickly responding to user calls and solving trivial cases. If the resources available to the 1^{st} Line are insufficient to solve the problem, functional escalation takes place and the case is referred to the second or third line. 2^{nd} Line consists of domain experts who have detailed knowledge in their fields and can solve cases using this knowledge. If privileged access to services or software code is needed, the tickets are passed to 3^{rd} Line Support – people with specific roles or credentials facilitating additional access to services or resources (for example, site administrators or programmers). If a bug is found in middleware, the relevant ticket can also be exported to GGUS for processing in an appropriate support unit in cooperation with EGI.

Hierarchical. When *Resolution and Recovery* steps take too long, hierarchical escalation is used. Issues not treated in accordance with procedures are first checked by 1^{st} Line Support representatives, who send email reminders to the support unit. If this does not produce a resolution, 1^{st} Line Support makes phone calls to support unit experts. The second escalation step is a discussion about the troublesome tickets (with invited experts) at an Operations Center meeting. If the expert does not attend the meeting and no representative from the expert resource center is willing to deal with the problem, contact with the expert's supervisor is foreseen. The last step involves reporting the issue to the PL-Grid Technical Deputy Director.

Incident Management Process. The Incident Management process in PL-Grid follows ITIL recommendations and consists of:

Incident Detection and Logging. Monitoring system performs regular checks of grid services. Monitoring probes examine important features from the user perspective. Probe results are reviewed by the Operations Center team. When a service fails it is referred to as an *alarm.* Since the location of the service is well known, a Helpdesk ticket can be assigned directly to the responsible support unit. Incidents affecting users are reported in a similar way i.e. as tickets in the Helpdesk. Such tickets are then routed by the 1^{st} Line Support team.

Categorization and Prioritization. Tickets based on monitoring system results can be routed directly to the responsible support unit. Priorities are set by the submitter who is usually an Operations Center team member. Ticket priority is understood as the user's recommendation concerning the urgency of the incident, as every support unit expert (site administrator) has autonomy in setting priorities. Tickets created by users are assigned to categories by the 1^{st} Line Support team (whether it is a request for information, bug report, request for a new feature etc.) Priorities are set by the submitter but it is the support unit expert who sets the final priority according to the current workload.

Initial Diagnosis. In the case of tickets created by users 1^{st} Line Support has to ascertain the actual symptoms of the incident (if not provided by the submitter). If a solution or workaround is known (there is an entry for this type of problem in the Knowledge Database) 1^{st} Line Support solves the ticket. If, however, the incident type is not recorded in the database, the ticket can be passed to 2^{nd} or 3^{rd} Line Support for investigation and diagnosis.

Investigation and Diagnosis. At this step of Incident Management tickets generated by the monitoring system as well as those filed by users are handled in the same way. Upon categorization of the incident the proper support unit is informed. That unit has 24 hours to acknowledge the ticket. Subsequently, each support group investigates and diagnoses what has gone wrong and documents progress by filing an event record every three working days.

Resolution and Recovery. In this step potential solutions are sought and, if found, they are applied and tested.

Incident Closure. Tickets created by users are tagged as *solved* only after ensuring that the submitter is satisfied or has not verified the solution for over 2 weeks. Within the following seven days the ticket is examined as a potential extension to the Knowledge Database. In case of recurring incidents, a problem record may be stored in order to expedite future problem management activities.

4.7 Problem Management

Problem Management is an ITIL term describing a set of processes which include diagnosis and analyzing the cause for occurring incidents, preventing their future occurrence as well as managing a Known Error Database which makes it easier to identify new incidents and solve problems when they occur. The term *problem* is defined as a cause of numerous incidents which may occur together at the same time, or non-simultaneously, at various intervals. An example of a *problem* is the unavailability of a central service which results in unavailability of other services at a number of computing centers.

Problem Management focuses on investigating and resolving general problems which cause many specific incidents. Resolving such problems usually requires introducing changes either in software or hardware, which may require time.

Problems in the PL-Grid infrastructure are identified on the basis of user calls and reports from various teams. Potential problems (expected causes of frequent incident recurrence) and complex cases are noted and discussed during biweekly Operations Center meetings. In some cases special groups may be dispatched to inquire and solve problems. All problems are then analyzed and results of such analysis are recorded in the Known Error Database, which contains information about known errors and problems that these errors may indicate. The Known Error Database is a product of the Problem Management process and its contents may be used to support Incident Management and Problem Management processes.

The PL-Grid Known Error Database is composed of two parts. The first part can be called the Operations Problems Database. It contains records submitted by Regional Technical Support following incidents. These are only available to administrators and help them solve operational problems. The second part of the database is a Solutions Database, whose records are created by 1^{st} Line Support and are used by users in solving typical problems.

5 Future Work

The main challenge from the operational perspective is to prepare processes and procedures that will ensure universality and flexibility, independently of changing circumstances and future plans. A well designed process should be able to adapt to new services without the need for major procedural changes. At present, two middleware packages are integrated in EGI as a result of PL-Grid contribution. The first is UNICORE support while the second is our own middleware product called QosCosGrid [23]. Both show that our procedures work well in circumstances which call for flexibility and adaptability.

Another significant challenge for Service Operations is the issue of computational grants. Introducing this innovation requires many changes within the infrastructure and will be quite noticeable for end users. A computing grant is a new entity affecting the way in which local site policies are set up. However, they are essential for some user groups which require resource guarantees. From the Service Operations point of view this calls for new procedures and tools as well as adapting existing ones.

Another interesting challenge is extending the infrastructure towards cloud computing. A new layer has to be implemented in the service stack due to the virtualization aspects which need to be taken into account. Moreover, new features within the monitoring and accounting system have to be implemented to handle include hypervisors and virtual machines. The user access model would need to be redesigned in order to support dynamically instantiated machines. We believe that extending our approach to cloud computing requires changes in all aspects of Service Operations, from Access Management to Service Level Management, including computational grants and accounting.

6 Summary

In this paper we have described the model of Service Operations as applied in PL-Grid. We have shown how business experience encapsulated by the ITIL standard can be leveraged to respond to the needs of our users. We also claim that adapting this model to our distributed infrastructure (which provides grid computing services to the academic society) is necessary in order to ensure a higher level of service.

We have developed procedures and set up instructions to deal with various fields of Service Operations. Moreover, we have found ways to integrate our procedures with EGI, which was not always easy as the requirements we faced in PL-Grid regarding, for example, user verification, were different from those in EGI. In the field of Incident Management we have worked out procedures for functional and hierarchical escalation as well as ways of determining general incident causes (problems). We are also capable of accommodating changes in the infrastructure, including deployment of new software and procedural changes, as well as handling minor requests filed by users.

We have developed custom tools that enable us to manage all these processes: the PL-Grid User Portal manages user access, the PL-Grid Helpdesk handles incidents and user requests and is compatible with the EGI Helpdesk, while the Grid Resource Bazaar takes care of computational grants both in the sense of resource provisioning negotiations and grant activity period management.

Thanks to ITIL our level of service has also grown beyond *best effort* – we are now able to not only provide our users with services, but also to meet service level guarantees. By taking advantage of computational grants our users can now plan their research, including the use of our services, and receive guarantees that resources will be available to them as agreed upon. At the same time, grants help administrators plan their work: knowing user needs they can now predict how busy their resources will be and how much can be made available to other users.

In the future we plan to further develop the Service Operations model currently in place. We can already see that most of our procedures continue to operate well under changing circumstances. Regarding the PL-Grid project, we intend to extend the selection of available middleware and eventually provide support for elements of cloud computing.

References

1. Kitowski, J., Turała, M., Wiatr, K., Dutka, Ł.: PL-Grid: Foundations and Perspectives of National Computing Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 1–14. Springer, Heidelberg (2012)
2. EGI.eu foundation web page, http://egi.eu/about
3. Szepieniec, T., Tomanek, M., Radecki, M., Szopa, M., Bubak, M.: Implementation of Service Level Management in PL-Grid Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 171–181. Springer, Heidelberg (2012)
4. Microsoft Corporation. Microsoft Operations Framework 4.0 (April 2008) (last updated July 28, 2010)

5. van der Veen, A., van Bon, J.: Foundations of IT Service Management Based on ITIL V3. Van Haren Publishing (2007) ISBN: 9789087530570
6. Wan, S.H.C., Chan, Y.: IT Service Management for Campus Environment – Practical Concerns in Implementation. In: 2007 10th IFIP/IEEE International Symposium on Integrated Network Management, pp. 709–712. IEEE (2007)
7. Yang, Z.: Research and Design of Digital Campus Operation & Maintenance Center (O&M Center) Based on ITIL. In: 2010 International Conference on Computer Application and System Modeling, vol. 5, pp. 598–601. IEEE (2010)
8. Shaaf, T., Metzker, M.G.: gSLM: Challenges in Service Delivery Management and Service Level Management in Grids. In: Cracow Grid Workshop Proceedings, pp. 102–109 (2010)
9. Barafort, B., Di Renzo, B., Merlan, O.: Benefits Resulting from the Combined Use of ISO/IEC 15504 with the Information Technology Infrastructure Library (ITIL). In: Oivo, M., Komi-Sirviö, S. (eds.) PROFES 2002. LNCS, vol. 2559, pp. 314–325. Springer, Heidelberg (2002)
10. Hochstein, A., Zarnekow, R., Brenner, W.: Evaluation of Service-Oriented IT Management in Practice. In: 2005 International Conference on Services Systems and Services Management, vol. 1, pp. 80–84. IEEE (2005)
11. Fieseler, T., Gurich, W.: Operation of the Core D-Grid Infrastructure. In: 8th IEEE International Symposium on Cluster Computing and the Grid, pp. 162–168. IEEE Computer Society, Washington, DC (2008)
12. Coles, J.: The Evolving Grid Deployment and Operations Model within EGEE, LCG and GridPP. In: First International Conference on e-Science and Grid Computing, pp. 90–97. IEEE Computer Society, Washington, DC (2005)
13. Enabling Grids for E-Science, http://www.eu-egee.org
14. Szepieniec, T., Tomanek, M., Twaróg, T.: Grid Resource Bazaar: Efficient SLA Management. In: Cracow Grid Workshop Proceedings, pp. 314–319 (2009)
15. Information Processing Centre web page, http://www.opi.org.pl/en
16. Radecki, M., Szymocha, T., Harężlak, D., Pawlik, M., Andrzejewski, J., Ziajka, W., Szelc, M.: Integrating Various Grid Middlewares and User Services into a Single Platform. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 15–26. Springer, Heidelberg (2012)
17. European Policy Management Authority for Grid Authentication (EUGridPMA) (2011), http://www.eugridpma.org
18. Polish Grid Certification Authority, http://www.man.poznan.pl/plgrid-ca
19. PL-Grid Simple CA, http://plgrid-sca.wcss.wroc.pl
20. Nagios – The Industry Standard In IT Infrastructure Monitoring, http://www.nagios.org
21. Bała, P., Benedyczak, K., Strzelecki, M.: Monitoring of the UNICORE middleware. In: Streit, A., Romberg, M., Mallmann, D. (eds.) Proceedings of 6th UNICORE Summit 2010, Jülich, Germany. IAS Series, vol. 5 (2010)
22. EGI Operations Portal, http://operations-portal.egi.eu
23. Kurowski, K., de Back, W., Dubitzky, W., Gulyás, L., Kampis, G., Mamonski, M., Szemes, G., Swain, M.: Complex System Simulations with QosCosGrid. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part I. LNCS, vol. 5544, pp. 387–396. Springer, Heidelberg (2009), doi:10.1007/978-3-642-01970-8_38

New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case

Bartosz Bosak[1], Jacek Komasa[2], Piotr Kopta[1], Krzysztof Kurowski[1], Mariusz Mamoński[1], and Tomasz Piontek[1]

[1] Poznań Supercomputing and Networking Center, Poznań, Poland
{bbosak,pkopta,krzysztof.kurowski,mamonski,piontek}@man.poznan.pl
[2] Adam Mickiewicz University, Faculty of Chemistry,
Grunwaldzka 6 Street, 60-780 Poznań, Poland
komasa@man.poznan.pl

Abstract. In this chapter we present the new capabilities of QosCosGrid (QCG) middleware for advanced job and resource management in the grid environment. By connecting many computing clusters together, QosCosGrid offers easy-to-use mapping, execution and monitoring capabilities for a variety of complex computations, such as parameter sweep, workflows, MPI or hybrid MPI-OpenMP as well as multiscale simulations. Thanks to QosCosGrid, large-scale programming models written in Fortran, C, C++ or Java can be automatically distributed over a network of computing resources with guaranteed Quality of Service – for example guaranteed startup time of a job. Consequently, applications can be run at specified periods with reduced execution time and waiting times. This enables more complex problem instances to be addressed. In order to prove the usefulness of the new functionality of QosCosGrid a detailed description of the system along with a real use case scenario from the quantum chemistry science domain will be presented in this chapter.

Keywords: parallel computing, MPI, metascheduling, advance reservation, QoS, High Performance Computing, High Throughput Computing.

1 Introduction

End users interested in the PL-Grid infrastructure have frequently voiced their demand for efficient programming and execution tools to run large parallel simulations and experiments requiring a certain Quality of Service. Highly parallel and coupled applications with significant inter-process communication that are not supported by existing grid infrastructures based on gLite [14] or UNICORE [19], represent a growing and promising class of simulations. To meet end-user needs, a new middleware infrastructure called QosCosGrid (or QCG for short) was designed, developed and deployed in the PL-Grid project [11]. QCG successfully integrates many new services and tools in order to deliver to PL-Grid users a new multilayered e-Infrastructure capable of dealing with computationally intensive simulations, including parameter sweep studies, workflows and, more importantly, large-scale parallel applications. QCG enables computing clusters in different administrative domains to be integrated into a single powerful virtual computing resource that can be treated as a quasi-opportunistic supercomputer, whose computational power exceeds the power offered by a single administrative domain (data center). In order to bring this supercomputer-like performance and structure (requested and expected by scientists) to bear on cross-cluster computations in an user-friendly way, various well-known application tools and services, including the OpenMPI and ProActive programming and execution environments, have been tuned to work in a multi-cluster QCG environment [1,11]. The cross-cluster scheduling enabled by QCG supports not only parallel simulations on many clusters (creating new possibilities and offering improvements for end users), but also – and more importantly from the resource owners' point of view – utilizing PL-Grid computing resources in a more efficient way, thus increasing the overall system throughput. The decomposition of large-scale tasks between many clusters decreases cluster "defragmentation" and results in better resource utilization.

The rest of the chapter is organized as follows. In Section 2 we present the key features of QCG. Section 2.1 describes QCG support for creating and running parallel applications. Section 2.2 describes how QCG enables execution of workflows. Section 2.3 presents QCG functions for Advance Reservation and Co-allocation. The following section, 3, describes the QCG architecture from a general point of view and introduces the main QCG components. A real usage scenario, based on a legacy application and exploiting the capabilities and features of QCG is discussed in Section 4. Finally, in Section 6, we present a short summary and list the ongoing and future work in the scope of QCG.

2 QCG Capabilities

2.1 QCG for Parallel Applications

To support large-scale applications in multi-cluster environments many novel mechanisms have been implemented. As presented in [9], QCG services are able to schedule and execute parallel applications consisting of groups of processes with different and often mutually contradictory resource requirements. For example, functional decomposition of the application and its implementation can result in a situation in which some processes should be run on a vector machine (for performance reasons) while others reside on a regular computing cluster. By defining groups of parallel processes it becomes possible to determine different resource requirements for each group and specify whether a given group can be split between resources or whether it should be allocated to a single resource. To avoid performance and communication bottlenecks caused by the fact that local connections have lower latency and higher bandwidth than long-distance ones (by two to four orders of magnitude), the QCG resource manager can either schedule the application in a topology-aware manner to meet its requirements or expose the physical topology to the application which then dynamically adapts itself to that topology. Such topology-awareness implies that, while matching the resource offers with requests, the scheduler has to take into account not only the computational properties of the resources, but also their interconnections. All these scenarios, except for the one involving self-adaptation of the application to match the available topology, do not require any changes in application code and are fully implementable owing to tight integration of QCG services with adapted OpenMPI and ProActive frameworks [11]. Running large-scale simulations, both sequential and parallel, in a multi-cluster environment requires not only launching and controlling processes on the available resources, but also some means of enabling inter-process communication between parts of a parallel application. Parallel processes running on different computing clusters must be able to communicate with one another without running afoul of the security mechanisms protecting each cluster (such as firewalls blocking connections and NATs reducing the number of public IPs required by the cluster). To address this primary requirement an open-source implementation of the MPI standard – OpenMPI – as well as the Java ProActive library have been extended with several basic and advanced connectivity techniques intended to bypass firewalls and NATs, and integrated with QCG services [1].

2.2 QCG for Workflow Applications

As has already been mentioned, QCG supports not only large-scale parallel applications but also other kinds of popular computational experiments. QCG is able to deal with complex applications defined as a set of tasks with precedence relationships (workflows). The workflow model is based on direct acyclic graphs (DAG). In this approach the end user has to specify (in advance) task precedence constraints in the form of task state relationships. A very interesting and novel feature of QCG, which distinguishes it from other middleware services supporting workflows, is that – in addition to being associated with input or output files – every task can be triggered by any combination of other tasks or conditional rules. This feature is very useful in many scenarios. For instance, one can imagine that a user would like to execute an application as soon as another one starts running, e.g. for client-server communication. Another example could involve redirecting the flow of computations in the event of a failure of one of the scheduled tasks (failover mechanisms). QCG also supports popular parameter sweep experiments and supports many instances of a single application, each with a different set of arguments. For every task in the collection, the value of one or more of the task parameters may be changed in some preordained fashion, creating a parameters space. This is also a very useful feature, providing the end user with an easy way to browse the parameters space in search for a specific set of parameters that meet the defined criterion. Parameter sweep tasks can be a part of a larger experiment, e.g. a workflow, and all parent-child dependencies are automatically converted by the system to take the whole collection of generated tasks into consideration. What distinguishes QCG from other middleware packages dealing with parameter sweep tasks, is its support for multi-dimensional parameter spaces, in which many variables can be regulated to construct the aforementioned space of parameters.

2.3 Advance Reservation and Co-allocation in QCG

The next feature distinguishing QCG from other e-Infrastructures offering access to PL-Grid resources like gLite and UNICORE is support for scenarios which call for a specific level of quality of service. As the only such infrastructure, QCG supports advance reservation of computational resources to guarantee the requested execution parameters. Advance reservation is used internally by QCG services in the case of cross-cluster execution but is also provided directly to end users. The reservation mechanism is applied in the scheduling process to co-allocate resources and then to synchronize execution of application parts in a multi-cluster environment. Cross-cluster scheduling and co-allocation of resources are tightly connected with support for groups of processes and communication topologies. When co-allocating resources and assigning tasks to specific resources, QCG may also consider user requirements regarding task execution time. Upon submitting a task to the system the user may specify resource requirements as well as the requested quality of service, including task duration and – optionally – the period when the task should be executed. QCG supports both the strict and best-effort approaches to resource reservation. In the former approach resources are reserved only if it is possible to fully meet user requirements (also known as the "all or nothing" approach), whereas in the the latter case the system reserves as much resources as possible and there is no guarantee that all requested resources (cores) will be reserved [13].

Recently, the QCG infrastructure has been extended to support new types of parallel applications based on the MPI/OpenMP hybrid programming approach. The user may request the application to execute on a given number of computational nodes with a predefined number of slots (cores) on each. In this solution, for each node participating in the computations, the user can specify the number of MPI processes that should be started.

The functional comparison of QCG middleware with gLite and UNICORE is presented in Table 1.

Table 1. Functional comparison of three grid middleware solutions: QCG, gLite and UNICORE.

Middleware	Single jobs	Workflows	MPI jobs	Cross-cluster MPI jobs	Interactive jobs	Parametric jobs
gLite	Yes	Yes	Yes	No	Yes	Yes
UNICORE	Yes	Yes	Yes	No	No	Yes
QCG	Yes	Yes	Yes	Yes	No	Yes

3 QosCosGrid Architecture

Though QCG middleware generally follows a multi-layered design approach, two main levels can be distinguished: grid domain and administrative domain. Services belonging to the grid domain provide a high-level interface to the grid or cloud environments (e.g. GridSpace [7], Nano-Science Gateway [8]) and control and schedule the execution of applications which are distributed over independent administrative domains. In turn, the administrative domain represents a single resource provider (e.g. HPC or data center) which contributes computational resources (e.g. clusters) to a particular grid or cloud infrastructure. Note that logical separation of administrative domains is intentional and corresponds to the fact that resources (and also users) may come from different institutions or resource owners. It is fully natural that each institution may need to preserve a certain level of independence and enforce its own resource allocation/sharing policies.

The general architecture of QCG is presented in Fig. 1. The critical service on the grid level is QCG-Broker; a meta-scheduling framework controlling execution of applications via services located in the administrative domains. On this level the QCG-Computing component (tightly connected with QCG-Broker) provides remote access to underlying queuing systems. Among others, QCG-Computing supports execution of jobs in several parallel execution environments, namely OpenMPI, ProActive and MUSCLE. Additionally, it exposes an interface for creation and management of advance reservations. Another relevant service in

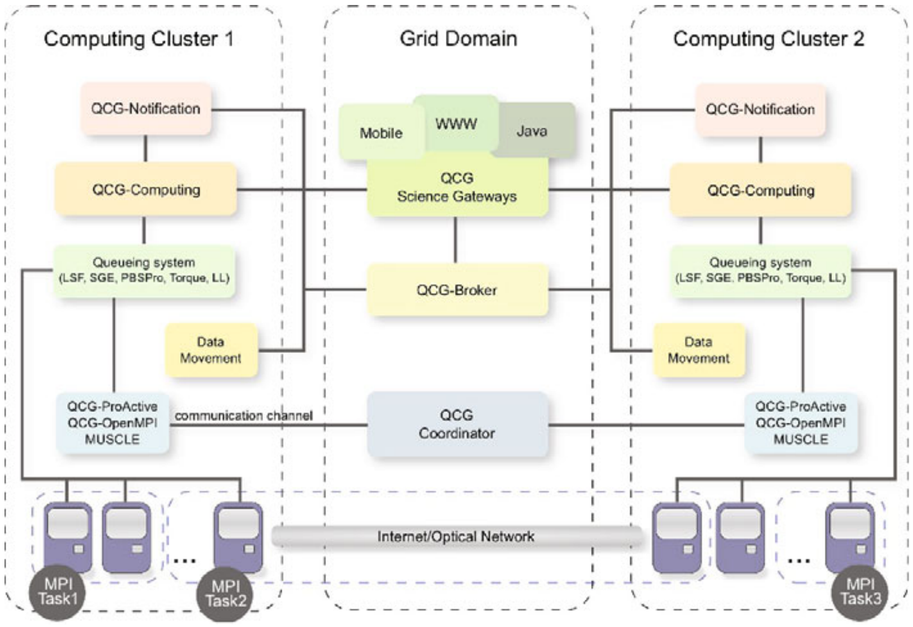


Fig. 1. General QCG middleware architecture

the administrative domain is called QCG-Notification: its task is to implement a notification mechanism. The QCG middleware structure is complemented by coordinator services controlling cross-cluster execution, as well as data movement services managing input and output data for applications.

3.1 QCG-Broker

The QCG-Broker[1] is an open-source metascheduling system which allows developers to build and deploy resource management systems for large-scale distributed computing infrastructures. Based on dynamic resource selection, mapping and advanced scheduling methodology combined with feedback control mechanisms, QCG-Broker deals efficiently with various metascheduling challenges, e.g. co-allocation, load balancing among clusters, remote job control, file staging support and job migration, as has been demonstrated in [10]. The main goal of QCG-Broker is to manage the whole process of remote job submission to administrative domains controlled by domain-level QCG components, and then to underlying clusters and computational resources. It has been designed as an independent core component for resource management processes which can take full advantage of various low-level core and grid services and existing technologies, such as QCG-Computing and QCG-Notification or GridFTP, as well as various grid middleware components, e.g. Grid Authorization Service, Data Management Service and others. All these services work together to provide a consistent, adaptive and robust grid middleware layer which fits dynamically to many different distributed computing infrastructures, enabling large scale simulations and providing the requested Quality of Service. One of the main assumptions for QCG-Broker is to perform remote jobs control and management in a way which satisfies users (Job Owners) and their applications while respecting the constraints and policies imposed by other stakeholders, i.e. resource owners and grid or Virtual Organization administrators. Simultaneously, Resource Administrators (Resource Owners) have full control over resources on which all jobs and operations are performed, through appropriate setup and installation of QCG components. Note that QCG-Broker, together with administrative-domain level QCG components, reduces the operational and integration costs for Administrators by enabling grid deployment across previously incompatible cluster and resources. The heart of QCG-Broker is its metascheduling framework, responsible for scheduling tasks in the controlled environment. QCG-Broker has been successfully integrated with the scheduling framework designed, implemented and used in the Grid Scheduling SIMulator [12], enabling grid administrators to modify scheduling policies in an easy and flexible way, using different scheduling plugins. All experiments controlled by QCG-Broker (including workflows, large-scale parallel applications with groups of processes and topology requirements, parameter sweep tasks and simple jobs) can be easily expressed in a formal way using the XML-based job definition language called Job Profile.

[1] `http://www.qoscosgrid.org/trac/qcg-broker`

3.2 QCG-Computing

The key component in the QCG administrative domain is the QCG-Computing service. Technically, QCG-Computing[2] is an open implementation of SOAP Web services for multiuser access and policy-based job control routines by various queuing and batch systems managing local computational resources. To communicate with underlying queuing systems, the service uses the Distributed Resource Management Application API (DRMAA) [20]. It has been successfully tested with many products and supports a variety of well-known queuing systems, including:

- Grid Engine,
- Platform LSF,
- Torque/Maui,
- PBS Pro,
- Condor,
- Apple XGrid,
- SLURM,
- LoadLeveler.

The QCG-Computing service is compliant with the OGF HPC Basic Profile specification [22], which serves as a profile over other Open Grid Forum standards like JSDL and OGSA Basic Execution Service. Moreover, it offers innovative remote interfaces for advance reservation management and supports basic file transfer mechanisms.

QCG-Computing has been designed to support a variety of plugins and modules for external communication as well as to handle a large number of concurrent requests from external clients and services. Consequently, it can be used and integrated with various authentication, authorization and accounting services. An example of integration with other PL-Grid services is described in Section 3.5.

3.3 QCG-Notification

QCG-Notification[3] is another service belonging to the administrative domain. Its main function in QCG is brokering asynchronous notifications concerning job state changes between the QCG-Computing and QCG-Broker services. Nevertheless, depending on demands, QCG-Notification may be variously configured and adapted to specific requirements. In general, QCG-Notification is based on the OASIS standards for Web Service notifications – WS-BaseNotification, WS-BrokeredNotification and WS-Topics [23], and provides an adjustable and efficient interface for message exchange between interested parties. Thanks to QCG-Notification, components which produce notifications may be logically separated from components interested in receiving those notifications, as presented in Fig. 2. Therefore, since some features required for communication between

[2] http://www.qoscosgrid.org/trac/qcg-computing
[3] http://www.qoscosgrid.org/trac/qcg-notification

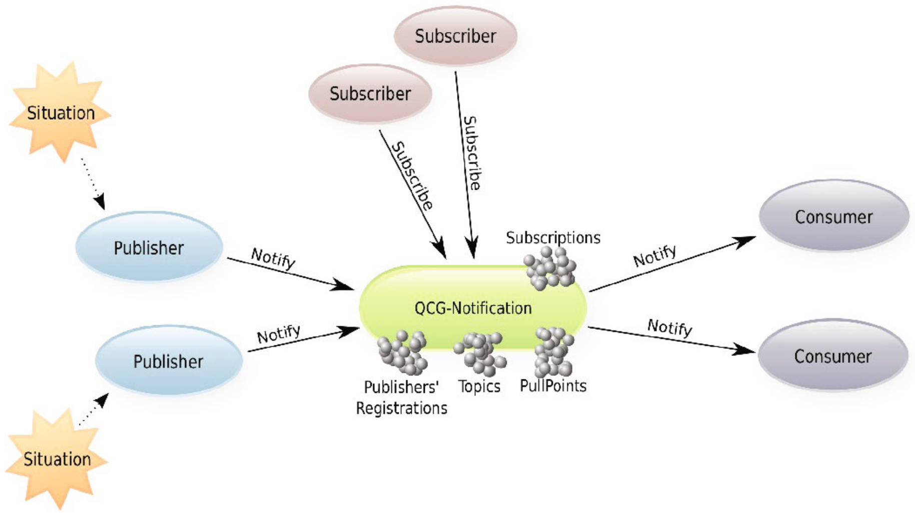


Fig. 2. Brokered notification scenario supported by QCG-Notification

specific components are delegated to an external entity, the overall performance of the system may significantly increase. Such a situation is also common in QCG. QCG-Notification implements many patterns defined in the WS-Notification specification and provides some extensions to the standard. Performed tests and comparisons with other well-known WS-Notification implementations have shown that QCG-Notification offers a rich feature set while remaining highly efficient. QCG-Notification is characterized by the following list of features:

– support for HTTP/HTTPS and XMPP transport protocols,
– subscription and publishers' registration handling,
– advanced two-level notification filtering based on hierarchical topic namespaces and notification message contents (XPath-based filters),
– pull and push styles of distributing notification messages with fault tolerance mechanisms,
– good performance owing to carefully selected data structures and internal algorithms,
– plenty of configuration and customization options, available through a bundled management interface,
– extensible architecture (pluggable modules for transport, authentication and authorization protocols).

3.4 Cross-Cluster Communication

QCG support for parallel cross-cluster execution consists of the three environments, namely QCG-OMPI [1], QCG-ProActive and MUSCLE, each targeting a different groups of application developers. The first environment, QCG-OMPI[4],

[4] http://www.qoscosgrid.org/trac/qcg-openmpi

is based on OpenMPI and preserves all standard properties of this library. Therefore it aims at C/C++ and FORTRAN code. QCG-ProActive, in turn, is an extended version of the ProActive Java library which may be easily used in new or existing parallel Java applications. The final environment, MUSCLE (still under development in the MAPPER [26] project), simplifies the development of multiscale computation scenarios.

Since the standard deployment methodologies used in OpenMPI or ProActive are limited to single-cluster runs, in order to support cross-cluster execution and spawning of parallel application processes on co-allocated computational resources, QCG (besides minor extensions to the OpenMPI and ProActive libraries) provides special services called coordinators. Coordinators are implemented as Web Services and should be accessible from all participating administrative domains. Depending on particular scenarios, coordinators may be configured in various ways. In general, two situations which influence deployment of QCG middleware can be distinguished:

1. All computing clusters have public IP addresses,
2. At least one computing cluster has private IP addresses.

In the former case, port range techniques can be applied to enable communication between processes executing in separate clusters. It is a simple approach founded upon the idea of using a predefined range of unprivileged ports. If, however, some clusters use private set(s) of IP addresses, a different solution is necessary. We have decided to take advantage of proxy mechanisms, where SOCKS services are deployed on frontend machines to route incoming traffic to the MPI and ProActive processes running inside clusters with private IP addresses.

3.5 Integration with PL-Grid Infrastructure

Almost every e-Infrastructure enforces its own authentication, authorization and accounting (called AAA for short) through a set of custom policies. Those polices are usually governed by a separate unit called the Operations Center. Therefore, any new middleware stack wishing to become a part of such an infrastructure must integrate itself with the existing AAA ecosystem.

The PL-Grid infrastructure offers two authentication mechanisms: password-based and X.509 certificate-based. The former is usually used while logging into the PL-Grid Portal, gLite UI machines and batch job submission hosts. The latter is required while contacting grid services. In PL-Grid every QCG-Computing instance is configured to accept RFC3820 [21] compliant proxy certificates.

The QCG-Computing services in PL-Grid are configured to use the plain grid-mapfile. This means that, much like UNICORE and contrary to gLite, QCG uses static accounts. The grid-mapfile is generated automatically based on information available in a local LDAP (Lightweight Directory Access Protocol) replica. Hence, each PL-Grid user who applies for QCG services and is cleared by local administrators, may be automatically added to this file. Moreover, system administrators are able to define their own lists of locally denied/accepted users.

In the PL-Grid project a completely new system called BAT[5], used for collecting accounting information, has been developed. The system consists of one central service (called BAT broker), which gathers resource usage records produced by clients (called BAT agents) deployed in every organizational unit. There exist two classes of BAT agents: local and grid. The former rely on information available in the batch system's (Torque and PBS Professional in PL-Grid) log files such as the job's wall-clock time, local id, etc., while the latter augment such information with high-level data – e.g. the user's certificate, distinguished name or grid job id. In PL-Grid a new BAT agent has been developed for QCG-Computing, alongside gLite and UNICORE agents. This agent periodically reads job data from the QCG-Computing accounting database and sends it over a secure channel to the BAT broker.

The QCG-Computing service also acts as a lightweight information service, providing (via a Web Service interface) information about PL-Grid users, groups and grants in a particular organizational unit. The consumer of this information is the QCG-Broker service, which later exploits it for scheduling purposes.

In addition, as every production infrastructure has to be monitored constantly, a set of Nagios[6] probes was provided for the QCG-Computing, QCG-Notification and QCG-Broker services.

The final requirement of the Operations Center was to provide binary RPM (Red Hat Package Manager) packages compatible with the Scientific Linux operating system.

4 Quantum Chemistry Application Using QCG

4.1 Motivation

This section presents the main assumptions and challenges for computationally demanding simulations for quantum chemistry, in particular an actual scientific application called NEL, representing numerical algorithms related to large-scale quantum-chemical calculations. The legacy quantum chemistry application discussed in [4], originally implemented in Fortran, has been redesigned and optimized to take full advantage of QCG. The optimization was performed in cooperation with the author of the original version, as part of the user support activity in the PL-Grid project.

The electronic structure of atoms and molecules is determined on the basis of the Schrödinger equation. However, in the case of many-electron systems this equation is not solvable and approximation schemes have to be employed. Moreover, the solving is often performed numerically with the use of advanced computations. Orbital methods, which posit that each electron moves in the average field of the other electrons, are quite ubiquitous but, in spite of their being generally successful, they are not applicable to a wide variety of quantum-chemical problems in which high precision of physical outcome is expected. Numerous

[5] `https://gforge.cyfronet.pl/projects/bat-plgrid/`
[6] `http://www.nagios.org/`

variational methods of solving the Schrödinger equation with high precision have been developed – for instance basing on the explicitly correlated Gaussian (ECG) functions, exploiting the growing computational power of state-of-the-art supercomputers and clusters. This is mostly associated with the ECG method's potential that might be used in the case of atoms with more than three electrons and many-electron, multicenter molecules [2,5].

There have also been variational attempts to solve the Schrödinger equation

$$H\Psi = E\Psi\,, \tag{1}$$

in which Ψ constitutes the wave function reflecting a particular state of a molecule or an atom and H is the clamped nuclei Hamiltonian, describing all the Coulomb interactions between electrons and nuclei as well as the electrons' kinetic energies. E stands for the electronic energy of the system. Both the energy and wave function are sought when trying to solve this equation for a given Hamiltonian. The following equation reflects the so-called trial wave function. This function will be represented as a K-term linear combination of N-electron ECG basis functions ϕ_i

$$\Psi = \sum_{i=1}^{K} c_i\,\phi_i\,, \tag{2}$$

in which

$$\phi_i = \mathcal{A}_N \left\{ \mathcal{P}\left[\exp\left(-\sum_{p=1}^{N-1} \sum_{q=p+1}^{N} A_{ipq}\,(\boldsymbol{r}_p - \boldsymbol{s}_{ip})\,(\boldsymbol{r}_q - \boldsymbol{s}_{iq}) \right) \right] \Theta \right\}. \tag{3}$$

Within this formula, spatial electronic coordinates are marked as $\boldsymbol{r}_k$, antisymmetrizer (working on space and spin coordinates) as $\mathcal{A}_N$, the spatial symmetry projector as $\mathcal{P}$, and Θ is the N-electron spin function.

Identical particles are indistinguishable from the point of view of their physical properties and the antisymmetry projector $\mathcal{A}_N$ is responsible for taking this feature into account. When electron coordinates are considered and the calculations performed on them, it returns a sum of $N!$ terms differing by electron coordinate permutations. All the elements of the Hamiltonian matrix are affected by the $N!$ explosion, which is one of the factors limiting the size of examined systems to a few electrons. The total number of nonlinear parameters, collected in $\boldsymbol{A}_i$ and $\boldsymbol{s}_i$ which are variables of the optimization process, depends on the size of atoms or molecules (assessed by the number of electrons N and nuclei) and the expansion (2). In the most advanced/complex cases, there are over 100 000 non-linear parameters which have impact on the wave function (and energy). In such cases determining the energy minimum becomes a very computationally demanding task. A solution of the Schrödinger equation presented in the matrix form of the general symmetric eigenvalue problem (GSEP) offers the optimal vector of the linear parameters c_i and the corresponding approximation ϵ for the exact electronic energy

$$\mathcal{H}\,\boldsymbol{c} = \epsilon\,\mathcal{S}\,\boldsymbol{c}\,. \tag{4}$$

The Hamiltonian $\mathcal{H}$ and the overlap $\mathcal{S}$ matrices are composed of elements defined using $4N$-dimensional integrals over all coordinates $dV_1 \dots dV_N$ of the electrons [3,18,5]

$$\mathcal{H}_{ij} = \int \phi_i H \phi_j \, dV_1 \dots dV_N \,, \tag{5}$$

$$\mathcal{S}_{ij} = \int \phi_i \phi_j \, dV_1 \dots dV_N \,. \tag{6}$$

Although the proposed method is general and can theoretically be applied to any N-electron atomic or molecular system, these calculations are commonly employed for systems containing no more than four electrons in order to preserve the accuracy level [5,6,17]. An extension of the applicability of the ECG method to larger systems is the challenge we face. QCG opens up yet another possibility of reaching this goal.

The ECG wave functions within the above mentioned conditions are accurate; however, any additional electron appearing in the system would require reassessment of the algorithms. When the number of electrons in a system is increased to five, accurate calculations call for $\sim 10^3 - 10^4$ ϕ_i functions. Moreover, the energy has to be minimized by the variational parameters within a multidimensional space. Arriving at a near-exact estimate of energy E would imply evaluations of the energy ϵ conducted $10^6 - 10^7$ times. On the grounds of the aforementioned theory, computations within a hybrid parallel computing model which involve the Message Passing Interface (MPI) are carried out so as to facilitate communication of processes over the network, along with a shared memory model (OpenMP) on local nodes.

4.2 Application Requirements

The scalar part of the program is responsible for the matrix elements of Hamiltonian $\mathcal{H}$, Eq. (5), and the overlap matrix $\mathcal{S}$, Eq. (6), whereas the vector part is composed of GSEP solutions, Eq. (4), and they are both deemed the most demanding parts with respect to computations. Cholesky decomposition of the matrix $\mathcal{H} - \varepsilon_t\,\mathcal{S}$ with a trial value ε_t (close to the desired energy ϵ) lays the foundations for the aforementioned demanding parts of the algorithm. As the next step on the way to optimizing the energy ε_t, an inverse iteration procedure is implemented [16]. In this computation a triangular system of equations is solved in each iteration and, consequently, the energy converges to ϵ following a few iterations. Optimization of the very large number of non-linear variational parameters $\boldsymbol{A}_i$ and $\boldsymbol{s}_i$ is yet another vital part of the algorithm. It is recommended to apply the iterative optimization procedure from $i = 1$ to $i = K$ tuning (in the i-th step) the parameters of a single-basis function ϕ_i. Therefore, during the optimization process and within the Powell's conjugate directions method [15], variational parameters reflecting the objective energy function undergo changes and if these changes do not improve the energy, the algorithm is stopped.

The optimization shot is another important concept which should be mentioned here. It involves a single execution of the above mentioned parts that is scalar- and vector-based, returning a single value of the energy ϵ. The number of shots required to achieve convergence approaches 100 times the number of nonlinear parameters. For instance, in the case of a four-electron wave function with 2400 terms and 14 nonlinear parameters per term (LiH molecule), these tasks have to be performed more than 10^6 times:

– scalar part – evaluation of the matrix elements $\mathcal{H}_{ij}$, $\mathcal{S}_{ij}$, with thousands of floating-point operations per each element,
– vector part – solution of the GSEP for matrices with $K = 2400$ rows and columns.

The whole application works as follows. First, the multithreaded master MPI process reads all input parameters. Subsequently, it forwards tasks to other MPI processes. Once data is received, an MPI process calculates $\mathcal{H}$ and $\mathcal{S}$ elements. Different matrix parts are computed by different processes. Having completed all the tasks, the initial energy is calculated by the master process with the use of the GSEP algorithm. The GotoBLAS2 library is employed in this part of the application.

The nonlinear optimization process for subsequent rows of the $\mathcal{H}$ and $\mathcal{S}$ can commence once the initial energy is known. Dozens of optimization shots are carried out, on average, for a single basis function. These shots are computations of the elements of a single row in the GSEP algorithm. All worker processes are responsible for calculating row elements while the master process manages the GSEP algorithm with the use of parallel threads.

A recent version of the presented application was successfully and efficiently executed in a multi-cluster environment managed by QCG services using geographically distributed resources in Poznań and Kraków. In both cases resources used in this experiment belonged to the production PL-Grid infrastructure and the main motivation to use co-allocated resources was to shorten the time which the task (requiring hundreds of computational cores) spends in the queue until it obtains the requested resources. We have observed that, especially in the case of the application implementing a master/slave paradigm with no intensive communication between processes, the performance overhead stemming from network delays incurred by the geographical distribution of participating computations is more than compensated for by the shorter queuing time. The requested number of cores (256) was co-allocated by the QCG-Broker on Reef and Zeus clusters respectively in Poznań and Kraków and parts of the application were started in a synchronous way by local QCG-Computing services. The main computing and communications steps in the hybrid parallel NEL application, with its decomposition between two clusters, are shown in Fig. 3.

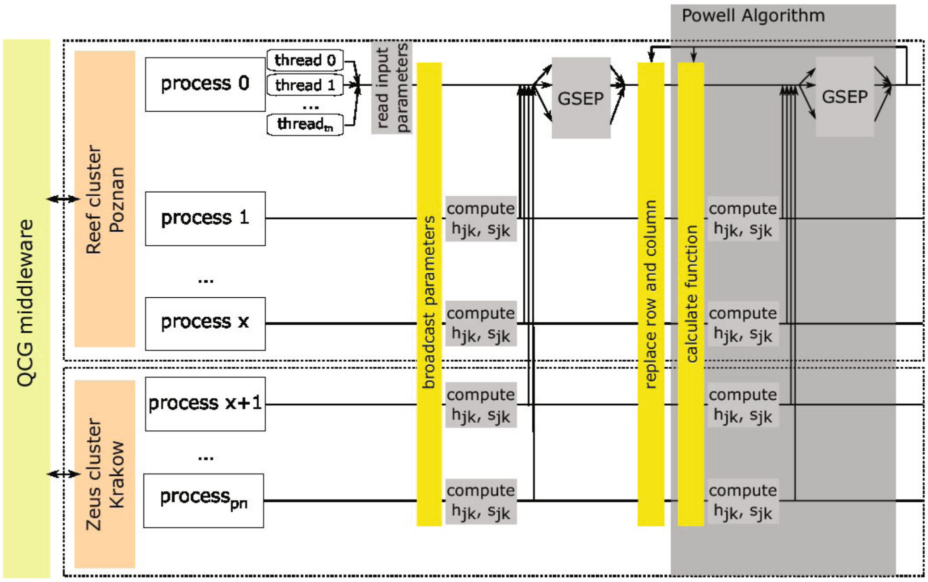


Fig. 3. Main computing and communications steps of the NEL application with its decomposition between two clusters

5 Deployment Status

The QCG middleware was successfully deployed in several production HPC environments belonging to the PL-Grid infrastructure (e.g. Poznań Supercomputing and Networking Center – PSNC, Academic Computer Centre Cyfronet AGH, TASK). Moreover, there are ongoing deployment activities at the Leibniz-Rechenzentrum – LRZ, University College London – UCL, National Institute for Research in Computer Science and Control – INRIA, and the Dortmund University.

6 Future Work

National and international grid e-Infrastructures provide resources comparable to the largest existing large-scale parallel computing environments. However, current grids typically do not address sophisticated scenarios which require specific Quality of Service guarantees to support simultaneous management of many kinds of resources, storage and networks. Existing grid middleware solutions, including the popular gLite [14] and UNICORE [19] systems, do not satisfy all demands of modern scientific simulations and computing models. One of the shortcomings of these systems is poor support for advance reservation, which makes it difficult (or indeed impossible) to run jobs on co-allocated resources. Basing on the outcome of earlier European projects (e.g. GridLab [25], BREIN

[24], QosCosGrid [28]) we propose QCG as an alternative grid middleware platform. QCG services support the latest open standards, including OGF HPC Basic Profile, JSDL, OGSA BES and WS-Notification, thereby providing a flexible, interoperable interface upon which to run, execute and monitor complex jobs as well as create advance reservations and co-allocations. To the best of our knowledge, QCG currently provides the most efficient and powerful multiuser access to job management and co-scheduling features, compared to other existing grid middleware services.

In order to meet the emerging end-user requirements, QCG will be integrated with various new services and application tools for distributed multiscale computing in the scope of the MAPPER project [26]. QCG middleware will also be evaluated by partners involved in other National Grid Infrastructures and PRACE partners [27].

References

1. Agullo, E., Coti, C., Herault, T., Langou, J., Peyronnet, S., Rezmerita, A., Cappello, F., Dongarra, J.: QCG-OMPI: MPI Applications on Grids. Future Gener. Comput. Syst. 27, 357–369 (2011)
2. Bachorz, R., Komasa, J.: Variational calculations on H2+ using exponentially correlated Gaussian wave functions. Computational Methods in Science and Technology 11(1), 5–9 (2005)
3. Boys, S.F.: The Integral Formulae for the Variational Solution of the Molecular Many-Electron Wave Equations in Terms of Gaussian Functions with Direct Electronic Correlation. Royal Society of London Proceedings Series A 258, 402–411 (1960)
4. Cencek, W., Komasa, J., Rychlewski, J.: High-performance Computing in Molecular Sciences. In: Handbook on Parallel and Distributed Processing, p. 205. Springer, Heidelberg (2000)
5. Cencek, W., Rychlewski, J.: Many-electron Explicitly Correlated Gaussian Functions. I. General Theory and Test Results 98(2), 1252–1261 (1993)
6. Cencek, W., Szalewicz, K.: Ultra-high Accuracy Calculations for Hydrogen Molecule and Helium Dimer. International Journal of Quantum Chemistry 108, 2191–2198 (2008)
7. Ciepiela, E., Nowakowski, P., Kocot, J., Harężlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing Entire Lifecycles of e-Science Applications in GridSpace2 Virtual Laboratory – from Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)
8. Dziubecki, P., Grabowski, P., Krysiński, M., Kuczyński, T., Kurowski, K., Piontek, T., Szejnfeld, D.: Online Web-Based Science Gateway for Nanotechnology Research. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 205–216. Springer, Heidelberg (2012)
9. Kravtsov, V., Bar, P., Carmeli, D., Schuster, A., Swain, M.: A scheduling framework for large-scale, parallel, and topology-aware applications. Journal of Parallel and Distributed Computing 70(9), 983–992 (2010)
10. Kurowski, K., Ludwiczak, B., Nabrzyski, J., Oleksiak, A., Pukacki, J.: Dynamic Grid Scheduling with Job Migration and Rescheduling in the GridLab Resource Management System. Sci. Program. 12, 263–273 (2004)

11. Kurowski, K., de Back, W., Dubitzky, W., Gulyás, L., Kampis, G., Mamonski, M., Szemes, G., Swain, M.: Complex System Simulations with QosCosGrid. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part I. LNCS, vol. 5544, pp. 387–396. Springer, Heidelberg (2009)
12. Kurowski, K., Nabrzyski, J., Oleksiak, A., Weglarz, J.: Grid Scheduling Simulations with GSSIM. In: Proceedings of the 13th International Conference on Parallel and Distributed Systems, vol. 02, pp. 1–8. IEEE Computer Society, Washington, DC, USA (2007)
13. Kurowski, K., Oleksiak, A., Weglarz, J.: Multicriteria, Multi-user Scheduling in Grids with Advance Reservation. J. of Scheduling 13, 493–508 (2010)
14. Laure, E., Grandi, C., Fisher, S., Frohner, A., Kunszt, P., Krenek, A., Mulmo, O., Pacini, F., Prelz, F., White, J., Barroso, M., Buncic, P., Byrom, R., Cornwall, L., Craig, M., Di Meglio, A., Djaoui, A., Giacomini, F., Hahkala, J., Hemmer, F., Hicks, S., Edlund, A., Maraschini, A., Middleton, R., Sgaravatto, M., Steenbakkers, M., Walk, J., Wilson, A.: Programming the Grid with gLite. In: Computational Methods in Science and Technology (2006)
15. Powell, M.J.D.: An Efficient Method for Finding the Minimum of a Function of Several Variables Without Calculating Derivatives. The Computer Journal 7(2), 155–162 (1964)
16. Press, W.H., Flannery, B.P., Teukolsky, S.A., Vetterling, W.T.: Numerical Recipes in Fortran 77: The Art of Scientific Computing, 2nd edn. Cambridge University Press (September 1992)
17. Przybytek, M., Cencek, W., Komasa, J., Łach, G., Jeziorski, B., Szalewicz, K.: Relativistic and Quantum Electrodynamics Effects in the Helium Pair Potential. Phys. Rev. Lett. 104(18), 183003 (2010)
18. Singer, K.: The Use of Gaussian (Exponential Quadratic) Wave Functions in Molecular Problems. I. General Formulae for the Evaluation of Integrals. Royal Society of London Proceedings Series A 258, 412–420 (1960)
19. Streit, A., Erwin, D., Mallmann, D., Menday, R., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Wieder, P.: UNICORE – From Project Results to Production Grids. In: Grid Computing and New Frontiers of High Performance Processing (2005)
20. Troger, P., Rajic, H., Haas, A., Domagalski, P.: Standardization of an API for Distributed Resource Management Systems. In: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid, CCGRID 2007, pp. 619–626. IEEE Computer Society, Washington, DC, USA (2007)
21. Tuecke, S., Welch, V., Engert, D., Pearlman, L., Thompson, M.: Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. RFC 3820 (Proposed Standard) (June 2004)
22. HPC Basic Profile Version 1.0, http://www.ogf.org/documents/GFD.114.pdf
23. OASIS Web Services Notification, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn
24. BREIN Project, http://www.eu-brein.com/
25. GridLab Project, http://www.gridlab.org
26. MAPPER – Multiscale Applications on European e-Infrastructures, http://www.mapper-project.eu/
27. PRACE – The Partnership for Advanced Computing in Europe, http://www.prace-project.eu/
28. QosCosGrid Project, http://biomed.science.ulster.ac.uk/archive/www.qoscosgrid.eu/

Seamless Access to the PL-Grid e-Infrastructure Using UNICORE Middleware

Krzysztof Benedyczak[1,2], Marcin Stolarek[1], Radosław Rowicki[1], Rafał Kluszczyński[1], Marcelina Borcz[1,2], Grzegorz Marczak[1,2], Maciej Filocha[1], and Piotr Bała[1,2]

[1] University of Warsaw, Interdisciplinary Centre for Mathematical and Computational Modelling, Pawińskiego 5a, 02-106 Warsaw, Poland
[2] Nicolaus Copernicus University, Department of Mathematics and Computer Science, Chopina 12/18, 87-100 Toruń, Poland
`bala@icm.edu.pl`

Abstract. This chapter provides a brief overview of the UNICORE grid middleware and its utilization in the large Distributed Computing Infrastructure. UNICORE framework, in its recent version implements key grid standards and specifications. The system architecture and capabilities, such as security, workflow and data management are described. The installation of the UNICORE environment in the PL-Grid is presented. Special attention is given to the integration of the UNICORE middleware with the PL-Grid authentication and authorization framework which allows for uniform infrastructure and user management across different middlewares. The solutions for monitoring and accounting of the UNICORE infrastructure is presented.

Keywords: UNICORE, Grid, PL-Grid.

1 Introduction, Motivation, Problem Statement

The main aim of the PL-Grid project is to build a sustainable grid infrastructure for the scientific community in Poland [1]. From its beginning the project considered utilization of the most popular and successful grid middlewares such as gLite and UNICORE. Both solutions provide basic necessary functionalities with different approaches to particular problems. Since the UNICORE middleware has been used in the creation of the distributed computing infrastructures connecting European supercomputer centers [2,3], we have decided to use similar approach in PL-Grid which gathers resources from the Polish supercomputer centers. In this chapter we provide a brief overview of the UNICORE grid middleware and its utilization in the large Distributed Computing Infrastructure. Then we describe the UNICORE infrastructure implemented in the PL-Grid project. Specific solutions motivated and developed within the project are also described.

Special attention is given to integration of the UNICORE middleware with the PL-Grid authentication and authorization framework which allows for uniform infrastructure and user management across different middlewares. The solutions for monitoring and accounting of the UNICORE infrastructure integrated with the PL-Grid framework are presented.

The main motivation of this work was to set up a production quality grid compatible with the existing European grid infrastructures, in particular DEISA. Therefore the UNICORE middleware has been chosen as a basis. The described solution has been developed to be compatible with the other parts of the PL-Grid infrastructure which was achieved by a proper set up and configuration of the UNICORE services.

The scope of this paper is to present an overall picture of the UNICORE installation in the PL-Grid project. The details of the middleware and particular components such as monitoring and accounting subsystems have been published elsewhere.

2 State of the Art

UNICORE 6 has been proven to be the middleware used to grant access to the distributed computational resources in a seamless way [4], which makes it attractive to build a national grid infrastructure. The UNICORE 6 middleware builds upon a number of concepts, such as Service Oriented Architecture (SOA) and messaging [5,6]. As defined by the OASIS SOA Reference Model [7], SOA is a "paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains". In effect, the capabilities of a distributed system are organized into well-defined chunks of functionality, called "services". Other parts of the system may then interact with the services to exploit their capabilities. To this end, both service and service consumer must agree on the service interface, the service semantics, and the real world effects of a service invocation. To promote loose coupling and reusability of services, consumers rarely bind to services directly, but usually engage in a process of dynamic discovery to 'discover' required services.

The web services technology enables building of document oriented distributed systems. In addition, web services cater to interoperability, an element that is of crucial importance for realizing SOA. Because web services use XML as their basic message exchange format, services and service consumers don't need to share the same implementation platform. The Web Services Resource Framework (WSRF) [8] is an application of web service technology to realize access and management of stateful resources. In grid computing, with its strong focus on heterogeneous resources that have to interoperate, web services have emerged as a key technology on which current grid middleware is built.

UNICORE, in its latest version 6, is compliant with the Open Grid Services Architecture [9] and uses standards such as WSRF 1.2 [10] and job submission definition language JSDL 1.0 [11]. The system is layered into three tiers: the client tier, the Gateway tier and the WSRF service container tier. Communication between these three tiers is carried out through the use of web services.

The Client tier consists of a number of diverse clients, varying in design and capabilities. The command line client can be used to submit simple script jobs or workflows described in the XML document according to the JSDL standard. The most advanced functionality, which includes workflow editing, is available with a dedicated client developed using Eclipse Rich Client Platform [12]. The web client for job or workflow submission has been developed using the UNICORE Basic Client libraries. The same tools are used to introduce grid capabilities to existing graphical applications. The Gateway authenticates messages and forwards them to the services. This mechanism provides a "web services firewall" and is used to protect several servers.

The services tier consists of a web services resource framework (WSRF) container hosting a set of basic grid services (atomic services), and additional services such as a registry. To access any service instance, clients need the Web Services Addressing End Point Reference (EPR) of that service instance. Via the Execution Management and Target System Interface (TSI) components, the services have access to physical resources such as batch systems or disks.

UNICORE 6 provides a generic web services hosting environment. Services can be deployed into this environment, in order to benefit from its general features: persistence, security infrastructure, and so on. Since all services are hosted in the same environment, a common web service level security infrastructure is achieved.

The basic grid services are often called "atomic" as they provide small, elementary services that may be composed into higher-level services. UNICORE 6 provides such basic services, most importantly job execution and file access. All services are WSRF services, so they provide a set of properties (WSRF resource properties), and the usual WSRF lifecycle and lifetime methods. Functions such as "job execution", "file management", "file transfer" and "service registry" are covered by these services. Abstraction and virtualization are used to hide target system specifics such as paths to binaries or storage mount points. In a typical system, many instances of most of these services run simultaneously.

The current UNICORE 6 security infrastructure [13] offers detailed access control, centralized user and role management and basic transport level security. UNICORE 6 provides basic communication security using secure socket layer and transport layer security (SSL and TLS). Users and server components are identified using a X.509 certificate issued by a trusted certification authority.

Requests are authorized using UNICORE authorization service (XUUDB) and a set of eXtensible Access Control Markup Language (XACML) policies, which issue the final decision whether the request is allowed to pass. Authorization policies can contain one or more rules and the XACML standard defines how rules are combined, and how policies can be composed of multiple subpolicies. Example rules expressed in natural language allow access to individual services for users with the "admin" role, allow querying the registry by everybody, deny access to job management instances to non-owners, or deny access to all users with a specific domain name in their email address. The policies for users and groups of users are stored in the UNICORE Virtual Organizations Server (UVOS) [14], which provides other services with the authorization and authentication data. As a result, flexible and powerful user management is obtained. Connections between components are protected using an SSL authenticated client. Client authentication means that both parties identify themselves using their certificates, and one party must trust the other party's certificate. For the communication between the Client layer and the Gateway, SSL is mandatory, whereas deployed services may communicate with the Gateway and the XUUDB using either SSL or unencrypted HTTP.

Workflow processing is at the core of the UNICORE 6 system [15]. The Workflow System is subdivided into two layers of abstraction: the Workflow Engine and Service Orchestration. The Workflow Engine processes the workflow on a logical level, whereas the Service Orchestrator deals with the actual execution and supervision of tasks using different services and resources on the Grid. Thus, the workflow processing logic is cleanly separated from the re-occurring invocations of lower level grid services.

In order to handle different domain specific workflow languages, the workflow engine translates incoming workflow descriptions into a generic workflow language using pluggable Domain Specific Logic (DSL) modules. The translation can be a complex process. For example, it may involve splitting of single activities into sets of activities that can be executed in parallel within the translated workflow. Splitting and distribution of computational effort is one of the core features of the system. Each activity of the translated workflow results in an atomic unit of work for the Grid, a so-called Work Assignment (WA). Work Assignments (WAs) are abstract, in the sense that they are not bound to specific service endpoints on the Grid. They are individually submitted to the Service Orchestrator for execution. Due to dependencies and conditions in the translated workflow, WAs cannot be executed in arbitrary order. For instance, one WA may depend on the output data of another WA. The Workflow Engine keeps track of such dependencies and does not submit WAs with unmet conditions. The Service Orchestrator transforms each incoming WA into a job, given in JSDL. It exchanges the logical names of input files for addresses of physical file locations, submits the job to a computing resource, supervises job execution and informs the workflow engine of job completion/failure.

3 Description of the Solution

Described above UNICORE 6 framework has been successfully implemented within the PL-Grid project giving access to the distributed computational infrastructure.

As presented above, the UNICORE system consists of a number of services which can be grouped as global, execution, workflow and data management services. Global services provide general information about the resources (registry) and authentication (UVOS). Workflow services are used to process and manage workflows submitted by users. Workflow and execution services require a wide variety of services in order to provide functional grid environment. Authorization services are web-service based and share common design patterns however they do not involve WSRF patterns. Execution services are responsible for the job execution process and have to be installed at target systems. All presented services must exist and be installed in order to build production quality infrastructure with the functionalities required by users.

The UNICORE PL-Grid infrastructure consists of a number of individual sites which are described by the list of provided hardware, services and installed applications. In PL-Grid most of the UNICORE services are installed at the main site located at ICM. The execution services are installed at the ICM as well as at other sites providing resources. The UVOS service is populated with the user data received from the PL-Grid LDAP repository which contains information about users, their priviledges and certificates. This data is synchronized with the UVOS and is used to authorize and authenticate users.

The UNICORE infrastructure consists of three installations dedicated to the different purposes and users communities. Because of that, the installations differ in scale and configuration. Namely we have the production infrastructure, which provides users with the resources in the computer centers. Then we have the test infrastructure, whose services (except UVOS) are independent of other infrastructures allowing for testing new versions of UNICORE, as well as solutions developed in the PL-Grid project. Finally there are training facilities intended for trainings (including remote ones), and allowing for the execution of short calculations on a specially prepared system. The training infrastructure includes all UNICORE services and is separated from the two other infrastructures.

3.1 UNICORE Production Infrastructure

Components of the production infrastructure can be separated into two sets depending on their role. The first set contains the central services which include components common to all facilities and services deployed on the distributed infrastructure. The second set is composed of the target system services which run independently on the computational systems distributed among the computational centers. These services are run by the centers which provide resources to PL-Grid. A typical configuration of the UNICORE services is presented in the Fig. 1.

The central services have been installed and run at ICM, while target system services have to be installed at each system connected to the UNICORE grid. Such configuration allows for fast connection of the additional resources even if the site administrators are not experienced with the UNICORE middleware. At the same time services crucial for the installation are controlled by the experienced team at ICM.

UNICORE Central Services. Central services are used by all the components of the UNICORE infrastructure and therefore they are accessed by all users and services installed on target systems. They store access addresses of the target systems, steer job submission process and maintain information associated with users. The central services include:

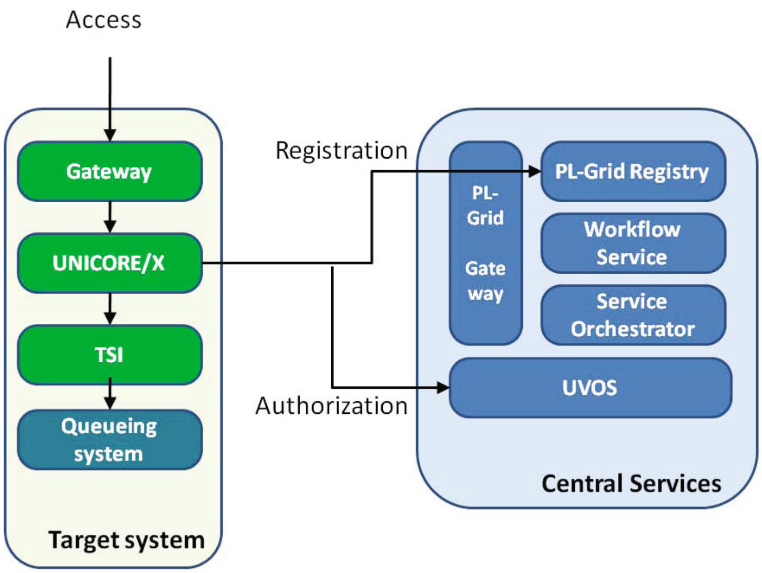


Fig. 1. Architecture of the UNICORE grid. Breakdown of components for the central services and the target system is marked. Communication flow is presented in a schematic way which does not describe details of the message exchange between the services.

- Registry,
- UVOS,
- Workflow Engine,
- Service Orchestrator.

The common register, amongst other functionalities, allows users to access all facilities through one address. UNICORE Clients and other middleware component services contact global registry to obtain detailed information on the available services and servers. This information is provided automatically while the registry location is known. Such solution is very convenient for the users as they have to be aware only of URL pointing to the registry. Such solution significantly simplifies access to the distributed resources.

Another important central service is UVOS, since it plays a crucial role in the authentication and authorization of the users. It is actively used by the target systems installed in all centers. For each computational center there were established separate groups in the PL-Grid virtual organization hierarchy. They allow for independent of the center determination of members belonging to the group and modification of all attributes within that group. In order to reduce the number of connections to the target system it is possible to configure a temporary cache that allows to speed up operations on the target system, and eliminates a possibility of calling multiple identical queries within a short period of time. This process is especially helpful where trust delegation is used and several connections to UVOS are performed in the short time.

The Workflow Engine and Service Orchestrator are necessary for the processing of the workflow jobs executed with the UNICORE system. The services use knowledge about the systems, getting the target system information from the global registry. Therefore, the use of workflows does not require any additional steps beyond the publication of the address in the central registry of services.

UNICORE Target System. The target system infrastructure provides access to the computational and storage resources available at the computational centers. This functionality is achieved through configuration and maintenance of the following services:

- UNICORE Gateway,
- UNICORE/X,
- TSI.

The first component provides secure access on the basis of certificate acceptance, in this case the certificate must be issued for the PL-Grid project by a dedicated Certification Authority. UNICORE/X is responsible for the services that allow access to the resources. Here we define which queues in the queue system are made available to users of the UNICORE infrastructure and which applications they can use. In addition, this component implements the communication with the TSI, which mediates with the queue system on the target system.

It should be noted that above services actively communicate with two central services available at ICM: the global registry and UVOS. Services store address of their instance in the global registry, which allows users to detect available resources and use them in the user generated workflows. UVOS provides authentication and authorization information to other services. It is configured in a way where each services provider (computational center) can block access for a particular user. It also allows for independent determination of additional attributes specific to the access to the resources, for example name of the queue which is accessible only to certain users.

The configuration of the global services and their maintenance is more complicated than target system services, therefore we have decided to minimize the number of instances. For the relatively small number of institutions providing resources in PL-Grid and therefore small number of target systems the central services are basically run in a single instance at ICM.

Reliability. In all grid systems the reliability of services is an extremely important point. In the case of UNICORE middleware the reliability is achieved based on the principle of redundancy of services. Currently supported solution enables application failover scenario. This involves setting up two copies of the same services, one of them is run as main and the second as a backup. In the absence of contact with the main instance, all traffic is routed to a service acting as a backup. In the case of UNICORE production infrastructure, such scenario should be used for both central services and target systems. It is worth noting that the main and backup central services, as well as the target system and

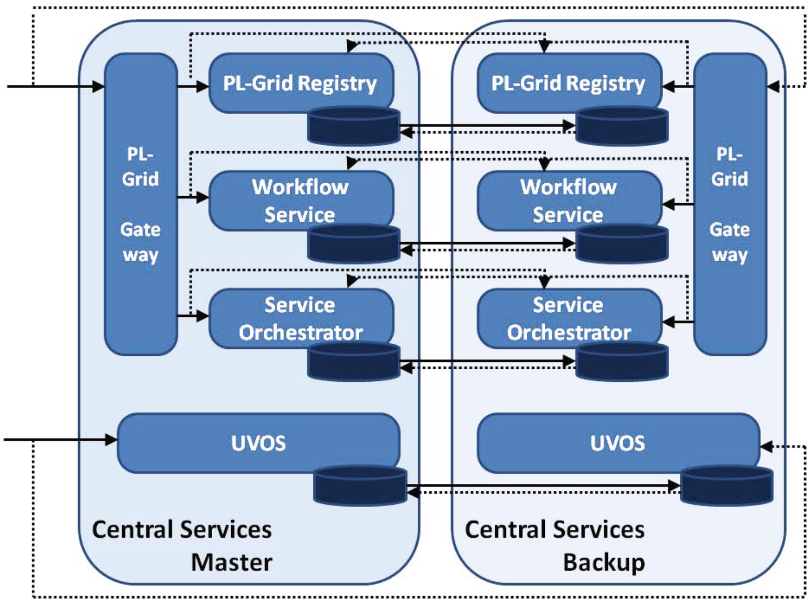


Fig. 2. Architecture of the UNICORE reliable central services

services, should be placed on physically separate machines to eliminate hardware problems. Redundant central services are configured to use the same data source. Furthermore, the reliability of the data source should be ensured, which in the case of the MySQL database can be accomplished in several ways using standard database tools and functionalities (see Fig. 2).

In order to achieve full reliability of the UNICORE grid, redundancy is also required at the target system. In the case of UNICORE/X, the most important thing is to define a common data source, e.g. MySQL database used by different instances. The reliability is then achieved by the redundancy of the data source, once more using standard mechanisms offered by the database engine (MySQL). One should notice that TSI does not require any additional configuration. Backup TSI instance works together with the UNICORE/X backup service. In order to achieve good performance TSI has to be configured to work with the queue system on the execution node. The architecture of the system is presented in Fig. 3.

The last component to which attention should be given is Gateway. Gateway is a stateless component that does not need additional source data. This allows solution at the network level. On both machines (master and backup) it should have the same configuration.

3.2 Training Facilities

In addition to the production infrastructure, it is very important to run infrastructure for both organized and remote trainings. The PL-Grid training

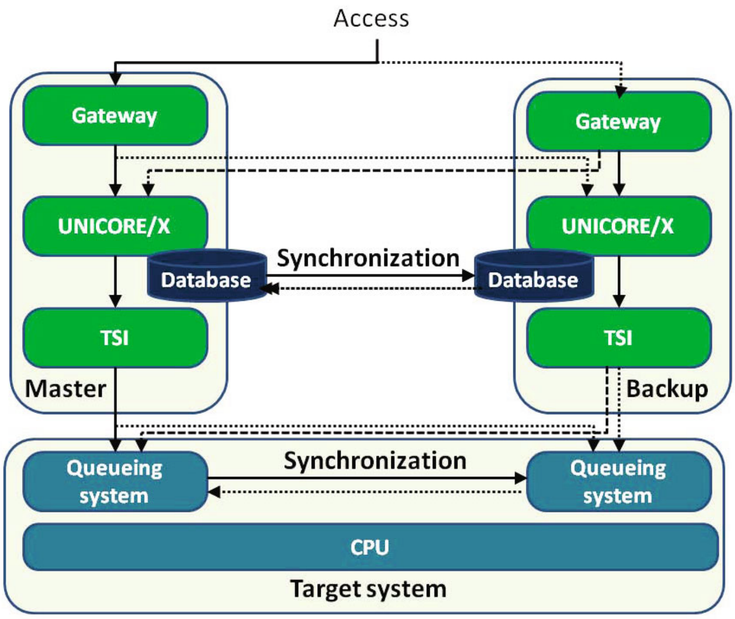


Fig. 3. Architecture of the UNICORE reliable target system

infrastructure was designed to allow temporary access for users not yet registered as PL-Grid users. The only requirement is to create an account in the PL-Grid portal. The training infrastructure has to emulate the production infrastructure, allowing users to familiarize themselves with how to carry out the calculations without the use of full resources in the centers.

Central services of the training infrastructure are separate instances of services used in production. The registry is independent and stores only target systems used for the training. A separate instance of UVOS includes only users who have access exclusively to the training infrastructure. Dedicated instances of Workflow Engine and Service Orchestrator are also installed to enable processing of task cascades.

For training purposes there are two target systems installed: ICM-TUTORIAL1 and ICM-TUTORIAL2. Two TSI services are configured to allow access to the computational resources. In order to optimize resource utilization both TSI are installed on the single system allowing users to run short jobs with the limited resources requirements. If necessary, additional resources can be attached to the test infrastructure. The training infrastructure has example applications installed: BLAST [16,17], Clustal [18], R [19,20] and PovRay [21].

The current version of the training infrastructure can be used by users registered in the PL-Grid User Portal. In order to familiarize users with how to use the UNICORE Grid system, a Certification Authority Center based on the PnP CA [22] has been prepared.

PnP CA is extremely useful for training, since it provides web interface to quickly approve new certificates for the users. The PnP CA server allows to generate certificate requests (CSR) and process them. The generated certificates are valid for a short period covering the training period. Additionally, through appropriate configuration of the notifications, CA automatically sends approved certificates to users, as well as the information about new applications to the persons responsible for training. This allows for partial verification of applications from individuals not registered in the PL-Grid project.

The PL-Grid Portal also provides interface for the users to obtain self-generated certificates issued by the PnP CA. The process itself uses an additional registration module registering applications in the virtual organization supplied with the UVOS server (uvos-webapp). It provides a customizable form, which defines the required data from the user, and then allows it to send a notification to the PnP CA server with a request to issue a certificate. Additionally, the registration determines the UVOS group to which the new user is attached.

This solution allows to hide from the user the less intuitive parts of the certificate issuing process and facilitates placement of additional information in UVOS, in the form of groups or attributes. These informations are necessary to grant users with proper privileges.

3.3 Test Infrastructure

The test infrastructure is also important for the PL-Grid UNICORE installation. It allows to test new solutions within the project as well as new developments of the UNICORE system. For example, this infrastructure has been used to test new UNICORE releases prepared within EMI project [23]. Access to this infrastructure is granted by the administrator to the people who in some way require the testing of new UNICORE system properties in conditions similar to those in the production infrastructure. Therefore access to test infrastructure is strictly limited to the administrators and developers.

As for the two other infrastructures, the central services, except UVOS are completely independent installations. The test infrastructure has its own registry and task components, transforming the workflow. Their configuration allows to test services before they are deployed in the production system.

The only difference is the use of the UVOS server. In this case the production instance of UVOS is used. For the purposes of the test infrastructure a separate group has been created in the PL-Grid virtual organization. Thus, the attributes and information about the users of the test infrastructure are kept in a separate branch.

It is also important to specify the use of the separate gateway which enables admission to the infrastructure for the UNICORE system developers and non-users of PL-Grid infrastructure.

The target system of the test infrastructure has to emulate the behavior of the target system in the production manufacture. For this purpose there are two virtual machines configured as computing nodes and two virtual machines for the UNICORE/X servers (main and backup installation). The test infrastructure has to ensure usage of the two compute nodes, which allows to simulate tasks using distributed communication between computing nodes. We have used separate hardware for both instances to check the reliability of the configuration based on the failover scenario. As in the production infrastructure for each UNICORE/X service there is a separate TSI server running on the same machine. In order to simulate the same behavior as in the production infrastructure, the target system uses the same queue system with the same service planning. For the ICM it is Torque server with the Maui scheduler.

Architecture of the test system is designed in the same way as the production infrastructure. Therefore, there is nothing preventing the centers from testing their solutions based on shared (test) central services. The only difference is a requirement to publish information on the target system in the test registry.

3.4 Monitoring

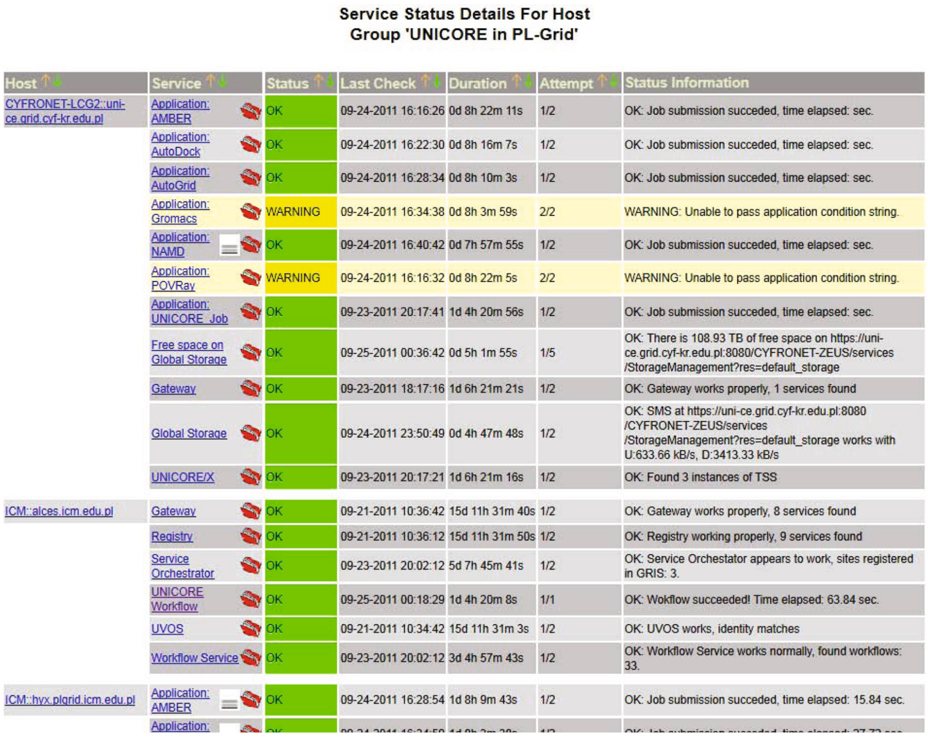
One of the basic principles of a high reliability system is to minimize the supplier or the administrator response time to the problems that prevent efficient operation of the system. A common phenomenon is lack of knowledge about problems arising in the system until the moment the need to use those elements arises. Unfortunately, this can sometimes be associated with loss of not only time, but also of important data. To avoid such problems the monitoring of the infrastructure is implemented to automatically perform periodic tests of the system components. In the case of UNICORE it has to check possible connection with services and check application performance on the target systems [24]. In the PL-Grid project the Nagios system [25] is used as the main monitoring framework. There is a single instance of Nagios which gathers information about status of all services: central and running in the providing resource centers. As a result the system administrators can consult each single service to see the current status of the UNICORE infrastructure.

Monitoring of the distributed resources is done using a number of tests (Nagios probes) developed to verify operation of the UNICORE services [27].

Among tested target system components one can find UNICORE/X which provides a range of functionalities. Nagios probes perform tests of the connection to the UNICORE/X server, tests for reading and writing a file in shared storage, checks of the status of the disk space and a simple task submitted through the queue system. Additionally, depending on the applications available at the target system, tests are executed in form of additional tasks.

In the case of central services, each has probes to test access to the service. A more complicated situation appears when workflow tests are performed, since this involves sending tasks and files between target systems. Nagios checks the ability to perform tasks by each of the target systems, as well as checks the automatic file transfer between them. To improve performance, a simple task on the target systems is not run when the test cascade of tasks performed correctly. This approach significantly reduces the number of jobs required by the monitoring system.

The PL-Grid project operates on the production version of Nagios which is responsible for monitoring of services of the production infrastructure only. To monitor the UNICORE test installation in PL-Grid, a dedicated installation of



Service Status Details For Host Group 'UNICORE in PL-Grid'

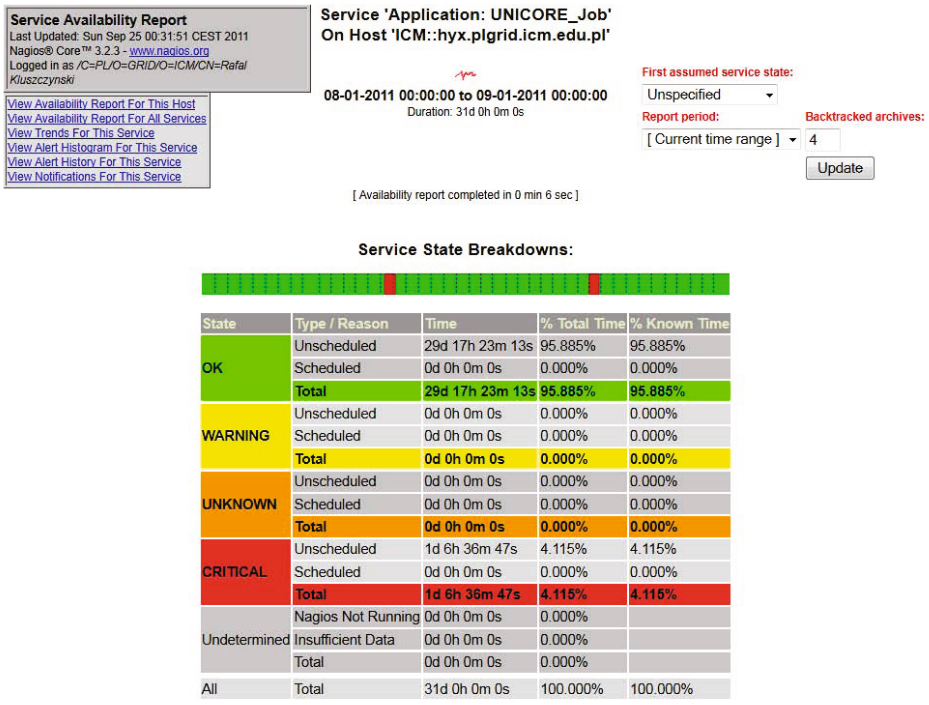
Host	Service	Status	Last Check	Duration	Attempt	Status Information
CYFRONET-LCG2::uni-ce.grid.cyf-kr.edu.pl	Application: AMBER	OK	09-24-2011 16:16:26	0d 8h 22m 11s	1/2	OK: Job submission succeded, time elapsed: sec.
	Application: AutoDock	OK	09-24-2011 16:22:30	0d 8h 16m 7s	1/2	OK: Job submission succeded, time elapsed: sec.
	Application: AutoGrid	OK	09-24-2011 16:28:34	0d 8h 10m 3s	1/2	OK: Job submission succeded, time elapsed: sec.
	Application: Gromacs	WARNING	09-24-2011 16:34:38	0d 8h 3m 59s	2/2	WARNING: Unable to pass application condition string.
	Application: NAMD	OK	09-24-2011 16:40:42	0d 7h 57m 55s	1/2	OK: Job submission succeded, time elapsed: sec.
	Application: POVRay	WARNING	09-24-2011 16:16:32	0d 8h 22m 5s	2/2	WARNING: Unable to pass application condition string.
	Application: UNICORE_Job	OK	09-23-2011 20:17:41	1d 4h 20m 56s	1/2	OK: Job submission succeded, time elapsed: sec.
	Free space on Global Storage	OK	09-25-2011 00:36:42	0d 5h 1m 55s	1/5	OK: There is 108.93 TB of free space on https://uni-ce.grid.cyf-kr.edu.pl:8080/CYFRONET-ZEUS/services/StorageManagement?res=default_storage
	Gateway	OK	09-23-2011 18:17:16	1d 6h 21m 21s	1/2	OK: Gateway works properly, 1 services found
	Global Storage	OK	09-24-2011 23:50:49	0d 4h 47m 48s	1/2	OK: SMS at https://uni-ce.grid.cyf-kr.edu.pl:8080/CYFRONET-ZEUS/services/StorageManagement?res=default_storage works with U:633.66 kB/s, D:3413.33 kB/s
	UNICORE/X	OK	09-23-2011 20:17:21	1d 6h 21m 16s	1/2	OK: Found 3 instances of TSS
ICM::alces.icm.edu.pl	Gateway	OK	09-21-2011 10:36:42	15d 11h 31m 40s	1/2	OK: Gateway works properly, 8 services found
	Registry	OK	09-21-2011 10:36:12	15d 11h 31m 50s	1/2	OK: Registry working properly, 9 services found
	Service Orchestrator	OK	09-23-2011 20:02:12	5d 7h 45m 41s	1/2	OK: Service Orchestator appears to work, sites registered in GRIS: 3.
	UNICORE Workflow	OK	09-25-2011 00:18:29	1d 4h 20m 8s	1/1	OK: Wokflow succeeded! Time elapsed: 63.84 sec.
	UVOS	OK	09-21-2011 10:34:42	15d 11h 31m 3s	1/2	OK: UVOS works, identity matches
	Workflow Service	OK	09-23-2011 20:02:12	3d 4h 57m 43s	1/2	OK: Workflow Service works normally, found workflows: 33.
ICM::hyx.plgrid.icm.edu.pl	Application: AMBER	OK	09-24-2011 16:28:54	1d 8h 9m 43s	1/2	OK: Job submission succeded, time elapsed: 15.84 sec.
	Application:					

Fig. 4. Screenshot of the status of the UNICORE infrastructure obtained with the Nagios monitoring system for the production infrastructure

Nagios is installed. It is also used to test monitoring components before installation in the production infrastructure. New Nagios probes and new versions of tests are checked here. Additionally, the test version of Nagios is used for monitoring of training and test UNICORE infrastructures. In addition, there are installed probes to analyze log files of each service. These tests work in a passive mode since they require access to log files and must be run on the machines that host services. Log analysis tests gather information of warnings and error messages appearing in the log files.

3.5 Accounting of the Use of the Resources

One of the aims of the PL-Grid project is to oversee allocation of the resources to individual users or research teams. In this process, accounting of the use of resources by individual users and users groups plays an important role. For the PL-Grid project there has been prepared a BAT portal in which each user is able to view their activity in different target systems. However, in order to provide such information to users, the history of jobs must be stored and gathered. For this purpose a number of services has been developed [28] for the UNICORE middleware.



Service Availability Report
Last Updated: Sun Sep 25 00:31:51 CEST 2011
Nagios® Core™ 3.2.3 - www.nagios.org
Logged in as */C=PL/O=GRID/O=ICM/CN=Rafal Kluszczynski*

View Availability Report For This Host
View Availability Report For All Services
View Trends For This Service
View Alert Histogram For This Service
View Alert History For This Service
View Notifications For This Service

Service 'Application: UNICORE_Job' On Host 'ICM::hyx.plgrid.icm.edu.pl'

08-01-2011 00:00:00 to 09-01-2011 00:00:00
Duration: 31d 0h 0m 0s

First assumed service state: Unspecified
Report period: [Current time range]
Backtracked archives: 4

[Availability report completed in 0 min 6 sec]

Service State Breakdowns:

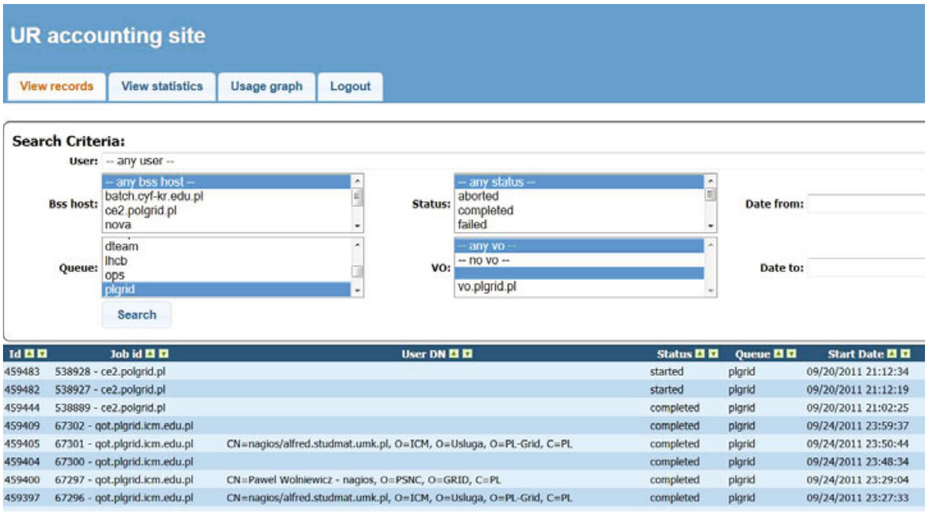
State	Type / Reason	Time	% Total Time	% Known Time
OK	Unscheduled	29d 17h 23m 13s	95.885%	95.885%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**29d 17h 23m 13s**	**95.885%**	**95.885%**
WARNING	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
UNKNOWN	Unscheduled	0d 0h 0m 0s	0.000%	0.000%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**0d 0h 0m 0s**	**0.000%**	**0.000%**
CRITICAL	Unscheduled	1d 6h 36m 47s	4.115%	4.115%
	Scheduled	0d 0h 0m 0s	0.000%	0.000%
	Total	**1d 6h 36m 47s**	**4.115%**	**4.115%**
Undetermined	Nagios Not Running	0d 0h 0m 0s	0.000%	
	Insufficient Data	0d 0h 0m 0s	0.000%	
	Total	0d 0h 0m 0s	0.000%	
All	Total	31d 0h 0m 0s	100.000%	100.000%

Fig. 5. Screenshot of the status of the UNICORE infrastructure obtained with the Nagios monitoring system for the production infrastructure. The detailed information on the test job execution on the site is presented. The graph shows timeline graph with information regarding the distribution of the jobs with different success levels.

Accounting system consists of several components, the most important are: **rus-bssadapter**, responsible for processing of the log files of the queue system in terms of consumption of the resources by the tasks performed, **rus-job-processor**, which is responsible for providing additional information about the tasks running with UNICORE infrastructure (for example, a DN of user certificate or virtual organization), **rus-service**, responsible for processing of accounting data, integration and preparation to visualization within the BAT website.

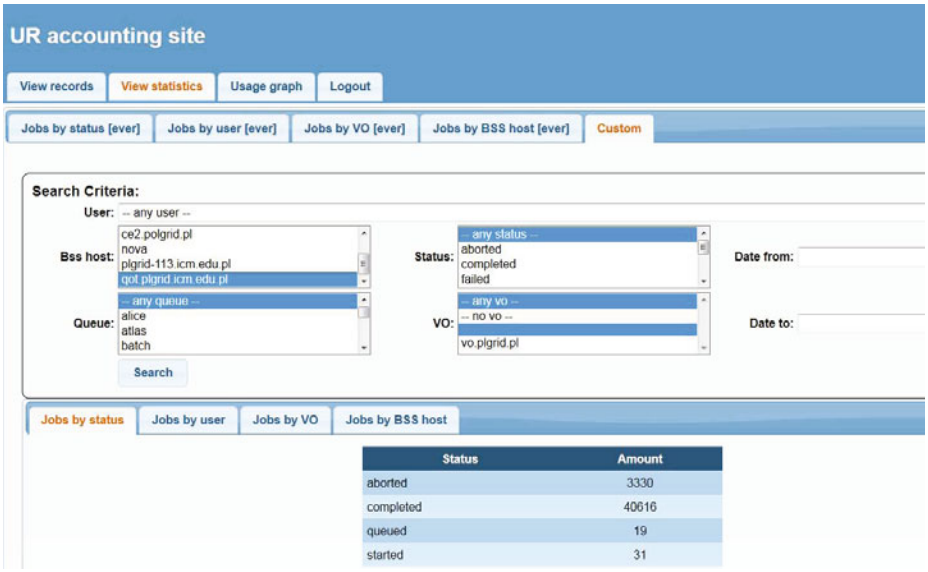
The first of these components is running on the same machine as the queue system since it requires access to log files with the resource consumption data. The second component must be installed on the server hosting UNICORE/X service. It allows for the transmission of the accounting data to the rus-service. Installation of the rus-service can be performed at any of the servers running UNICORE Services Environment (USE). However, it is suggested to run this component on the server hosting central services.

For communication between components of the accounting system additional service (called Broker) has been used. In this particular case Apache ActiveMQ [26] has been used. It works together with the UNICORE central services and


UR accounting site
View records
View statistics
Usage graph
Logout
Search Criteria:
User: -- any user --
Bss host: -- any bss host -- batch.cyf-kr.edu.pl ce2.polgrid.pl nova
Status: -- any status -- aborted completed failed
Date from:
Queue: dteam lhcb ops plgrid
VO: -- any vo -- -- no vo -- vo.plgrid.pl
Date to:
Search

Id	Job id	User DN	Status	Queue	Start Date
459483	538928 - ce2.polgrid.pl		started	plgrid	09/20/2011 21:12:34
459482	538927 - ce2.polgrid.pl		started	plgrid	09/20/2011 21:12:19
459444	538889 - ce2.polgrid.pl		completed	plgrid	09/20/2011 21:02:25
459409	67302 - qot.plgrid.icm.edu.pl		completed	plgrid	09/24/2011 23:59:37
459405	67301 - qot.plgrid.icm.edu.pl	CN=nagios/alfred.studmat.umk.pl, O=ICM, O=Usluga, O=PL-Grid, C=PL	completed	plgrid	09/24/2011 23:50:44
459404	67300 - qot.plgrid.icm.edu.pl		completed	plgrid	09/24/2011 23:48:34
459400	67297 - qot.plgrid.icm.edu.pl	CN=Pawel Wolniewicz - nagios, O=PSNC, O=GRID, C=PL	completed	plgrid	09/24/2011 23:29:04
459397	67296 - qot.plgrid.icm.edu.pl	CN=nagios/alfred.studmat.umk.pl, O=ICM, O=Usluga, O=PL-Grid, C=PL	completed	plgrid	09/24/2011 23:27:33

Fig. 6. List of the user jobs reported by the accounting system



Status	Amount
aborted	3330
completed	40616
queued	19
started	31

Fig. 7. Job accounting of a particular virtual organization grouped by the job status

contains separate queues for the data coming from test and manufacturing infrastructures.

In addition to the components of the accounting system described above, rus-UCC-plugin and rus-ur-site are used. The first one is an extension of the capabilities of the UNICORE Commandline Client (UCC). The second one allows for visualization of the data collected. The presentation is realized in the form of an additional web application. In order to run additional services servlet container such as Apache Tomcat is required.

4 Results

The UNICORE infrastructure has been successfully installed and integrated with the PL-Grid infrastructure. In the paper we have described in detail the UNICORE middleware deployment process on the existing infrastructure. Services planning and configuration has been developed. Monitoring and accounting of job submission through UNICORE is implemented. Monitoring and accounting information are integrated with the rest of the PL-Grid infrastructure. The procedure of attaching new resources to the UNICORE grid has been prepared and tested. Training facilities have been installed and made available to users. The training material was prepared and used in the form of traditional hands-on sessions as well as on-line tutorials.

As a result we have obtained a stable infrastructure which offers PL-Grid users and administrators flexible access to the resources such as CPU and storage. From the users' point of view, UNICORE is one of the access modes which does not require any additional activities or elements such as special keys, procedures, etc.

5 Conclusions and Future Work

The UNICORE middleware has been successfully used as a framework providing uniform access to resources and applications important for life sciences such as computational chemistry and molecular biology. This includes seamless access to computational resources, databases, as well as a framework for application specific interface development. Modular architecture and demonstrated visualization capabilities offer the possibility for a number of applications in diverse areas. The overall system architecture and the resulting software are innovative in providing rich and dynamic environment for the user interface, together with a lightweight framework for application developers. For users, the software offers powerful means to discover and access complex applications running on large-scale computational clusters, and to integrate applications in creative ways. For developers, the software combines support for grid technology, adherence to open standards, and integration with back-end service-oriented architectures.

References

1. Kitowski, J., Turała, M., Wiatr, K., Dutka, Ł.: PL-Grid: Foundations and Perspectives of National Computing Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 1–14. Springer, Heidelberg (2012)
2. Wypychowski, J., Pytlinski, J., Skorwider, Ł., Nazaruk, M., Benedyczak, K., Wroński, M., Bała, P.: Life Sciences Grid in EUROGRID and GRIP projects. New Generation Computing 22(2), 147–156 (2004)

3. Lederer, H., Pringle, G.J., Girou, D., Hermanns, M., Erbacci, G.: DEISA: Extreme Computing in an Advanced Supercomputing Environment. In: Bischof, C., Bücker, M., Gibbon, P., Joubert, G.R., Lippert, T., Mohr, B., Peters, F. (eds.) Parallel Computing: Architectures, Algorithms and Applications, John von Neumann Institute for Computing, Juelich. NIC Series, vol. 38, pp. 687–688 (2007)
4. Bala, P., Baldridge, K., Benfenati, E., Casalegno, M., Maran, U., Rasch, K., Schuller, B.: UNICORE – a successful middleware for Life Sciences Grids. In: Cannataro, M. (ed.) Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and HealthCare IGI, pp. 615–643 (2009)
5. Erwin, D.W., Snelling, D.F.: UNICORE: A Grid Computing Environment. In: Sakellariou, R., Gurd, J., Freeman, L., Keane, J. (eds.) Euro-Par 2001. LNCS, vol. 2150, pp. 825–834. Springer, Heidelberg (2001)
6. Streit, A., Bala, P., Beck-Ratzka, A., Benedyczak, K., Bergmann, S., Breu, R., Daivandy, J.M., Demuth, B., Eifer, A., Giesler, A., Hagemeier, B., Holl, S., Huber, V., Lamla, N., Mallmann, D., Shiraz Memon, A., Shahbaz Memon, M., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Schlauch, T., Schreiber, A., Soddemann, T., Ziegler, W.: UNICORE 6 – Recent and Future Advancements. Annales des Télécommunications 65(11-12), 757–762 (2010)
7. Reference Model for Service Oriented Architecture, Committee Draft 1.0 (2006), http://www.oasisopen.org/committees/download.php/16587/wd-soa-rm-cd1ED.pdf
8. Foster, I., Czajkowski, K., Ferguson, D.E., Frey, J., Graham, S., Maguire, T., Snelling, D., Tuecke, S.: Modeling and Managing State in Distributed Systems: The Role of OGSI and WSRF. Proceedings of the IEEE 93(3), 604–612 (2005)
9. Foster, I., Kesselman, C., Nick, J., Tuecke, S.: The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration. Open Grid Service Infrastructure WG, Global Grid Forum (2002), http://www.globus.org/alliance/publications/papers/ogsa.pdf
10. WSRF Web Services Resource Framework, http://docs.oasisopen.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf
11. JSDL 1.0. Job Submission Language Specification (2006), http://www.gridforum.org/documents/GFD.56.pdf (retrieved on March 6, 2008)
12. Jeff, M.: Eclipse Rich Client Platform: designing, coding, and packaging JAVA applications. Addison-Wesley (2006)
13. Benedyczak, K., Bała, P., Berghe, S., Menday, R., Schuller, B.: Key aspects of the UNICORE 6 security model. Future Generation Comp. Syst. 27(2), 195–201 (2011)
14. Benedyczak, K., Lewandowski, M., Nowiński, A., Bała, P.: UNICORE Virtual Organizations System. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Wasniewski, J. (eds.) PPAM 2009. LNCS, vol. 6068, pp. 155–164. Springer, Heidelberg (2010)
15. Sild, S., Maran, U., Romberg, M., Schuller, B., Benfenati, E.: OpenMolGRID: Using Automated Workflows in GRID Computing Environment. In: Sloot, P.M.A., Hoekstra, A.G., Priol, T., Reinefeld, A., Bubak, M. (eds.) EGC 2005. LNCS, vol. 3470, pp. 464–473. Springer, Heidelberg (2005)
16. http://www.ncbi.nih.gov/BLAST
17. Borcz, M., Kluszczyński, R., Bała, P.: BLAST Application on the GPE/UnicoreGS Grid. In: Lehner, W., Meyer, N., Streit, A., Stewart, C. (eds.) Euro-Par Workshops 2006. LNCS, vol. 4375, pp. 245–253. Springer, Heidelberg (2007)
18. http://www.clustal.org
19. http://www.r-project.org/

20. Borcz, M., Kluszczyński, R., Bała, P.: Statistical Analysis of Biomolecular Data Using UNICORE Workflows. In: Fred, A.L.N., Filipe, J., Gamboa, H. (eds.) BIOINFORMATICS 2010 - Proceedings of the First International Conference on Bioinformatics, Valencia, Spain, January 20-23. INSTICC Press (2010) ISBN 978-989-674-019-1
21. http://www.povray.org/
22. http://pnp-ca.sf.net
23. http://www.eu-emi.eu/
24. Bała, P., Benedyczak, K., Strzelecki, M.: Monitoring of the UNICORE middleware. In: Streit, A., Romberg, M., Mallman, D. (eds.) UNICORE Summit 2010 Proceedings, Jülich, Germany, May 18-19. IAS Series, vol. 5, pp. 117–123. Forschungszentrum Jülich GmbH Zentralbibliothek, Verlag (2010)
25. Barth, W.: Nagios: System and Network Monitoring. Open Source Press Gmbh, Münich (2008)
26. http://activemq.apache.org
27. http://unicore-life.svn.sourceforge.net/viewvc/unicore-life/monitoring/
28. Bała, P., Benedyczak, K., Lewandowski, M.: Resource Usage Accounting for UNICORE. In: Streit, A., Romberg, M., Mallman, D. (eds.) UNICORE Summit 2010 Proceedings, Jülich, Germany, May 18-19. IAS Series, vol. 5, pp. 75–82. Forschungszentrum Jülich GmbH Zentralbibliothek, Verlag (2010)
29. http://unicore-life.svn.sourceforge.net/viewvc/unicore-life/accounting/

User-Oriented Provisioning of Secure Virtualized Infrastructure

Marcin Jarząb, Jacek Kosiński, Krzysztof Zieliński, and Sławomir Zieliński

AGH University of Science and Technology,
Faculty of Electrical Engineering, Automatics, IT and Electronics,
Department of Computer Science,
al. Mickiewicza 30, 30-059 Kraków, Poland
`{mj,jgk,kz,slawek}@agh.edu.pl`

Abstract. The chapter presents a system for effective provisioning of VM Sets and supporting the dialog between providers and end users. VM Sets are networks of interconnected virtual appliances. Requirements for user-oriented provisioning of VM Sets are presented and an infrastructure supporting this activity is described. Stages of the VM Set provisioning process are defined and their functionality is described. Subsequently, manageability aspects are presented with special focus on runtime aspects. The implementation status of the VM Set provisioning environment within the PL-Grid Project is also reported.

Keywords: virtualization, virtual appliances, virtual networks, security, provisioning, VM Set, Cell-as-a-Service.

1 Introduction

On-demand provisioning of computational services has been a research goal for many years. According to the definition proposed by the American National Institute of Standards and Technology, on-demand provisioning is one of the five essential characteristics of cloud computing [1].

Cloud services are typically specified as belonging to one of three classes: IaaS (which stands for Infrastructure as a Service), PaaS (Platform as a Service) and SaaS (Software as a Service). This chapter presents the results of research experiments in the area of provisioning virtualized infrastructures to end users, conducted as part of the PL-Grid project. The experiments focused on providing IaaS/PaaS classes of services on top of the underlying PL-Grid infrastructure [27].

In the presented case users are offered the ability to express their needs as "VM Sets", which are networks of interconnected virtual appliances (virtual machines with preinstalled applications). The complexity of the internal organization of virtual appliances constituting a single VM Set varies depending on user needs [28].

The goal of the chapter is to present a system for effective provisioning of VM Sets as well as supporting the dialog between providers and end-users. The chapter describes a model of VM Sets and the process of automating their construction.

Network virtualization is one of the most important steps in VM Set implementation. The techniques involved in implementing virtual networks which interconnect VM Set entities enable introduction of a higher level of security. Security improvements result from the fact that network traffic (for different VM Sets) is isolated by using VLANs and tunnelling. Advanced network virtualization techniques may even support guaranteed quality of service (QoS) parameters.

Practical aspects of VM Set provisioning involve new requirements and technical challenges associated with standard features of the cloud computing paradigm, such as multi-tenancy, dynamic assignment of virtual resources and their reassignment according to consumer demands.

More mature scenarios of VM Set provisioning have to address rapid elasticity. This feature concerns the ability to rapidly and elastically provide resources (in some cases, in an automated manner), and quickly scale them up or down. This feature is important as cloud systems should automatically control and optimize resource usage. Such activity requires leveraging the metering capability at some level of abstraction, appropriate to the concept of VM Sets (e.g. processing, bandwidth, storage, etc.) Therefore, provisioning support for VM Sets should constitute a central element of cloud computing middleware.

The structure of the paper is as follows. In Section 2 requirements for user-oriented provisioning of VM Sets are presented. They are then compared to the IaaS/PaaS provisioning procedure. In this context a definition of a VM Set is introduced. User-oriented provisioning of VM Sets is described. In Section 3 the process of provisioning VM Sets is defined and its steps described. Suggestions for automating this process are also presented. In Section 4 the architecture of the VM Set provisioning infrastructure is described and all required system services are specified. In Section 5 manageability aspects of VM Sets are described, with special focus on runtime aspects. Section 6 discusses related work. The paper ends with conclusions contained in Section 7.

2 Requirements for User-Oriented Provisioning of VM Sets

The key concept introduced in the presented work is the VM Set, which comprises a set of VM Appliances interconnected by a virtual network. The VM Set can be implemented upon a provider's infrastructure and offered to the user for a specified period of time.

In its simplest form, the process of VM Set creation is initiated by the users who describe their requests in a formal or informal way, submitting a VM Set Requirements Specification documents to the provider. This specification is then processed by the provider, who creates a VM Set Deployment Description document. This, in turn, forms a basis for the instantiation of VM Appliances and configuration of virtual links between such appliances.

It should be noted that in order to construct a suitable VM Deployment Description the provider may need to gain better understanding of the functionality of the VM Set components. Thus, the process of constructing the VM Specification and Deployment Description (for a particular VM Set) should involve the user-provider negotiation phase.

The process of VM Set provisioning will be presented in more detail in Section 3. Here, we intend to focus on user input, i.e. the VM Set Specification.

At first glance it might appear convenient (from the user's perspective) for the VM Set Specification to involve as few technical details as possible. It seems that the specification should be a simple document, perhaps presented as a form, easy to fill out and submit. However, as some of the users' applications might present very specific requirements, the specification should not be a closed form; rather, its design should provide for extendibility – especially given that the VM Set concept may refer to IaaS as well as PaaS.

Roughly speaking, the specification of a VM Set consists of five parts, covering the properties of the desired (i) VM Appliances, (ii) network interconnections, (iii) shared storage, (iv) user access policy, and (v) lease period (see Fig. 1).

The specification of a VM Appliance consists of the following items:

– Required processor architecture, including the number of cores,
– CPU time requirements,
– Memory requirements,

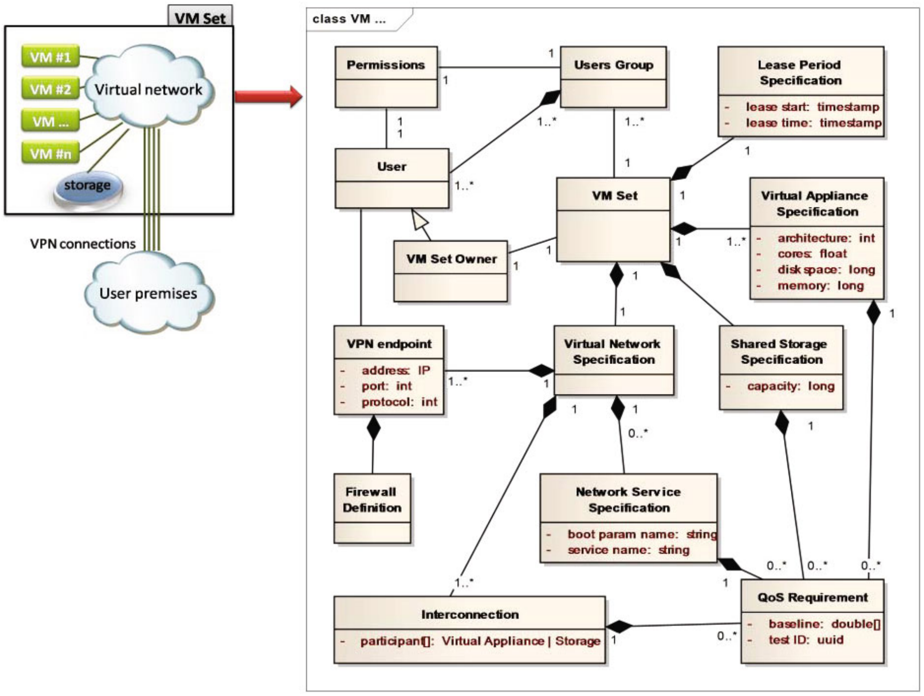


Fig. 1. VM Set specification

- Local storage requirements,
- Application software platform specification, when PaaS is requested.

Specification of VM Appliances could be augmented with QoS requirements. Such requirements may be expressed in a very generic form (a test identifier and an array representing the desired result), which subsequently needs to be narrowed down to a specific case, e.g. a set of micro-benchmark results for a machine on which a particular VM appliance is deployed.

Regarding the inter-VM Set network, a requesting user can specify a graph connecting VM Appliances and shared storage locations. Each edge of this graph represents a link, capable of routing flows characterized by the required:

a) bandwidth – the rate at which an application's (or VM's) traffic must be carried by the network,
b) latency – the network delay in delivering a data packet,
c) jitter – the variation in latency,
d) loss – the percentage of lost data.

The process of expressing these parameters in the definition of an inter-VM Set network should also support creating additional virtual network QoS policies. A policy regulating inter-VM traffic could include assurances regarding the ability to (i) handle traffic that meets the service needs of user applications (running inside VMs), (ii) allot resources preferentially to certain VMs and applications, (iii) simulate network conditions not available in LAN environments (e.g. greatly increased packet loss ratios).

In addition, a virtual network specification should describe:

- a set of VPN endpoints that will be used to access the created virtualized infrastructure (a minimum of one endpoint is required),
- a set of required firewall rules (e.g. for the applications to communicate with external nodes),
- a set of network services to be provided for appliances (e.g. DNS, HTTP proxy, etc.)

It is necessary to point out that some parts of the proposed virtual network specification touch upon organizational aspects; in particular the VPN endpoints and firewall rules are user-specific.

The third element of a VM Set specification refers to shared storage which can be of use for computational tasks. This description may contain such parameters as (i) storage location (ii) capacity, (iii) access network, etc.

The fourth element of the specification refers to access control policies. As the VM Set should be capable of serving multiple users over a specified period of time, the requestor (who then becomes the owner of the VM Set), needs to specify the groups of users who will use the VM Set, along with their privileges.

Access control policy relates to the final part of the VM Set specification – the preferred lease period. This parameter needs to be considered during the VM Set resource allocation process since any agreed-upon QoS requirements should be met throughout the operation of the VM Set. To achieve this goal, a resource scheduler module should be provided, along with efficient resource (re)allocation mechanisms.

The most important non-functional requirement regarding the provisioned VM Set is that the logical topologies of the VPN and QoS parameters are to be preserved regardless of problems affecting the provider's infrastructure. Events such as VM migration, resource (re)allocation etc., though unavoidable, cannot influence the logical view and parameters of the VM Set.

From the user's point of view, a typical provisioning scenario begins with specification of required virtual appliances and their properties. At this stage the user may be supported by a virtual appliance template repository, from which a range of predefined virtual appliance images can be selected. The images may be generic (having only an operating system installed), or profiled (equipped with additional components, e.g. specialized computational libraries). Following customization, a document describing the choice made by the user is submitted to the provider for further processing, as described in the following section.

3 Organization of the Provisioning Process

This section presents the VM Set provisioning process model, which sets out specific stages (Fig. 2) and defines a management model for virtualization of computing, network and storage resources on distributed physical servers, enabling flexible allocation of such resources in accordance with user requirements.

At the Requirements Specification stage the user asks the provider to create and expose a VM Set with parameters listed in Section 2. The requirements specification standard is defined by the infrastructure provider and may be very straightforward (e.g. filling out a predefined request form).

The Requirements Validation stage ensures requirements are validated against the provider's capabilities and restrictions according to the provisioned VM Set (deployment description document). In case of a negative result, the request must be rejected and specified once more. The user specification must be validated against the security policy imposed by the provider, and the existence of physical resources which could satisfy the user's requests must be confirmed. Another important aspect is the existence of a VM Set template which satisfies the stated requirements. These activities can be partly automated with custom administrative software; however some steps need to be performed manually.

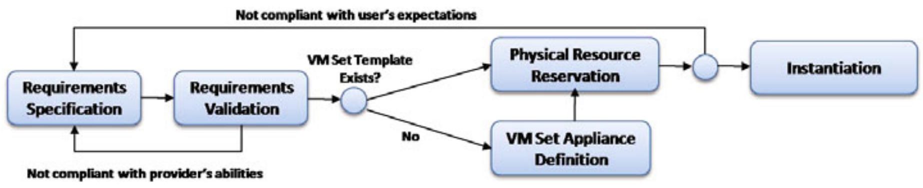

Not compliant with user's expectations
Requirements Specification
Requirements Validation
VM Set Template Exists?
Physical Resource Reservation
Instantiation
No
VM Set Appliance Definition
Not compliant with provider's abilities

Fig. 2. VM Set provisioning stages

The VM Set Appliance Definition stage exploits the reference servers for the VM Set configuration. VM Appliances containing PaaS elements require tools which, besides operating on raw OS distributions (relevant in the scope of virtualized infrastructure provisioning), must also capture knowledge about the application to be deployed, with special emphasis on elements such as infrastructure and application model, resource configuration management and dependency management [5]. These tools are technology-neutral (in terms of OS and middleware) and ensure that the prepared VM Appliances suit different requirements of a particular VM Set provisioning process instance. The infrastructure and application model provides data-driven representation of VM Appliances, with application-specific configuration templates applicable to different environments that might require different configuration parameters such as port numbers, credentials or computing resource requirements.

These parameters must be assigned appropriate values at the VM Set Instantiation stage (based either on user input or default values specified in the model). The resource configuration management capabilities are required to determine the desired configuration of an application's virtualized runtime environment through specification of minimal requirements for computing, storage and network resources and middleware. This step requires in-depth analysis, with a capacity planning process supported by performance tests or exploitation of computer system models like queuing networks [7] or others [8]. Some built-in services, specific to particular middleware packages [9], may apply heuristics to determine the best configuration depending on the assigned computing resources. Such services can also be used at the instantiation stage. Both the model itself and the resource configuration steps should be supported with a VM Set Assembly Builder for automation purposes. Such a solution is provided, for instance, by the VMware vApp [10] and Oracle tools [11].

The proper set of VM Appliances constitutes a reference configuration for the VM Set which can be used for provisioning virtualized infrastructures (IaaS) with middleware (PaaS). Dependency management ensures that, following initial configuration, the system verifies that all dependencies for future successful instantiation of a given VM Set are met. Upon validation and certification approval, snapshots of VMs running on the reference servers are created and stored in VM Appliance and configuration repositories, with metadata describing the installed software, OS version and type of hypervisor for each appliance within the VM Set.

The Physical Resource Reservation stage refers to tasks required of the provider to implement the logical representation defined by the user upon the physical infrastructure. In this case, the user's infrastructure provisioning request is dispatched to the operations centre, which, based on the number of instances of the VM Set and resource requirements, must marshal the required physical resources. If the required resources are not available, the instantiation must remain in the pending state until the problem is resolved. If such a scenario is not accepted by the user, the process can be rejected and initiated again, with new requirements.

VM Set instantiation is activated by the infrastructure provider and involves deployment of specific VMs with the required configurations of OS and application resources. In such an environment the instantiation of the requested VM Set consists of the following steps: (i) Creation of Virtual Machines based on plain or customized VM appliances, (ii) Configuration of a dedicated VLAN for the user connecting the VM appliances, and (iii) Configuration of the VPN connection to the user's site. The user specifies the configuration of computing resources (amount of RAM, number of CPUs), required storage and required network bandwidth (network configuration, including interfaces and IPs, remains transparent) which must satisfy the minimal requirements. If the specified resources exceed the minimum requirements, activities related to adjusting the internal middleware configuration should be performed. As described earlier, the configuration can be performed automatically or manually, by the user or by the infrastructure provider. Once the instantiation process completes, the user – by using a VPN connection – can access the virtual machines as if they were available locally.

4 Provisioning Infrastructure Architecture

In as much as possible, each stage of the VM Set provisioning process should be automated and therefore involve orchestration of Infrastructure Services according to provisioning procedures [6] expressing policies which govern access to computing, storage and network resources. As proposed in [14], the Service Oriented Infrastructure (SOI) defines a conceptual model which specifies the building blocks of the so-called Infrastructure Management Framework (IMF). IMF merges Infrastructure Services, IT process frameworks and user requirements, providing a modern computing infrastructure on which effective VM Set provisioning can be established. The provided functionality must automate provisioning and management activities across the entire computing infrastructure, including virtualized resources, operating systems, middleware, storage and network through user-defined workflows that implement "best-practices", defined within some IT process framework (ITIL [12]). Such an approach is exploited in the context of SOI provisioning with lightweight virtualization [22] and partly used in the proposed architecture (Fig. 3).

- User Access Services, supporting secure external user connectivity.
- Boot Services, supporting addition of new hardware to the provider's infrastructure.
- Repositories containing configuration data about available hardware and virtualized infrastructure resources, along with definitions of VM sets and VM Appliance images. The properties of these repositories are described in more detail in our previous paper [4].
- Infrastructure Management Services which provide an abstraction layer for the computing infrastructure provisioning process, allowing resource allocation to be managed in accordance with VM Set configuration requirements.

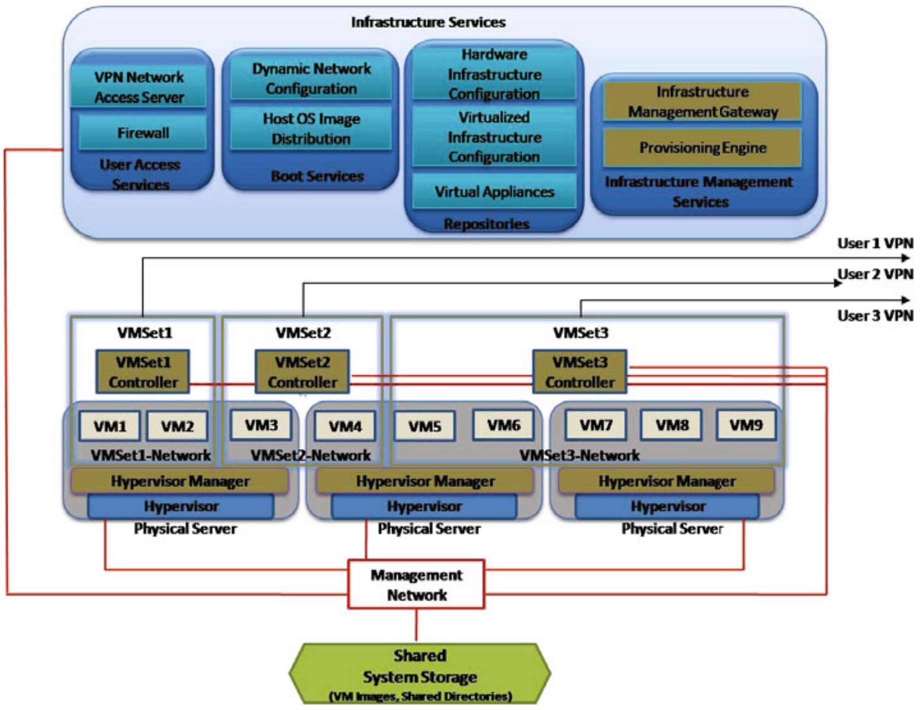


Fig. 3. System architecture for VM Set provisioning

The Infrastructure Management Services (IaaS, PaaS) are responsible for exposing vendor-specific interfaces as a common API that provides a set of interfaces used for provisioning, monitoring and management of VM Sets. They support discovery of the current network topology and current assignment of resources (list of available CPUs, memory, number of NICs) which can be partitioned between instances of specific VM Sets.

IaaS provides services required for VM provisioning, capacity and resource utilization monitoring by the VMs and controlling CPU, network and I/O assignments. Many other complex infrastructure-related tasks such as discovery of physical servers with running hypervisors and VM Set Controllers supervising activities of a particular VM Set should be supported without administrator intervention. Physical servers, storage systems, switches, routers, and other devices are pooled and made available to handle workloads generated by the VM Sets. Assignment of physical servers to particular instances of a VM Set is very challenging due to the fact that the infrastructure is shared and requires sophisticated scheduling algorithms implemented, for instance, by HAIZEA [13].

PaaS delivers platform services which include preconfigured OS images, databases and application servers with deployed service components, enabling software delivery in an on-demand business model.

In the proposed solution, Infrastructure Management Services are provided by four software modules: Infrastructure Management Gateway (IMG), Provisioning Engine (PE), VM Set Controllers and Hypervisor Manager (HM) which supports virtualized infrastructure provisioning, management and monitoring, ensuring consistency of the running VM Sets.

The outcome of VM Set provisioning is a runtime infrastructure which comprises a set of VM instances with middleware such as application servers, databases, load balancers, etc. Runtime infrastructure activity must be monitored and managed – a task which falls to the VM Set Controller, ensuring that the VM Set operates according to user expectations. The VM Set Controller is a configuration management component: a single instance of this component represents a single instance of the VM Set. This component stores configuration information and implements the required operations associated with managing the state of the VM Set.

The Hypervisor Manager exposes services for VM lifecycle management (creating, deleting, shutting down and migrating VMs), modification of runtime parameters including CPU and memory allocation and monitoring. The Hypervisor Manager also enables virtualization of network resources, including creation of VNICs and defining bandwidth limits for a particular VNIC assigned to a VM as required by the VLAN configuration of a particular VM Set. Furthermore, it contains services for self-organization of hypervisors (based on dynamic discovery) with a number of running VMs, and provides notifications about VM lifecycle events or exceeding attribute thresholds through computing resource monitoring. The information collected by monitoring services ensures fault detection and can be used to determine if a particular VM Set or a VM instance (within the VM Set) is active.

The Infrastructure Management Gateway (IMG) provides services required to retrieve information about the state of the infrastructure, such as a list of physical servers and VM Sets managed by the VM Set Controller. IMG also aggregates notifications about events in the distributed virtualized infrastructure (physical servers and hosted VMs). It is therefore able to manage data within the Configuration Repository.

The Provisioning Engine is responsible for managing the provisioning procedures which orchestrate Hypervisor Managers. It also carries out initial verification of the availability of resources at the level required by the infrastructure user. The provisioning procedure can be defined as a regular shell script or, in more advanced cases, through Business Process Management (BPM) which is used in NGOSS frameworks [16].

The Shared System Storage element is necessary for VM appliances to implement their features (including mobility and persistency) and access directories which store data used by the workloads within a given VM Set.

5 Implementation Status

As outlined in the preceding section, successful provisioning of VM Sets depends on collaboration of many complex services. Therefore, the current implementation of the VM Set provisioning environment does not offer full functionality to end users. Instead, key components are made available. The goal of this section is to provide insight into the state of system development. We will first provide an overview of the hardware infrastructure and follow up with a brief description of the implemented components.

5.1 Experimental Infrastructure Hardware

The presented system is based on the PL-Grid experimental infrastructure. Technically, the infrastructure consists of 16 HP ProLiant BL2x220c G5 blades (2x Quad-Core Intel Xeon 2500MHz, 16GB RAM) interconnected by a switched Gigabit Ethernet LAN, and a router for external connectivity.

As depicted in Fig. 4, the current implementation of the provisioning environment features a set of services required for the environment operations. These services were introduced in the "Provisioning Infrastructure Architecture" section. In order to provide the required services, two of the available blades were selected. These are called the infrastructure services provider and the storage server respectively. Both blades work under the Solaris 10u9 operating system.

The remaining 14 blades are used as computational nodes, with Slackware Linux and XenSource hypervisor installed as the basis for their operation. Additionally, the Hypervisor Manager components were installed on each node. The dependencies between individual elements of this environment are described in the following subsections, focusing on support for hardware configuration changes and VM Set operation respectively.

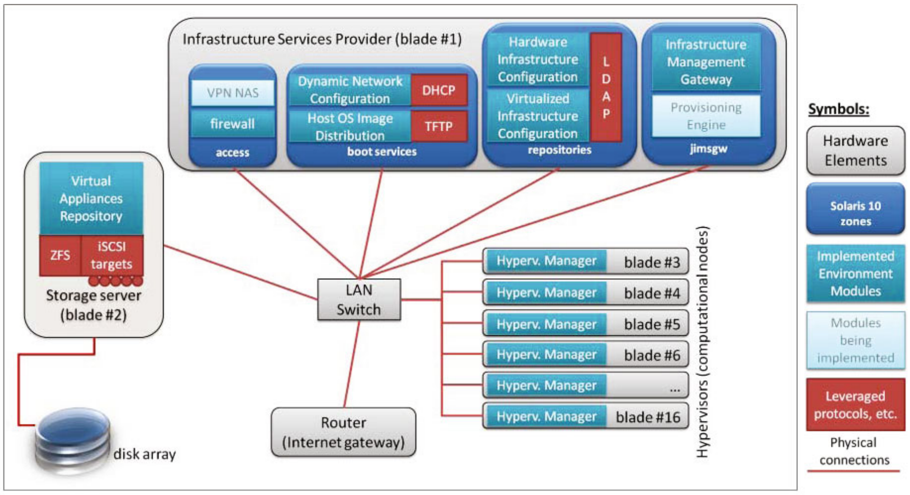

Infrastructure Services Provider (blade #1)
VPN NAS
firewall
access
Dynamic Network Configuration
DHCP
Host OS Image Distribution
TFTP
boot services
Hardware Infrastructure Configuration
Virtualized Infrastructure Configuration
LDAP
repositories
Infrastructure Management Gateway
Provisioning Engine
jimsgw
Virtual Appliances Repository
ZFS
iSCSI targets
Storage server (blade #2)
disk array
LAN Switch
Router (Internet gateway)
Hyperv. Manager
blade #3
blade #4
blade #5
blade #6
...
blade #16
Hypervisors (computational nodes)
Symbols:
Hardware Elements
Solaris 10 zones
Implemented Environment Modules
Modules being implemented
Leveraged protocols, etc.
Physical connections

Fig. 4. Key elements of the PL-Grid experimental infrastructure

5.2 Support for Hardware Configuration Changes

Although the key process to be supported by the environment is the provisioning of a user-specified VM Set, in order to function properly the environment needs to be aware of the underlying hardware configuration. Section 4 introduced a set of services which support hardware configuration changes. As the services are not expected to be computationally intensive, they were set up in the "boot services" zone of the infrastructure services provider blade. The list of services directly corresponds to the "Boot Services" block of the environment architecture (see Fig. 3) and includes:

– A DHCP service for configuring PXE clients,
– A TFTP service for distributing hypervisor operating system images.

The TFTP service allows the newly installed machines to download an image of the computational node operating system. The image is already augmented with a Hypervisor Manager instance which updates the Infrastructure Management Gateway and, in turn, the Hardware Configuration Repository as soon as the system is installed and started. Thus, the repository is able to keep track of the hardware computational nodes assigned to the provider, starting with the first boot-up of the hypervisor.

5.3 Support for VM Set Operation

The process of provisioning the VM Sets results in a set of separate, user-defined networks of interconnected virtual machines to be deployed by the provider upon the physical infrastructure. Due to the simplicity of the underlying network, the initial implementation of the VM Set provisioning environment assumes that the VM Appliances are interconnected using a plain, Ethernet LAN-like logical network. Logical networks are mapped to VM Set VLANs in order to separate network traffic.

To facilitate deployment of VM Appliances and their interconnections, the following services were set up on the storage server:

– Virtual Appliance Repository, which stores the images of virtual appliances' operating systems in a ZFS-based hierarchical organized structure of data volumes,
– iSCSI server, exposing a set of targets to be used as VM Appliances' logical disk drives.

The storage server blade therefore implements a portion of the "Repositories" block of the environment architecture. The blade uses 4TB of disk space to manage a ZFS-based repository of images of VM Appliances. Images are kept in a hierarchical dataset structure. Dataset names reflect the operating system names and version numbers. For example, the full repository path to the volume containing a recent version of Ubuntu Linux is `vdisks/linux/ubuntu/10.04.x86.20100426`.

In order to make the VM Set VLANs accessible to end users, the "access" zone of the infrastructure service provider was configured to provide the following services:

– VPN access server, for setting up dynamic lightweight VPN tunnels for users,
– firewalls, to secure the resulting infrastructure.

Additionally, the Virtual Infrastructure Configuration Repository was deployed in the "repositories" zone of the Infrastructure Services Provider blade to store the deployment description and the VM Appliances' configurations. The repository is implemented as an LDAP directory information tree.

The infrastructure ensures separation of provider and user network traffic. Separate virtual networks (VLANs) are constructed for provider and user operations. The set of hosts participating in a VLAN depends on the VLAN application (see Fig. 5).

As depicted in Fig. 5, not all nodes are needed in each of the configured VLANs. In particular, only the hypervisors which host specific user virtual machines are required to participate in the user's VLAN. On the other hand, all hypervisors need to participate in the management VLAN (which cannot be accessed by the virtual machines). Currently, the VLAN participation scheme is defined by the administrator, based on information stored in the Virtual Infrastructure Configuration Repository.

In order to provide an elastic and secure communication infrastructure for VM Sets, classic networking techniques which reflect the functionality of the "User Access Services" architecture block, are used. The services are implemented either by network switches or by the "access" zone of the Infrastructure Services Provider.

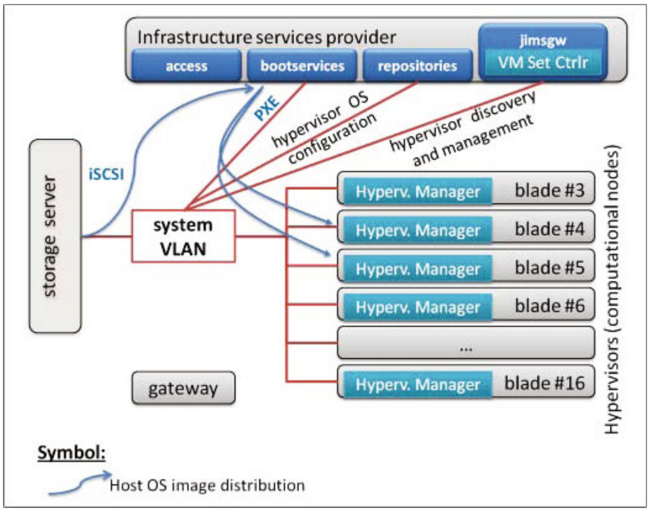


Fig. 5. Function-based machine participation in virtual networks

Inside the provider's domain the most important security-related aspect is user traffic separation. To achieve this goal, VLANs composed of virtual network interface cards (VNICs) are configured. However, as VLAN configuration is typically limited to a single LAN, the solution needs to be extended to remain usable outside a single provider's domain. To secure communication between end users and their VM Sets, traffic separation should also be maintained in the public Internet – for this reason on-demand Virtual Private Networks are used. The configuration of VLANs and VPNs is based on user credentials stored in the virtual infrastructure configuration repository.

To summarize, most of the features specified by the environment architecture have already been implemented. However, tools which automate typical provisioning and maintenance tasks still need to be developed. In particular, work on virtualized environment discovery and management [26], as well as on automated virtual machine provisioning, is ongoing.

6 Related Work

Selected aspects of the VM Set provisioning infrastructure development have been addressed in the authors' previous publications. These studies, however, are mostly provider-oriented (rather than user-oriented). The JMX-based VEI environment for managing VPN-connected VM collections is described in [1]. This work also addresses the problem of ensuring VM mobility while preserving secure links. A more in-depth study of this issue in the context of the Crossbow network virtualization library can be found in [3].

Effective management and automation of VM Set provisioning requires metadata describing the configuration of VM Sets. This metadata should be provided by the end user, stored in a metadata repository and kept up-to-date, mirroring any VM Set configuration changes. To maintain consistency between an existing VM Set configuration and its metadata, discovery services should be deployed and a dedicated monitoring service launched. The architecture of these services is presented in [4].

The concept of a VM Set, as introduced in this chapter, might be compared to the recently-proposed Cells-as-a-Service (CaaS) paradigm [23,25]. This approach is built upon the fundamental concept of a cell that contains virtual machines (VMs), storage volumes and subnets, all of which are declared as cell model elements. The model also describes how these components mesh together to create the desired virtual infrastructure. Each definition of a component or connection includes a set of relevant attributes. As systems providing CaaS are still under development, many issues remain unresolved. The VM specification elements include specifications for memory requirements, bus addresses, subnet connections and behaviour in the event of a failure. An XML document is used to express the cell model description. Such a document could be compared to metadata describing a VM Set. Similarly to our approach, the infrastructure designer can handle model changes by submitting an updated model document or by using an API that supports incremental updates. The cell controller is responsible for securely interacting with the service provider and for monitoring the virtual infrastructure's status.

As proposed, the VM Set is controlled by the VM Set controller which could be compared to the Autonomic Manager introduced in [17,18]. In the case of CaaS cell management activities are performed within the privileged system cell, responsible for creating and deleting all the virtual components and managing their connectivity, enforcing all the connectivity policies defined within each cell, cell interaction, cell recovery and any scalability limits associated with a cell.

Even though the concepts of VM Sets and CaaS have evolved independently, they share many similarities. There is no doubt that they constitute a new wave in cloud computing research. The complexity of provisioning the investigated systems shows that useful solutions should exploit an adaptive provisioning model such as the one used in SOA architectures [19,20,21].

7 Conclusions

User-oriented provisioning of secure virtualized infrastructures is a very complex process which should result in instantiation of a VM Set satisfying user requirements [29]. In shared environments provisioning a new VM Set should not compromise the QoS of already-running VM Sets. This is why, in addition to the previously discussed infrastructure services supporting VM Set provisioning, an admission control service has to be developed. The construction of such service is challenging as it should take into account current resource consumption and attempt to predict future demand.

Another challenging issue is ensuring scalability. Managing VM Sets containing thousands of interconnected VMs requires a distributed VM Set controller. It is also important to provide VM Sets which can be dynamic and capable of horizontal scaling. Management policy definition and enforcement is another interesting area of research. This aspect is related to transformation of high-level end-user requirements into equivalent sets of managements rules performed on the system level [23].

Additional effort has to be invested in the area of VM Set definition specifications. Standardization of such specifications would support portability of VM Sets between different cloud systems and enable development of multi-cloud VM Sets.

The proposed VM Set user-oriented provisioning process is an important milestone in this emerging research area. It captures the most important requirements and proposes a base architecture which can be further enhanced to address the issues identified above.

References

1. Liang-Jie, Z., Zhou, Q.: CCOA: Cloud Computing Open Architecture. In: IEEE International Conference on Web Services, ICWS 2009, pp. 6–10 (July 2009)
2. Kosińska, J., Kosiński, J., Zieliński, K.: Virtual Grid Resource Management System with Virtualization Technology. In: KU KDM 2009, Zakopane, March 12-13 (2009)

3. Jarząb, M., Kosiński, J., Zieliński, K.: Virtualization of Grid Networking Resources for Computation Mobility Support. In: Computational Methods in Science and Technology Special Issue, pp. 35–44 (2010)
4. Kosińska, J., Kosiński, J., Zieliński, K., Zieliński, S.: Flexible Organization of Repositories for Provisioning Cloud Infrastructures. In: KU KDM 2010, Zakopane (2010)
5. N1 Grid Service Provisioning System – Application Awareness: Automating the Data Center, SUN Microsystems Whitepaper (2004)
6. Zieliński, K., Jarzab, M., Kosinski, J.: Role of N1 Technology in the Next Generation Grids Middleware. In: Sloot, P.M.A., Hoekstra, A.G., Priol, T., Reinefeld, A., Bubak, M. (eds.) EGC 2005. LNCS, vol. 3470, pp. 942–951. Springer, Heidelberg (2005)
7. Gunther, N.J.: Analyzing Computer System Performance with Perl:: PDQ. Springer, Heidelberg ISBN 3-540-20865-8
8. Hellerstein, J.L., Diao, Y., Parekh, S., Tilbury, D.M.: Feedback Control of Computing Systems, August 24. Wiley-IEEE Press (2004) ISBN-13: 978-0471266372
9. Kou, X.: GlassFish Administration – Performance advisor, December 15. Packt Publishing (2009) ISBN-13: 978-1-847196-50-7
10. VMware vApp, `http://www.vmware.com/products/cloud-os/application.html`
11. Oracle Virtual Assembly Builder, `http://www.oracleimg.com/technetwork/middleware/ovab/overview/index.html`
12. Powell, B.: IT Infrastructure Library (ITIL) V3: Overview and Impact, Global Technology Services, ITS Strategy and Architecture
13. Sotomayor, B., Santiago Montero, R., Llorente, I., Foster, I.: Virtual Infrastructure Management in Private and Hybrid Clouds. IEEE Internet Computing (2009)
14. Jarząb, M., Kosińska, J., Kosiński, J., Zieliński, K., Zieliński, S.: The SOA infrastructure tools: concepts and tools. In: Virtualized Infrastructure for SOA. Poznan University of Economics Press (2010)
15. Zieliński, K., Jarząb, M., Wieczorek, D., Balos, K.: JIMS Extensions for Resource Monitoring and Management of Solaris 10. In: Alexandrov, V.N., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2006, Part IV. LNCS, vol. 3994, pp. 1039–1046. Springer, Heidelberg (2006)
16. Kosiński, J., Nawrocki, P., Radziszowski, D., Zieliński, K., Zieliński, S., Przybylski, G., Wnęk, P.: SLA Monitoring and Management Framework for Telecommunication Services. In: Fourth International Conference on Networking and Services (2008)
17. Adamczyk, J., Chojnacki, R., Jarząb, M., Zieliński, K.: Rule Engine Based Lightweight Framework for Adaptive and Autonomic Computing. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008, Part I. LNCS, vol. 5101, pp. 355–364. Springer, Heidelberg (2008)
18. Balos, K., Jarząb, M., Wieczorek, D., Zieliński, K.: Open Interface for Autonomic Management of Virtualized Resources in Complex Systems – Construction Methodology. Future Generation Computer Systems Journal 24(5), 390–401 (2008)
19. Psiuk, M., Bujok, T., Zieliński, K.: Enterprise Service Bus Monitoring Framework for SOA Systems. IEEE Transactions on Services Computing PP(99), 1, doi:10.1109/TSC.2011.32
20. Zieliński, K., Szydło, T., Szymacha, R., Kosiński, J., Kosińska, J., Jarząb, M.: Adaptive SOA Solution Stack. IEEE Transactions on Services Computing, February 07 (2011)

21. Szydło, T., Zieliński, K.: Method of Adaptive Quality Control in Service Oriented Architectures. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008, Part I. LNCS, vol. 5101, pp. 307–316. Springer, Heidelberg (2008)
22. Jarząb, M.: Adaptable SOI Provisioning with Lightweight Containers Virtualization Technology, PhD Dissertation submitted for acceptance, AGH University of Science and Technology, Department of Computer Science, Kraków (2011)
23. de Leusse, P., Zieliński, K.: Towards Governance of Rule and Policy Driven Components in Distributed Systems. In: Abramowicz, W., Llorente, I.M., Surridge, M., Zisman, A., Vayssière, J. (eds.) ServiceWave 2011. LNCS, vol. 6994, pp. 317–318. Springer, Heidelberg (2011)
24. Coles, A.: Cells-as-a-Service: A Cloud Infrastructure Service, tech. report HPL-2011-7, HP Laboratories (January 2011)
25. Banerjee, P.: Everything as a Service: Powering New Information Economy. IEEE Computer (March 2011)
26. Zieliński, S., Handzlik, T., Lewicki, T.: UPnP-Based Discovery and Management of Hypervisors and Virtual Machines. In: KU KDM 2011, Zakopane, March 9-11 (2011)
27. Kitowski, J., Turała, M., Wiatr, K., Dutka, Ł.: PL-Grid: Foundations and Perspectives of National Computing Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 1–14. Springer, Heidelberg (2012)
28. Radecki, M., Szepieniec, T., Szymocha, T., Szopa, M., Krakowian, M.: Towards Professional Service Operations in Grids. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 27–39. Springer, Heidelberg (2012)
29. Meizner, J., Ciepiela, E., Nowakowski, P., Kocot, J., Malawski, M., Bubak, M.: Flexible and Extendable Mechanism Enabling Secure Access to e-Infrastructures and Storage of Confidential Data for the GridSpace2 Virtual Laboratory. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 89–101. Springer, Heidelberg (2012)

Flexible and Extendable Mechanism Enabling Secure Access to e-Infrastructures and Storage of Confidential Data for the GridSpace2 Virtual Laboratory

Jan Meizner[1], Eryk Ciepiela[1], Piotr Nowakowski[1], Joanna Kocot[1],
Maciej Malawski[2], and Marian Bubak[2,3]

[1] AGH University of Science and Technology, ACC Cyfronet AGH,
Kraków, Poland
jan.meizner@cyfronet.pl
[2] AGH University of Science and Technology, Faculty of Electrical Engineering,
Automatics, Computer Science and Electronics, Department of Computer Science,
Kraków, Poland
[3] Informatics Institute, University of Amsterdam, The Netherlands

Abstract. This paper describes the security framework used by the GridSpace2 platform. The framework needs to restrict access to the Experiment Workbench to authorized users only as well as enable scientists to use multiple resources – computers (such as clusters, grids and clouds) and data, like e.g. relational database management systems (RDBMSes). The paper analyzes the appropriate technologies, details the proposed solution and summarizes the results of our research and development of flexible and extensible security solutions for scientists who need transparent access to heterogeneous compute and data resources. Additionally, as part of this paper, a wallet mechanism is described which enables secure storage of arbitrary confidential data such as credentials for external services.

Keywords: security, clusters, grids, virtual laboratory, wallet, clouds.

1 Introduction

Creating adequate security mechanisms for the GridSpace2 [2], [28] Virtual Laboratory [1] requires addressing many problems associated with key features of the target environment. The chief goal is to provide secure access to a large variety of heterogeneous computing and data storage resources commonly used by scientists. Meeting another crucial objective of the GridSpace2 platform – the ability to share experiments between groups of people – also requires us to take into account the security of confidential data accompanying these experiments.

The key issue which must be overcome when developing such a system is the heterogeneous nature of the resources from the security perspective, i.e. different resources often use different access mechanisms and credentials. Moreover, despite some standardization efforts, full interoperability between all e-infrastructures still has not been realized. Additionally, due to the highly dynamic nature of e-infrastructures – especially modern ones (like Clouds) – the developed solution must be generic and extensible, and not limited to a small set of systems. Finally, experiments should be self-contained so that they can be shared. However, sharing credentials in experiments would be highly insecure, and may lead to policy violations and loss of accountability. On the other hand, requesting users to manually enter credentials each time an experiment is run would reduce the usability of the solutions. This is why it was necessary to provide a mechanism for injecting appropriate user credentials when enacting experiments.

The solution described in this paper enables scientists to use e-infrastructures provided by the PL-Grid project [3], [29] as well as external ones (such as international grids or clouds) that may be available to the users. Reaching this goal required an analysis of the security mechanisms employed by these e-infrastructures and mechanisms that could be used to access them. Results of this research are presented in Section 2. Subsequently, in Section 3, we describe the proposed solution for securing the platform and providing unified access to various types of underlying resources, including those requiring custom credentials (such as RDBMSes) through the wallet mechanism. The overall architecture of our solution and a description of the technologies applied can be found in Section 4. Finally, we draw some conclusions and present a plan of our future work in Section 5.

2 Related Work

To provide proper security solutions we had to analyze multiple technologies. Those include various e-infrastructures such as clusters or grids with their security mechanisms, Shibboleth-based security solutions used by the previous versions of the GridSpace [4] as well as third-party tools and libraries that could prove useful for accessing the resources.

In addition to scientific High Performence Computing (HPC) resources provided through some form of grid middleware, some scientists prefer or require direct access to computing clusters. In this scenario they access them through a dedicated node commonly referred as "Head" or "User Interface" (UI) Node. This allows submitting jobs (in a batch fashion) to other nodes – so-called "Worker" Nodes – using workload managers such as the Portable Batch System (including original OpenPBS [5], its fork – TORQUE [6] and a commercial product called PBSPro [7]), Oracle (formerly Sun) Grid Engine (SGE) [8] or Univa Grid Engine [9]. In the PL-Grid project, access to User Interface nodes is available through the Secure Shell (SSH) protocol. It is possible to submit jobs to the cluster from UIs without additional authentication.

In addition to local clusters, users who require larger pools of resources may need to use Grids such as those developed in the Enabling Grids for E-science (EGEE) or European Grid Infrastructure (EGI) European Union (EU) projects [10]. Access to those e-infrastructures is provided through appropriate middleware such as the Globus Toolkit and gLite, which both employ the Grid Security Infrastructure (GSI) as security mechanism. GSI is based on the concept of private keys and X.509 certificates [26] – providing public user keys with additional information signed with a Certificate Authority's (CA) private key. In addition to regular certificates, GSI introduces the concept of proxy certificates [26] – a short-lived certificate signed by the user with a private key associated with a personal grid certificate. The proxy itself, its private key and the original user grid certificate can be used to delegate user credentials. They can also be stored using mechanisms such as MyProxy [11]. Notable examples of interoperability between GSI and other systems include GridShib [13] and ShibGrid [14] which both aim at providing integration with the Shibboleth Single Sign-On (SSO) mechanism for authentication and attribute-based authorization. GridShib (whose support has now been taken over by the CILogon project [18]) allows access to infrastructures either by generating certificates based on Shibboleth credentials (through GridShib CA) or by binding Security Assertion Markup Language (SAML) assertions to the Proxy Certificate (GridShib SAML Tools) and using them for authorizing access to Java-based GT services (ShibGrid for Globus Toolkit). A prominent scientific user of GridShib CA and the Shibboleth is the TeraGrid project which provides access to its resources through the GO TeraGrid Portal [20]. In addition to Shibboleth, GridShib CA [19] can also use open identity management mechanisms such as OpenID. ShibGrid works on a similar basis, either by generating short-lived certificates based on Shibboleth credentials or by allowing users to protect standard grid key/certificate pairs with Shibboleth credentials. Another solution providing more manageable interoperability with other systems is GSI-OpenSSH [21]. It allows using standard GSI credentials to access computing nodes (e.g. the above-mentioned cluster UI nodes) with standard GSI credentials. Its drawback is that it cannot be used unless appropriate software (a modified OpenSSH server) is installed by administrators.

Another type of HPC resource, namely supercomputers, is provided by the Distributed European Infrastructure for Supercomputing Applications (DEISA) and Partnership for Advanced Computing in Europe (PRACE) [15]. Accessing DEISA sites is possible through various mechanisms. Interactive access is provided through GSI-SSH. As it is one of the so-called "Core" services, it should be available on all sites. There is also standard grid middleware called the Uniform Interface to Computing Resources (UNICORE) [16] as well as a toolkit called DEISA Services for Heterogeneous management Layer (DESHL) providing a command-line (CLI) interface and an API for UNICORE. Those services are designated "Additional Services" and, as such, should also be available on all sites (like "Core" services) but with more lenient reliability guarantees. Like GSI, UNICORE bases on X.509 certificates. Finally, there is a group of services called "Optional", including WS-GRAM from the Globus Toolkit, which use the GSI security mechanism described above but do not have to be provided on all sites. Before developing a new solution we also had to analyze the previous version of the GridSpace platform. It is based on the Shibboleth framework and supports securing both Web-based software (by design) and non-Web-based applications (following appropriate adaptation). Later on we have also developed a tool called pam_shib [12] which enables secure access to entire computing nodes, in turn exposing software installed on these nodes without any modifications. Despite the fact that it would certainly satisfy most of our requirements, some serious drawbacks have prevented us from adapting the described solution. The most key issue is the requirement to install the pam_shib module on UI nodes, which would be infeasible. This is due to the fact that such an action would significantly affect the OS configuration. Additionally, it would require constant effort to maintain the operation of a critical service, as failing to do so could create serious security risks – including the possibility of compromising the entire project infrastructure. Finally, additional, complex (Shibboleth-based) security infrastructure, not used by other PL-Grid partners, would have to be deployed. Due to these reasons we have decided to look for other, more generic solutions.

3 Description of the Solution

To provide scientists with secure access to the GridSpace2 platform, as well as all resources described in Section 2 a special security solution was developed. This section describes its various aspects, including access to the GS2 Experiment Workbench (EW) itself, working with clusters and grids as well as accessing other resources such as databases which may require arbitrary credentials. The architecture of the solution is shown in Fig. 1. The depicted resources serve as examples – it should be noted that GridSpace2 allows access to huge varieties of resources with the assistance of the wallet mechanism. The dashed line on the diagram represents the ability to access credentials stored in ReCeW from external nodes (not only from the Experiment Host).

3.1 GridSpace2 Experiment Workbench Security

As shown in Fig. 2 the GS2 Experiment Workbench exposes 3 entry points that had to be secured. The main one is, of course, used by the scientists to develop and execute experiments. The second one is used by the platform administrator to edit its configuration, including experiment hosts and interpreters, register certificates used by the ReCeW remote wallet (described later on in this paper) as well as customize look and feel of the EW. The final endpoint is used to run internal diagnostics, for example by the PL-Grid Operations Center during automated (Nagios-based) testing.

For the sake of simplicity (given the lack of IT expertise among our end users) we have selected a user/password authentication method for the main entry point. To avoid the need for separate registration or complex synchronization of EW accounts with the main PL-Grid security infrastructure, we have performed full integration with this infrastructure using an authorization mechanism based on the SSH protocol. As a result, any user with access to UI nodes (like those

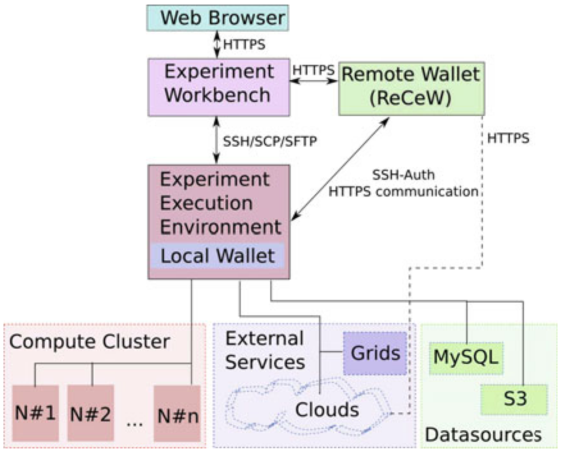


Fig. 1. Architecture of the GridSpace2 Security system, depicting components of the platform as well as examples of services that may be accessed using EW either directly (clusters), or using credentials stored in the Wallet

provided by the project partners) may access the EW using the same credentials. As PL-Grid provides integrated Lightweight Directory Access Protocol (LDAP) access mechanisms, any PL-Grid user may use the EW (following activation of UI access in the PL-Grid Portal). SSH integration was implemented using the Ganymede SSH-2 for Java [22] library providing a pure-Java implementation of the SSH protocol (client side).

Access to the configuration entry point is enabled by two mechanisms. The first one uses a preconfigured password for authentication and authorization. The second allows GSI certificate-based authentication using a Tomcat connector developed by Globus. Following authentication, the user is authorized to access the configuration entry point based on a preconfigured list of Distinguished Names (DNs).

These mechanisms are also used for the test entry point. The password is the same as the one used for accessing configuration, however the DN list is separate so it is possible to restrict access to just one of the entry points, basing on the presented certificate.

3.2 Accessing Computing Clusters

The UI nodes offer the ability to run jobs on the clusters they are part of without additional security requirements. It is therefore sufficient to enable access to those nodes which, in PL-Grid, may be effected by using the SSH protocol. Logging to EW requires the user to provide appropriate credentials for the selected UI. These credentials are then used to open an SSH session. In turn, authentication and authorization to the EW is sufficient for immediate access to the corresponding computing cluster.

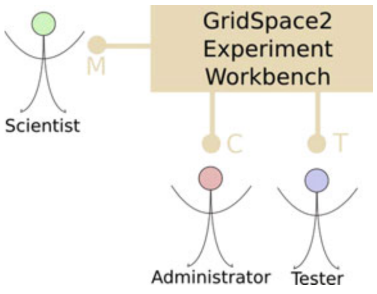


Fig. 2. The three entry points exposed by the GridSpace2 Experiment Workbench (EW) that need to be secured – the main one (M) used by the scientists (users), configuration (C) used by EW administrators and testing (T) used to run automatic self tests

Table 1. Key differences between Local and Remote Wallet mechanisms provided by the GridSpace2 Experiment workbench

	Local	Remote
Accessible from	Single Exp. Host	Any location in the Internet
External API	none	REST based
Additional infrastructure	none	Small UNIX daemon
Additional authentication	No	Yes - SSH (integrated with EW)
Security measures	UNIX file permissions	AES256 encryption, external location

3.3 The Wallet Mechanism

Scientific users may also need to access various external resources (requiring non-standard credentials), such as RDBMses, Representational State Transfer (REST) / Simple Object Access Protocol (SOAP) services like Amazon S3 Cloud Storage API [27], etc. As it is not possible to foresee and integrate all of them, we have decided to provide a generic solution that could facilitate secure storage of credentials through a so-called wallet mechanism. At present it allows storage of both plaintext credentials ("String secrets"), like passwords, as well as more complex "File secrets", like certificates. During the project we have created two types of wallets which we designated "Local" and "Remote". Key differences between them are outlined in Tab. 1. The "Local" wallet is simpler to set up and should be sufficient unless it is necessary to access the stored credentials from multiple locations – either from a different experiment host (using standard mechanisms provided by the EW) or an arbitrary locations (using the API).

The Local Wallet uses files stored in a hidden directory on the Experiment Host (EH), which is automatically created whenever needed. As such it does not require any preconfiguration and is instantly available after the specific EH is added to the EW configuration.

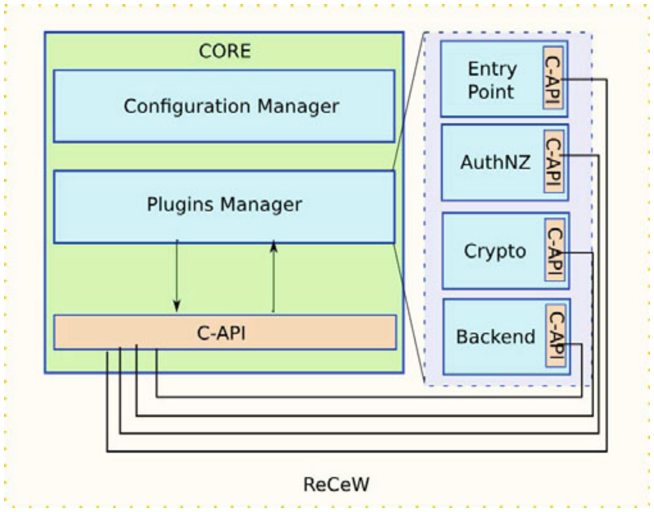


Fig. 3. Overview of ReCeW, including its main module (CORE) responsible for maintaining configuration, managing plugins and facilitating communication with the four available plugins

The Remote Wallet offers much more advanced features, but at the price of a more complex deployment process. First of all, a special server (UNIX daemon) called the Remote Central Wallet (ReCeW) must be deployed on a designated host. It provides flexible mechanisms for secure storage of user credentials. The software consists of the main module (the actual daemon) as well as four types of plugins implemented as standard Linux dynamic libraries (.so). The CORE is responsible for maintaining ReCeW configuration (Configuration Manager) as well as plugin management (Plugin Manager). This includes loading all plugins from a preconfigured location, wrapping their functions and allowing communication between plugins inside the CORE. Communication between plugins and the CORE is done through a specific API written in ANSI C (C-API), which has been chosen over C++ for portability reasons (to allow building CORE and plugins using different compilers). The rest of the code is written in C++. All communication between plugins is done via the CORE module (C-API and Plugin Manager wrappers). The decision to use C and C++ instead of an interpreted or VM-based language has been made to ensure maximum efficiency.

The "Entry Point" plugin is responsible for providing a custom API for ReCeW users – in PL-Grid it is based on REST, but may be replaced by any other protocol, e.g. SOAP. Hypertext Transfer Protocol (HTTP) server functionality of the module is provided through the GNU libmicrohttpd library [23]. Communication is secured using the Transport Layer Security (TLS) protocol provided by the gcrypt and gnutls libraries. The API is designed to be manageable. It is divided into sections – *auth* used for authentication and authorization and *cred* for credential management. Required attributes are sent using the HTTP POST mechanism with appropriate encoding (*application/x-www-form-urlencoded*). Requests that require prior user authentication need to include an additional custom HTTP header *X-ReCeW-SID* which is used to authorize them. Accessing ReCeW through a RESTful API takes the form of the following sequence of steps:

1. Call */auth/login* with appropriate parameters – *user_name* and *credential* to get Session ID (SID) – which is set as *X-ReCeW-SID* and returned as the body (for convenience) in response.
2. Call the appropriate method used to load or store credentials – for example */cred/load* with the *cred_id* parameter or */cred/loadall* without any parameter to get specific/all secrets as JavaScript Object Notation (JSON) encoded object/table of objects stored on behalf of the authenticated user.
3. Call any other method to load/store/check size or obtain additional metadata.
4. Finally, call */auth/logout* without parameters.

The "AuthNZ" plugin controls authentication and authorization of ReCeW users. In the project we have implemented a mechanism based on SSH access to one of the UI nodes – just like the one used for the EW itself. This allows smooth integration with the EW on the one hand, and with the PL-Grid security infrastructure on the other hand. However, ReCeW and EW security configurations remain separate so it is possible to restrict external access to either of them, as required. Much like in the case of the API, it is, of course, possible to allow any other authentication and authorization mechanism (e.g. based on internal user/password, LDAP, etc.) by creating a new plugin.

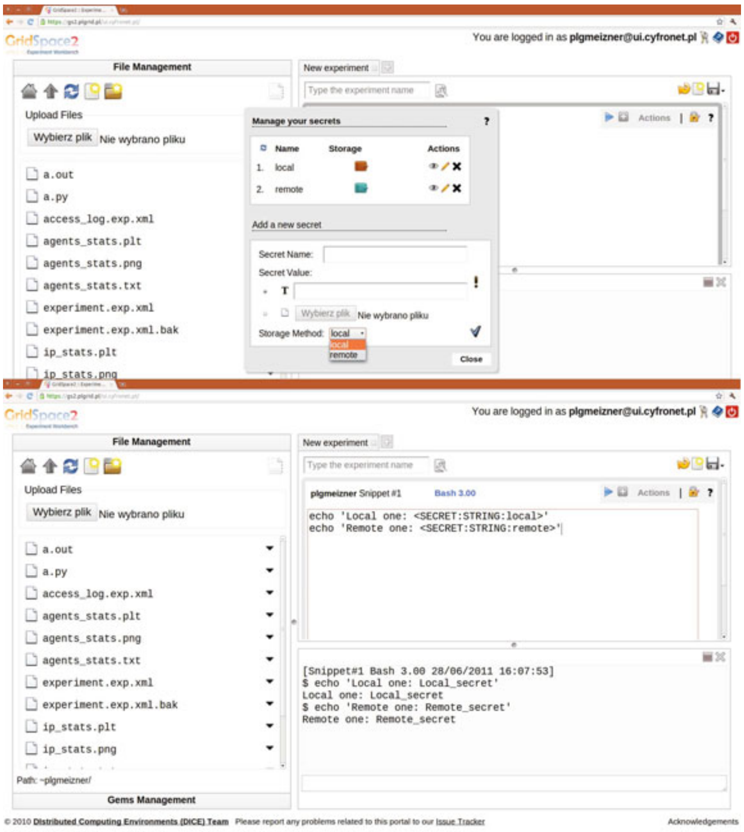
The "Crypto" plugin is responsible for providing encryption for ReCeW. At present, the strong Advanced Encryption Standard (AES) 256 encryption standard is used, as provided by the libgcrypt library [24] – a commonly used cryptographic library based on GnuPG. Should other solutions become necessary (such as e.g. hardware cryptographic devices) they can be implemented as replacement plugins.

The final plugin – "Backend" – is responsible for storing encrypted credentials. At present, SQLite – a lightweight standalone database engine – is used, however it could be replaced by any other RDBMS (like MySQL, PostgreSQL) or even a completely different solution (like Amazon S3).

Both wallets are fully integrated with the Experiment Workbench, which provides a Graphical User Interface (GUI), supports using credential in snippets and ensures integrated security for ReCeW. The GUI allows adding, viewing, editing and removing credentials to/in/from both wallets. Credentials stored in wallets may be accessed from snippets by using special formulas:

– <SECRET:STRING:secret_name> – replaced during execution by the content of the secret,
– <SECRET:FILE:secret_name> – replaced by the location of the secret file on the experiment host.

Finally, EW provides smooth integration with ReCeW security so it is not necessary to perform any additional authentication when accessing the "Remote"


GridSpace2
You are logged in as plgmeizner@ui.cyfronet.pl
File Management
Upload Files
Wybierz plik Nie wybrano pliku
New experiment
Type the experiment name
Actions
Path: ~plgmeizner/
Gems Management
© 2010 Distributed Computing Environments (DICE) Team Please report any problems related to this portal to our Issue Tracker
Acknowledgements

```
a.out
a.py
access_log.exp.xml
agents_stats.plt
agents_stats.png
agents_stats.txt
experiment.exp.xml
experiment.exp.xml.bak
ip_stats.plt
ip_stats.png
```

Manage your secrets

	Name	Storage	Actions
1.	local	🟧	👁 ✎ ✗
2.	remote	🟦	👁 ✎ ✗

Add a new secret

Secret Name:

Secret Value:

Wybierz plik Nie wybrano pliku

Storage Method: local (local, remote)

Close

plgmeizner Snippet #1 Bash 3.00

```
echo 'Local one: <SECRET:STRING:local>'
echo 'Remote one: <SECRET:STRING:remote>'
```

```
[Snippet#1 Bash 3.00 28/06/2011 16:07:53]
$ echo 'Local one: Local_secret'
Local one: Local_secret
$ echo 'Remote one: Remote_secret'
Remote one: Remote_secret
```

Fig. 4. Depiction of the Experiment Workbench wallet management interface (top) allowing viewing and manipulating stored credentials, as well as fragments of code and execution result of snippets which use secrets drawn from local and remote wallets (bottom)

Wallet. Fig. 4 depicts the graphical interface used to manage both types of wallets as well as an experiment using (displaying) credentials from these wallets (both the code itself and the results of execution).

3.4 Accessing European e-Infrastructures

As already described, it is possible to access any arbitrary system with the help of the wallet mechanism. This mechanism could also be used for accessing e-infrastructures such as EGEE/EGI grids. Accessing those resources requires uploading generated proxy certificates as "File" credentials, which can later on be used in snippets.

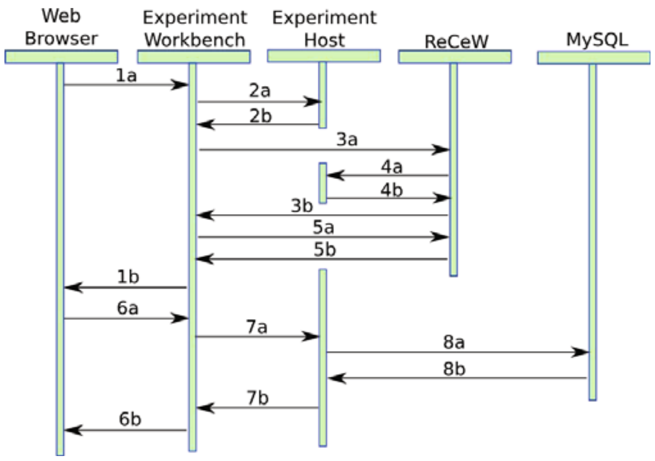


Fig. 5. Sequence diagram for an experiment involving computations based on data retrieved from a MySQL server with the help of user credentials stored in the Remote Wallet (ReCeW)

4 Results

As a result of our research a complete solution described in Section 3 has been developed and deployed. All architectural blocks from Fig. 1 have been implemented and integrated. We thus offer a fully integrated mechanism for scientists, supporting secure yet problem-free development and enactment of experiments. The platform is based on well-known industry standards such as TLS and SSH, ensuring secure access to various types of e-infrastructures.

Basic access to the Workbench is highly manageable, as it does not require complex procedures such as requesting certificates and uploading them to the browser. After completing the unsophisticated and well documented registration process, any PL-Grid user may access GridSpace2 without any additional effort.

By obtaining access to the Workbench the user can use any of the resources described in Section 2, such as:

– computing clusters – instantly, without meeting any additional security criteria,
– grids – after uploading (to the Wallet) credentials appropriate for the middleware being requested, such as GSI Proxy certificates,
– any arbitrary infrastructure (including Clouds) and data source (such as RDBMses) – after uploading appropriate credentials to the Wallet.

A sample diagram representing an experiment which uses data from an external data source (MySQL database) accessed using credentials retrieved from the Remote Wallet (ReCeW) is shown in Fig. 5.

The experiment is executed in the following stages:

– (1a) User logs into the Experiment Workbench using selected UI user and password.

- (2a) EW verifies user credentials (authorizing him/her) by opening an SSH session to the Experiment Host.
- (2b) Session is successfully opened – the user is now authorized.
- (3a) EW contacts ReCeW on behalf of the logged-in user (using user credentials).
- (4a) ReCeW authorizes the user by attempting to log into the Experiment Host via SSH.
- (4b) Authorization to EH succeeds; hence the user is authorized by ReCeW.
- (3b) ReCeW returns the session ID to EW for the authorized user.
- (5a) EW queries user credentials using the appropriate SID.
- (5b) ReCeW returns requested credentials to the EW.
- (1b) All procedures related to user authorization are now complete – control is returned to the user's browser.
- (6a) User requests an experiment to be executed in EW.
- (7a) EW executes the experiment (through the previously established SSH session) on the Experiment Host (previously retrieved MySQL credentials are passed along).
- (8a) EH accesses an external service (MySQL) using credentials provided by the EW.
- (8b) MySQL returns data.
- (7b) EH returns results of the performed computations to the EW.
- (6b) EW presents experiment results by sending them to the user's browser.

The GridSpace2 platform, along with its security framework described in this paper, is already deployed as an official production service in the PL-Grid project and, as such, is available to all users of the project. The whole platform has successfully passed the thorough security audit performed by the Security Center of Pl-Grid project. It has already been used both by scientists for real calculations as well as students attending courses related to distributed/grid systems.

5 Conclusions and Future Work

As shown in the paper, we have managed to create a flexible security solution ensuring access to various e-infrastructures as well as storage of confidential data (such as external credential) using the Wallet mechanism. The GridSpace2 platform is deployed as an official service in PL-Grid and the security mechanism described in this paper is fully integrated with the general security framework of the project. This allows any PL-Grid user to access the infrastructure without additional registration. However, our security system is not tightly bound to the PL-Grid project – it is equally possible to use GridSpace2 with other projects' security frameworks. For example, we presently host a development version of the system integrated with our own internal infrastructure as well as a version geared for another project – MAPPER [17] which uses different computing middleware, such as QosCosGrid [25],[30] and DEISA resources.

In the future we plan to further extend the security components of the platform, including the ReCeW tool. We aim for better integration with Cloud platforms. As already mentioned, it is possible to access Cloud systems such as

Amazon EC2 using credentials stored in the Wallet, but our goal is to provide even better integration, enabling the use of Cloud hosts to host parts of the platform itself (e.g. Cloud-based version of ReCeW using security-as-a-service model).

References

1. Ciepiela, E., Harężlak, D., Kocot, J., Bartyński, T., Kasztelnik, M., Nowakowski, P., Gubała, T., Malawski, M., Bubak, M.: Exploratory programming in the virtual laboratory. In: Proceedings of the International Multiconference on Computer Science and Information Technology, Wisla, Poland, pp. 621–628 (2010)
2. GridSpace technology homepage, http://dice.cyfronet.pl/gridspace
3. The PL-Grid Project, http://www.plgrid.pl/en
4. Meizner, J., Malawski, M., Ciepiela, E., Kasztelnik, M., Harężlak, D., Nowakowski, P., Król, D., Gubała, T., Funika, W., Bubak, M., Mikołajczyk, T., Płaszczak, P., Wilk, K., Assel, M.: ViroLab Security and Virtual Organization Infrastructure. In: Dou, Y., Gruber, R., Joller, J.M. (eds.) APPT 2009. LNCS, vol. 5737, pp. 230–245. Springer, Heidelberg (2009)
5. OpenPBS (original homepage not maintained), http://www.mcs.anl.gov/research/projects/openpbs/
6. TORQUE Resource Manager, http://www.clusterresources.com/pages/products/torque-resource-manager.php
7. PBSPro, http://www.pbsworks.com/
8. Oracle Grid Engine, http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html
9. Univa Grid Engine, http://www.univa.com/products/grid-engine.php
10. European Grid Infrastructure, http://www.egi.eu/
11. Basney, J., Humphrey, M., Welch, V.: The MyProxy online credential repository. Softw., Pract. Exper. 35(9), 801–816 (2005)
12. Meizner, J., Malawski, M., Bubak, M.: Flexible and Secure Access to Computing Clusters. Computer Science, Annual of University of Science and Technology 11, 21–36 (2010)
13. Barton, T., Basney, J., Freeman, T., Scavo, T., Siebenlist, F., Welch, V., Ananthakrishnan, R., Baker, B., Goode, M., Keahey, K.: Identity federation and attribute-based authorization through the globus toolkit, shibboleth, gridshib, and MyProxy. In: 5th Annual PKI R&D Workshop (April 2006)
14. Spence, D., Geddes, N., Jensen, J., Richards, A., Viljoen, M., Martin, A., Dovey, M., Norman, M., Tang, K., Trefethen, A., Wallom, D., Allan, R., Meredith, D.: ShibGrid: Shibboleth access for the UK national grid service, p. 75 (December 2006), http://dx.doi.org/10.1109/E-SCIENCE.2006.261159
15. Distributed European Infrastructure for Supercomputing Applications, http://www.deisa.eu/
16. Uniform Interface to Computing Resources, http://www.unicore.eu/
17. Multiscale Applications on European e-Infrastructures, http://www.mapper-project.eu/
18. CILogon – GridShib, http://www.cilogon.org/gridshib/
19. CILogon – GridShib-CA, http://gridshibca.cilogon.org/
20. GO Teragrid Portal, https://go.teragrid.org/
21. GSI-Enabled OpenSSH, http://grid.ncsa.illinois.edu/ssh/

22. Ganymed SSH-2 for Java, http://www.cleondris.ch/opensource/ssh2/
23. Christian Grothoff, GNU libmicrohttpd, http://www.gnu.org/software/libmicrohttpd/
24. Free Software Foundation, GnuPG – Libraries (libgcrypt), http://www.gnupg.org/related_software/libraries.html
25. Kurowski, K., de Back, W., Dubitzky, W., Gulyás, L., Kampis, G., Mamonski, M., Szemes, G., Swain, M.: Complex System Simulations with QosCosGrid. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part I. LNCS, vol. 5544, pp. 387–396. Springer, Heidelberg (2009)
26. University of Chicago, Overview of the Grid Security Infrastructure, http://www.globus.org/security/overview.html
27. Amazon, Amazon S3 APIs, http://docs.amazonwebservices.com/AmazonS3/latest/API/
28. Ciepiela, E., Nowakowski, P., Kocot, J., Harężlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing Entire Lifecycles of e-Science Applications in the GridSpace2 Virtual Laboratory – From Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)
29. Radecki, M., Szepieniec, T., Szymocha, T., Szopa, M., Krakowian, M.: Towards Professional Service Operations in Grids. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 27–39. Springer, Heidelberg (2012)
30. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)

SARA – System for Inventory and Static Security Control in a Grid Infrastructure

Gerard Frankowski and Michał Rzepka

Poznań Supercomputing and Networking Center,
Institute of Bioorganic Chemistry of the Polish Academy of Sciences,
ul. Z. Noskowskiego 12/14, 61-704 Poznań, Poland
{gerard.frankowski,michal.rzepka}@man.poznan.pl
http://www.pcss.pl

Abstract. IT security, even if once achieved, is not a permanent state but rather a process. One of the main factors that impact this process is the ability to identify security vulnerabilities in the software. Disclosing such a flaw is usually followed by issuing a patch. However, for maintainers of a heterogeneous and compound environment, being up to date with all necessary fixes, may be an unexpectedly difficult task. Developing custom software in a grid project introduces another dimension to this problem. The SARA system for static security control has been developed to help the administrators with that issue.

Keywords: IT security, attacks, vulnerabilities, security measure, security standards, CVE, CPE, CVSS, NVD, SARA.

1 Introduction

Security is considered to be a process rather than a permanent state [1], as computer systems undergo numerous changes: new software is installed (or new versions of already used applications), new user accounts are created, etc. Their environment is inconstant as well – previously unknown methods of network attacks are invented or disclosed and new software vulnerabilities are found. In case of distributed and heterogeneous environments, maintaining an appropriate security level is especially significant, since one weak link in the chain may give unauthorized access to resources and data to an attacker. Dynamic environments, like grids, make this task more difficult, as these infrastructures may change over time. When new clusters or single computational nodes are temporarily attached to the infrastructure, they may not always be consistent with general security systems or rules applied to the grid.

There are also other fundamental problems to secure such an infrastructure – like the fact that most of the grid communities use non-proprietary, specific software. Usually, these are applications like batch systems, grid monitoring software, and resource discovery tools. Many of those are created by participants of research and development (R&D) projects, and are not supported by quality assurance departments like in the case of large IT vendors. This makes it difficult to find and apply a general, automated security system to a grid environment. Taking into account the grid's dynamic nature, this should be a system for continuous monitoring with respect to individual security of each of the infrastructure's subsystems. Additionally, the monitoring system should make use of the information about known (publicly disclosed) security vulnerabilities that may threaten the environment. However, the final difficulty is that security is hard to be quantified. There are several standards for assessing single vulnerabilities, but it is an overwhelming task to measure the overall infrastructure security level, especially with respect to large and distributed networks.

2 State-of-the-Art

2.1 Standards and Repositories

As stated in the previous section, the problem of accurate, formal and consistent description of the actual security state of an infrastructure is very challenging. A security report containing such a description, should be appropriately structured, but easy to understand (also by non-security specialists) and exchange (e.g. between different organizations within the grid). There are several standards and repositories that may support this task – usually relatively new or still under construction – like Common Vulnerabilities and Exposures (CVE [2]) repository, Common Platform Enumeration (CPE [3]), Common Vulnerability Scoring System(CVSS [4]) or National Vulnerability Database (NVD [5]). The aims of particular standards can be different, but grouped together, they provide a consistent way to describe the security of software used in a specific environment.

CVE – Common Vulnerabilities and Exposures. CVE is one of the most widely recognized repositories of security vulnerabilities. It is assumed to be a dictionary of publicly known information about security vulnerabilities and exposures. CVE's common identifiers enable data exchange between security products and provide a baseline index point for evaluating security coverage of tools and services. A "vulnerability" is here assumed to be a software security flaw that allows an attacker to gain direct access to the system or network being attacked, while an "exposure" means a security weakness that results in revealing sensitive information which might be used for further attacks. Before a CVE entry is officially confirmed (after being thoroughly investigated and assessed), it is first categorized as a "candidate".

CVE provides a broad enumeration of publicly known security weaknesses of the software produced by worldwide vendors, in an understandable, uniform manner. Currently, there are over 46 thousand entries gathered in the CVE database. It grows substantially, although the number of security flaws disclosed every year started to descend – possibly due to increased security awareness amongst the designers and developers of the software. Fig. 1 [6] shows the number of discovered security flaws in recent years, making it clear that their multitude makes maintaining a desired security level of a complex software set a very hard task.

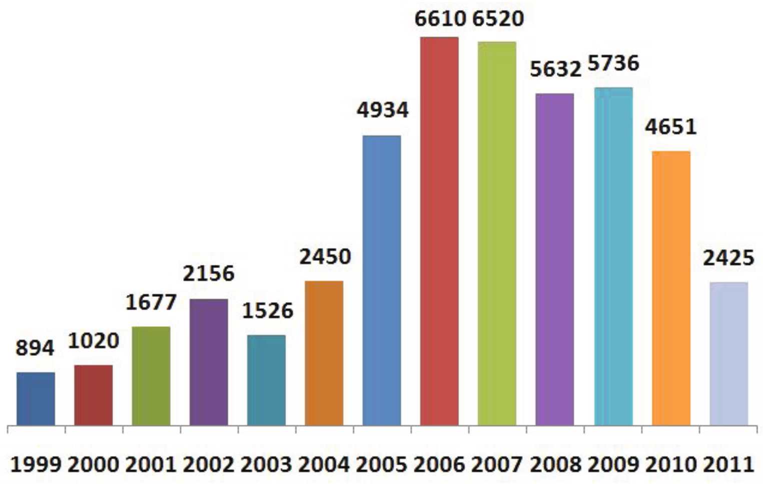


Fig. 1. Number of discovered CVE vulnerabilities by year (based on [6])

CPE – Common Platform Enumeration. The information about all existing vulnerabilities, even if properly structured, is not sufficient for managing an IT infrastructure. The vulnerabilities must be suitably identified for the software actually installed on systems belonging to the infrastructure. Therefore, it is necessary to have a specification that would facilitate building a consistent, aggregated description of this software.

CPE provides sufficient means for that purpose, proposing a structured naming scheme based on the URI (Uniform Resource Identifiers) syntax for known IT systems and platforms. The CPE specification consists of a formalized format of names, a language for providing descriptions of complex platforms (CPE Language), a method for checking assignments of names to particular systems, and, finally, a format describing the assignment of text and tests to a CPE name.

A CPE instance is represented by an URI constructed according to the following scheme:

```
cpe:/{part}:{vendor}:{product}:{version}:{update}:{edition}:{language}
```

The rules of building a CPE URI are more strict for some components – e.g. for the “part” name, there are three defined values: “a” for application, “h” for hardware and “o” for operating system – and less strict for others – like the “product” component, where the most widespread and recognizable product name should be used. More details on building CPE names may be found in [7].

CVSS – Common Vulnerability Scoring System. In an IT infrastructure, there may be a number of security vulnerabilities. As it is not possible to address all of them simultaneously, they should be prioritized to emphasize those that involve the greatest threat. CVSS is a platform for describing characteristics of vulnerabilities and their potential impact on networks/systems. It provides a common language for assessing the severity of vulnerabilities that may exist in an IT infrastructure. CVSS uses three compound metrics (Base, Temporal and Environmental) and takes into account factors like: access complexity, potential impact, ease of exploitability, or whether authentication is necessary, etc. The CVSS metrics are presented in Fig. 2.

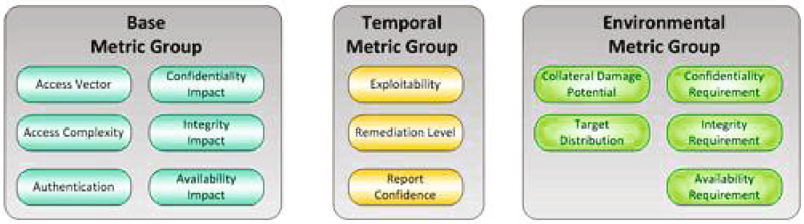


Fig. 2. Common Vulnerability Scoring System metrics (source: [8])

The severity of vulnerabilities may be evaluated by assigning 0 to 10 points for each metric to them. The final score is computed using a chosen algorithm. In order to facilitate the computations, several CVSS calculators are accessible, like [9]. They provide a set of drop down lists with values to be assigned (like "Low", "Medium", "High", etc.) and their additional explanations.

NVD – National Vulnerability Database. The three sets of information described in the previous paragraphs, can be combined in order to create a full description of the software security. NVD is a repository provided for the purpose of vulnerability management. It integrates a general description of the vulnerability (taken from the CVE database), with list of the affected software (based on an appropriate CPE entry) and the severity level (assessed with the help of the CVSS score).

NVD is a product of the National Institute of Standards and Technology (NIST), Computer Security Division and is sponsored by the Department of Homeland Security's National Cyber Security Division. Besides the CVE entries, it contains over 30 thousand CPE names, almost 200 security checklists, and about 6 thousand OVAL (Open Vulnerability and Assessment Language) entries.

2.2 Systems and Software

Measuring Security in Static and Dynamic Manner. To measure the infrastructure security, data concerning the software installed on it must be gathered and, when necessary, updated. There are two approaches to this operation: static and dynamic security control. The static approach bases on manual updates of the database of the software installed on the infrastructure, while the dynamic approach requires presence of a set of agents (or plug-ins) – tiny client applications, installed on each system that should be protected. The goal of an agent may be e.g. to collect information about the software that is utilized by the users. In a particular deployment, it may be easier to apply either static or dynamic approach, but usually a combination of both assures the best results.

Review of Systems for Monitoring and Controlling Security. One of the most widely known tools for monitoring systems and networks is Nagios [10]. It is a system for dynamic monitoring, actively developed by a wide community since 1999. Nagios uses monitoring applications (plugins, agents) that may be customized to a particular operating system or application, to perform specific tasks. Nagios plugins are available for different operating systems, including Windows. The software has become an industry-standard in IT infrastructure monitoring (for the dynamic approach). Nagios is an Open Source software, already used in PL-Grid to monitor whether the infrastructure is functioning properly.

Another tool, closer to the software version controlling approach, is Pakiti [11]. It is a monitoring tool that controls the patching status of an infrastructure. Installed on a client host, Pakiti periodically sends a list of installed packages to the Pakiti Server. The server compares the versions with versions obtained from package repository OVAL definitions from MITRE (Massachusetts Institute of Technology Research Establishment). The client is informed which packages should be upgraded, with emphasis on security updates. The Pakiti software has been used in many R&D projects, e.g. EGEE. From our point of view it has two main drawbacks. Firstly, it has been prepared only for Linux distributions that use package repositories based on rpm (yum) or dpkg (apt). Although this is not an issue for the PL-Grid infrastructure, we have intended to prepare more general solution. The second disadvantage (less important comparing to the previous one) is the need to affect the client installation, though only with a tiny bash script.

An interesting solution for monitoring of the patching status is the Secunia Personal Software Inspector [12]. The software is free. After installation it is able to automatically detect several dozens of popular software installed on the infrastructure – including operating system, browser, and a particular set of applications. The scan results are provided in a user-friendly manner, with additional suggestions concerning necessary updates. What is more, Secunia PSI does not merely suggest a newer version – it analyzes whether a newer version of the software addresses known security vulnerabilities found in the currently installed version [13]. Although the approach is optimal, the tool is available only for certain versions of Microsoft Windows. Therefore, it cannot be used for PL-Grid servers and nodes.

Finally, we analyzed also the Decision Aid Tool (DAT). This comprehensive tool has been implemented within the INSPIRE (INcreasing Security and Protection through Infrastructure REsilience) project [14]. The system applies an ontology-based approach to secure SCADA infrastructures. One of the goals of the system is to enumerate potential threats on the base of the current set of resources [15] – e.g. by detecting versions of the installed software packages and verifying them against the contents of vulnerability repositories, including NVD.

However, DAT appears too complex to be directly applied, especially as detecting known vulnerabilities based on the software versions is only a small part of its functionality.

Summary. None of the solutions described above fully meet our needs and expectations, especially in terms of providing comprehensive security information with only minimum interference with the working infrastructure. We have intentionally chosen a static approach to controlling and measuring security, as we prefer an approach where no additional software updates are necessary on the monitored systems. Obviously, this approach requires additional effort for entering and maintaining inventory information, but does not require e.g. preparing new, custom agents for certain operating systems (like those installed on network hardware: in similar cases, often no custom software may be installed due to licensing issues, while security vulnerabilities are also likely to occur there, and patches for these systems are also issued). Additionally, there are already dynamic systems used for security control in PL-Grid, like ACARM [16]; therefore, a static system would be a suitable complement for them, providing protection on another level.

For the abovementioned reasons, we proposed a new security system, developed from scratch. We intend it to be a basis for a comprehensive security toolset for securing large internal infrastructures (like R&D project deployments, large organizations using custom software, etc.).

3 Solution – The SARA System

3.1 System Overview

SARA (System for Automated Reporting and Administration) combines information on vulnerabilities stored (with the help of the standards mentioned in previous section) in the National Vulnerability Database.

The system has been implemented as a Web application in the web2py environment. Web2py is a Python-based framework for rapid development of Web applications conforming to the MVC (Model – View – Controller) paradigm. SARA is deployed on a web2py application server (the solution is currently provided as web2py environment with SARA integrated into it). The environment supports built-in access control mechanisms based on the RBAC (Role Based Access Control) model. Two basic groups of its users have been defined: administrators and supervisors – who take care of the infrastructure as a whole. The system uses a PostgreSQL database.

The main system functionalities are:

- downloading information about vulnerabilities and storing them in a local database,
- automatic updates of the local vulnerability database,
- storing information about the whole infrastructure (including the software installed on it),

- performing static security assessments (by combining infrastructure information and data taken from NVD),
- generating reports (about required updates and assessments performed) and sending them to respective responsible people [7].

In its most basic form, SARA works as it may be seen in Fig. 3. Firstly, it downloads information about the most recent security vulnerabilities from NVD and processes it, storing in the local database. This action is repeated periodically with use of a cron-like solution. The second key source of information required to assess the security of the infrastructure is a list of the software used within it, that must be prepared by the administrators (who use CPE names for particular applications installed on the servers). Having this information, SARA uses the CPE Language Matching algorithm (that is a part of CPE specification) to build a security assessment. The algorithm verifies CPE entries for each controlled system and compares them with appropriate records from the NVD database. Both updating the NVD records and the assessment procedure may be configured by the SARA administrator (a supervisor). When the security assessment is ready, appropriate reports on the discovered vulnerabilities are generated and sent via an e-mail to the supervisor (all reports) and to system administrators (reports concerning their particular servers) [17].

The system consists of four main components: a downloading module, security assessment module, report generation module and notification module. In addition, an inventory interface is necessary – as a static system, SARA requires a certain amount of initial work of an administrator who has to fill the local database with information on their systems (although it may be decided,

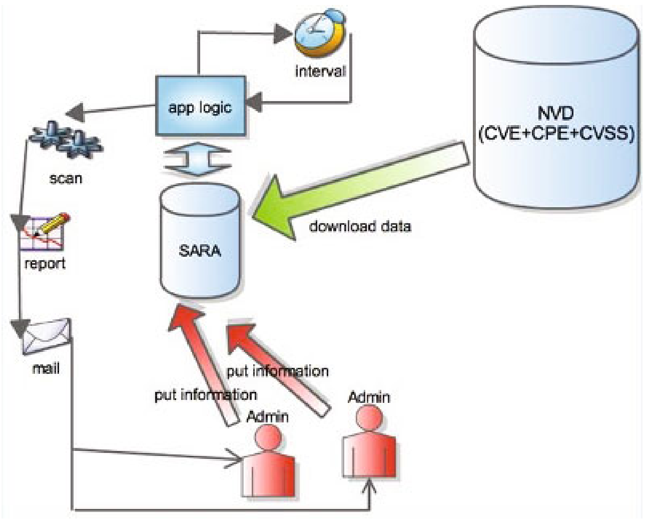


Fig. 3. SARA – the basic algorithm of the SARA functioning

according to the local security rules, that not all the maintained applications must be entered into the system). This overhead will be minimized in the future versions of the system; however, it will never be completely avoided, especially in environments containing heterogeneous, uncommon or outdated systems. The SARA system is also an excellent tool for an inventory – it allows to store information about physical location and the responsibility of each of the systems. In contrast to the aforementioned dynamic systems, like Pakiti or Nagios, SARA works not only for servers, but also for network devices, like routers, switches and others [7].

Being a static security control system, SARA was not intended to replace, but rather complement the dynamic security control solutions, thus, realizing security in depth principle.

3.2 Vulnerability Database Problems and SARA Proxy

The NVD repository is a large XML file (though available also in other formats), that often suffers from encoding or formatting errors caused by the vendor side. During the work with the system, a number of unexpected parsing errors were encountered, which caused SARA not to be able to use the most recent version of the repository.

To solve this problem, a separate module called "proxy" was provided. Its main part is an XML validator. Apart from that, the module has the following functionalities (provided in an automatic or semi-automatic way):

– assuring access for multiple client databases,
– copying vulnerability information between different databases,
– possibility of applying online manual corrections to the XML content in case of an unexpected error that cannot be handled by the validator,
– distribution of vulnerability information.

The last improvement is especially important, as the main vulnerability database may be installed either on the client side or in a central location. Therefore, a user may select an optimal approach, optimizing either storage facilities or network bandwidth requirements. Fig. 4 shows the idea of the SARA proxy module.

The data transmitted between the proxy module and SARA is no longer sent in XML format, but via direct database transfer, which improves the information exchange efficiency. The proxy module is implemented in Python, and uses XML parsers built into that programming language.

Apart from the NVD repository, SARA is able to use a custom database (which may be a subset of the NVD file). An important factor related to this facility is the ability to provide an additional filtering layer that would allow to extract only entries concerning the installed software components from the whole NVD content. Avoiding the processing of the whole NVD is another advantage of SARA that improves the system efficiency.

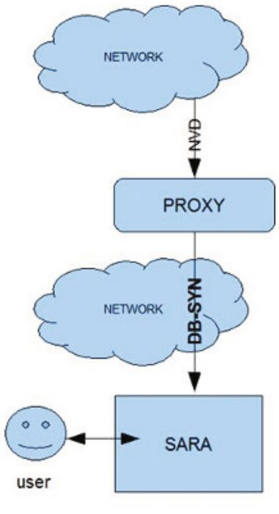
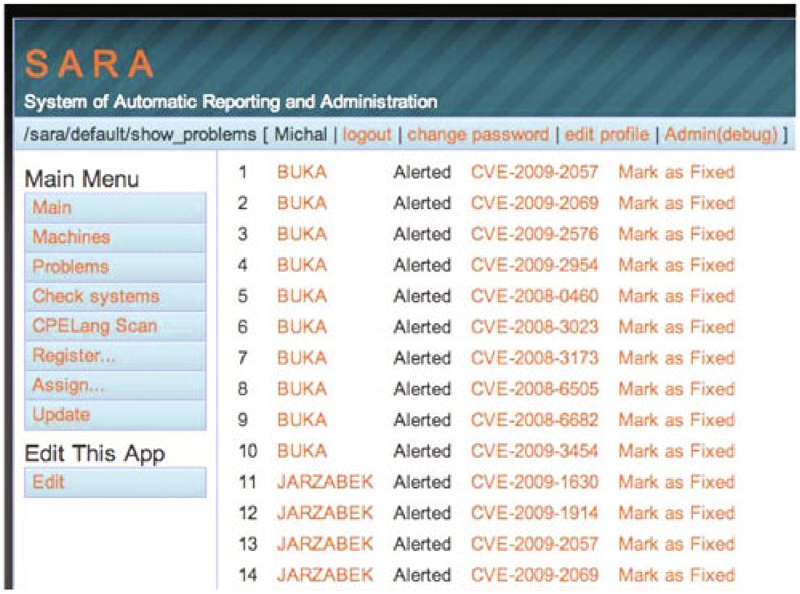


Fig. 4. The idea of the proxy module in SARA

4 Results

The current version of SARA was accepted as an optional solution for single clusters within a national R&D project – PL-Grid. The main assumptions of


SARA
System of Automatic Reporting and Administration
/sara/default/show_problems [Michal | logout | change password | edit profile | Admin(debug)]
Main Menu
Main
Machines
Problems
Check systems
CPELang Scan
Register...
Assign...
Update
Edit This App
Edit

1	BUKA	Alerted	CVE-2009-2057	Mark as Fixed
2	BUKA	Alerted	CVE-2009-2069	Mark as Fixed
3	BUKA	Alerted	CVE-2009-2576	Mark as Fixed
4	BUKA	Alerted	CVE-2009-2954	Mark as Fixed
5	BUKA	Alerted	CVE-2008-0460	Mark as Fixed
6	BUKA	Alerted	CVE-2008-3023	Mark as Fixed
7	BUKA	Alerted	CVE-2008-3173	Mark as Fixed
8	BUKA	Alerted	CVE-2008-6505	Mark as Fixed
9	BUKA	Alerted	CVE-2008-6682	Mark as Fixed
10	BUKA	Alerted	CVE-2009-3454	Mark as Fixed
11	JARZABEK	Alerted	CVE-2009-1630	Mark as Fixed
12	JARZABEK	Alerted	CVE-2009-1914	Mark as Fixed
13	JARZABEK	Alerted	CVE-2009-2057	Mark as Fixed
14	JARZABEK	Alerted	CVE-2009-2069	Mark as Fixed

Fig. 5. SARA – a list of security vulnerabilities within an infrastructure

the system and current state of work were presented during Cracow Grid Workshop 2010 [7] and the i3 2010 Conference in Wrocław [17]. A proof-of-concept system was presented within a workshop for system administrators during the PL-Grid project meeting in March 2011. The first feedback from potential users was positive; however, since the application is in its early stage of development, identification of some drawbacks was unavoidable. Fig. 5 shows a sample screenshot of the SARA interface.

Currently, the system is undergoing intensive functional and security tests. In addition, a list of change requests was gathered internally and from the system administrators, during the project workshop. On this base new features are implemented – e.g. the possibility of importing CPE names from a text file. The implemented changes and improvements will be included into initial PL-Grid deployments in Poznań Supercomputing and Networking Center and Wrocław Centre for Networking and Supercomputing.

It should be noted, however, that the current version of SARA is assumed to be a proof-of-concept tool. The resources that could be assigned to this task within the PL-Grid project – in the context of all other security tasks mentioned within this book in [18] – were not sufficient to build a full version of the system.

5 Conclusions and Future Work

The SARA system leverages on a static approach to the problem of correlation of known security vulnerabilities with the contents of the infrastructure system configurations. It complements dynamic systems like Nagios or Pakiti, especially for nodes where agents required by these tools cannot be installed. Other deployments that are planned, include securing virtual laboratories prepared within the WLIN project [19], [20] and the Integrated IT Platform for Polish Police.

In future versions of the SARA system, the main issues extending the functionality and usability of the software are planned to be addressed. Currently, SARA uses public repository of vulnerabilities, but it also offers facilities to attach custom repositories. Therefore, SARA is an ideal solution to support static security control in research infrastructures and corporate or governmental networks (where at least a certain part of software vulnerabilities may not or must not be disclosed publicly). It is also planned to propose further SARA deployments together with support for initial populating of the software repositories and suitable trainings.

To improve usability, it is planned e.g. to apply a CPE Dictionary (a set of the most popular CPE names) to speed up the process of populating the infrastructure database and decrease the number of accidental errors. System templates are also going to be introduced, in order to enable one-click cluster definition. In the future (long-term plans), the inventory functionality is planned to be further extended by restructuring the application logic into a set of independent, loosely coupled services – according to Service Oriented Architecture (SOA) principles.

The changes described above will transform the current version of SARA into a universal solution that allows to support dynamic monitoring systems within heterogeneous networks, minimizing impact of the human factor. The starting point for such a solution would be the deployment of SARA with support for initial populating of the repository with the inventory information. The above would be then supported on a regular basis with security audits of custom software (e.g. produced within an R&D project, or dedicated for the needs of a particular user – thus, without publicly available NVD entries) and adding the information about the discovered vulnerabilities to the repository. Finally, a board of experts would, independently from penetration testers, assess the threat level assigned to a particular vulnerability in the context of a specific environment of the infrastructure. The threat level would also be added to the repository, thus assuring the future security assessments to be more consistent.

The approach described above could actually be applied to every R&D project, assuring the necessary protection against network attacks. Advantages of an infrastructure protected in such a way could be summarized as follows:

– assuring an initial, basic security level (information on software that needs patching),
– inventory facilities minimizing the opportunity of an accidental error,
– possibility of controlling the software that is developed only within a particular infrastructure, including specification of a consistent description of a security vulnerability that does not have to be publicly disclosed;
– support for applying CVSS metrics to determine the severity level of vulnerabilities found during internal security reviews of the software, which will decrease inconsistencies between assessments prepared by different security teams;
– ability to customize deployment for a particular network or project, by assuming only a particular subset of NVD entries (according to the software used within the infrastructure).

Technically, the final version of SARA will be implemented according to the SOA paradigm. This will facilitate adjusting the deployment to the needs of a particular user. SARA should not be then treated merely as software, as the SOA paradigm assumes that the business facet is the key aspect of using the application. This means that SOA may be described not as a technology, but rather as an approach to using other technologies, adjusted to the current business use case. Therefore, we intend to create a whole ecosystem in which the software (SARA with additional modules) is the core, but where there is also a board of security experts and penetration testers equipped with custom repositories, and, finally, the users.

References

1. Schneier, B.: Crypto-gram newsletter, `http://www.schneier.com/crypto-gram-0005.html`
2. Common vulnerabilities and exposures, `http://cve.mitre.org`
3. Common platform enumeration, `http://cpe.mitre.org`

4. Common vulnerabilities scoring system, http://www.first.org/cvss
5. National vulnerability database, http://nvd.nist.gov
6. Cve details – browse vulnerabilities by date, http://www.cvedetails.com/browse-by-date.php
7. Rzepka, M.: An approach to monitoring grids with system of automatic reporting and administration (SARA). In: CGW 2010 Conference (October 2010)
8. A complete guide to the common vulnerability scoring system version 2.0, http://www.first.org/cvss/cvss-guide.html
9. Cvss version 2 calculator, http://nvd.nist.gov/cvss.cfm
10. Nagios monitoring system, http://www.nagios.org
11. Patching status monitoring tool pakiti, http://pakiti.sourceforge.net
12. Secunia personal software inspector (psi), http://secunia.com/vulnerability_scanning/personal
13. Secunia psi 2.0 – setup and usage guide, http://secunia.com/gfx/pdf/SecuniaPSI2.0-Setupandusageguide.pdf
14. Inspire project web page, http://www.inspire-strep.eu
15. Choraś, M., Flizikowski, A., Kozik, R., Hołubowicz, W.: Decision Aid Tool and Ontology-Based Reasoning for Critical Infrastructure Vulnerabilities and Threats Analysis. In: Rome, E., Bloomfield, R. (eds.) CRITIS 2009. LNCS, vol. 6027, pp. 98–110. Springer, Heidelberg (2010)
16. Balcerek, B., Szurgot, B., Uchroński, M., Waga, W.: ACARM-ng: Next Generation Correlation Framework. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 114–127. Springer, Heidelberg (2012)
17. Rzepka, M.: Monitorowanie bezpieczeństwa złożonych infrastruktur przy pomocy systemu SARA, i3 2010 Conference (December 2010), http://www.i3conference.net/online/2010/prezentacje/58.pdf
18. Balcerek, B., Frankowski, G., Kwiecień, A., Smutnicki, A., Teodorczyk, M.: Security Best Practices: Applying Defense-in-Depth Strategy to Protect the NGI_PL. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 128–141. Springer, Heidelberg (2012)
19. Adamski, M., Frankowski, G., Jerzak, M., Stoklosa, D., Rzepka, M.: Defense in depth strategy – a use case scenario of securing a virtual laboratory. In: Davoli, F., Lawenda, M., Meyer, N., Pugliese, R., Weglarz, J., Zappatore, S. (eds.) Remote Instrumentation for eScience and Related Aspects (2012)
20. Virtual laboratory of interactive learning (wlin) project, http://www.wlin.pl

ACARM-ng: Next Generation Correlation Framework

Bartłomiej Balcerek, Bartosz Szurgot,
Mariusz Uchroński, and Wojciech Waga

Wrocław University of Technology, WCSS,
Wrocław, Poland
acarm@kdm.wcss.wroc.pl
http://www.wcss.pl

Abstract. ACARM-ng is an extensible, plug-in-based alert correlation framework. It introduces abstractions over correlation, reporting, reaction, gathering data from multiple sources and data storage. ACARM-ng supports real-time reporting, meaning that alerts can be reported while still being correlated. For an administrator, a Web User Interface is provided, to present gathered and correlated data in a consistent way. The system makes use of multi-core architectures and is written in C++.

Keywords: correlation, alerts, IDMEF, framework, IDS, IPS.

1 Introduction

Although computer networks have been ubiquitous for more than a decade, there is still no ultimate solution to ensure proper security within them. In October 2010 William J. Lynn III, the United States Deputy Secretary of Defense, claimed that there were more than 100 foreign intelligence organizations trying to hack into the digital networks that undergird U.S. military operations [3]. After this, the security of government networks came to the fore once again as a hot topic.

When we speak about security, we usually mean protecting our data against falling into wrong hands or protecting our infrastructure against exploitation. The problem is that both of these can remain undetected for a long time. It is clear that there is a need for a tool that can facilitate, or even make possible, accurate and rapid detection of intrusion.

There are two major approaches to intrusion detection: anomaly [11] and misuse [9] detection. While the first is based on statistics and attempts to find some kind of deviation from what is defined as normal behavior, the second works by pattern matching and detects strictly defined attack scenarios. As both of them have their weak and strong points, neither is a perfect solution. What is common for the two, is that for a large network one can get a number of reports that cannot be processed in real time. The best solution to this problem is to use a correlation engine which is able to merge similar events together, as well as to discard irrelevant data.

This paper presents ACARM-ng [2] – a real-world implementation of an extensible framework for alert correlation. Some effects of its operation in network environment at the Wrocław Center for Networking and Supercomputing (WCSS) [23] are presented as well.

2 Intrusion Detection

The event correlation problem is widely present throughout the literature [5], [6], [12]. Apart from theoretical studies, several implementations have been proposed, different in concept of processing, scope of interest and correlation handling. Some of them are described in this section.

2.1 Event Correlation

Among numerous Intrusion Detection Systems (IDS) available [17], [20], many do not perform correlation, but use rules for detection of specific situations instead. They base on single reports, like incorrect login attempts [20] or appearance of a specific entry in configuration files [18].

Since intrusion detection is a wide topic, describing all the existing tools and approaches is beyond the scope of this paper. Therefore, the following subsections describe only the most important, open-source solutions available.

Prelude Correlator. The first example of a correlation engine described here is Prelude Correlator [17], which ships with Prelude-Manager [17]. For the sake of flexibility, Python scripting language is used within it, instead of a one that needs compilation, like C or C++. Although writing new rules for this tool is pretty straightforward, the performance that comes with this scripting language is poor, and its memory footprint can be large.

Prelude-Correlator consists of a set of simple rules of correlation, all launched upon arrival of a new alert. When a required number of alerts is gathered (usually 5), they are reported as a correlated entry. Such an approach is insufficient for several reasons. First of all, important alerts that are fewer in number than the threshold, will not be reported at all. Secondly, the usual case is to have many similar alerts, which do not tell the administrator anything new. A typical example is login information, which in our installation appears (tens of) thousand times a day. Receiving $10k$ alerts instead of $50k$ is an improvement, but the overall benefit is still insufficient.

OSSEC. OSSEC [14] is a “Swiss Army knife” of host-based intrusion detection. It is able not only to monitor system logs, but also to check files integrity or to search for rootkits. OSSEC is also capable of taking immediate actions in response to a predefined malicious activity. What is missing in it, is the possibility to monitor more than one machine and correlating alerts between networked computers.

OSSIM. The most advanced system found by the authors at the time of writing this paper is OSSIM [15]. It is an open source counterpart of AlienVault's commercially available Unified SIEM. It is a complex SIM system, that includes an event correlation engine and a Web interface for data visualization. A rarely found feature of the included correlator is recurrent processing. This means that the effects of the correlation re-enter the system to be processed once more in case they can become a part of another alert. This can further reduce the number of events reported to the administrator.

Unfortunately, in spite of OSSIM's great popularity, it is still poorly documented. Another disadvantage of the system is that although OSSIM launches new threads at the start-up, the correlation is done only in a single thread. This is probably why AlienVault, the company behind OSSIM, describes its performance only as *moderate*. OSSIM is also lacking a logger which would save information about system crashes or its proper operation.

ACARM. A correlation engine that has been widely used at WCSS is ACARM [1]. It is based on the concept described in [6], which is devoted exclusively to efficient alert correlation techniques.

ACARM's alert life-cycle starts when a new alert is collected by an input module. After being tested against discrimination rules in *prefilter*, the alert is processed by a fixed chain of filters. Because a notification can only be sent when all processing steps are finished, it can take more than ten minutes to notify the user.

ACARM, in contrast to the aforementioned examples, is written in Java and has a fixed architecture. It is not possible to add new correlation rules, remove them or even change the order of processing. Apart from this, ACARM uses a very inefficient framework for creating network protocols, namely BEEP [19], which is a huge performance issue. As a result, ACARM was withdrawn in the WCSS and the project was discontinued.

2.2 Outcome

During the preparatory phase of the PL-Grid project it was decided that there is a need for a fast correlation engine that would protect the whole grid environment against hostilities. After some debates and performance measurements, three main features that would be required from the new system were suggested, namely: multi-threaded architecture, native language implementation and native Prelude protocol. Since ACARM was created without taking those into account, a new project called ACARM-ng (ACARM – next generation) was launched.

3 Proposed Framework

Having in mind the limitations of the existing solutions, a new framework for correlation of events has been proposed. This section describes in detail the architecture of the proposed system.

3.1 Main Goals

The conceptual foundations of the new architecture were derived mostly from the experience gained during the development and use of the first version of the ACARM system, as well as from the evaluation of Prelude-Correlator [17] IDS at WCSS [23]. They are described in the following sections, along with a rationale for their importance. Since real-world tests were performed on the "first" ACARM and Prelude-Correlator, we will focus mostly on these two engines when explaining particular goals.

Generic Framework. One of the main issues with the "first" ACARM was the lack of a generic approach towards adding new correlation techniques – there was no easy way of extending the existing implementation with new algorithms, due to the system's closed design. It has been decided to overcome this by designing the whole system as a framework with correlation methods provided as plug-ins.

Efficiency. During a pilot usage in production environment, both ACARM and Prelude-Correlator proved to be inefficient in at least some cases.

First of all, ACARM had a bottleneck of the "BEEP" protocol's throughput that limited the number of processed alerts to just a few per second. With its lack of parallelism, after being flooded with thousands of alerts, ACARM could freeze for hours, especially after a few days' break. Because of this, we decided to put more effort on efficiency of the new incarnation of the system, considering both its design and implementation.

Asynchronous Reporting. Another important goal for ACARM-ng was to remove the time gap between an alert's arrival/processing and its reporting. Both tested systems [1], [17] required an alert to go through the whole path of correlation before deciding whether it was to be reported, delaying even very important alerts.

In order to have long time windows for analysis, the correlation must be separate from the reporting mechanism. To achieve this, reporting must be asynchronous, so that we do not have to wait for the correlation to finish before reporting the issue.

Surprisingly, though there seems to be a fundamental need for it, this feature has not been addressed by any Open Source correlation engine known to the authors [14], [17], except for OSSIM [15].

Flexible Reaction. To provide as much customization as possible, not only the data processing mechanisms of the system must be flexible, but also its reactions to alerts and reporting. Depending on the situation and configuration, different actions may be acceptable – e.g. we can either notify the user or perform a fully-automated action.

Friendly User's Interface. The ability to watch the working system and statistics concerning it, is a must-have to any system administrator. It is worth to note that although Prelude-Correlator [17] has a Web User Interface (called Prewikka – it is an HTTP server with WUI written in Python), the tool is, unfortunately, slow beyond the bounds of acceptable usability. Some pages take as long as two minutes to load, and sometimes do not load at all (timeouts). The system suffers also from stability issues – i.e. server crashes under heavy load. ACARM-ng comes with a handy Web User Interface, which enables the user to monitor, not only if the system is fully operational, but also the outcome of the correlation and the number of incoming events.

Scalability. For a few years it has been observed that single-core performance does not increase much over time. However, overall performance of the whole processor is rising because of multi-core architectures. Today, desktop computers have from 2 to 4 cores in standard, while it is possible to buy a stock CPU with as many as 12 cores [4] with even more cores expected in the nearest future [22].

As the processing model moves from sequential (single-core) to parallel (multi-core), some extra design effort must be made in order to utilize the new hardware effectively [22]. In order to achieve good scalability on modern hardware, ACARM-ng from the beginning was designed as a multi-threaded piece of software.

Database Independence. It is common to keep a large amount of data in databases. There are multiple engines available and some are better suited to a specific kind of applications than others. Therefore, it is desirable to separate the data storage method (persistent storage) from its implementation.

Generic Input. The correlation engine, by design, should be able to gather data (alerts) from different sources. To cover as many different systems as possible, it has been decided to use Prelude-Manager [17] as an input source, since it supports many different devices and software packages, and can provide output in a consistent form.

Nevertheless, the authors of the new architecture decided not to limit implementation to a single source, and, therefore, to build a powerful plug-in mechanism that allows data to be read from virtually any sensor.

Recursiveness. In order to extract as many relations as possible from a given data set, recursive data processing is preferred over linear pass-through, as used in the first ACARM [6]. Recursive processing is fairly common among many correlation engines [14], [15], however, not all of them use this mechanism [17].

Fault Tolerance. For security-related tools, reliability is a great concern. With regards to the correlation engine, fault tolerance can be understood as a secure handling of incorrect data (i.e. range checks, pointer validity, etc.) or recovery from hard errors like system crash or power loss.

Since incorrect data can usually be dealt with, hardware errors are of greater importance, and we will focus on them exclusively. When a hardware problem occurs, its effect is usually killing all the user-space processes unexpectedly. This issue has been addressed in ACARM-ng by saving the current state of the system in a database. After the hardware problem is fixed, ACARM-ng can be re-run from the point at which it was stopped.

Long Correlation Windows. Making the correlation window short, makes it impossible to correlate issues that are spread over time, and the attacker is able to bypass the system by simply waiting for a while before performing certain steps. Therefore, it is highly desirable to have as long a correlation window as possible and/or required.

3.2 Architecture Overview

To fulfill the requirements defined in Section 3.1, a parallel queue processing mechanism has been proposed. The main part of the system is a FIFO queue of events (meta-alerts in ACARM-ng's terminology). All other parts of the system have been logically split into the following categories:

- filters – data processing abstraction,
- triggers – notification and reaction abstraction,
- inputs – data collecting abstraction,
- persistency – data storage abstraction.

Each category has a generic interface so that user-provided implementations can be loaded as plug-ins. All categories are described shortly in separate subsections.

All data enters the system via the inputs. Every new element is added to the processing queue and, then, passed to all triggers and running filters, each of them having a separate thread of execution. Filters may change the processed elements as well as correlate – to form new ones. The elements that are newly created and the elements changed by filters are added back to the main queue to be processed again. This implements recursive processing (see Section 3.1).

At the same time, meta-alerts can also be reported with triggers. This implements the relation between correlation and reporting – see Section 3.1.

An overview of data processing in ACARM-ng is presented in Fig. 1. Note, that there are dedicated queues for each thread (filters and triggers), to allow independent, non-blocking processing.

It should also be noted, that, since reporting and correlation are separate and independent processes, it is possible (and natural) to use long observing time windows. While other correlation engines [14], [15], [17] require windows of at most few minutes, ACARM-ng can use time windows for correlation with duration of hours, still keeping the administrator up-to-date on what is happening within the whole system (see Section 3.1).

As seen in Fig. 1, alerts enter the system via inputs and are stored in the main system's queue. Each meta-alert entering that queue is sent to every filter and trigger (or simply, processor) registered in the system. Triggers can react to events by performing actions, and can interact with outside world. Filters, on the other hand, process meta-alerts, correlate them, reprioritize them and write

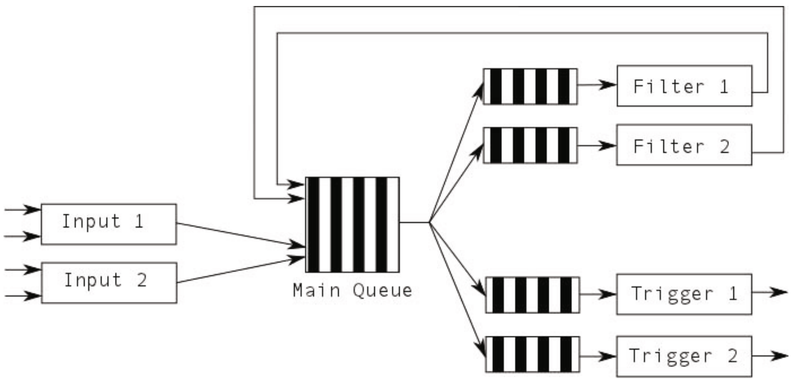


Fig. 1. Data (i.e. meta-alerts) processing flow overview in ACARM-ng. Alerts are gathered by the inputs and inserted to the main processing queue. From there, events are distributed to all filters and triggers. Additionally the filters can correlate new events, that are added back to the queue, while triggers can report them to the external systems.

new data to them. If there is any change, the alert automatically goes back to the main processing queue for further analyses by other filters.

The system automatically saves its state after every operation, making it possible to observe the correlation in the WUI in real time. This also allows the system to restart from the last-saved point after any hardware, software or power failure, making it fault tolerant (see Section 3.1).

Filters. A Filter is a generic mechanism for meta-alert processing. Since ACARM-ng is an alert correlation framework, most of the implementations of the filter's interface correlate alerts and meta-alerts based on a defined rule set.

The API of a filter is generic. It uses inheritance for the actual implementations to be done. The processing call takes the meta-alert to be processed as an argument and returns a set of changed elements as a result. This allows very flexible actions to be implemented. Though most of the filter types perform a "regular" correlation, there are ones designed to do other actions like:

– changing priorities, based on given rules,
– DNS name resolving.

To make the development of new, typical correlation plug-ins even simpler, a common case when correlation is performed based on a match between meta-alerts being observed and the incoming ones, has a helper implementation called "simple strategy". It wraps the actions of saving meta-alerts for observation for a given amount of time, pruning old entries, creating new correlated entries and adding elements to the already correlated ones. For interaction with the user's implementation, a callback is used.

Triggers. Triggers have a similar interface to filters, but they are not allowed to change meta-alerts, nor to correlate with other ones. Having a given input, they can only react to it.

The main purpose of triggers is to react to and report on meta-alerts – sample triggers can send messages via e-mail or instant messengers like Jabber [10] or Gadu-Gadu [7]. They can also react to reported events by taking any other action. One of the existing triggers, for example, executes a user-provided binary or script. This allows easy implementation of user actions, even when the user has a near-zero knowledge of the ACARM-ng system.

It is also important for the trigger to keep observing which meta-alerts it has already "triggered". This prevents continuous reporting of a given meta-alert after it reaches a predefined threshold (e.g. priority), and something else is done with it later on (e.g. new alert added). Without this functionality, the administrators would have been flooded with reports after one of meta-alerts had reached a defined level of importance.

Each trigger should only observe a meta-alert but not keep ownership of its shared object. This prevents a situation where meta-alerts are kept in memory whilst no filter is observing them any longer; when the last filter releases the meta-alert it should be deallocated, and triggers should remove it from their set of triggered meta-alerts, if it was originally there.

Triggers are an important feature of ACARM-ng, making it an Intrusion Prevention System (IPS). They provide the user with a powerful mechanism for automating security management.

Processors. A Processor is a common interface that the trigger (section 3.2) and the filter implement. It can be observed, that both abstractions have very similar, top-level behavior. Each takes a new element (meta-alert) to process, performs some actions on it, and returns the result set consisting of elements changed and/or newly created.

It should be noted, however, that triggers do not change anything – they are just meant to perform some action when a defined event is spotted. Taking this into account we can narrow their implementation down to getting one meta-alert and returning none.

Inputs. The input provides a generic interface for gathering alerts from external sources. It does not make any assumptions on the format of input data nor the channel of arrival. It is up to specific input implementation to translate its input data to a piece of information compatible with ACARM-ng.

Sample inputs already implemented in ACARM-ng are IDMEF [8] file input and Prelude-Manager [17] input. The first allows easy integration with other, non-standard tools, migration and testing; while the second covers all the most important alert sources that are typically used[1].

Input is also an interface that has to be provided with an implementation. There can be any number of different input instances running at the same time

[1] Full list of the sources supported by Prelude-Manager, has many positions and is available in [17].

in the system. The interface can be summarized as a call that returns an alert captured from a given source.

Note, that each input is also run in a separate thread, to avoid blocking the whole system on the I/O, which is usually much slower than the processing mechanisms.

At this point, it is worth mentioning, that alert representation in ACARM-ng is similar to the one proposed by IDMEF [8], but not identical. Internal representation is limited to fields known to be useful for correlation. This model may be expanded in the future.

Preprocessor. The experience with using the "first" ACARM in production environment showed a need for an extra component to filter out insignificant reports. Such an approach would decrease the amount of data to be processed by the main part of the system and improve the quality of generated results by minimizing false positives.

The ACARM-ng framework implements this feature as a processing phase between input modules and the main system queue. It checks if a given alert matches some of the defined rules and acts accordingly, rejecting or accepting the event for further processing.

The preprocessor's behavior is defined by a configuration file. This configuration consists of default behavior specification (reject or accept) and a sequence of rules that decide if an alert should be accepted, rejected or passed on to the next rule. Each rule can be a combination of *and*, *or* and *not* logical operations. The term is a constant (true/false) or a comparison between some field(s) and a given value. This set provides the user with full Boolean logic.

The ability to create a sequence of rules, where not every one has to make a final decision is also very user-friendly, allowing the user to create short, intuitive rules that can be easily understood.

For example, consider the following sequence of rules:

1. source.ip="10.0.0.1" ⇒ ACCEPT
2. name contains "something common" ⇒ REJECT
3. ⇒ ACCEPT (default behavior)

The sequence causes the alerts that have the string "something common" in their name to be rejected, while accepting all the others. The only exception are alerts that have source IP set to 10.0.0.1, which is always accepted.

An alert's representation may as well contain collections (for example, a set of source hosts). To allow this, in preprocessor a special notation has been created. Whenever a collection is spotted, one can use "*" or "$" syntax, meaning "every entry" or "at least one entry", respectively.

Persistency. In order to be database-independent, it was decided to create an abstraction layer providing interfaces for common persistency-related tasks. The complete API for this module consists of multiple classes and methods, and, therefore, will not be presented here in detail – the general ideas will be described instead.

Since persistency abstraction provides a generic interface in form of simple methods, defining how a given call will be handled is a fully implementation-specific issue. The only real requirement of the provided back-end is the support for transactions. The actual persistency plug-in can even use regular disk files, though in practice databases are a more convenient choice.

Using a defined interface, a developer needs not know what persistency type they will be using, making code simpler, less error-prone and easily maintainable. Such an approach allows also for transparent change of implementation type – i.e. changing data storage does not require any recompilation. In fact it is enough to just change an appropriate configuration entry.

The current implementation provides PostgreSQL [16] back-end support, as well as the "stub" (no write operations are performed). There are plans of providing support for SQLite [21] too, since it requires barely no configuration, thus, making it suitable for smaller or test installations.

Heartbeats. For security-related systems, it is important that there is an easy way to check if the security tool is running – monitoring/protecting the infrastructure. In order to be able to check this, a periodical "heartbeat" signal is required. As long as the signals arrive on time, the system is considered to be running. If the system dies, signals are no longer received.

ACARM-ng is a framework composed of many components (plug-ins) working together. Each of them requires monitoring to ensure they have not died or been blocked. In order to make the plug-in's code as simple as possible, sending heartbeats was added to the basic functionality of the *Processor* interface. As long as the user's code returns a signal to wait for new events, heartbeats are sent. If the user's code somehow stops the thread or hangs, heartbeats will no longer be sent.

Web User Interface. To facilitate the use of the system, a Web User Interface that gives a brief look at the system's state, correlation results, etc. is provided. A good UI should be easy to use and powerful at the same time, and the ACARM-ng WUI seems to meet these requirements.

The features of the WUI include:

– incoming alerts list,
– heatmap of the most often reported source and destination hosts (Fig. 2),
– correlated meta alerts list,
– alert types (Fig. 3),
– analyzers list (i.e. facilities reporting alerts),
– alert time series.

Using the provided Web User Interface, an administrator can keep an eye on what the system is currently processing, what the top-most correlation results are, how often a new alert arrives, and more.

The WUI also improves real-time reporting, since it is just enough to provide an administrator with basic pieces of information about the reported incident, along with a link to the page that will display full information on this particular

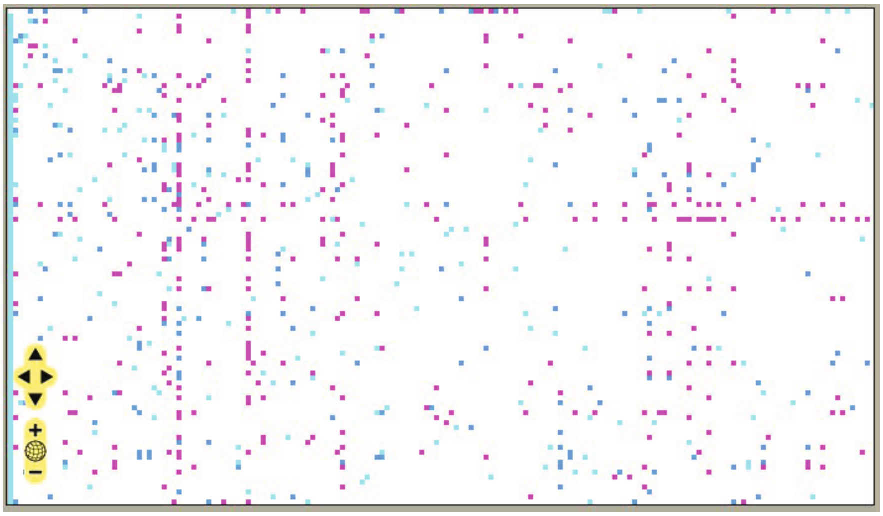


Fig. 2. Heatmap depicts the most troublesome host interactions. All hosts are represented on the map, where each dot represents interaction between two, specific hosts. The color of the dot represents intensity of the activity.

event. If the administrator decides that the report is relevant, they can follow the link to learn more about it.

4 Results

ACARM-ng is currently being tested at WCSS [23]. The main purpose of these tests is to measure the system's performance and the ability to reduce the number of alerts in a real-world environment. To achieve these goals, different system configurations have been tested, and the correlation between (meta-) alerts as well as the number of reports sent to the administrator have been analyzed.

4.1 Test Environment

The tests were performed on WCSS' supercomputers, namely: Nova and Super-Nova [13]. In addition, some of the workstations and local servers were connected to the monitored infrastructure as well. This made a total of over 700 physical machines.

Our installation uses XEN-server with two Dual Core 2.6GHz AMD Opteron 285 CPUs. ACARM-ng runs on a XEN-based virtual machine, with 1GB RAM and three cores assigned.

4.2 Performance

The real-world load of a security system is, on average, one incoming alert every 10s. This load makes ACARM-ng idle nearly all the time in real configuration

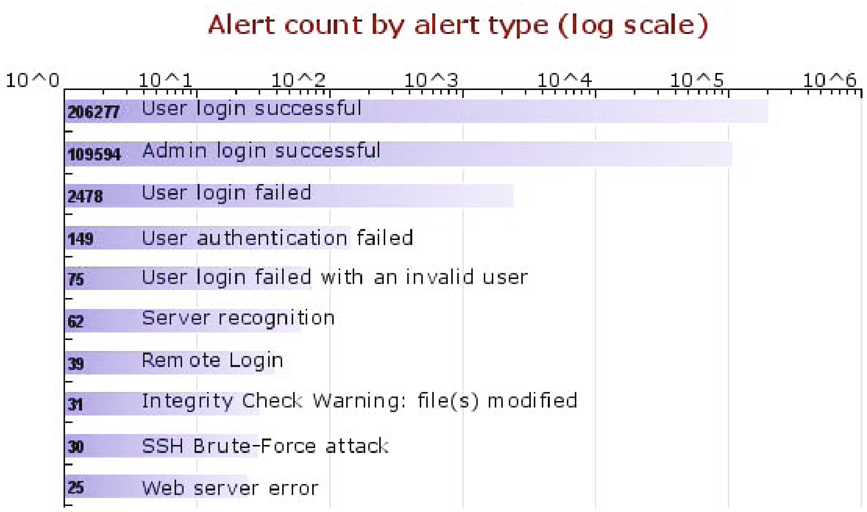


Fig. 3. The alert types figure shows the most frequently seen alert types in a given period. The alerts are grouped together by their names. Notice the usage of logarithmic scale, which is required because of the huge differences between the first and the last entries in the figure – usually there are 2-3 types of reports that make 80-95% of all reports arriving.

(i.e. feasible and useful). However, performance of different configurations have been compared. For test purposes, inputs were read from IDMEF files. ACARM-ng running with no persistent storage and no filters was able to process as many as 12,000 alerts per second. After turning on all the filters, performance dropped to 1,100 alerts per second, which is still an outstanding result. Finally, enabling persistent storage in a database limited the total number of alerts processed per second to 90, which is still 900 times larger than the number of incoming alerts at WCSS. This makes ACARM-ng a good solution even for large grids.

4.3 Alert Reduction

As with the performance, alert reduction of ACARM-ng depends heavily on the configuration. Using longer time windows usually results in much higher correlation rates. For time windows of 2h, as typically used in our test installation, the correlation of 1k alerts together is not unusual.

After running for six weeks, the system had gathered about 414k alerts. This makes on average 9k alerts a day. The output set from all filters is 153k events. From 414k input alerts, 1.5k events were reported, and out of the 1.5k reported, 57 were reported as important, giving, on average, less than two reports per day. These figures show how ACARM-ng can make an administrator's work easier.

Table 1 contains the number of output meta-alerts and reports for each filter, along with average number of alerts merged into one meta-alert. In our

Table 1. The number of output meta-alerts and reports produced by chosen filters along with the average number of correlated alerts. The entries are sorted by the number of meta-alerts created. Notice, that high correlation rate filters make it simple for an administrator to process all the alerts. On the other hand, "small", yet accurately selected and reported alerts make it easier to react on specific situations.

filter name	1000*meta-alerts	reports	average correlation
one-to-one	54.0	1	6.3
many-to-one	44.4	1	8.0
similarity	17.4	6	23.1
one-to-many	13.3	1	31.1
users' monitor	12.2	1	20.2
event chain	10.7	0	2.2
same name	1.2	9	355.9

configuration, the "one-to-one" filter produced about 54k meta-alerts consisting of, on average, 6.3 alerts each. Out of these, only one report was sent to the administrator.

5 Conclusions and Further Work

In this paper, a novel approach to alert correlation is described. The main goals and feature-requests are discussed in detail. For each of them, a proper solution is proposed along with discussion of its advantages and possible drawbacks, if present. A comparison with other available Open Source tools shows the advantages of the proposed system.

Based on the presented concept, an actual framework has been implemented [2]; and, to show it at work, sample correlation techniques have been introduced as well. Unique features of independent reporting and correlating, along with multi-threaded processing have been proved to work.

Further work mostly involves extending ACARM-ng implementation to include new correlation techniques. New filters that are planned to be implemented include testing artificial intelligence techniques for correlation and using topology-based reasoning (i.e. introducing extra knowledge of the network layout).

References

1. ACARM home page (first version) (2009), http://www.acarm.wcss.wroc.pl
2. ACARM-ng home page (2010), http://www.acarm.wcss.wroc.pl
3. Adams, J., William J.: Lynn meets with NATO leaders for Cybersecurity Discussions (2010)
4. AMD Opteron 12-core CPU (2011), http://www.amd.com/us/products/pricing/Pages/server-opteron.aspx
5. Cuppens, F., Miège, A.: Alert correlation in a cooperative intrusion detection framework. In: IEEE Symposium on Security and Privacy, pp. 202–215 (2002)

6. Valeur, F., Vigna, G., Kruegel, C., Kemmerer, R.A.: A comprehensive approach to intrusion detection alert correlation. IEEE Transactions on Dependable and Secure Computing 1 (2004)
7. Gadu-Gadu instant messaging protocol, http://www.gadu-gadu.pl
8. Debar, H., Curry, D., Feinstein, B.: RFC 4765: The intrusion detection message exchange format (IDMEF) (2007)
9. Helman, P., Liepins, G., Richards, W.: Foundations of intrusion detection. In: The IEEE Computer Security Foundations Workshop V (1992)
10. Jabber XMMP-based communicator, http://www.jabber.org
11. Jones, A.K., Sielken, R.S.: Computer system intrusion detection: A survey. Tech. rep., University of Virginia, Charlottesville, VA (1999)
12. Ning, P., Cui, Y., Reeves, D.S.: Analyzing Intensive Intrusion Alerts via Correlation. In: Wespi, A., Vigna, G., Deri, L. (eds.) RAID 2002. LNCS, vol. 2516, pp. 74–94. Springer, Heidelberg (2002)
13. Nova – computing cluster at WCSS, http://www.kdm.wcss.wroc.pl/wiki/Nova
14. OSSEC – host-based intrusion detection system, http://www.ossec.net
15. OSSIM – open source security information management, http://www.ossim.net
16. Postgresql open source database, http://www.postgresql.org
17. Prelude intrusion detection system, http://www.prelude-technologies.com
18. Rootkit Hunter project, http://rkhunter.sourceforge.net
19. Rose, M.: RFC 3080: BEEP – The Blocks Extensible Exchange Protocol (2011)
20. Snort intrusion detection system, http://www.snort.org
21. SQLite – server-less, transactional database, http://www.sqlite.org
22. Sutter, H.: The free lunch is over: a fundamental turn toward concurrency in software (2009)
23. Wrocław Centre for Networking and Supercomputing, http://www.wcss.wroc.pl

Security Best Practices: Applying Defense-in-Depth Strategy to Protect the NGI_PL

Bartłomiej Balcerek[1], Gerard Frankowski[2], Agnieszka Kwiecień[1], Adam Smutnicki[1], and Marcin Teodorczyk[1]

[1] Wrocław University of Technology, WCSS
bartlomiej.balcerek@pwr.wroc.pl
http://www.wcss.pl

[2] Poznań Supercomputing and Networking Center,
Institute of Bioorganic Chemistry of the Polish Academy of Sciences
gerard.frankowski@man.poznan.pl
http://www.pcss.pl

Abstract. The role of security in modern IT systems is continuously growing. Large infrastructures have to be protected against sophisticated attacks on organizational, technical and logical levels. Achieving sufficient security level becomes even more difficult for distributed and, often, heterogeneous environments that involve valuable assets and data – like grids. The main goal of the work described within this paper is to provide maximum level of protection against network attackers to the PL-Grid – Polish National Grid Initiative – infrastructure.

Keywords: IT security, attack, defense-in-depth, procedures, penetration tests, static analysis, PKI, NGI, NGI_PL.

1 Introduction

1.1 IT Security Today

Contemporary IT services are becoming more and more ubiquitous, which rapidly brings new possibilities for network attackers, due to at least two facts. Firstly, increasing the number of running services causes the attack vector to grow too, as every service contains a number of security vulnerabilities. Secondly, an average user of an IT service cannot be considered an IT expert, especially a security specialist. Therefore, IT services available for the public must not only be free from security vulnerabilities to the maximum possible degree, but also provide embedded security mechanisms that would require no more than general consciousness of reasonable behavior on the Internet from their users. This trend is especially visible in most recent computational models currently recognized by the users, like cloud computing. However, a need for built-in security mechanisms applies to a more mature – grid – model as well.

A number of reports show that all organizations may be prone to network attacks. For instance, in the United Kingdom, 75% of large organizations suffered a security incident in 2010, while the average number of incidents was 45 per year, and the average cost of the worst incident was between £280,000 and £690,000. The statistics show also that all the numbers have significantly worsened since 2008 [31]. Even within the area where one would expect a reasonable level of security, significant improvements are necessary – only 21% of Polish e-commerce sites are secured with SSL certificates. Moreover, the available technologies are often used inappropriately or insufficiently: 7% of certificates out of these 21% are invalid or do not realize their expected function [10].

Today, network attackers think in terms of financial gain. They not only try to find personal or sensitive data that may be stolen and sold, but also are eager to create botnets and offer them on the black market. A botnet may be used e.g. for sending spam or DDoS attacks, but its computing power may be sold as well. For instance, it was estimated, that in January 2007, a 1,000-host botnet could be bought for $10,000, but in February it was worth only $2,000 [13].

1.2 The PL-Grid Project and Its Potential Security Threats

The infrastructures of R&D projects are becoming an attractive goal for network attackers, especially if they provide powerful computational facilities for researchers, and PL-Grid may be an example of such.

The basic characteristics of such an infrastructure that may induce security threats are as follows:

– The project involves a significant number of resources (ca. 15 thousand of CPUs with 215 TFlops and ca. 2,500 TB of storage space [28]).
– The users of the computational infrastructure store a lot of important or sensitive data, e.g. results of their research.
– Not all users of the infrastructure are sufficiently security aware, as it cannot be expected from the researchers to act like IT security experts.
– In order to make access to resources more user-friendly, a set of custom software tools is being developed within the project. This software does not undergo analysis of external "white hat" experts, and its vulnerabilities may be discovered by attackers.
– The infrastructure is distributed and heterogeneous, which makes the task of maintaining high security level even more difficult and time-consuming.

A successful attack (e.g. stealing and disclosing research data) would not only have a direct impact on the victims, but also can cause a PR disaster, which could decrease trust towards PL-Grid and grid infrastructures in general. Therefore, the problem of an appropriate design and implementation of IT security has become essential and meaningful since the project proposal stage.

2 State-of-the-Art

2.1 Addressing Security in Complex Infrastructures

As security is rather a process than a stable state, it, even if once achieved, requires a security policy to be maintained. A good implementation of a security system has to involve two levels: organizational and technical. The first is not possible to be applied alone (no measures to implement it). The technical implementation, in turn, without organizational support will merely assure the desired security level at the application time, but after any changes to the protected environment (e.g. new vulnerabilities disclosed) new threats may appear.

Therefore, security must be carefully planned and implemented according to these plans. To protect a large and complex infrastructure, one of the well known methodologies and models should be applied. According to the best practices of security testing [16], the protected system should be decomposed into smaller (functionally separated) pieces, with interactions between them identified. After that, a selected threat modeling approach should be applied to obtain structured description of potential security threats. When the whole system is ready, regular tests should address particular threats identified within it [5].

Usually, three objectives of information security are enumerated: Confidentiality, Integrity and Availability (CIA) [32]. They enable a good understanding of the issue, but are not sufficient for the threats specification. Therefore, a more formal approach should be applied. One of the security models worth considering is the STRIDE model [15], that has been used to assess the concept of National Data Storage R&D project [1].

The next step in applying security should be risk analysis. Particular security vulnerabilities may be hardly exploitable – visible only to a trusted user, or the cost of their mitigation may be enormously high. In such cases, it may be decided not to address them, but the decision should be made by a management body. Another model – DREAD [24] may be used to support such tasks.

Each service is built upon compiled and executed source code, which inevitably contains bugs. The research shows that a thousand lines of source code (called KLOC) may contain up to 20-30 bugs [23]. The reason for that is mainly the level of code complexity, often combined with pressure to deliver fully functional software on time. Achieving better bug coefficients involves costs. It is estimated that providing an appropriate quality industry software costs about $5,000 per KLOC, while for the space shuttle management software (1 bug per 250 KLOCs) each KLOC is worth $850,000 [20]. Evaluating the software for the Common Criteria EAL4 (out of EAL7) may cost between $150,000 and $300,000 and last for between 9 and 24 months [26]. Those approaches are therefore not applicable in the PL-Grid project.

The hope lies in the fact that the protected systems do not have to be absolutely secure. The attackers are forced to use some resources (e.g. time or money) to prepare and perform attacks, and these are not unlimited. Thus, in order to avoid attacks, it is usually enough to make a successful attack consume too many of the attacker's resources. When the gain from breaking into the system is smaller than the cost of the attack, the system may be considered secure [25]. It may be difficult to precisely calculate those values, but, in general, providing consistent security measures decreases the probability of an attack.

2.2 Security Tasks in Selected Polish R&D Projects

In this section, handling security issues in several Polish R&D projects will be shortly described, based on the authors' experience. Some of the security solutions implemented in PL-Grid are based on research performed during these projects.

Clusterix. The primary goal of the Clusterix project was to build a grid-type environment consistent with local PC clusters placed at geographically independent research centers [2]. The project architecture was security-oriented, even if this approach increased its complexity level and introduced potential bottlenecks. A significant security challenge was the possibility of dynamically attaching and detaching additional clusters to the basic infrastructure. An adequate protection was applied automatically just after attaching the cluster, thanks to suitable procedures that had been defined for that purpose. A separate work package within the project was devoted to security issues. Security was multilayered, but there were no software security assessments nor penetration testing of the whole infrastructure; and security trainings were not provided.

National Data Storage. The purpose of the National Data Storage (NDS) project was to provide large secondary storage facilities for education and government in Poland [1]. As in many other R&D projects, limited project scope caused that security measures could only be applied to some extent. They covered security assessment of the project assumptions and reviews of custom and third-party software. Although a separate task was devoted to security issues, the largest drawback in handling security within the project was the lack of dedicated security trainings (however, they were partially provided on-demand during the project meetings).

Security matters have been addressed more thoroughly in the NDS2 project launched in May 2011. The project has introduced new functionality as well as enhanced security and performance stability, comparing to the basic NDS model [8].

3 Solution – Security Center

In order to protect the PL-Grid resources, a complex, top-down approach has been agreed and accepted, beginning with high-level, organizational and procedural issues. To provide appropriate control over IT security within the project, a separate work package has been created (WP6 – Infrastructure Security). The Security Center (SC) for the project, led by Security Coordinator, was established. The Security Coordinator is responsible for all security-oriented activities (either within WP6 or the cross-activity ones), as well as for communication between the Center's members. PL-Grid SC represents Polish grid as NGI_PL in the European Grid Initiative (EGI) project's security activities.

The Security Center is engaged at every stage of development of the target infrastructure. According to the project guidelines, actions performed by SC must not only be reactive, but also proactive – they should prevent incidents, rather than just handle them. Therefore, the first goal of the Center was to review the current security standards, prepare and deploy suitable security policies and procedures that would be used in further work. A number of documents were released, containing security requirements and recommendations for administrators and programmers, as well as operational policies, e.g. concerning security monitoring and assessment. Further tasks of the Security Center involved development of a methodology for grid-specific penetration tests, based on well known security testing approaches. Since keeping a Certificate Authority is the main part of grid authority management, a separate team was appointed for this purpose, and two Certificate Authorities were assigned – each of them different in terms of policy strictness and scope.

The tasks of the PL-Grid Security Center are described in respective subsections below.

3.1 Operational Actions

As mentioned before, IT systems are subject to continuous development and changes, which leads to treating security as a process rather than a state. Therefore, there is no such thing as a secure system which does not need any supervision. For this reason, operational actions are the most common tasks performed by security teams. Unlike other actions, these tasks have to be performed on a day-to-day basis. They cover three main aspects: infrastructure monitoring, patching (or mitigating threats) and dealing with incidents. The first two are preventive actions, while the latter is a reaction to a certain situation.

Extensive and effective monitoring is the best way to find newly appearing vulnerabilities in a system before someone is able to exploit them. Yet, if such a situation occurs, the affected parts of the system can be isolated until they are properly secured. In most cases, a security patch for vulnerable source code or configuration already exists, and applying it should be enough. If there is no patch, the only way of dealing with the threat is its mitigation – i.e. lowering the chance of exploiting the vulnerability.

PL-Grid is a large project and needs effective tools and processes for monitoring its infrastructure. SC uses Pakiti as the main tool for this purpose. Pakiti allows to search for all known vulnerabilities in every system package of each host available within the infrastructure. When a vulnerable host is found, a security officer notifies the administrator responsible for this host, in order to secure the system as quickly as possible.

As the PL-Grid infrastructure is not isolated from other infrastructures, SC cooperates with the EGI Computer Security Incident Response Team (EGI CSIRT) and Polish NREN – the Pionier Computer Emergency Response Team.

In order to support the security team and grid administrators in identifying attacks as fast as possible or preventing them, ACARM-ng (the Alert Correlation Assessment and Reaction Module [7]) – was deployed for the infrastructure. Additionally, the SARA system (System for Automated Reporting and Administration [14]) is being developed for supporting certain facets of static security monitoring within PL-Grid.

3.2 Simple CA – Security and Usability

X.509 digital certificates are at the base of grid security. By using them, the users can confirm their identity, claim and achieve integrity and confidentiality of their data. Thanks to these certificates, administrators can implement required authorization, accountability and non-repudiation policies. With all their advantages, digital certificates have at least one major disadvantage – most of the users find the process of obtaining and using such a certificate a time consuming, inconvenient and confusing task. The PL-Grid Simple Certification Authority (Simple CA) was designed to minimize those disadvantages with as small as possible security-usability trade-off.

User Verification Procedures. The international organization that coordinates the trust fabric for e-Science grid authentication in Europe, the Middle East and Africa is EUGridPMA (European Grid Policy Management Authority) [11]. This organization is a member of the International Grid Trust Federation (IGTF). Grid Policy Management Authorities (PMAs) are bodies that establish requirements and best practices for grid identity providers, to enable common trust domain applicable to authentication of end-entities that access distributed resources. They coordinate Public Key Infrastructures (PKIs) for use with Grid authentication middleware.

The IGTF is a body that establishes common policies and guidelines between its PMA members and that ensures compliance with its Federation Document [18] among the participating PMAs. All these coordination activities simplify the usage of PKI in grids, mainly from the relying parties' point of view (e.g. infrastructure administrators). The advantage that affects the users is a centralized and well-structured information source about relevant certificate authorities. However, it still requires a lot of effort from a user to learn the policies, requirements and process details needed to obtain and maintain a certificate. Typically, users who want to obtain a certificate, have to request a certificate from their national or regional issuing authority (Certification Authority) and prove their identity to the local representative of Registration Authority (RA).

In its guidelines for traditional X.509 PKI CAs [12], EUGridPMA requires from the users a face-to-face meeting with local representative of an RA and presenting passport or other ID with a photo. At the same time, one-statement certificate policies (1SCP) [17] managed by EUGridPMA define two Certificate Policies with different identity vetting rules: Face to Face and Trusted Third Party mediated. The latter policy relies on the identity vetting mediated by a trusted third party (TTP). An adoption of this option results in less restrictive policies and procedures for a particular CA and possibility to avoid face-to-face meetings with every end-user. Still, there is a recommendation that TTP should be personally known to the RA; and a requirement, that the end user must be personally known to the TTP. This points another way to build a trust relation between the end user and the CA – upon a chain of personal contacts.

Currently, the only Polish national certificate authority accredited by EU-GridPMA is the Polish Grid Certification Authority [29]. In principle, it follows the 1SCP Identity Vetting: Face to Face and has local representative RAs in 13 cities. This situation discourages some users, especially the ones that come from cities that do not have an RA.

Simple CA User Verification Procedure. The PL-Grid Simple CA is an alternative to the Polish Grid Certification Authority. The PL-Grid users are not able to use certificates signed by PL-Grid Simple CA outside the PL-Grid infrastructure, but, in return, these certificates can be obtained in a faster and more convenient way. The identity verification here is based on a PL-Grid user registration process in the PL-Grid Portal [30].

The Portal is a Web application, which allows users to manage their account information and certificates, apply for services and to access the grid. The registration procedure implements identity verification sufficient for granting user access to the grid services, and membership in the PL-Grid Virtual Organization. In general, the procedure relies on checking entries in an external database of Polish scientists (the OPI database), e-mail messages and telephone calls.

The scenario of a successful registration in the Portal may be summarized as follows [21]:

1. A user fills in a registration form – name, organization, OPI number and e-mail address (initial e-mail address) are required. The OPI number is an identification number in the database of Polish scientists. If the user does not have his/her own OPI number, he/she can specify the number of his/her supervisor.
2. The Portal sends an e-mail (to the initial e-mail address) with a confirmation request.
3. The User confirms his/her request using a unique URL from the message.
4. The Portal Operator verifies the data specified by the user. He/she compares the data from the registration form and the OPI database.
5. The Portal Operator finds the telephone number to the user's/supervisor's institute and the user's/supervisor's email address (registration e-mail address). This e-mail address does not have to be the same as the initial e-mail address.
6. The Portal Operator calls the user's organization and calls or e-mails his/her supervisor to confirm the user's identity and registration e-mail address.
7. The Portal Operator sends an e-mail message to the registration e-mail address. This message contains a URL and a four-digit PIN code.
8. The User follows the URL and enters his/her PIN code.
9. The Portal creates a new account.

3.3 Penetration Testing

Penetration testing is a method of infrastructure testing, conducted from the point of view of a potential attacker. Thus, the major advantage of penetration testing is high degree of realism – the tester does what a real attacker would do. A set of different scenarios may be defined for penetration testing, e.g.: a user who has no access to the service tries to gain any access (e.g. a standard user account), or a user with a standard user account attempts to become an administrator. The biggest disadvantage of this form of testing is a relatively small number of attack vectors being attempted – as not all the security flaws are visible from external networks or may be much more difficult to detect from there. Obviously, such vulnerabilities are much harder to be exploited (or it may be even impossible) in the "external attacker" scenario, but they are still vulnerabilities and should be highlighted as such. They may be exploited in other scenarios or may become exploitable when the environment of the service changes. It is therefore recommended to complement penetration testing with other forms of security assessment.

Penetration tests can be divided into two main classes: blackbox and whitebox. The distinction between them is the point of view, from which tests are performed. Intermediate classes can also be defined, but they are seldom used.

In the blackbox approach, one does not have any information about the tested system. The tester needs to gather all information about the system by himself/herself and then find a way to break in or bypass the security protection. Such an approach simulates a real case scenario, when an attacker wants to break into an unknown system. A potential disadvantage of this method is the fact, that if the tester does not find sufficient information about a potential vulnerability, he/she would not be able to test it – however, it does not mean that the system is secure. In this case, breaking into the system is not a matter of security but only finding adequate information. Another disadvantage of this approach is that, if the tester stops on one protection mechanism, it does not automatically mean that there are no further vulnerabilities. It only means that he/she could not reach them at the moment of testing. If someone else bypasses this secure mechanism, he/she will reach untested parts of the system.

The second approach, called whitebox or security system auditing, does not have any of the aforementioned blackbox approach disadvantages. In this approach, all tests are performed with full knowledge of the system. They can be performed in a variety of ways, e.g. as a full configuration audit or as testing of all security protection mechanisms listed by a client. During whitebox testing, administrators may disable some security protection mechanisms, allowing testers to penetrate all parts of the system. Such tests affect all parts of the system, providing more detailed checking, but require much more resources.

Blackbox. Blackbox security testing of the PL-Grid infrastructure is performed by the Security Center in two identical turns, according to the following procedure: target identification, information gathering, vulnerability assessment, vulnerability verification and reporting.

In the target identification phase, the tester obtains the address of an organization's home page and gathers as much information about its infrastructure as he/she can. He/she prepares a list of machines from IP neighborhood, with domain name keywords match or with typical grid services, verifies them with Whois and sends to the organization for confirmation. After that, a list of target machines is prepared.

In the information gathering phase, the tester uses passive and active techniques to gather information about each host from the target list. This includes at least Google hacking and full nmap scan.

In the vulnerability assessment phase, using the information from the previous phase, the tester searches for known vulnerabilities. This includes at least: searching through public vulnerability databases, such as CVE (Common Vulnerabilities and Exposures) Mitre or BugTraq, scanning machines with OpenVAS only for the previously determined services and testing the services manually.

In the vulnerability verification phase, the tester checks if the vulnerabilities that were found are valid. This can be done either by testing the actual machine or by setting up a sandbox environment and testing the service in it. The latter approach should be chosen if the vulnerability exploitation may pose any risk for the target.

In the report phase, the tester prepares a report with a list of vulnerabilities discovered in the system. The list should contain the following information for each vulnerability: description, priority of fixing, how it was discovered, whether it is publicly accessible, how easy it is to discover it, how easy it is to exploit it and how to fix it. The report is then sent to responsible administrators.

Whitebox. Due to the size of the PL-Grid project, the potential threats caused by unprivileged access can be very harmful – not only for our infrastructure, but also for its victims. For this reason, SC has decided to perform whitebox penetration testing, as a full configuration audit, for the whole infrastructure. Additionally, during such an audit, SC checks whether administrators have implemented the measures required for securing the infrastructure, which were published in early 2010 [6].

A full configuration audit for such a large number of machines within the infrastructure required a dedicated approach and very good planning. As a starting point, SC gathered the following requirements for successful whitebox testing:

1. A range of the configuration to be tested needs to be defined.
2. The whole infrastructure is based on GNU/Linux systems, therefore, the tests should target these platforms.
3. The whole process must be automated as much as possible.
4. Tests will be run periodically to check newly appearing system configurations.

The first step from the list above was the most important, as the overall quality of the tests performed depended on it. An insufficiently prepared configuration check would only be a waste of time and would not improve the system security.

During the process of preparing the configuration tests, we took into consideration: grid-specific tools and their security, the Center for Internet Security benchmarks [9], the Information Systems Security Assessment Framework [27], the Open Source Security Testing Methodology Manual [19] and requirements for securing PL-Grid systems [6].

Finally, SC decided that the whitebox penetration tests would be performed in the following stages:

1. The administrators gather configurations from all the hosts within their site and send them to SC.
2. SC performs analysis of the received materials and prepares detailed reports.
3. Reports are sent to the administrators.
4. The administrators have 30 days to update vulnerable configurations.
5. SC validates whether all configuration updates have been performed properly.

Steps 1 and 2 are the main parts of whitebox penetration testing, since they produce lists of vulnerabilities in the configurations. Performing these two steps consumes over 90% of the time dedicated to the whole task. Step 5 requires repeating the whole process, but in simpler way. For this reason, SC decided to implement steps 1 and 2 as a set of scripts.

The scripts for step 1 are prepared so that they require as little time from administrators as possible, while gathering as much information as possible about the tested system. This tool has been prepared in cooperation between SC and administrators. Scripts created for step 2 check the gathered information, searching for vulnerable configurations.

3.4 Software Deployment Procedures

Software deployment procedures have been created to ensure the quality of the software available within the PL-Grid infrastructure. Before deployment, the software packages have to be evaluated as useful for the project and then pass operational and security validations.

Detailed Description. During the security validation, SC evaluates software security using blackbox and whitebox approaches (auditing), and prepares a detailed report, which is then sent to Helpdesk. If any of the operational or security validations is negative, the software will not be deployed. It can be improved and pass the validation process again, though. Finally, if the validation result is positive, the software is deployed in the production system. In case of software updates, the Provider should inform the Operation Center (OC) and provide updated version of the software package, description of the changes that were made, and updated version of the documentation. Then, OC and SC validate the updated version of the software and may approve it for deployment in the production system or reject it.

Use of Ticketing System. The whole software deployment procedure is carried out using Helpdesk – a ticketing system based on RT. This approach ensures effective and secure information flow between the parties, guarantees defined response time, provides e-mail notifications and reminders and facilitates task transfer between SC team members. It can also be searched, making the data for later requests easily available.

3.5 Auditing

A source code review is another way to evaluate the level of security. One can distinguish here between static and dynamic approach (formal and randomized, called fuzzing). As stated before, not every security vulnerability may be easily and rapidly discovered during the penetration tests, therefore, the main advantage of source code testing is a more complete review of the software. The source code may also contain errors or weaknesses, which are results of insecure programming practices and that are not a vulnerability at the moment of testing, but may cause it in the future – e.g. when new functionality is added. The disadvantage of such a control is additional (and significant) effort that must be exerted for the source code review.

Regarding the software development life cycle, a source code review should be performed after the implementation is finished, but before the software is released. This enables developers to implement corrections, if needed. Therefore, embedding the security review phase in the ticketing system that supports the software development process, seems an optimal solution.

In PL-Grid, the tests were performed both with source code analyzers, like Findbugs for Java-based software packages (e.g. Migrating Desktop), and manually. The former approach is very fast, but may reliably detect only vulnerabilities with clear structure. Moreover, the tools may report flaws while the code is actually correct ("false positives"). Thus, the results of these analyzers must still be manually verified. On the other hand, a full manual review takes much more time but is extremely reliable. It allows detecting flaws that will not be recognized by automatic scanners. Good examples may be flaws that depend on the program logic or functionality, or those that are associated with handling sensitive data in memory. Source code reviews were partially complemented by penetration tests of the software under investigation. In those cases, the penetration tests were conducted before the code review, or different experts performed the tasks.

4 Results

The code reviews of particular software packages, as well as their penetration testing, allowed to discover a significant number of security vulnerabilities and functional remarks. For instance, a report concerning one of the applications contained descriptions of 58 issues. In other cases, also a number of security vulnerabilities were detected. Nevertheless, no critical issues were identified – it was not possible to compromise any of the servers nor applications without having a user account, still, a number of moderately critical flaws, especially OS Command Injection, were reported.

All the discovered issues were successively reported to the developers, which sped up their resolving. However, for some modules, it was necessary to temporarily withhold the deployment, until the suggested recommendations were implemented.

Some general recommendations issued after a number of source code reviews and penetration testing, included: improving data sanitization patterns, optimizing monitoring of versions of external software and libraries used, putting more emphasis on warnings of the compiler, and using static code analyzers that may be integrated with the development environment, like Findbugs.

Apart from the software developed in PL-Grid, some third party software was assessed. This included Liferay Enterprise Portal and Torque Resource Manager. The Community Edition of the Liferay portal is Open Source and one of its versions has been applied in PL-Grid to build portal for the users. The research (both penetration testing and source code review) revealed a number of security vulnerabilities, partially known to the security experts from their earlier research. The vulnerabilities were first announced to the vendor, and then disclosed, based on standard rules [3] [4]. One of the reported flaws allowed to hijack a standard user account without having any credentials, while an other facilitated elevation of privileges from a standard user level to administrator rights; therefore, by combining those scenarios, an attacker could have compromised the whole portal [22]. In spite of the severity level, the vendor did not address the reported flaws. However, suitable workarounds were applied to the PL-Grid portal in order to mitigate the threats. The security analysis of the Torque Resource Manager code and its behavior revealed that it, too, was prone to two dangerous types of attacks: buffer overflow and authorization bypass. Both of them were addressed by authors in successive software versions.

The PL-Grid Simple CA has been in use for over a year now. During this period, more than a half of the PL-Grid users – 174 out of 250 – have decided to use certificates signed by PL-Grid Simple CA. We have had no compromised certificates and only few certificates revoked upon the user request. By simplifying and integrating the identity vetting rules with the Portal's registration procedures, the PL-Grid Simple CA made it easier for the PL-Grid users to start with the grid technology.

5 Conclusions and Future Work

The solutions described in this chapter allow to assure an almost optimal handling of security in the PL-Grid infrastructure, also thanks to the experience gained by the PL-Grid partners in previous R&D projects. Some minor improvements may still be identified, e.g. optimizing the procedures of spreading the knowledge of certain (not all) security vulnerabilities or elaborating ways to achieve full consistency of security reports issued by all assessment teams. Additionally, more security trainings are recommended, optimally starting from the very beginning of such a project. However, the available set of procedures and practices has been absolutely sufficient. So far, PL-Grid has not suffered from any known IT security attack.

The PL-Grid security model has already been a template for other R&D projects. It was a basis for the model proposed in the National Data Storage 2 project, which builds also on the findings of the GN3 European R&D project. Security trainings are considered there a vital part of handling security, but the crucial organizational part was based on PL-Grid.

References

1. National Data Storage Project, `http://nds.psnc.pl`
2. Research projects of WCNS, `http://www.wcss.wroc.pl//english/r.php`
3. Liferay portal multiple vulnerabilities. Secunia Advisory SA28742 (February 2008), `http://secunia.com/advisories/28742`
4. Liferay portal script insertion and jsp code execution vulnerabilities. Secunia Advisory SA38088 (January 2010), `http://secunia.com/advisories/38088`
5. Adamski, M., Frankowski, G., Jerzak, M., Stokłosa, D., Rzepka, M.: Defense in depth strategy – a use case scenario of security a virtual laboratory (2011)
6. Balcerek, B., Kosicki, G., Smutnicki, A., Teodorczyk, M.: Zalecenia bezpieczeństwa dotyczące instalacji klastrów lokalnych v0.95 (2010)
7. Balcerek, B., Szurgot, B., Uchroński, M., Waga, W.: ACARM-ng – Next Generation Correlation Framework. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 114–127. Springer, Heidelberg (2012)
8. Brzeźniak, M., Jankowski, G., Meyer, N.: National Data Storage 2 – Secure sparing, publishing and exchanging data (February 2011), `http://www.terena.org/activities/tf-storage/ws10/slides/20110204-nds2.pdf`
9. Center for Internet Security, `http://www.cisecurity.org/`
10. Domeny.pl, CertytfikatySSL.pl: Bezpieczeństwo zakupów w polskich serwisach internetowych, p. 10 (2011), `https://certyfikatyssl.pl/resources/bezpieczenstwo_zakupow_w_polskich_e-sklepach_raport.pdf`
11. EUGridPMA (2010), `http://www.eugridpma.org`
12. EuGridPMA: Authentication Profile for Classic X.509 Public Key Certification Authorities with secured infrastructure (2010)
13. Franklin, J., Paxson, V., Perrig, A., Savage, S.: An inquiry into the nature and causes of the wealth of internet miscreants, p. 12
14. Frankowski, G., Rzepka, M.: SARA – System for Inventory and Static Security Control in a Grid Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 102–113. Springer, Heidelberg (2012)
15. Herman, S., Lambert, S., Ostwald, T., Shostack, A.: Thread modeling – uncover security design flaws using the STRIDE approach. MSDN Magazine (November 2006)
16. Howard, M., LeBlanc, D.: Writing secure code, p. 347. Microsoft Press (2002)
17. IGTF: IGTF One Statement Certificate Policies (2011), `http://www.eugridpma.org/guidelines/1scp`
18. IGTF: International Grid Trust Federation, version 1.1 (2011), `http://www.igtf.net/new-doc/IGTF-Federation-20051005-1-igtf.pdf`

19. Institute for Security and Open Methodologies, http://www.isecom.org/osstmm/
20. Jain, N., Swaminathan, B.: Agile overview – embrace uncertainty, http://www.slideshare.net/nashjain/agile-overview
21. Krakowian, M.: Procedura rejestracji użytkowników v1.0.1 (2010)
22. Kuczyński, T., Nowak, T.: Conference i3, badania poziomu bezpieczeństwa portalu dostępowego do infrastruktury PL-Grid (December 2010)
23. McConnell, S.: Code Complete – A Practical Handbook of Software Construction, 2nd edn. Microsoft Press (2004)
24. Meier, J., Mackman, A., Dunner, M., Vasireddy, S., Escamilla, R., Murukan, A.: Improving web application security: Threats and countermeasures. MSDN Magazine (June 2003)
25. Odlyzko, A.: Economics, psychology and sociology of security (2003)
26. United States Government Accountability Office. Information assurance – national partnership offers benefits, but faces considerable challenges. Tech. rep. (March 2006)
27. Open Information Systems Security Group, http://www.oissg.org/issaf/
28. PL-Grid: Introduction to PL-Grid project, http://www.plgrid.pl/en/project/introduction
29. Polish Grid CA (2010), http://www.man.poznan.pl/plgrid-ca/
30. Portal (2010), http://www.portal.plgrid.pl/
31. PricewaterhouseCoopers: Information security breaches survey 2010 – technical report, p. 2, http://www.infosec.co.uk/files/isbs_2010_technical_report_single_pages.pdf
32. National Institute of Standards and Technology: Standards for Security Categorization of Federal Information and Information Systems (February 2004)

Automation of System Monitoring Based on Fuzzy Logic or Rules; Comparison of Two Designed Approaches with Regard to Computational Infrastructures

Włodzimierz Funika[1,2], Filip Szura[1,2], and Jacek Kitowski[1,2]

[1] AGH University of Science and Technology, ACC Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków, Poland
[2] AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Department of Computer Science, al. Mickiewicza 30, 30-059 Kraków, Poland
{funika,szura,kito}@agh.edu.pl

Abstract. This paper is focused on monitoring automation of distributed systems. In the presented research, AI-based approaches to distributed monitoring related to large distributed systems such as grids, were explored. In both presented concepts knowledge is used to make decisions regarding management actions, using rules and fuzzy logic. The first concept is an agent-less rule-based solution, implemented in a high-level monitoring system called Saude-Net. It allows to define actions for monitored resources, using a kind of expert system. The second solution, which exploits agents and fuzzy logic, is realized in a system called SAMM Compliant Agent. Both presented systems are capable of reacting to observed failures and of modifying their knowledge to better fit possible problems with resources. We also present a short comparison of the two concepts, and an analysis of their usage.

Keywords: system monitoring, automation, rules, artificial intelligence, fuzzy logic, fuzzy sets, error reporting.

1 Introduction

Since the PL-Grid infrastructure [1] is meant to be a country-wide compute- and data-intensive platform, facilitating its administrator's operations, e.g. due to the size of infrastructure, is one of the key issues. An installation of this size can be endangered by malfunctioning of its resources, which needs detection of their failures and fast reactions (responses) to them.

To accomplish this task it is necessary to provide support for monitoring and reacting to observed errors. To do this the system administrator is usually assumed to check any raised alerts and respond on them. It has to be done continuously. This paper presents two approaches aimed to help administrators in their daily work. These solutions use knowledge described by rules or fuzzy sets.

Many existing systems like Autopilot exploit this type of mechanism related to Artificial Intelligence (AI) which is commonly considered as a very broad research area that focuses on "Making computers think like people" and includes disciplines such as Neural Networks, Genetic Algorithms, Decision Trees, Frame Systems and Expert Systems. Fuzzy logic can be also regarded as an AI supporting solution and is used in Multi Criteria Decision Analysis (MCDA) [2]. In the context of resource monitoring/management activities, fuzzy logic may be used to reply to the following question: *"which action is better for the values of the parameters under measurement (if these parameters are influencing decision-making, e.g. related to resource management)?"*. Both concepts, rules and fuzzy logic, are used in existing systems as their knowledge engine, not only in computer science but also in areas such as medicine, process control or financial service. Fuzzy logic may be used when the designer does not have enough knowledge to model the whole area. In these types of systems it is possible to use these sets and possess only a limited amount of information to begin with.

In contrast to fuzzy logic, rules are often used when the designer wants to model relationships with "IF ... THEN ...". In these situations it is possible to describe this by rules which are understandable for humans and machines. Rules allow to define a sequence of conditions which have to be fulfilled. The rules need to be well defined and should cover the whole considered area. The usefulness of expert systems made them very popular in medicine, computer science, robotics. On the market many expert systems are available, such as Gideon (used for diagnostics of diseases), CaDet (cancer detection), Thorask (selection of injured people). Expert systems are also becoming a noticeable tool for computer science itself.

In this paper we present two concepts which were developed as intelligent software which would be able to help the administrator of a large computational infrastructure to cope with everyday duties at lower cost. These systems are targeted to provide automation of system administrator's functions, e.g. data storage usage, services accessibility. Both of them try to address the discussed issue using knowledge mechanisms. Based on these two approaches we focus on building a monitoring tool aimed at handling faulty operations of infrastructure resources.

The rest of the paper is organized as follows: Section 2 gives a brief overview of some of the existing monitoring tools. Section 3 presents our approach to the issue of monitoring automation and our solutions implemented in two different monitoring systems. Section 4 presents some details regarding tests which were performed on both our systems. The last section summarizes the discussed solutions and shows ideas regarding future work.

2 Related Work

At present there are a lot of monitoring systems for large distributed systems, both commercial such as Intellipool Network Monitoring, and free such as Zabbix or Nagios. These applications were designed for different purposes. As mentioned in [3] "the goal of any monitoring system is to provide a centralized view of the monitored resources so that systems administrators can analyze conditions and prevent or fix problems quickly and efficiently. Generally, a separate system is set up to host the monitoring system and is placed in a centrally-located point". All monitoring systems may be divided into two classes. One of them uses agents in monitoring, another group uses only commands and protocols. The former, agent-based approach is used in many of existing systems [4,5] because it allows to monitor various parameters and does not affect the network load too much. The latter solution, the agent-less one is used when the agents cannot be installed due to system restrictions. There exist many facilities which allow to monitor large distributed systems such as grids.

The first system which is going to be presented is Nagios [6]. It is often used as a basis for other monitoring facilities, e.g. EGEE Grid Infrastructure Monitoring. Nagios provides many plugins which allows for customizing for particular purposes. It is able to monitor many distributed system parameters, it also can be easily extended. Nagios monitors the status of host systems and network services and notifies the user of problems. The monitoring daemon runs intermittent checks on hosts and services one specifies using external plugins, which return status information to Nagios. It provides only a basic set of sensors, but custom sensors can be developed by using any existing programming language. Nagios functionality may be extended by many plugins. It is also possible to define system commands (scripts or executables) which are run when a host or service state is changed. This ability is called an "event handler" and it gives Nagios an ability to e.g. restart services, log event information, enter a trouble ticket into a helpdesk system. This tool is able to react to failures but its reactions have to be earlier well defined by the system administrator.

The second interesting system is Zabbix. This tool is similar to Nagios. Like the previous one this system is designed to monitor various network services. This solution uses agents to monitor statistics of Unix and Windows hosts. In contrast to Nagios the Zabbix monitoring tool is much easier to configure and manipulate. All these functions may be done easily during network monitoring using a simple and easy-to-use web user interface. It doesn't require any modifications in its configuration files like does Nagios in its basic form.

Another tool is Ganglia, which – as described in [7] – is designed to monitor large infrastructures. It is a scalable distributed monitoring system designed for high performance computing systems such as large localized clusters and even widely distributed Grids. It relies on a multicast-based listen/announce protocol to monitor the state within clusters and uses point-to-point connections among representative cluster nodes to federate clusters into a Grid and aggregate their states. Ganglia has the advantages of low per-node overhead, high concurrency and robustness. This tool is a very good scalable monitoring facility, but like the previous one this system does not allow to automate the monitoring, nor uses it knowledge to adjust its work to the observed situations.

A different monitoring solution is Autopilot. It is one of the first systems which uses knowledge to choose appropriate actions. As described in [8] Autopilot is a novel infrastructure for dynamic performance tuning of heterogeneous computational grids based on closed loop control. This software, in contrast to the above, can take actions which can optimize data centers and distributed applications.

Another system which allows for automation is Intellipool Network Monitor [10] which enables to define a set of rules which will be used when an error will be observed in a monitored network. This system is designed for large enterprise networks and can monitor distributed systems. It uses predefined actions to solve observed problems. It also provides functions which are used to alert the system administrator about reported failures. A next system which allows to define reactions on failures is Hyperic HQ [11]. It allows to automatically perform simple system administration tasks to prevent and resolve a broad range of issues without human intervention.

The above overview shows that there are very few systems such as Autopilot which are able to make decisions on actions in an adaptive way. Most of the existing systems are designed only to monitor networks and show visual information to the system administrator – it is up to the administrator how they are going to tackle emerging problems. The tools are mostly used to inform about system (resource) failures. They allow to define simple actions such as service restart as a reaction on the failures reported, but this is done in form of static definitions. There also exist other monitoring tools, e.g. QStorMan [9]. This tool enables data-bound monitoring related to the infrastructure workload and users' activities.

3 Concepts of Our Solution

This section presents two concepts of automation of system monitoring [12] used: rule-based and semantic-oriented agent-based approaches. These concepts are implemented by the Saude-Net system [14] and SAMM Compliant Agent (SAMM-CA) [15], respectively, both of them are meant to react to captured system failures. Both concepts allow to manage monitored resources to optimize their work and usage by re-arranging a system configuration which is in the scope of the system administrator's responsibility. In this section we present further details regarding the concepts and relevant systems' architectures.

3.1 Saude-Net System

The first solution to be presented is an automation system which uses rule-oriented approach. This system allows to define multiple actions for observed monitored resources that we call *services*. This system is on top of external low-level monitoring facilities. In its current version it uses the Zabbix monitoring system to provide Saude-Net with data, and to perform appropriate actions based or this data when necessary. The back-end of the Saude-Net tool Zabbix was chosen because it is much easier to configure or manage than other tools of similar functionality. Zabbix is also an agent-based solution. There is no need to write a complex XML description of a new resource when it is added to the monitored resources list. Zabbix only requires a single line in the agent configuration.

Saude-Net was developed to automate monitoring using rules and actions. This system is allowed to extend or change predefined actions on-line as well as redefine rules on-line. These changes do not bring the system to a restart or a suspend – the user is unaware that the monitoring system has to modify, validate and reload its knowledge. The Saude-Net tool monitores the whole network using Zabbix as a low-level layer. When a failure is observed in one of monitored resources (host or service), Saude-Net tries to perform some actions. To accomplish this task it uses its knowledge, which is described by rules. At the beginning it creates a list of all possible actions for an observed failure. This list is generated with help of rules. For each action there is an associated preference value called Preference Value (PV). This value has to be set by the system administrator when the action is added to Saude-Net. This value should be from the interval between 0 and 1. The default value of PV is 0,5 as a center of the mentioned interval. The PV is used to determine which action is best. Saude-Net sorts the list of possible actions to choose the best one from all available ones. If there is only one action on the list it is chosen regardless of the value of PV. In the worst scenario an action with a value of PV close to zero might be used. The Saude-Net picks the first action on the list – it is an action with the highest PV value. In the next step Saude-Net performs a selected action, and after that it changes the PV value for all the actions which were added to the list including the action which was executed. This modification consists of increasing each PV by the Performance Tuning Value (PTV) using the following formula:

$$PV = oldPV + PTV \tag{1}$$

After all modifications the list is cleared for a next failure to be handled. PTV means “preference tuning value” which is dependent on the count of actions which were considered when an error occurred. This value is calculated according to the following formula:

$$PTV = \frac{N_s}{D} * const * \frac{I_w}{D} \tag{2}$$

where:

- D is the number of instances of any action in the created ranking. It is the count of elements which occur in the ranking returned by the knowledge engine.
- N_s is the number of services which are currently modified. They are defined as services which have to be optimized or repaired.
- I_w is one of the following values: 1 implies a situation when the action wins and -1 when it does not.
- *const* is a constant value which equals 1/700. This value was determined for the monitored infrastructure empirically through a series of tests. For this value the modification of the PV value was enough to manage actions. This value may be changed in configuration for different networks. It is dependent on the monitored system and should be set by the system administrator. The value 1/700 is the default value for which PTV was large enough to fit actions to the observed failures.

The above functions are used to choose the best action. They allow our system to learn which action is best and which one is worst. This is dependent on the statistics of the usage and on the history of each action. The preference value can be also changed by the system administrator when Saude-Net is working. The PTV formula is dependent on the count of all actions because when the system is able to choose many actions with similar PV values the modification should be smaller to avoid big errors.

A system which implements the above idea is designed as a modular tool, divided into three separate components: Saude-Net GUI, Saude-Net Server and Saude-Net Local-Monitor. These components are shown in Fig. 1 as layers.

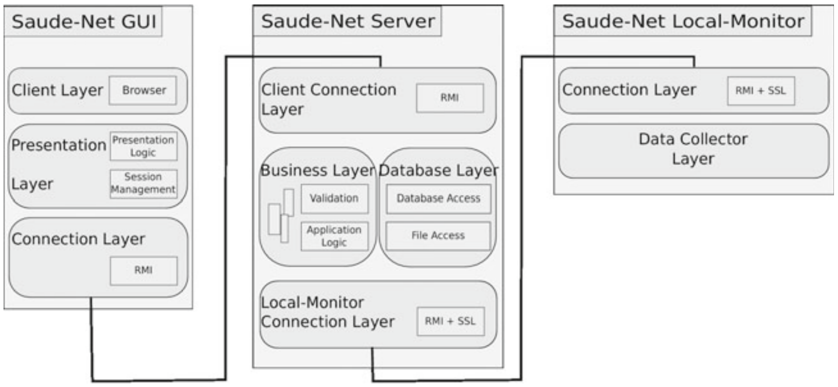


Fig. 1. Layers of modules of Saude-Net system

The first component is designed to allow the administrator to operate on rules and actions. It is responsible for simple validation and data presentation. The second module is the main component of the developed system, which is responsible for validation of rules and actions. It chooses the best action as a reaction to the failures reported. The last component is responsible for collecting of monitoring data. This data may be obtained from more than a single Zabbix monitoring server. This system is also able to exploit other monitoring tools. In order to do this it is necessary to adapt the implementation of the data collector to a low-level monitoring facility. Saude-Net is customizable by the system administrator.

Interactions between the modules as well as the basic architecture of the Saude-Net system are depicted in Fig. 2.

This system is developed as a tool on top of external low-level systems. It requires the system administrator to define both rules and actions for the whole distributed system under monitoring. Each type of monitored resource should have relevant actions associated with it. Nonetheless it is possible to deploy default actions which will be good enough for more than one type of resource, e.g. a function which is able to restart the appropriate resource, or a function which enables another instance of the monitored service.

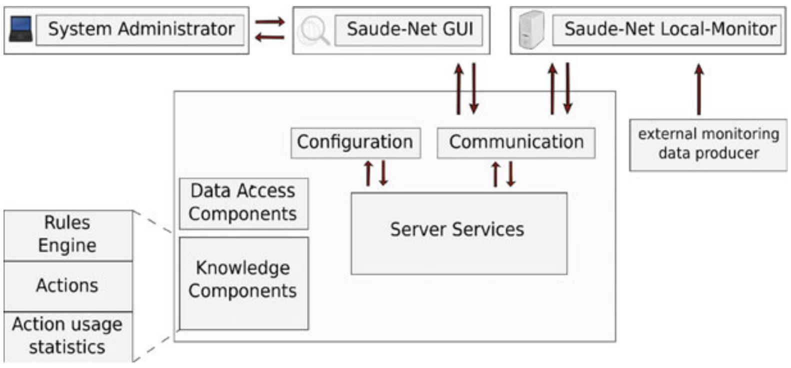


Fig. 2. Architecture of the Saude-Net system

3.2 SAMM Compliant Agent

The SAMM Compliant Agent (SAMM-CA) is the second system that is going to be presented. SAMM-CA is the solution for automation of system monitoring, based on semantics and agents. It is an extension to the Semantic-based Autonomic Monitoring and Management (SAMM) monitoring tool [16], which provides description of monitored resources in a flexible form of ontologies. SAMM is used as a low-level monitoring tool because it uses ontologies to describe monitored resources. It also allows to provide many statistics of system resources like CPU usage. SAMM exploits methods for reasoning using ontologies and data mining [13], which will be helpful for intelligent agents. SAMM-CA implements an agent-based approach for the automation of system management. The agent uses predefined actions and knowledge. Unlike Saude-Net, SAMM-CA uses knowledge described using fuzzy logic. This approach goes beyond the static nature of Saude-Net. The actions supported by SAMM-CA are associated with fuzzy sets. It allows to evaluate the purposefulness of actions described in terms of membership function with a few function types, like Gauss function or trapezoidal function. The use of fuzzy logic does not require to know the whole system model. It also allows to manage dynamically the borders of the membership function. Each set describes a single action. An action may be described by many sets. The measured parameters are used to define which action can be used for the observed failure. The agent allows to define the ranges of sets for which the associated action cannot be used. These sets are depicted in Fig. 3.

The borders of each fuzzy set are determined by the following set of formulas. For the left border there are:

$$\begin{aligned} y &= ax + b \\ a &= 1(x_1 - x_0) \\ b &= (-x_0)/(x_1 - x_0) \end{aligned} \tag{3}$$

The value x_0 describes the point where the set is starting with a value equal to 0 and the x_1 point defines the first point where the set possesses the

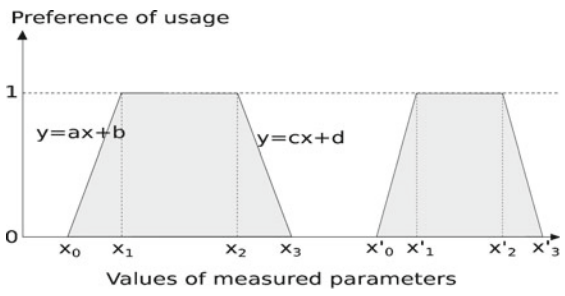


Fig. 3. Representation of fuzzy sets for actions in SAMM-CA

maximum value, which equals 1. For the right border the formulas are similar to the previous ones:

$$\begin{aligned} y &= cx + d \\ c &= -1(x_4 - x_3) \\ d &= x_3/(x_4 - x_3) \end{aligned} \tag{4}$$

The above formulas are used to determine the borders of these sets. The borders are modified while SAMM-CA works, whenever an action is performed. If this action ends with a positive result in the monitored system, the ranges of the set which describes that action may be extended. Otherwise this set may be reduced. Due to this fact the width of the range of the monitored parameter may be changed to fit the observed failures better.

The concept of SAMM-CA was to develop a solution which will enable resources management, especially decision-making, as close to the resource as possible. The developed agents possess their own knowledge. They are also allowed to exchange their knowledge and obtained information. The behavior of the agents is defined by associated modes. These modes are meant to enable the agents to send information about the failures observed, obtain information, and exchange their knowledge (see Table 1).

Table 1. Functions associated with agent modes

Mode	Functionality		
	Possess History	Learn	Exchange information
1	No	No	No
2	Yes	No	No
3	Yes	Yes	No
4	Yes	Yes	Yes
5	Yes	No	Yes

The global solution to the issue of the way SAMM-CA realizes its monitoring tasks is to optimize agents' knowledge by dividing it to separate entities. Each one of them learns separately but is connected with others, communicates with them and exchanges parts of knowledge. In some situations when one agent is unable to resolve the observed problem it has to cooperate with others to respond. All agents can have an effect on the observed environment.

Fig. 4 depicting the main modules of SAMM-CA presents connections between modules and actions which are performed on these connections.

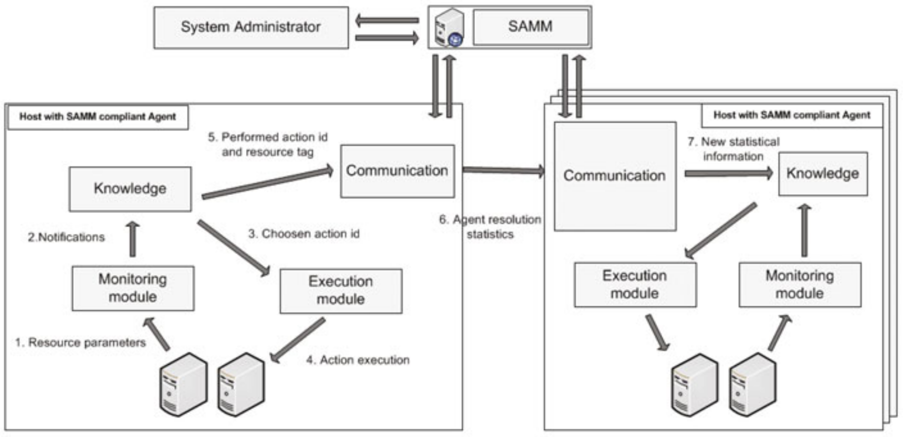


Fig. 4. Architecture of SAMM Compliant Agent

As mentioned before, each agent is able to perform its independent actions. One of the goals of this system is to provide a sufficient number of agents-experts to cover the whole scope of resources monitoring/management for a given infrastructure, each agent being an expert in a narrow area. The agents are assumed to exchange their knowledge and problems to be solved whenever necessary.

The agent-based solution presented in this paper is aimed at performing independent actions which are currently based on statistics and historical data. Each action is associated with a history of its usage. When a failure is observed in a monitored resource, SAMM-CA tries to use these actions whose fuzzy sets include the most recent value of the monitored parameter. The actions which are used most often may have larger widths of their fuzzy sets. These actions are more suitable for a wider scope of measured parameters and may be used when it is possible. When a value from within a wider range is observed, a relevant action can be performed and should be good according to the current knowledge of the agent.

The ranges of sets are closely related to the history of actions usage and they may be modified after each action execution.

Due to this fact these actions will be better suited for the observed problems – the sets are modified to better indicate, in which situations and for what value of the monitored parameter, the action may be performed.

3.3 Comparison of Presented Solutions

Table 2 presents a short comparison of both presented approaches, limited to the most important features.

Table 2. Comparison of the approaches implemented in Saude-Net and SAMM-CA

No.	*Saude-Net*	*SAMM-CA*
1	agent-less solution	agent-based solution
2	using rules engine	using fuzzy logic
3	centralized point of knowledge	knowledge distributed into separate agents
4	static expert system which requires administrator to describe all possible actions	dynamic knowledge, an agent is able to modify its knowledge and to add new fuzzy sets which will describe possible actions
5	actions have to be designed for the whole monitored system	actions may be defined individually for the separate areas of the monitored systems
6	developed for medium size systems	designed for all sizes of distributed systems
7	increase in the number of resources leads to modification in the central knowledge base	increase in the number of monitored resources implies modification of agents' knowledge or a need to start another agent for these resources

Both presented solutions were developed to automate monitoring of distributed systems such as grids. They were preliminarily tested on the PL-Grid test infrastructure. In the future it will be possible to run these solutions on grids for monitoring data storage. The SAMM-CA will be able to monitor a distributed data storage oriented infrastructure.

4 Evaluation Results

Both solutions presented above have their pros and cons. The first one, which is Saude-Net, may prefer only one action. In some cases it may be dangerous since there is a chance that a single action will be used all the time. Due to this fact, the monitored system might be unable to cope with some failures. It is caused by the lack of feedback from the monitored system.

The tests presented in this section were performed on a small infrastructure which contains only four hosts with LAN interfaces. On each host with Unix there was installed an Apache Server. This test infrastructure contains also one router and a switch. All four hosts were deployed in the same subnet. In all the tests this infrastructure was treated as one distributed system.

The behavior of the Saude-Net system through a sequence of failures is presented in Fig. 5.

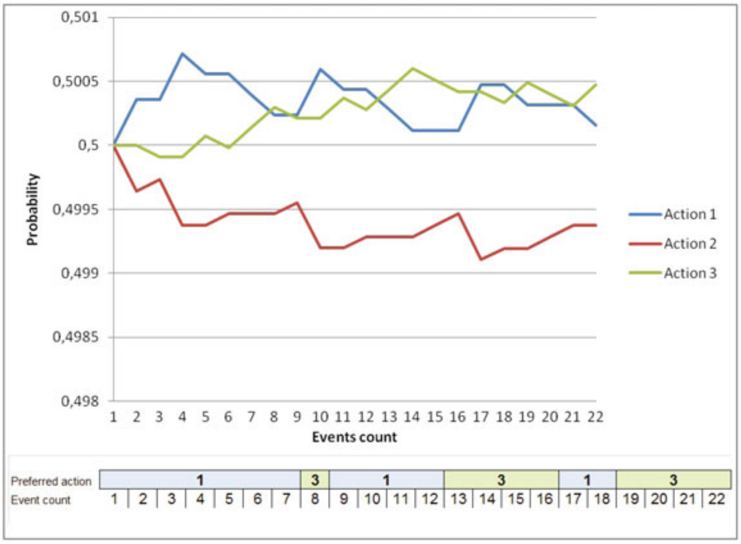


Fig. 5. Preference value course for three independent actions

The probability value in the chart shown in Fig. 5 defines which action is more suitable for the monitored resource. This chart represents available actions as a function of time for the monitored resource, which was the Apache Server, run on one of the experimental hosts. For this resource three possible actions were defined:

– Action 1 – server restarts on the same host.
– Action 2 – server starts on the alternative host.
– Action 3 – server is suspended for two minutes and after that this action has to restart the server on the same host.

All the actions were defined as Unix shell scripts. During the run of the system the preference values were changed due to the failures of two other servers for which the system administrator has defined only two of the above actions as available. During the test the preference values of all three actions were changed. On the presented chart the PV values of each action for Apache Server are shown. The system tried to manage its actions. It changed their preference values after it had tried to resolve each error notification. In the presented solution, action 2 is considered the worst one. Our system should use other actions but when there are no other choices it has to choose action 1. Saude-Net is able to cope with the actual situation because it not only uses the actions with the highest preference value but it can perform actions which are described as the worst ones if they are the only choice.

The second approach presented in this paper is SAMM-CA. Using this solution it is possible to perform one of the available actions and to fit them to the problem observed. In its current version it is able to exchange statistical information about the observed failures and knowledge. When comparing these two solutions it might be noted that the first one, Saude-Net, uses rules which determine possible actions for each resource. This solution is not apparently as dynamic as SAMM-CA, which uses fuzzy sets. It allows to match a better action with a failure at a lower cost. Another difference is partially related to the system architecture: while Saude-Net uses the external monitoring tools to obtain data transmitted to a central point, which is a Saude-Net server, SAMM-CA is able to monitor resources by itself and to decide on actions locally. Saude-Net responses are slower than those performed by SAMM-CA, mostly due to communication costs. SAMM-CA uses local actions so they may be better suited for the monitored resources. These actions may be personalized for these resources like in Saude-Net, but in this case the system administrator is able to provide different actions for each agent. The second solution, SAMM-CA, was tested in the monitoring environment as well. These tests were dependent on the mode of each agent. In mode 1 the agents behave like a static expert system. When they are run in a different mode they are able to learn and communicate. During tests SAMM-CA was able to react on Apache Server failures like the Saude-Net system but its reactions were better suited when the borders of sets were fuzzy shaped. When the function which describes sets was more similar to a rectangle, the agent behaved like Saude-Net or any other rules-based expert system. The results collected from the agents are shown in Fig. 6.

This chart presents the test results where the x axis presents the count of the tested data to all data and the y axis shows the percentage of the correctly classified actions. This chart presents two tests. Each one was started with basic

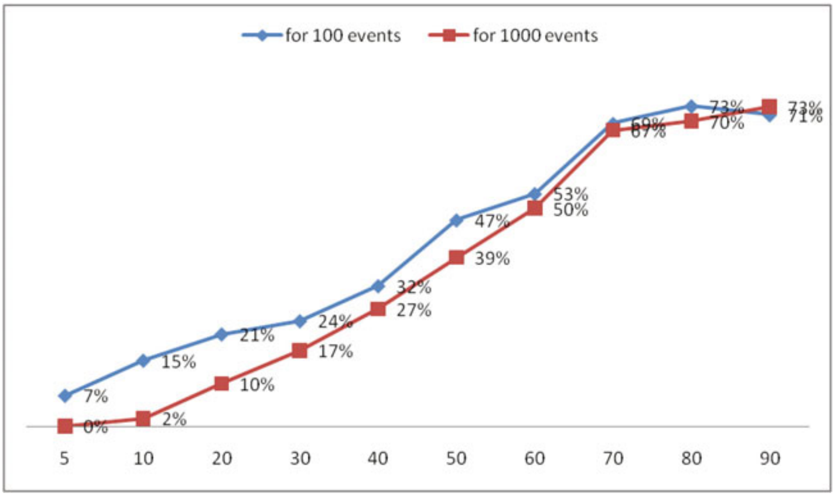


Fig. 6. Percentage of right classification of actions as a function of the size of the learning set with respect to the whole data set used for tests. The results for two independent tests are involved.

knowledge (the same for both of them). The results evidence that the developed system is able to learn and fit to the observed failures.

5 Conclusions and Future Work

The approaches presented in this paper were aimed at creating a solution for automation of system monitoring, i.e. decision-making what action should be taken to cope with a failure of a resource. The first solution is a rule based approach implemented in the Saude-Net system built on top-level of existing low-level monitoring systems, e.g. Zabbix. The second solution is a fuzzy logic-based approach called SAMM-CA. It uses agents to manage resources in a decentralized manner. It is implemented as an extension to the SAMM monitoring tool. In the Saude-Net system we introduced a rule-based knowledge engine. This facility is aimed to be used in the systems where the administrator can define multiple rules for each monitored resource under consideration. This tool is designed to learn which solution is the best for the observed situation, based on the concept of so called preference value. This system has a few drawbacks which may make it a little bit unstable due to the fact that the point of knowledge is central. The Saude-Net approach is an agent-less solution. There is a central point where data from external monitoring tools are first analyzed and afterwards this system can choose an action best suited to the observed failure.

The second approach is an agent-based solution implemented in SAMM Compliant Agent, SAMM-CA. Like the previous one, it extends another monitoring tool, specifically, the SAMM monitoring system which uses ontologies to represent knowledge on monitored resources. Unlike Saude-Net, which only uses rules, this system exploits the agent-based approach and fuzzy sets which are more flexible and should be more suitable for monitoring automation. This approach is not assumed to possess a centralized knowledge engine. Instead, it features distributed nature of its knowledge which allows to create a set of independent decision makers.

The SAMM-CA tool is able to better fit the monitored resources. It decides on actions locally so it is safer because it does not have to send an action description through the network. The Saude-Net system stores the whole knowledge in one central point. In large distributed systems this tool responds to captured failures rather slowly. In contrast to Saude-Net, the second presented solution is able to react much faster because its knowledge is distributed and is closer to resources. This allows to manage fragments of a system by one agent only.

To summarize the above, the SAMM Compliant Agent is a more suitable solution for the automation of monitoring of large grid infrastructures such PL-Grid: it uses fuzzy sets which underly a more flexible mechanism than rules, thus allowing for better action matching. On the other hand, the Saude-Net system is quite sufficient for the automation of infrastructure parts monitoring, where the communication costs can be outweighed by efficient event handling.

In the future the functionality of SAMM-CA is going to be extended. The system will be able to learn only one domain of the network resources. Another modification – which will be implemented – is related to the agents: they will be able to exchange descriptions of the observed failures. This solution should make these agents more flexible. Each agent shall be a kind of expert system. Also in SAMM-CA an ability to manage its agents will be implemented. The system administrator will be able to choose which agent will be responsible for what kind of resources. For example it will be possible to configure one agent to monitor only storage devices and another agent to monitor only computational units or servers. The agents will be able to react only to a selected set of failures. If one agent will be unable to resolve an observed problem, it will delegate the problem to another agent.

References

1. PL-Grid project site, http://www.plgrid.pl
2. Figueria, J., Greco, S., Ehrgott, M.: Multiple Criteria Decision Analysis. Springer Science + Business Media (2005)
3. Stone, S.: Monitoring System Comparison. The Forbin Group (2007)
4. Cetnarowicz, K., Kozlak, J.: Multi-agent system for decentralized computer network management. In: Binder, Z. (ed.) Proc. MCPL 2000: Management and Control of Production and Logistics, 2nd IFAC/IFIP/IEEE Conference, vol. 1, pp. 469–474. Pergamon, Oxford (2001)
5. Cetnarowicz, K.: From Algorithm to Agent. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part II. LNCS, vol. 5545, pp. 825–834. Springer, Heidelberg (2009)
6. Comparision report on network monitoring systems (nagios and zabbix). Technical report, Malaysian Administrative Modernisation and Management Planning Unit (MAMPU) (2010)
7. Massie, M.L., Chun, B.N., Culler, D.E.: The ganglia distributed monitoring system: Design, implementation, and experience. Parallel Computing (2004)
8. Ribler, R.L., Simitci, H., Reed, D.A.: The Autopilot Performance-Directed Adaptive Control System. FGCS 18(1), 175–187 (2001)
9. Słota, R., Król, D., Skałkowski, K., Kryza, B., Nikołow, D., Orzechowski, M., Kitowski, J.: A Toolkit for Storage QoS Provisioning for Data-Intensive Applications. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 157–170. Springer, Heidelberg (2012)
10. Intellipool Network Monitor, Intellipool AB (2011), http://www.intellipool.se/4_0/doc/index.html (access: June 15, 2011)
11. Hyperic HQ, System Monitoring Software, http://support.hyperic.com/display/DOC/HQ+Documentation (access: June 11, 2011)
12. Xiao Luo, X., Wang, Y.: The Research of Software Automation on Monitoring System. In: ISCID 2010 Proceedings (2010)

13. Han, J., Kamber, M.: Data Mining Concepts and Techniques. Academic Press (2001)
14. Funika, W., Szura, F.: Automation of decision making for monitoring systems. In: Bubak, M., Turała, M., Wiatr, K. (eds.) Proc. CGW 2010, October 11-13, pp. 164–171. ACC Cyfronet AGH, Kraków (2010)
15. Funika, W., Szura, F.: Agent-Based Monitoring Using Fuzzy Logic and Rules. In: Proc. 4th ACC Cyfronet AGH Users' Conference KU KDM 2011, March 9-11, pp. 28–30. ACC Cyfronet AGH, Kraków (2011)
16. Funika, W., Kupisz, M., Koperek, P.: Towards autonomic semantic-based management of distributed applications. In: Computer Science Annual of AGH-UST, vol. 11, pp. 51–63. AGH Press, Kraków (2010)

A Toolkit for Storage QoS Provisioning for Data-Intensive Applications

Renata Słota[1,2], Dariusz Król[1], Kornel Skałkowski[1,2], Bartosz Kryza[1], Darin Nikołow[1,2], Michał Orzechowski[2], and Jacek Kitowski[1,2]

[1] AGH University of Science and Technology, ACC Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków, Poland
[2] AGH University of Science and Technology, Faculty of Electrical Engineering, Automatics, Computer Science and Electronics, Department of Computer Science, al. Mickiewicza 30, 30-059 Kraków, Poland

Abstract. In this paper, we present a toolkit named FiVO/QStorMan for supporting data-intensive applications in distributed environments with storage-related Quality of Service provisioning. Main features of the toolkit are: explicit definition of non-functional requirements regarding storage resources, usage of semantic descriptions of the available storage infrastructure and active monitoring data concerning the infrastructure workload and users operations. In particular, we describe user interfaces of the toolkit along with descriptions of main parts of the toolkit. In addition, the paper describes results of the performed experimental evaluation of the toolkit which confirm the effectiveness of the proposed approach for the storage-related QoS provisioning.

Keywords: storage management, SLA, QoS requirements, Virtual Organization, ontology.

1 Introduction

The concept of Virtual Organization has been already around for several years, ever since the vision of the Grid computing paradigm was sketched in [1]. However, over the years this concept was significantly reduced to simple management of user credentials which often was sufficient for scientific applications; where communities of users tended to trust each other. As Grid and Cloud computing infrastructures are becoming applied to commercial scenarios the need for more mature approach to Virtual Organization concept is necessary. This fact was also identified in a recent report [2], where issues, such as security, policies, contracts, monitoring of Service Level Agreement (SLA), VO requirements, and introduction or adoption of proper standards, are discussed.

One of the solutions addressing this problem is the FiVO framework (Framework for Intelligent Virtual Organizations) [3]. This framework consists of several components, which, among other things, allow users to include publication of metadata about organizations resources and capabilities, discover partners for creating Virtual Organizations, invite partners to join VO, negotiate VO contracts in a distributed manner (including specification of security and SLA related statements), automatically configure security and monitoring components of the Grid infrastructure as well as to enforce contracts during the VO operation. Apart from some specific frameworks which address particular issues [4,5], the FiVO framework aims at comprehensive support of various aspects of grids usage.

In this paper, we present an extended version of the FiVO framework with ability to manage and monitor SLA rules, with focus on Grid based applications that heavily exploit storage resources. All this and more was combined to form a toolkit named QStorMan from Quality-based Storage Management.

The rest of the paper is organized as follows. In Section 2, we describe the most important features of data-intensive scientific applications. Then, in Section 3, we discuss a number of existing solutions related to optimization of data access time, ensurance of QoS, and SLA management. In Section 4, we describe the main goals of the QStorMan toolkit, supported use cases, user interface and implementation details. Next, in Section 6 we present an experiment evaluation of the presented toolkit. The paper is concluded in Section 7 along with planned future research.

2 Data-Intensive Applications

Generally the term *data-intensive* applications conforms to applications which perform many I/O operations; including both commercial/enterprise transaction systems, which execute many short I/O operations, as well as heavy scientific applications, which perform small number of long-lasting I/O operations. Since the FiVO/QStorMan toolkit constitutes a grid middleware extension, its design is oriented on heavy data-intensive grid applications, rather than on transaction systems. Data-intensive scientific applications are usually associated with execution of computations on large data sets (e.g. medical data or data provided by sensor networks) or large data structures (e.g. trees or arrays).

The algorithm executed by such applications relies on reading portions of data, processing it, and dumping computations results. Obviously, some applications perform mostly write or read operations, in that case they respectively require high data transfer rates during output or input operations. If the amount of transferred data is so large that its loading/saving time has significant impact on the whole application execution time, then the application can be considered a data-intensive one. A special type of data-intensive applications constitute the *out-of-core* applications [6], [7] which operate on very large data structures such as trees or arrays. Data structures processed by that kind of applications are so large, that it is impossible to load them wholly into the operating memory. In order to cope with their processing they are only loaded partially into the memory, while the rest of the data remains on storage devices.

Large data structures can be processed by the out-of-core applications in two ways. The first relies on explicit loading of subsequent parts of a structure into the system's memory from an input file. The second does not require explicit part-by-part loading of a structure, instead it assumes that the structure is wholly loaded into the memory. Partial loading of the data structure in that case is accomplished implicitly by the operating system virtual memory mechanisms. Naturally, provisioning of storage resources with QoS requirements for that kind of out-of-core applications is much more difficult due to inherited unpredictability of the operating systems' virtual memory mechanisms.

3 Related Work

The related research studies presented further are focused on two main fields:

1. data access time optimization for distributed data-intensive applications,
2. QoS support for data storage systems.

When it comes to the data access optimization, some studies refer to the performance of data access of MPI applications. In [8] a method for increasing the efficiency of data access for collective I/O for MPI applications using Lustre is proposed. The idea relies on using subfiles which are stripped to a few disks only. The subfiles are joined as a single file using the Lustre file joining mechanism. The experiments show significant improvements for collective write operations. In [9] the performance and scalability problems of MPI-IO with Lustre are addressed. The authors proposed a software library, called Y-lib, which optimizes data access by redistributing the data. The data distribution in the studies mentioned above is done without monitoring of the current state of storage resources.

As for the second field, QoS issues, being critical for many storage systems, have been widely studied. In [10] a framework based on pNFS is proposed for supporting storage QoS in Grid environments. Three additional modules enhancing the pNFS have been defined – QoS broker, economic module and storage system monitor. An implementation for Linux is announced. In [11] the replica placement problem in grid environments for achieving QoS is studied. Two heuristic algorithms for approximating the optimal solution have been proposed. In [12] a performance interface is proposed enabling efficient QoS support for storage systems. The interface, which represents a reference storage system, delivers data for a given I/O workload with performance not worse than that of the real referenced storage system. The implementation of the interface is based on a machine learning model. The presented papers propose solutions for selected storage systems narrowing the usability range to these storage systems. In contrast, the approach presented in this paper is a general one as it takes into account various heterogeneous storage devices and systems. Additionally, we use semantic technologies for defining QoS metrics what makes our approach independent from storage resources.

4 QStorMan Toolkit

The QStorMan toolkit constitutes a set of tools providing QoS for data-intensive applications in Grid environments. As input, QStorMan accepts non-functional requirements for storage resources. The requirements can be defined for the following classes: Virtual Organizations, users, and applications. As a VO contains a number of users, each user "inherits" all the requirements defined for his/her VO automatically. A similar rule applies to a user and his/her applications. Obviously, additional requirements can be defined for a user which will override the requirements of the VO.

If Grid applications perform on worker nodes then QStorMan responds to the following requests:

– finding a worker node, for an application dispatching, with access to storage resources which meet basic requirements of a data-intensive application, e.g. storage capacity,
– finding a storage resource accessible from a certain worker node where data produced by an application should be placed during its execution based on non-functional requirements and current workload on the resources.

The decision of selection either a worker node or a storage resource is made based on a storage environment description and information received from a monitoring system dedicated for storage resources. The environment description contains rarely updated information, e.g. storage resource capacity, while the monitoring system provides information about dynamic parameters of storage resources, e.g. current read/write transfer rate. QStorMan provides a "best-effort" type of service since there is no guarantee that the defined non-functional requirements will be met by the selected storage resources.

The QStorMan toolkit consists of a number of components, some of them are directly used by the user or user applications while others reside on the server side. From the end user's point of view, the QStorMan toolkit provides a web interface for storing non-functional requirements and two programming libraries for supporting data-intensive applications. By interacting with these two components, the end user can influence distribution of their data by specifying QoS high-level requirements only.

4.1 QStorMan Supported Use Cases

The QStorman toolkit supports a set of use cases which depend on the possibility of source code manipulation of a Grid application:

– The most basic use case supported by the QStorMan toolkit includes selection of a worker node where a Grid job should be dispatched depending on non-functional requirements of the application regarding storage resources. The user's requirements are provided through the QStorMan WebFace and the toolkit returns information about the most suitable worker node in form of a part of the Job Description Language (JDL) file. The returned part of the JDL file can be easily merged with the rest of the job description file and sent to a Grid batch system. No other changes in the job description file of the application are required. This use can be also easily integrated with some grid middlewares which support management of jobs dispatching [13,14].

- The second use case conforms to selection of a storage resource within a single supercomputing center, based on local information. In this scenario, the QStorMan toolkit intercepts all requests to the underlying cluster file system and decides where the produced data should be put. This is achieved by a technique called library pre-loading. By using this technique, we can support applications whose source code can not be modified, most notably, all the legacy applications. The storage resource selection process is based on requirements stored by users in an ontological knowledge base through the QStorMan WebFace. The requirements relate to a concrete application, every application of a concrete user or to every application of every user who is a member of a concrete Virtual Organization.
- The last use case supports development of new applications. By using a dedicated programming library, a grid application developer can provide requirements for each new file created by the application explicitly in its source code. It is also possible to assign different requirements to different files. Nonetheless, this way of the QStorMan toolkit usage requires applications source code modification, thus its application area is tightly constrained.

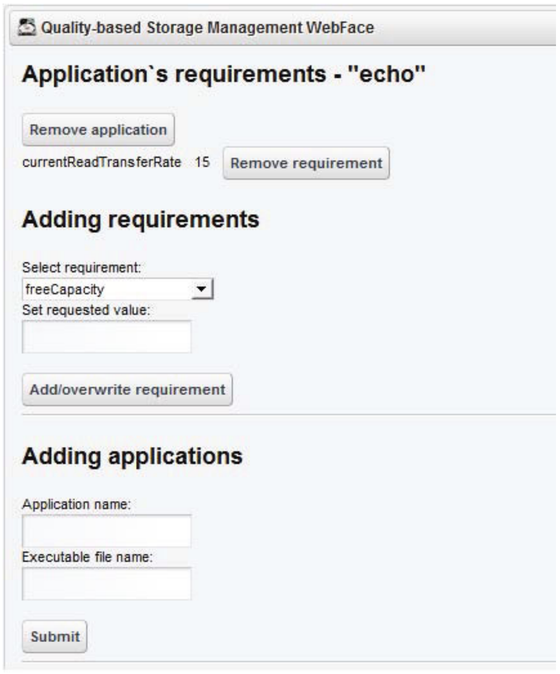
The first use case concerns the scheduling phase of an application. The user can imply this process by finding the most suitable worker node based on given requirements. The last two use cases concern application runtime during which a storage resource is selected based on non-functional requirements. Thus, the first use case can occur with one of the last two, i.e. the user first decides to which worker node the application should be dispatched and then QStorMan decides to which storage resources the data generated by the application should be put.

4.2 Toolkit User Interfaces

QStorMan provides two types of user interfaces: a web-based graphical user interface for Grid end-users who execute data-intensive applications on the Grid and an Application Programming Interface (API) for Grid application developers who create new data-intensive applications.

The QStorMan WebFace – the first user interface – is a portlet which can be embedded in a web portal, e.g. the PL-Grid portal, as a separate element along with other portlets. By using the QStorMan WebFace, a user can find the most suitable worker node for an application based on given requirements. The second functionality provided by the QStorMan WebFace is management of non-functional requirements for users applications, i.e. a user can register an application and requirements for it. A screenshot of the QStorMan WebFace is presented in Fig. 1.

At the top of the figure we can see the form which is used for providing QoS requirements. The requirements are then used for searching for an appropriate worker node, according to the specified requirements. After the form submission, the QStorMan WebFace returns a part the JDL file which has to be attached to the submitted job. Below the initial form, non-functional requirements can


Quality-based Storage Management WebFace

Application`s requirements - "echo"

Remove application

currentReadTransferRate 15 Remove requirement

Adding requirements

Select requirement: freeCapacity

Set requested value:

Add/overwrite requirement

Adding applications

Application name:

Executable file name:

Submit

Fig. 1. The QStorMan WebFace

be defined, either at the user level, i.e. these requirements concern all the user applications, or for a specific application.

While the QStorMan WebFace provides a graphical user interface, the Storage Element Selection (SES) libraries equip user with an API for new software development as well as for existing applications adaptation without introducing any code modifications. In the first case, a developer uses the API for creating files within the distributed file system based on a specified set of non-functional requirements regarding storage resources. The requirements are encapsulated in an instance of the `StoragePolicy` class.

In the latter case, i.e. supporting a legacy application, QStorMan provides a system library which intercepts all the IO requests coming from the application to a distributed file system automatically. During the interception, the library communicates with the rest of the QStorMan toolkit to select the most suitable storage resource at the given moment based on non-functional requirements assigned to the application.

5 Implementation

An overview of the QStorMan architecture is depicted in Fig. 2. A set of non-functional requirements must be given as a framework input. The user interacts with the QStorMan toolkit either via the QStorMan WebFace or SES libraries.

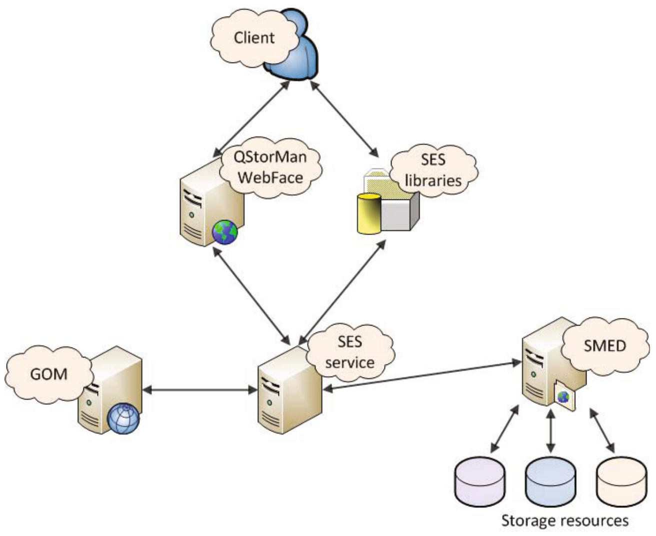


Fig. 2. The architecture of the QStorMan toolkit

The requirements of users application are sent to the SES service, which – according to the requirements, the information from monitoring service (SMED), and knowledge base (GOM) subsystems – finds the most appropriate storage resource.

The QStorMan toolkit has been implemented using modern technologies and open standards. It follows the SOA paradigm principles in order to provide its functionality with a number of loosely coupled services, each of which is exchangeable as long as it possesses a necessary interface. The following subsection provides information about internal implementation of each element of the QStorMan toolkit.

5.1 Semantic Descriptions

All information about storage resources and user requirements is stored in an ontological knowledge base for further use. The knowledge base component is called GOM (Grid Organizational Memory) [15] and it constitutes a distributed semantic knowledge base, supporting evolution and queries concerning semantic information stored in the Web Ontology Language (OWL). GOM provides a standard Web Service interface for querying the underlying knowledge using the SPARQL language and updating the knowledge through a custom protocol. The knowledge evolution is fully supported by both fine grained as well as coarse grained modifications of the ontologies. Additionally, GOM has been integrated with the X2R tool [16] which supports dynamic import of knowledge from relational databases, LDAP repositories or XML data sources into a predefined OWL ontology, based on custom mappings.

The QStorMan toolkit exploits semantic descriptions to integrate different components with a single, shared data model. By using ontologies stored in the GOM knowledge base, each of the QStorMan toolkit subsystem uses the same set of concepts with a defined meaning. In ontologies, the QStorMan toolkit stores information about available storage resources and defined non-functional requirements for different subjects, i.e. VOs, users or applications.

Definition of different storage resources, available in a distributed environment, are described in a single ontology named *Storage.owl*. It contains types of storage resources along with their specific features. The ontology is based on the previously created ontology within the OntoStor project [17] and it is compatible with the C2SM model [18]. Each type of storage resource has a number of attributes attached, which are represented by the *Attribute* class. Each attribute represents a measurable quality parameter of the resource, e.g. `averageWriteTransferRate` or `freeCapacity`. The ontology divides storage resources into two groups: *physical* and *virtual* storage resources. *Physical* resources represent hardware devices dedicated to store data, e.g. hard drives or disk arrays. Such devices are often accessible via a standard file system. However, if there is a need to provide users with a coherent view of storage parameters, for example global storage capacity, one can aggregate several physically distributed hardware devices into a single system. To describe such a concept, a *virtual* storage resource has been introduced. This class encompasses e.g. distributed file systems which can consist of several physical devices. An example of such a resource is a concept of a *pool* in the Lustre file system [19] which simply describes a set of disks.

The second ontology defines non-functional requirements at different levels of abstraction. Non-functional requirements (represented by the *NonFunctionalRequirement* class) can be linked with: a Virtual Organization, a user or an application via an instance of the *DataSLA* class. At the Virtual Organization level, the defined requirements can be treated as global settings for all users who are members of a concrete VO. Currently, only an administrator of Virtual Organization can define requirements at this level directly in the VO knowledge base. At the user level, the defined requirements concern all users applications. The lowest level concerns applications themselves, i.e. each application can have different requirements defined.

Both the user and the application levels are supported by the *QStorMan* portlet. Basic operations such as creation, overwriting and removing are provided by clickable and intuitive Graphical User Interface in the portal. An instance of the ontology was developed to describe sample requirements for test users. The instance ontology is used by the SES subsystem, i.e. *libSES* and *SES service*, to:

– get requirements of an application or a Virtual Organization for a given user,
– find a Virtual Organization whose member a given user is.

5.2 Storage Resources Monitoring

The SMED subsystem implementation follows the SOA paradigm principles in order to utilize its benefits such as adaptability, flexibility and others. Architecture of the subsystem assumes division of a deployment environment on two types of nodes: storage nodes (which possess directly mounted file systems) and share nodes (which exposes gathered data for external subsystems). The SMED subsystem consists of two types of services:

– The monitoring service – the service is deployed on storage nodes and is responsible for collecting parameters of monitored storage resources according to the mentioned C2SM model. Current version of the service supports monitoring of disk arrays, HSM systems, local disks and cluster file systems. Since the service has a plugin architecture, it can be easily extended on new types of storage services by implementation of appropriate plugins.
– The sharing service – the service provides an external interface for sharing data delivered by the monitoring services to other subsystems. Single instance of the service is usually associated with a few instances of the monitoring service.

Communication among the system services utilizes the Enterprise Service Bus (ESB) architecture which is currently the most promising integration technology for the SOA-oriented systems. The external interface exposed by the sharing service is implemented using the RESTful approach, since it constitutes the most convenient way of representing resources through the web.

Storage resources monitoring is performed by the SMED system in order to provide up-to-date information about the resources state. Parameters provided by the SMED subsystem can be of twofold type: static and dynamic. Static parameters have constant values and are independent from time, thus they can be retrieved only once during the subsystem startup and then cached in memory. Examples of such parameters are total resource capacity, disk array RAID level, etc. In turn, dynamic parameters can change in every moment, hence they have to be retrieved afresh for each new request. Examples of dynamic parameters are: the read/write transfer rate (which is affected by ongoing I/O operations), free resource capacity, etc. The way of retrieving resources parameters depends on a specific type of a resource and parameter type.

Current version of the subsystem retrieves parameters in three ways: from the subsystem configuration file, from system commands invocations and from active measurements. The subsystem configuration file provides some basic parameters about monitored resources which have fixed value and are set by the system administrator (for example resources mount points). Most of static and dynamic parameters such as total/free capacity are provided by the Unix system commands invocations like **df** or their counterparts for virtualized storage resources (e.g. the **lfs df** command for the Lustre FS [20]). Parameters specific for a concrete type of a storage resource are obtained using API or system commands provided by the resource vendor. For instance, parameters of the Lustre FS pools are obtained using the following two commands: **lfs pool_list**, **lfs df -pool**. Parameters resulting from system commands invocations are passed to the subsystem using the *screen-scripting* technique applied to the commands output. This technique of parameters acquisition is not very convenient and effective, but unfortunately most of storage resources vendors do not provide any programming API for the resources parameters acquisition. The last way of parameters retrieving are active measurements. This way is used in case of transfer rate parameters, since they can not be retrieved in any other way. In order to measure storage resource read and write transfer rates, the SMED subsystem periodically writes and reads certain amount of data to the resource, and measures duration of these operations. The measurements are executed by means of the **dd** Unix system command with the **direct** flag in order to avoid data caching. The intervals between succeeding measurements and amount of written data are configured by the system administrator, since they are dependent on a specific type of a storage resource. This way of parameters acquisition is the most invasive of all ones listed, however the transfer parameters are essential for the whole QStorMan system logic.

5.3 Storage Resources Selection

The process of storage resource selection is handled by the Storage Element Selection (SES) subsystem which consists of two elements: two programming libraries and a server side service. The two libraries are called *libses-wrapper* and *libses* respectively. The *libses-wrapper* supports legacy applications, whilst the *libses* is aimed at development applications. The libraries are implemented using C and C++ languages. The programming libraries provide a pure programming interface for the QStorMan toolkit, all computations are executed on the server side by the SES service which is implemented in Python.

The service exposes its functionality using the REST model and it is accessed using the standard HTTP protocol, thus it can be used from almost every programming language. The service input constitutes some of the following information:

- non-functional requirements in an explicit form, i.e. names and values of a number of requirements,
- information about the user and the host name which is sending the request. The information is used by the service to retrieve requirements defined for a Virtual Organization to which the user belongs,
- information about the user, the host and the application for which the non-functional requirements should be retrieved from an ontological knowledge base.

In all cases, after collecting non-functional requirements, the SES service communicates with GOM to retrieve a current description of a storage environment and with SMED to retrieve current values of the metrics which represent the requirements. The communication with GOM is implemented with SOAP-based web services and the communication with SMED is implemented using the JSON notation.

Based on the retrieved information, the SES service computes a “distance” between the given requirements and each of the available storage resources. The smaller the distance is, the better the storage element.

6 Performance Evaluation

To evaluate the QStorMan toolkit, we exploited a storage infrastructure within the Academic Computer Centre (ACC) Cyfronet AGH which comprises an enterprise class installation of the Lustre file system. The infrastructure is depicted in Fig. 3. The Lustre file system installation consists of 12 FATA Hard Drives managed by 4 Object Storage Servers (OSS) and Meta-Data Server (MDS). The network fabric used within the installation is FibreChannel between storage devices and Lustre OSS/MDS servers, 10 GbE between Lustre OSS/MDS servers and network switches and 1 GbE to each worker node.

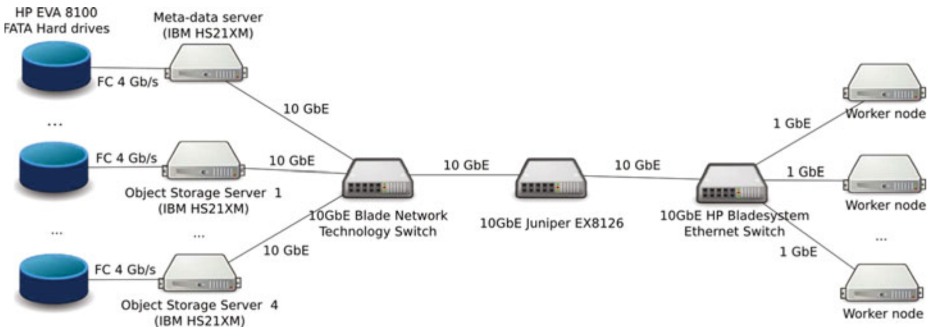


Fig. 3. Storage infrastructure used in the QStorMan performance evaluation

Using this storage infrastructure, we implemented a test case to evaluate the QStorMan toolkit. The test case simulates a number of users who run data-intensive applications using a standard Grid batch system at ACC Cyfronet AGH. The batch system schedules Grid jobs to worker nodes which have access to the Lustre installation presented above. The infrastructure, which was used during the evaluation, is a part of the PL-Grid production infrastructure provided by ACC Cyfronet AGH. Tests were scheduled during a period of time when the infrastructure was less loaded based on information from the Cyfronet's monitoring system. Also, in order to provide enough data for averaging the write time, a few series of tests were performed.

A test script simulated data writing to the Lustre file alternately in a loop with performing computation similarly to known data-intensive applications described in Section 2. The number of iterations in the loop was configurable as well as the size of data stored in each iteration. In the presented results, the iteration count was set to 70 and the data size to 1 GB.

During our experimental evaluation we simulated 4 users. Two of these users used the QStorMan toolkit (via a system-level SES library) while other two selected storage resources with the Monte-Carlo algorithm. The only non-functional requirement was to find the fastest device to store the data. The obtained results are depicted in Fig. 4. The users who were using QStorMan managed to

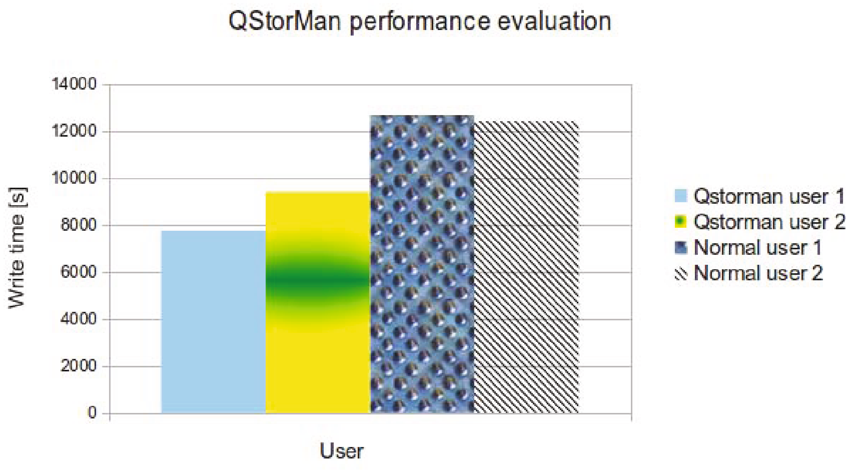


Fig. 4. Results of QStorMan performance evaluation

finish the test script before the users who were not using our system. Due to information about current workload, QStorMan was able to write data of its users to less loaded resources while normal users did not have such information. As a result QStorMan users wrote data even 37% faster than normal users. It should be noted that the presented scenario did not imply any modifications to the application source code.

7 Conclusions and Future Work

In this paper we have described a toolkit for data management in Grid environments based on non-functional requirements. The results show that making it possible for the VO members to define their requirements related to storage at different levels of abstraction using semantic descriptions, allows the system to decrease the data access times significantly.

The FiVO/QStorMan toolkit provides a few ways to define the non-functional requirements regarding data storage as well as a few user interfaces: from a graphical user interface called QStorMan WebFace to a programming library, which allows the user to decide how each of the files created by the application should be managed.

The performed tests show a possible speed up of data write time up to 37% for jobs scheduled with the QStorMan toolkit, compared to the same jobs scheduled without taking into account the information regarding the non-functional requirements of the job and the current workload of the infrastructure.

Future work will include further experiments with the proposed system with real users of data-intensive applications, in the framework of the PL-Grid project. Also, enhancements of the area of requirements definition are planned. By further exploiting the semantic technologies, we plan to enable users to define their requirements at a higher, more application-specific level of abstraction.

Acknowledgments. We thank Michał Wrzeszcz for help in the QStorMan portlet implementation. We thank Patryk Lasoń, Marek Magryś and Łukasz Flis (ACC Cyfronet AGH) for help in preparation of the testing environment based on the PL-Grid testing infrastructure.

References

1. Foster, I., Kesselman, C., Tuecke, S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. Int. J. High Perform. Comput. Appl. 15(3), 200–222 (2001)
2. Cummings, J., Finholt, T., Foster, I., Kesselman, C., Lawrence, K.A.: Beyond being there: A blueprint for advancing the design, development, and evaluation of virtual organizations. Technical report, National Science Foundation, `http://www.ci.uchicago.edu/events/VirtOrg2008` (as of May 26, 2011)
3. Kryza, B., Dutka, L., Slota, R., Kitowski, J.: Dynamic VO Establishment in Distributed Heterogeneous Business Environments. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part II. LNCS, vol. 5545, pp. 709–718. Springer, Heidelberg (2009)
4. Palak, B., Wolniewicz, P., Płóciennik, M., Owsiak, M., Żok, T.: User-Friendly Frameworks for Accessing Computational Resources. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 191–204. Springer, Heidelberg (2012)
5. Balcerek, B., Szurgot, B., Uchroński, M., Waga, W.: ACARM-ng: Next Generation Correlation Framework. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 114–127. Springer, Heidelberg (2012)
6. Paszyński, M., Pardo, D., Torres-Verdín, C., Demkowicz, L., Calo, A.: A parallel direct solver for the self-adaptive hp Finite Element Method. Journal of Parallel and Distributed Computing 70(3), 270–281 (2010)
7. Cettei, M., Ligon, W., Ross, R.: Support for parallel out of core applications on Beowulf workstations. In: Proceedings of the Aerospace Conference, vol. 4, pp. 355–365. IEEE (1998)
8. Weikuan, Y., Vetter, J., Canon, R., Jiang, S.: Exploiting Lustre File Joining for Effective Collective IO. In: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid, pp. 267–274. IEEE Computer Society (2007)
9. Dickens, P., Logan, J.: Y-Lib: A User Level Library to Increase the Performance of MPI-IO in a Lustre File System Environment. In: Proceedings of the 18th ACM International Symposium on High Performance Distributed Computing, HPDC 2009, Garching, Germany, pp. 31–38. ACM (2009)
10. Vijay, V., Anthony, S.: Quality of Service support for Grid Storage Environments. In: Proceedings of GCA, pp. 34–140 (2006)
11. Cheng, C., Wu, J., Liu, P.: QoS-aware, access-efficient, and storage-efficient replica placement in grid environments. Journal of Supercomputing 49(1), 42–63 (2009)
12. Zhang, X., Xu, Y., Jiang, S.: YouChoose: A Performance Interface Enabling Convenient and Efficient QoS Support for Consolidated Storage Systems. In: Proceedings of MSST, Denver, Colorado (2011)
13. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)

14. Bosak, B., Konczak, J., Kurowski, K., Mamoński, M., Piontek, T.: Highly Integrated Environment for Parallel Application Development Using QosCosGrid Middleware. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 182–190. Springer, Heidelberg (2012)
15. Kryza, B., Slota, R., Majewska, M., Pieczykolan, J., Kitowski, J.: Grid organizational memory-provision of a high-level Grid abstraction layer supported by ontology alignment. Future Generation Computer Systems 23(3), 348–358 (2007)
16. Mylka, A., Swiderska, A., Kryza, B., Kitowski, J.: Integration of heterogeneous data sources into an ontological knowledge base. Computing and Informatics 31(1) (2012)
17. The OntoStor project (June 20, 2011), `http://www.icsr.agh.edu.pl/ontostor/en.html`
18. Polak, S., Nikolow, D., Slota, R., Kitowski, J.: Modeling Storage System Performance for Data Management in Cloud Environment using Ontology. In: Pileggi, S. (ed.) Proceeding of IWSI 2011 International Workshop on Semantic Interoperability, in conjuction with ICAART 2011, Rome, Italy, pp. 54–63. SciTePress (2011)
19. The Lustre pool mechanism, `http://wiki.lustre.org/index.php/Creating_and_Managing_OST_Pools` (as of June 20, 2011)
20. The Lustre system, `http://wiki.lustre.org/index.php/Main_Page` (as of June 20, 2011)

Implementation of Service Level Management in PL-Grid Infrastructure

Tomasz Szepieniec, Małgorzata Tomanek, Marcin Radecki, Magda Szopa, and Marian Bubak

AGH University of Science and Technology, ACC Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków, Poland
t.szepieniec@cyfronet.pl

Abstract. One of the key concept of Grids – as stated in their definition – is providing nontrivial quality of service. However, there are many users whose specific needs for guaranteed resources (depending on the type of their applications and research schedule) are not satisfied. This fact is the main motivation for introducing Service Level Management in the Polish National Grid Infrastructure (PL-Grid). The paper presents a model of SLAs with quality properties required in HPC Grid. In order to properly manage these SLAs a negotiation process has been proposed. Finally, an implementation of the procedures and tools needed to support this processes is summarized.

Keywords: Service Level Management, Service Level Agreement, Grid.

1 Introduction

One of the key concepts of Grids – as stated in their definition [1] – is providing nontrivial quality of service. Several infrastructures have recently reached "production" quality, according to their claims. In parallel, there are several ongoing research projects to work out appropriate business model [7] and define related Service Level Management processes [4,5]. However, providing resources for scientific teams must take into account specific guarantees which are currently not available to users [8]. Such special needs might be related to the scientific teams' particular type of applications or, more commonly, their schedule of research actions. The main goal of introducing SLM in PL-Grid was to maximize scientific results achieved with the infrastructure. This is done by providing resource quality and availability guarantees for scientific groups, facilitating their research.

The primary challenge on the way to implementing this idea comes from the fact that PL-Grid provides services related to computations and data management on a large scale. In order to expose such services to users, vast amounts of elements related to computational and storage resources are needed. The crucial fact is that those elements are distributed and spread across different administrative domains. Each resource owner can be a provider in terms of SLM, and may introduce specific limitations and requirements related to SLA management. The resources themselves are of various types and thus their support levels may differ. Grid users, who are customers in terms of SLM, may have access to any grid resource on a technical level. The actual allocation of resources to customers is subject to Service Level Agreements (SLA). SLAs provide the users with certain guarantees related to the capacity and conditions of use of specified service elements. Detailed information is necessary to select suitable resources and then sign (conclude) an SLA. The complexity of the problem comes from the fact that, in typical scenarios, there are many providers for a user to contact and negotiate an SLA with, while at the same time each provider may provide resources to many (groups of) users. Therefore, SLM in a Grid forms a many-to-many network joining providers and clients. Two parties signing an agreement must be able to use the same language in terms of procedures, obligations and expectations. The above issues translate into requirements faced by the SLM model and any suitable support systems.

In this article we present an attempt at instituting service level management in the PL-Grid infrastructure. At the same time, we contribute to the discussion on how SLM should be structured for any national grid infrastructure or any federated e-infrastructure. Section 2 provides a short introduction to SLM which is subsequently mapped to PL-Grid actors. In Section 4 we list specific features of the SLA in terms of its lifecycle and quality metrics used in PL-Grid. Subsequently, we provide details on how the SLM-related processes, particularly the negotiation process, are implemented in PL-Grid operations. A summary of the tool environment is given in Section 7. The article ends with a description of related work and a summary.

2 Benefits of SLM

Service Level Management (SLM) is one of the processes defined within IT service management (ITSM), which is a discipline aiming at efficient delivery of quality IT services in a constantly changing environment. The standard process framework of ITSM is specified in ISO/IEC 20000 [15]. Best practices relevant to this field are listed in a set of handbooks collectively known as the IT Infrastructure Library (ITIL) [14]. The processes described there have been implemented by hundreds of IT providers worldwide. SLM includes the following actions:

- defining a catalog of IT service offerings,
- specifying services along with their service level options,
- negotiating and signing Service Level Agreements (SLAs) with customers,
- ensuring that each SLA is mirrored by Operation Level Agreements (OLAs) with suitable suppliers,
- monitoring and reporting on the fulfillment or violation of SLAs [13].

SLM is a vital part of customer-oriented provisioning of quality-aware IT services. The goal of SLM is to introduce a relationship between IT service providers and their customers. Typically, those two communities are separate and speak different languages. Thus, SLM promotes common understanding of customers' expectations, mutual responsibilities, communication channels as well as any constraints.

3 Actors and Relations in SLM Process

In classical SLM, relations are specified between a customer and a provider. The provider often relies on other (second-level) providers to deliver the agreed-upon services. In this section we specify how these roles are assigned in SLM for PL-Grid, in which case three types of actors are defined:

– *Customer: User Groups or Virtual Organizations* – sets of individual users who cooperate in order to achieve some scientific results; in PL-Grid user groups are registered in the PL-Grid Portal [16] and can apply for resources and access to specific domain services to support their research projects.
– *PL-Grid Operations Center (OC)* – integrates a robust set of technical services and provides a single point of contact for user groups and virtual organizations to apply for those services; it cooperates with 2^{nd} level providers to collect resource requests from customers, provide all the necessary tools and processes to monitor the use of resources and mediate in case of service delivery problems.
– *Individual Sites* – infrastructure providers who offer access to computing and storage resources through well defined protocols. Sites retain the right to manage their own resources, including allocating them to specific customers in accordance with site-specific procedures. Moreover, most sites do not provide the entire set of technical services required to support grid computing.

In order to introduce SLM, it is first necessary to define the relation between the provider and the customer, and reach an agreement upon services that are needed by customers. The crucial decision regarding our framework was to appropriately structure this agreement. The difficulty comes from the fact that services are delivered by sites, while the grid model assumes some level of unification under the umbrella of PL-Grid. It is also important that customers need

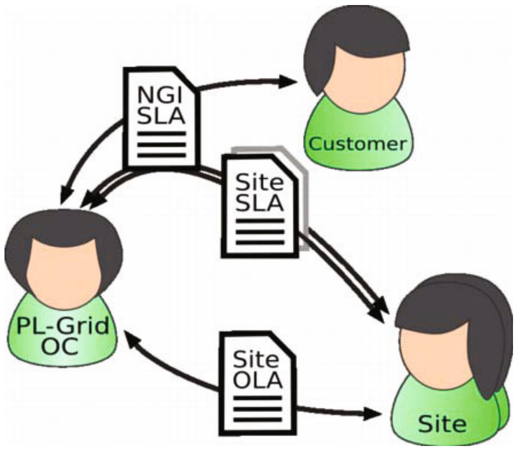


Fig. 1. SLAs and OLAs defining relations between actors in Grids

a single contact point for requesting resources, which, in our case, is the PL-Grid OC. The solution chosen for PL-Grid customizes the standard hierarchical structure of providers by adding some level of transparency in the SLA/OLA framework which enables convenient communication of customers with actual service providers. This resulted in three kinds of agreements, shown in Fig. 1, while the PL-Grid SLM framework is built on agreements defined below:

– *NGI Service Level Agreement* – the main SLA that is concluded between a customer and PL-Grid, i.e. the National Grid Initiative (NGI). It specifies the set of services that need to be delivered for a given customer, including service level descriptions and support information. The agreement must also include references to Site SLAs that provide the actual services.
– *Site Service Level Agreement* – concluded between a site (as a provider) and PL-Grid (representing the customer); it specifies the details of services, including instances of service elements. This agreement is linked to a specific NGI SLA.
– *Site Operations Level Agreement* – concluded between a site and PL-Grid to specify integration details and services that both parties need to deliver. This agreement is signed when a site joins PL-Grid; it also provides a base for integrated operations and efficient Site SLA settlement process.

It is important how the PL-Grid layer affects the guarantees offered to customers. In principle, any resource allocation and guarantee can be given by sites only based on their ability to manage their own resources. PL-Grid OC does not operate any resources directly. However, PL-Grid may upgrade customer guarantees by brokering Site SLAs to deliver the agreed-upon service level even in cases when a individual site violates its own Site SLA. The main benefit of maintaining an NGI SLA in addition to single-site SLAs lies in this increased level of guarantees. Other benefits are organizational in nature and related to the fact that customers can share a single point of contact for many sites.

This is a simplified model built upon the concepts presented in [13]. In the future, the model may be extended to support international VOs by adding an extra layer of hierarchy which can materialize as an international body, e.g. EGI.

4 SLA Life Cycle

Having identified the main types of SLAs, in this section we will focus on how the SLA state is described and how state transitions are performed during the SLA lifetime. To keep the description short we will focus on Site SLAs and then summarize how the NGI SLA differs from Site SLAs. Site OLAs are standard operation agreements – thus, we don't need to specify them in detail.

The state of a Site SLA is characterized by the pair $\alpha = (\alpha_1, \alpha_2)$. Individual elements are later referred to as sub-states. Some of the states make sense only if a previous state in the tuple has a specific value.

The first sub-state, α_1, is also called the *main sub-state*. Possible values of α_1 are as follows: `PROPOSAL`, `PREAGREED`, `AGREED` or `CANCELLED`. This sub-state

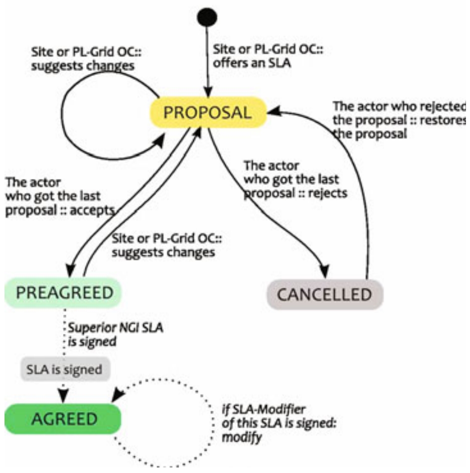

Site or PL-Grid OC:: suggests changes
Site or PL-Grid OC:: offers an SLA
PROPOSAL
The actor who rejected the proposal :: restores the proposal
The actor who got the last proposal :: accepts
The actor who got the last proposal :: rejects
Site or PL-Grid OC:: suggests changes
PREAGREED
CANCELLED
Superior NGI SLA is signed
SLA is signed
AGREED
if SLA-Modifier of this SLA is signed: modify

Fig. 2. Sub-states of SLA: main sub-states (α_1)

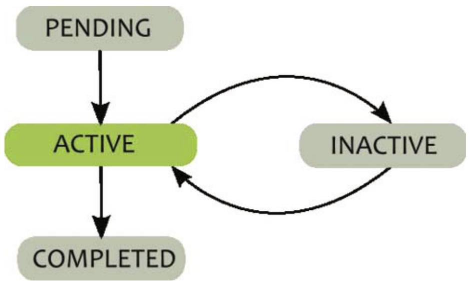


Fig. 3. Sub-states of SLA: activity sub-states (α_2)

determines whether the SLA is binding. If so, the sub-state is `AGREED`. To reach this sub-state both parties need to accede to each other's conditions and also the related NGI SLA needs to be in the `AGREED` state. Once the first condition is fulfilled, α_1 is switched to `PREAGREED`. The `PROPOSAL` value means that the SLA is in the process of negotiation. Finally, when an SLA is cancelled, its state switches to `CANCELLED`. The allowed transitions are presented in Fig. 2. Renegotiation is possible in the model by negotiating a new SLA (whose proposal can be based on the currently agreed version) and cancelling the current one.

The α_2 sub-state, also called the *activity sub-state*, is used to define the current status of SLA in relation to the time when the user utilizes the resources covered by the SLA. This sub-state is valid only if α_1 equals `AGREED`. When α_2 is `ACTIVE`, user is able to use the resources on demand (according to the SLA). Otherwise, this sub-state can be `PENDING` – which occurs before the first `ACTIVE` period, `INACTIVE` – between two `ACTIVE` periods or `COMPLETED` – following the last active period. These states are configured on the basis of the timeframe defined in the

SLA and additional administrative factors that may result in inactivating an otherwise `ACTIVE` SLA.

The NGI SLA lifecycle follows quite similar rules, however transitions between states depend on the dependent Site SLAs. Moreover, in the NGI SLA α_1 cannot be set to `PREAGREED`. Setting α_1 to `AGREED` is possible only when all associated Site SLAs are `AGREED` or `CANCELLED`. α_2 is set according to the following rules, applied in the presented order:

1. `ACTIVE` if at least one associated site SLAs is also `ACTIVE`;
2. `INACTIVE` if at least one associated site SLAs is `INACTIVE`;
3. set to the state which is shared by all dependent site SLAs.

It can be easily shown that it is not possible for the first rules to not apply and for Site SLAs to differ in their sub-states.

5 SLA Negotiation Process

The conceptual model of the SLA negotiation process is shown in Fig. 4. In the case of hierachical SLA models which have been introduced in PL-Grid, we can distinguish two types of related negotiation processes:

– NGI SLA Negotiation – initiated upon customer request. The SLA is negotiated with PL-Grid represented by the Operator. Each actor may change the SLA proposal but any modification must be confirmed by the other side.

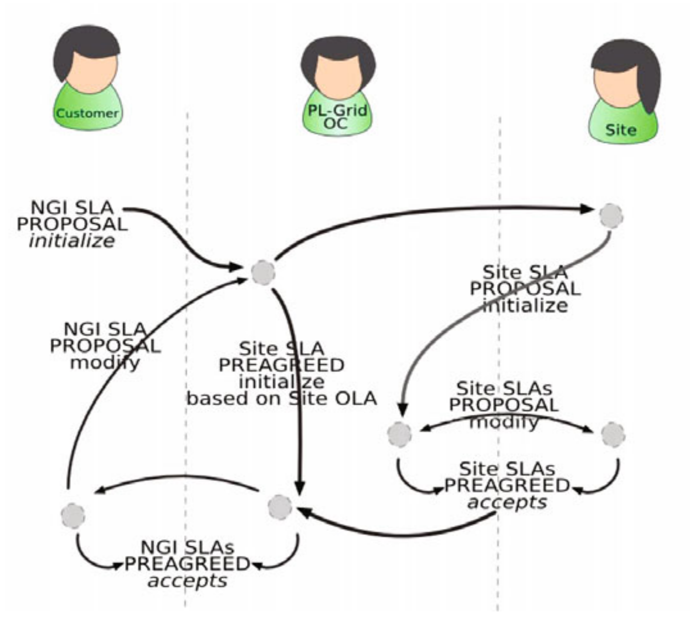

Customer
PL-Grid OC
Site
NGI SLA PROPOSAL *initialize*
Site SLA PROPOSAL initialize
NGI SLA PROPOSAL modify
Site SLA PREAGREED initialize based on Site OLA
Site SLAs PROPOSAL modify
Site SLAs PREAGREED *accepts*
NGI SLAs PREAGREED *accepts*

Fig. 4. SLAs negotiation process in PL-Grid

– Site SLA Negotiations – initialized as the site's answer to a customer request. The Site SLA is negotiated between a site and NGI. Agreement is reflected by setting the Site SLA status to `PREAGREED`. Several Site SLAs may be involved in negotiations at the same time, each covering a part of the customer's request and each realized by different providers.

The key task in this process falls to the PL-Grid Operator who brokers the site offers and aggregates them into a single SLA. The SLA is understood as a set of dependent Site SLAs; moreover, the SLA itself can contain an additional set of quality-related properties. The process is shaped in such a way that the act of accepting the NGI SLA is associated with final acceptance of the underlying Site SLAs. Without an accepted SLA those Site SLAs are not binding. Moreover, customers can review details of Site SLAs and propose changes to them in the process of SLA negotiations.

6 SLA Quality Properties

The usability of the proposed SLM model strongly depends on how SLA details are formulated and, in particular, on what resource quality properties are expressed in such agreements. The main requirement in defining them is that they should be both measurable and easy to verify in post-mortem analysis. Additionally, they should be configurable, which means that there should exist a method of translating these properties into service configuration. The set of properties included in SLA should not be too large, but should remain meaningful for the parties. Thus, they should have practical meaning for users as well as for providers. As different users' needs may vary, the properties included in a particular SLA can be selected from a larger set of available properties. If no properties are specified, a minimum service level should be guaranteed by some general documents. It is important to define the responsibilities for quality properties, each of which may refer to:

– provider – to define details of the offer and guarantees which the provider agrees to deliver,
– customer – to define limits within which the offer is valid or to restrict malpractice.

The responsibility of both actors related to a single measurable value may create conditions under which utilization of resources is expected and guaranteed.

While dealing with multi-level SLAs we need to distinguish between the following types of quality properties:

– *qualitative* – defining services, service elements or available service features;
– *quantitative summable*, called *capacity metrics* – measurable properties that specify the parameters of a service or service level; for metrics of this kind the NGI SLA value should not exceed the sum of values in the Site SLAs; usually these properties relate to the capacity of some resource;

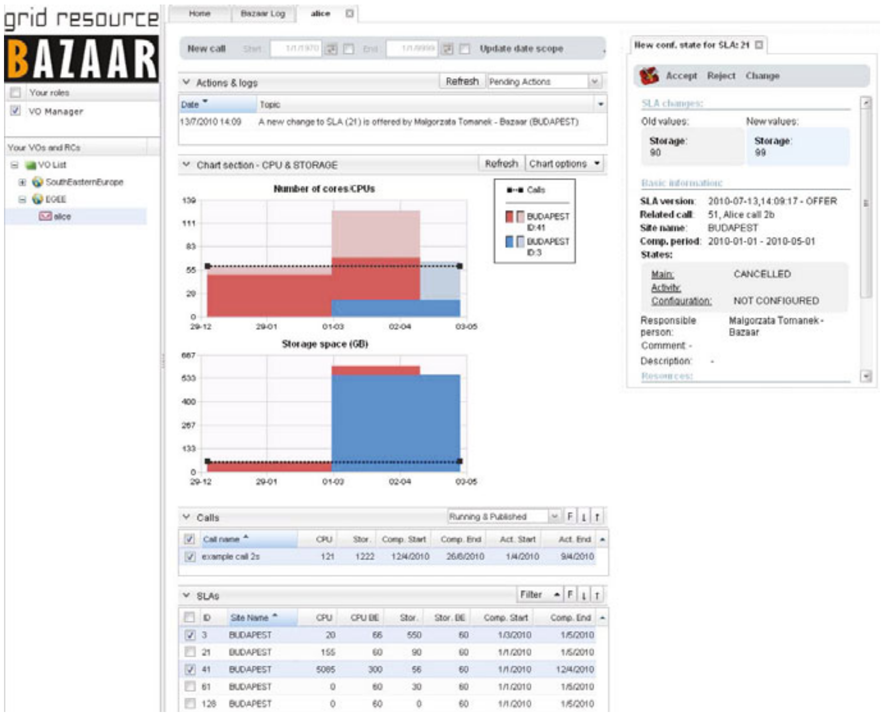
– *quantitative threshold*, called *quality metrics* – measurable properties that specify parameters of services or service levels; for such properties the NGI SLA value should not exceed the minimum value listed in the Site SLAs unless service elements are configured to increase the overall threshold; usually these properties are related to service level limits, deadlines, etc.

Table 1 defines both capacity and quality metrics describing the main types of resources applied in PL-Grid. In the case of computational resources, the capacity is expressed by "total normalized wall clock time", which limits the aggregated use of computational power defined in the SLA. Normalization is performed on the basis of benchmarks run on each type of available machines. Wall clock time values are used because concurrency is limited by the number of so-called machine slots, thus minimal computing efficiency for each job should be guaranteed and reflected in the benchmark. Other properties listed in the table define conditions for using computational power, including minimum machine configurations and time-dependent distribution of computations.

Similarly, a list of metrics for storage resources is provided. Not surprisingly, the capacity metric here is defined as "soft quota". Quality metrics specify the types of resources, assessment parameters, additional restrictions and extra features (e.g. backup processes). Apart from specific properties, there are several metrics that are common to all types of resources. Those metrics, presented in

Table 1. SLA quality properties for key types of resources

Name [unit]	Description/Notes	Resp.
Computational resources		
Total normalized wall time [normalized hours]	Sum of allocated time for user jobs in the queue system. The sum is normalized according to slot efficiency	provider
Max wall clock time for a single job [hours]	Limits the wall clock time for a single job. When this value is exceeded the RP is allowed to abort the job.	customer
Max single job parallelism [no. of slots]	Peak number of slots used by a single parallel job.	customer
Max slots used concurrently [no. of slots]	Number of slots that are used at the same time	customer
No. of slots reserved	Number of slots available for computation with no delay	provider
Storage Resources		
Soft Quota [GB]	Maximum volume of stored data that should not be exceeded	provider
Type of storage	Disks or tapes	provider
Hard Quota [GB]	Maximum volume of stored data that may not be exceeded	customer
Grace Period [days]	Maximum time allowed for soft quota breaches	customer



Date	Topic
13/7/2010 14:09	A new change to SLA (21) is offered by Malgorzata Tomanek - Bazaar (BUDAPEST)

SLA changes:

Old values:	New values:
Storage: 90	Storage: 99

Basic information:

SLA version: 2010-07-13,14:09:17 - OFFER
Related call: 51, Alice call 2b
Site name: BUDAPEST
Comp. period: 2010-01-01 - 2010-05-01
States:

Main:	CANCELLED
Activity:	
Configuration:	NOT CONFIGURED

Responsible person: Malgorzata Tomanek - Bazaar
Comment: -
Description: -
Resources:

Calls

	Call name	CPU	Stor.	Comp. Start	Comp. End	Act. Start	Act. End
✓	example call 2s	121	1222	12/4/2010	26/6/2010	1/4/2010	9/4/2010

SLAs

	ID	Site Name	CPU	CPU BE	Stor.	Stor. BE	Comp. Start	Comp. End
✓	3	BUDAPEST	20	66	550	60	1/3/2010	1/5/2010
	21	BUDAPEST	155	60	90	60	1/1/2010	1/5/2010
✓	41	BUDAPEST	5085	300	56	60	1/1/2010	12/4/2010
	61	BUDAPEST	0	60	30	60	1/1/2010	1/5/2010
	128	BUDAPEST	0	60	0	60	1/1/2010	1/5/2010

Fig. 5. Grid Resource Bazaar graphical user interface

the last section of the table, are mainly inherited from EGEE SLA and address service availability and related staff support.

7 Tool Support

The main requirement regarding SLM was to simplify the communication related to the process of SLA negotiation. Additionally, managing many SLAs requires support related to managing site capacity over time. This motivation inspired the development of the Grid Resource Bazaar [10] – a web platform which, once integrated with other PL-Grid operational tools, provides a convenient SLA and OLA negotiation framework. This collaborative tool simplifies communication by implementing communication patterns crafted according to the SLM model. The Bazaar GUI was designed for easy SLA handling, using complexity management techniques. As can be seen in Fig. 5, the portal is organized in the form of a dashboard with resource allocation views for customers and providers. The resources are visualized on a chart and SLAs are listed beneath. The user can manage SLA proposals and examine the influence of new SLAs on the resources.

Bazaar is integrated with the PL-Grid Portal authorization service, therefore in Bazaar users can negotiate resources only on behalf of user teams, which can be defined in the PL-Grid Portal. Information regarding SLA status is also made available via APIs which enable integration with other services, including automatic site configuration tools. Additionally, SLA metric monitoring is being integrated with the PL-Grid monitoring infrastructure and an accounting module is under development, but these features are out of scope of this article.

8 Related Work

There is currently no coordinated effort to introduce Service Level Managements in the main European Grid Initiatives, besides providing an initial business model [7]. There have been several attempts to implement some of its aspects in the infrastructures, mainly at the national level. Moreover, the EU-funded gSLM Project intends to stimulate the development of SLMs for Grids. The arguments supporting the need of such actions are collected in [13]. Leff et al. in [6] draw the conclusion that Grids without an SLM framework can only have very limited use.

Examples of such projects are SLA@SOI[1] and SLA4D-Grid[2]. The former project is concerned mainly with Service Oriented Infrastructures and aimed at industrial use cases. Its main concern is ensuring predictability and dependability for business partners. These goals are achieved by introducing an SLA framework for automatic SLA negotiation and management, which may not be possible in such large infrastructures as Grids. The aim of the second project is to design and implement a Service Level Agreement layer in the middleware stack of the German national Grid initiative D-Grid. The target of introducing SLAs into the project was to guarantee the quality of service and fulfillment of prenegotiated business conditions. The SLA4D-Grid project focuses on tools for automatic SLA creation and for negotiations, also offering support for monitoring and accounting. It does not, however, provide a model of an integrated SLA framework which would enable interaction with grid infrastructures other than D-Grid. An important aspects of its activities involves standardizing SLA negotiation protocols on the basis of WS-Agreement [17].

In general, SLA awareness in grids and other computing infrastructures remains a challenge [12].

9 Summary

This article presented several important parts of the PL-Grid SLM concept, including a model SLA/OLA framework, as well as details regarding internal SLA states and resource usage metrics. We described the negotiation process and how it is supported by specialized tools. We believe that the presented solution remains valid for other federated infrastructures such as grids and clouds.

[1] http://www.sla-at-soi.eu/
[2] http://www.sla4d-grid.de/

References

1. Foster, I.: What is the Grid: A Three Point Checklist. Grid Today (2002)
2. Gerndt, H.M., Rana, O.F., Von Laszewski, G., Ziegler, W.: Service level agreements in grids, Schloss Dagstuhl, Germany, March 22-27 (2009), abstracts collection
3. Battré, D., Hovestadt, M., Keller, A., Kao, O., Voss, K.: Virtual Execution Environments and the Negotiation of Service Level Agreements in Grid Systems. In: Boursas, L., Carlson, M., Hommel, W., Sibilla, M., Wold, K. (eds.) SVM 2008. CCIS, vol. 18, pp. 1–12. Springer, Heidelberg (2008)
4. Koller, J., Oliveros, E., Sanchez-Macian, A.: Service Level Agreements (SLAs) in the Grid Environment. In: Buyya, R., Bubendorfer, K. (eds.) Market-Oriented Grid and Utility Computing. John Wiley (2010)
5. Guitart, J., Macias, M., Rana, O., Wieder, P., Yahyapour, R., Ziegler, W.: SLA-Based Resource Management and Allocation. In: Buyya, R., Bubendorfer, K. (eds.) Market-Oriented Grid and Utility Computing. John Wiley (2010)
6. Leff, A., Rayfield, J.T., Dias, D.M.: Service-level agreements and commercial grids. IEEE Internet Computing 7(4), 44–50 (2003)
7. Candiello, A., Cresti, D., Ferrari, T., et al.: A Business Model for the Establishment of the European Grid Infrastructure. In: Proceedings of the 17th Int. Conference on Computing in High Energy and Nuclear Physics (CHEP 2009), Prague (March 2009)
8. Schwiegelshohn, U., Badia, R.M., Bubak, M., Danelutto, M., Dustdar, S., Gagliardi, F., Geiger, A., Hluchy, L., Kranzlmüller, D., Laure, E., Priol, T., Reinefeld, A., Resch, M.M., Reuter, A., Rienhoff, O., Rüter, T., Sloot, P.M.A., Talia, D., Ullmann, K., Yahyapour, R.: Perspectives on grid computing. Future Generation Comp. Syst., 1104–1115 (2010)
9. EGEE-III: Service Level Agreement Between ROC and Sites (template) (2008), https://edms.cern.ch/file/860386/0.7/EGEE-ROC-Site-SLA-v1.6.pdf
10. Szepieniec, T., Tomanek, M., Twarog, T.: Grid Resource Bazaar. In: Proc. of Cracow Grid Workshop 2009, Kraków (2010)
11. von Laszewski, G., Alunkal, B.K., Veljkovic, I.: Toward Reputable Grids. Scalable Computing: Practice and Experience, Scientific International Journal for Parallel and Distributed Computing 6(3), 95–106 (2005)
12. Sakellariou, R., Yarmolenko, V.: Job Scheduling on the Grid: Towards SLA-Based Scheduling. In: Grandinetti, L. (ed.) High Performance Computing and Grids in Action. The Advances in Parallel Computing series, vol. 16, pp. 207–222. IOS Press (2008)
13. Szepieniec, T., Kocot, J., Schaaf, T., Appleton, O., Heikkurinen, M., Belloum, A., Serrat-Fernandez, J., Metzker, M.: On Importance of Service Level Management in Grids. Presented in Euro-PAR 2011. LNCS. Springer, Heidelberg (2011)
14. ITIL Lifecycle Publications Suite, ISBN 13: 9780113313235
15. ISO/IEC 20000-1:2011 IT Service Management System (SMS) Standard
16. Radecki, M., Szymocha, T., Harężlak, D., Pawlik, M., Andrzejewski, J., Ziajka, W., Szelc, M.: Integrating Various Grid Middleware Components and User Services into a Single Platform. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 15–26. Springer, Heidelberg (2012)
17. Andrieux, A., Czajkowski, K., Dan, A., Keahey, K., Ludwig, H., Pruyne, J., Rofrano, J., Tuecke, S., Xu, M.: Web Services Agreement Specification (WS-Agreement). Global Grid Forum (May 2004)

Highly Integrated Environment for Parallel Application Development Using QosCosGrid Middleware

Bartosz Bosak, Jan Konczak, Krzysztof Kurowski,
Mariusz Mamoński, and Tomasz Piontek

Poznań Supercomputing and Networking Center, Poznań, Poland
{bosak,krzysztof.kurowski,mamonski,piontek}@man.poznan.pl,
jan.konczak@student.put.poznan.pl

Abstract. QosCosGrid infrastructure has been prepared, introduced successfully on multiple clusters and comprehensively tested. In order to enhance user experience and increase usability, a set of additional tools must be prepared. Profiting from distributed software becomes more popular nowadays, but the number of tools supporting developers of parallel programs is still smaller than developers' needs. Eclipse PTP is the leading open source tool among Integrated Development Environments for distributed software. In this chapter we shortly present Eclipse PTP and related Eclipse plug-ins aiding programmers in developing parallel software. Later we describe how QosCosGrid middleware has been integrated with Eclipse PTP. We present the plug-in basic concepts, such as the Resource Manager or Remote Services, general architecture, functionality of the PTP QCG plug-in and advantages over other solutions of this kind.

Keywords: parallel computing, debugging, MPI, Integrated Development Environments.

1 Introduction

Currently, a large portfolio of simulation software packages, both commercial and open source, is available to the scientific community. Despite the maturity and comprehensiveness of many of such packages, various scientists have to develop their own code because of the uniqueness of their research problem. Our work has been oriented to ensure a convenient environment for the scientists and provide them with tools serving as aid in developing parallel software. In order to take advantage of rich integrated development environments and users' acquaintance with them we decided to extend the Eclipse Parallel Tools Platform [5] – the Eclipse IDE parallel computing plug-in – to enable the cooperation with QCG, thereby enhancing users' experience with programming and running jobs on distributed resources governed by the QosCosGrid [9] middleware.

2 Related Work

Currently the majority of HPC toolkits are mixtures of command-line tools. If GUI tools exist, they are usually focused on single functionalities such as profiling (e.g. Vampir) or debugging (e.g. TotalView). On the other side, more comprehensive toolkits such as HPC Intel Cluster Studio or Oracle Solaris Studio are missing one important feature: the ability to assist users in running parallel applications on remote clusters.

A notable exception is MATLAB Distributed Computing Server[1] (an extension to the MATLAB Parallel Computing Toolbox[2]) which supports running MATLAB scripts on a remote cluster. However this tool is obviously limited to MATLAB appliances.

3 Motivation

The command-line based tools that are common in the HPC world may seem archaic. In many cases, using terminal based editors and managing the whole development process manually may seriously slow it down. Moreover, introducing a new person to the command-line tools, especially someone with high demand for advanced uses, is time-consuming. Many scientists are still reluctant to the possibilities offered by the HPC world due to difficulties they encounter while using non-intuitive text-based tools.

Graphical Integrated Development Environments have become standard outside of HPC due to the productivity aid they provide such as context aware help not only for basic language features, but also for a wide array of libraries and toolkits. The IDEs include tools for source code control and bug tracking, graphical debuggers, and a lot of other useful tools in a single workbench with a consistent user interface.

4 The Eclipse IDE Plug-ins for Parallel Computing

Eclipse Parallel Tools Platform tries to address all aspects of parallel application development cycle, namely: coding and analysis, running and monitoring of parallel application, debugging and performance tuning.

Additionally, there are two other Eclipse sub-projects: CDT (C/C++ Development Tooling) and Photran which provide support for two most popular languages in the HPC world: C and Fortran. This includes dedicated project types, context help, templates and samples for the language and launch configurations, as well as plug-ins providing integration with popular compilers, debuggers and other tools dedicated to C/C++ or Fortran.

Moreover, the first of the two aforementioned environments supports so-called 'C/C++ remote projects' in which the source code is located and the build process takes place on the remote machine. This allows users to develop applications

[1] `http://www.mathworks.com/products/distriben/`
[2] `http://www.mathworks.com/products/parallel-computing/`

on a remote machine using a local Eclipse IDE, solving most problems with architecture differences and libraries availability. Moreover, it allows users to easily employ commercial compilers and software packages usually available only on the target machine. However, the CDT remote project alone does not provide support for parallel programming.

The PTP extends those environments with the support for Message Passing Interface (MPI) specific content and templates simplifying MPI programming. If an MPI implementation is available on a local machine, or accessible via shell connection, the PTP is able to execute, debug and profile MPI applications owing to its integration with the most common MPI implementations (OpenMPI and MPICH2).

In order to execute jobs on remote resources using a certain service, a set of plug-ins providing a so-called Resource Manager, must be available. Each Resource Manager is responsible for interconnecting Eclipse with a certain batch system or other tool enabling running parallel applications, either interactively or in a batch mode. The developer can view the output of the application directly in Eclipse, as well as manage the submission by checking its state or canceling it.

Another valuable tool built in Eclipse PTP is SDM (Scalable Debug Manager) – a parallel debugger. While traditional debuggers apply operations only to single processes, parallel debugging offers applying it to arbitrary collections of processes. SDM allows grouping processes into user-defined sets, which simplifies this process. SDM may automatically be started alongside with an MPI program, which greatly simplifies debugging of parallel applications.

Other, more advanced tools integrated with Eclipse PTP are TAU-based [3] performance tools framework and Graphical Explorer of MPI Programs (GEM). The latter can be used for the formal dynamic verification of MPI programs, which helps detecting hard-to-find concurrency bugs (e.g. deadlock). Other tools can be integrated relatively easily with the use of External Tools Framework.

Other important parts of the Eclipse project for developing software on remote machines are plug-ins providing access to the filesystem and terminal on the remote resource. So called Remote Services are certain plug-ins that can be used to create a connection to a remote system in order to exchange files or create new processes on a remote machine. One of the Remote Service implementations, the Remote Development Tools (RDT) is a part of PTP project. Another one, Remote System Explorer (RSE), is a separate Eclipse subproject dedicated to integrate any sort of heterogeneous remote resources with the IDE.

At present, the Parallel Tools Platform is being constantly improved owing to the funding received from the Software Infrastructure for Sustained Innovation (SI2) projects funded by the National Science Foundation.

5 The QCG-Computing Plug-in for Eclipse PTP

Our work with the Parallel Tools Platform was focused on the implementation of a Resource Manager plug-in for the QCG-Computing service, an administrative level component of the QosCosGrid stack – a new grid middleware developed and deployed within the PL-Grid project [8,1].

5.1 QCG-Computing

The QCG-Computing is an open architecture implementation of SOAP Web Service for multi-user access and policy-based job control routines offered by various Distributed Resource Management systems.

It uses Distributed Resource Management Application API (DRMAA) [4] to communicate with the underlying DRM systems. The service interface is compliant with the OGF HPC Basic Profile [6] specification, which serves as a profile over the JSDL and OGSA® Basic Execution Service Open Grid Forum standards. The overview of the QCG-Computing architecture is presented in Fig. 1.

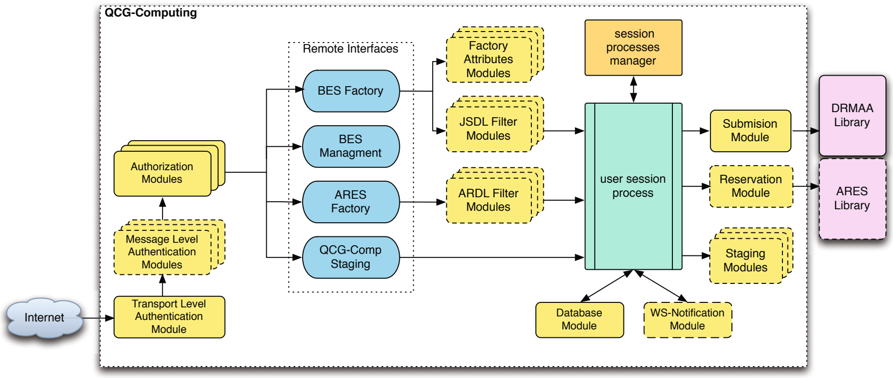


Fig. 1. Overview of the QCG-Computing architecture

The service interface is composed of four Web Service ports: BES-Factory – interface for job creation, monitoring and management; BES-Management – interface for managing the service; ARES-Factory – interface for advanced reservation creation and management; QCG-Staging – interface for direct (client-service) file transfer via SOAP attachments.

The system architecture is based on dynamically loadable modules and it is in accordance with the privilege separation model. Thus only a relatively small part of the overall system (the Session Process Manager in Fig. 1) runs with superuser privileges. This component is responsible for creation of new processes (called User Session Processes) that run with mapped users effective privileges.

The service was successfully tested with the following Distributed Resources Management systems: Grid Engine, Platform LSF, Torque/PBSPro, PBS Pro, Condor, Apple XGrid, SLURM and IBM Tivoli LoadLeveler.

5.2 Resource Manager

Within the Parallel Tools Platform the term *resource manager* means any system that controls the resources required for launching a parallel job (e.g. batch system). The QCG-Computing plug-in provides the following resource manager capabilities: basic cluster monitoring; job submission, monitoring and control; providing Eclipse with application's standard output and error streams in real-time. It also implements all the methods required by the Scalable Debug Manager in order to debug parallel MPI applications remotely. It is worth mentioning that the debugging functionality is not available in all remote resource manager plug-ins (e.g. the PBS plug-in is currently missing this feature).

What is also important, especially in grid environments such as PL-Grid, is that the QCG-Computing Resource Manager plug-in supports authentication mechanisms based on proxy certificates, which is a unique feature among all the PTP resource manager plug-ins and a requirement of the PL-Grid e-infrastructure.

5.3 Internal Plug-in Architecture

In order to add support for the QosCosGrid infrastructure to Eclipse PTP, three plug-ins combined into one meta-package (called *feature* in the Eclipse terminology) were created. First the QCG-Computing Java SDK has been turned into an Eclipse plug-in, while the two other plug-ins, the core plug-in and the UI plug-in, were written from scratch. The core plug-in groups core classes and utilities, while the UI plug-in covers user interaction details. This common approach allows for separation of the plug-in logic and the user interface.

Fig. 2 presents the internals of each plug-in. Core plug-in contains an implementation of Remote Services interface, required to remember and manage connections to the remote system. QCG plug-in does not rely on shell connection or proxy, thus a dedicated implementation has been prepared. The QCG Remote Services also contain the implementation of file transfer routines. The core plug-in is responsible for executing actions requested by the user, such as starting or stopping the Resource Manager, submitting, monitoring and canceling jobs. These actions are forwarded from the UI plug-in, processed accordingly to the state of Resource Manager and executed on the remote resource using the SDK plug-in.

The QCG UI plug-in persistently stores the configured Resource Managers and user settings. It also provides all wizards and dialogs for interaction with the user – user interface elements responsible for creating and configuring the Resource Manager, launching tasks, monitoring submitted jobs and transferring files. However, in order to simplify the architecture, some minor dialogs reporting warnings are not included in the UI plug-in, but in the core class.

5.4 Access to Remote Filesystem

On the contrary to other resource managers, the QCG Resource Manager does not need a shell account or any external connection to the remote system – it can work properly without using the Remote Services infrastructure.

Access to the remote filesystem is provided by the QCG Resource Manager itself, as the QosCosGrid is able to transport files to and from the remote resource. It is therefore possible for the user to access a resource using only

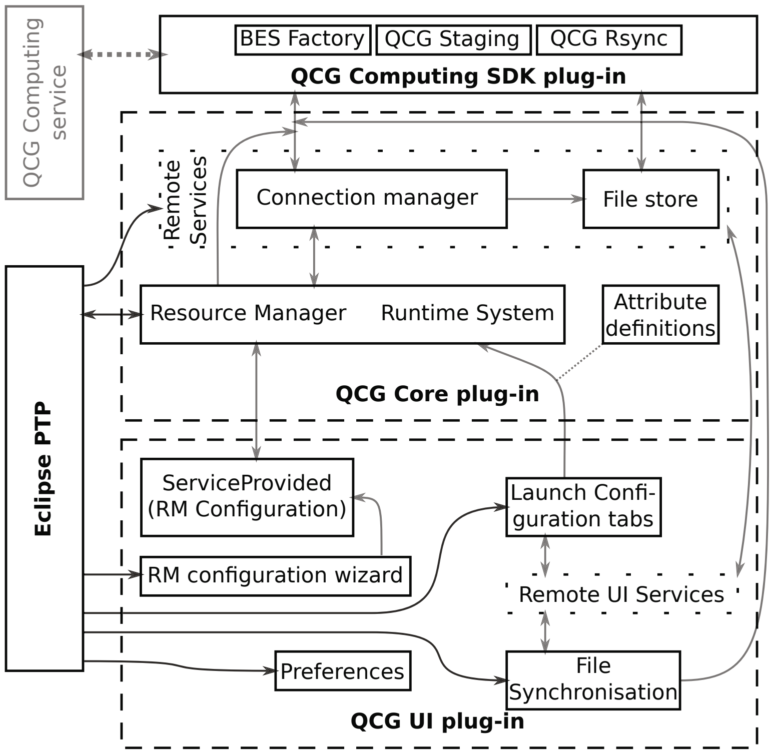


Fig. 2. Architecture of QCG plug-ins for Eclipse PTP

a X.509 certificate, without the need to possess shell account on the requested resource.

By using the QCG plug-in the user may transfer single files on demand, stage files with a task or use the rsync protocol [10] to synchronize local and remote directory. The remote terminal is not provided – user cannot pass commands directly to the resource and every new process must be submitted as a standard job.

However, if the user has an account on the remote machine, the PTP QCG plug-in can take advantage of this fact. As long as the same filesystem is used by shell connection and QCG, external remote services may be used for transferring data. It enables the use of more optimized routines for filesystem access.

An important advantage of providing an SSH connection to the plug-in is the ability of tunneling ports. During debugging the master debugger process needs to be accessible for the Eclipse IDE. Usually only the front-end machine is accessible from the outside. If the machine with a master debugger process is inaccessible from the outside but accessible from the front-end, a port forwarding is the only solution to connect to the remote debugger.

In such a setup tunneling data from a remote debugger to Eclipse is currently the only way to use debugging features.

5.5 Compiling the Source Code

Eclipse PTP enforces a certain algorithm while executing a program. Firstly, the source code must be compiled locally, later on, the program is sent to the remote machine and then, on demand, compiled again. Once these steps are fullfiled, the requested action (either debugging or execution) is performed.

For most PL-Grid users this is not a good solution as the libraries and compilers are available on remote machines only. Also both system and architecture of the target machine and the user's workstation do not have to match, which causes more problems with proper compilation of the source code.

In such a case, a custom empty build schema must be created, so that the automatic build would always succeed. As far as the remote build is concerned, a user may choose what steps should be executed in order to build the program: either standard `make` with accompanying Makefile, or a custom build command. The custom command may also be a path to user script that executes a certain, possibly complex, toolchain.

5.6 Submitting MPI Jobs

The job created in Eclipse can be given various attributes, some of which are purely descriptive (like the name or the job description), other decide how the task will be executed. The user may choose a number of cores, a queue for the job, a set of preferred nodes, a working directory and whether a special wrapper script should be used to start the job.

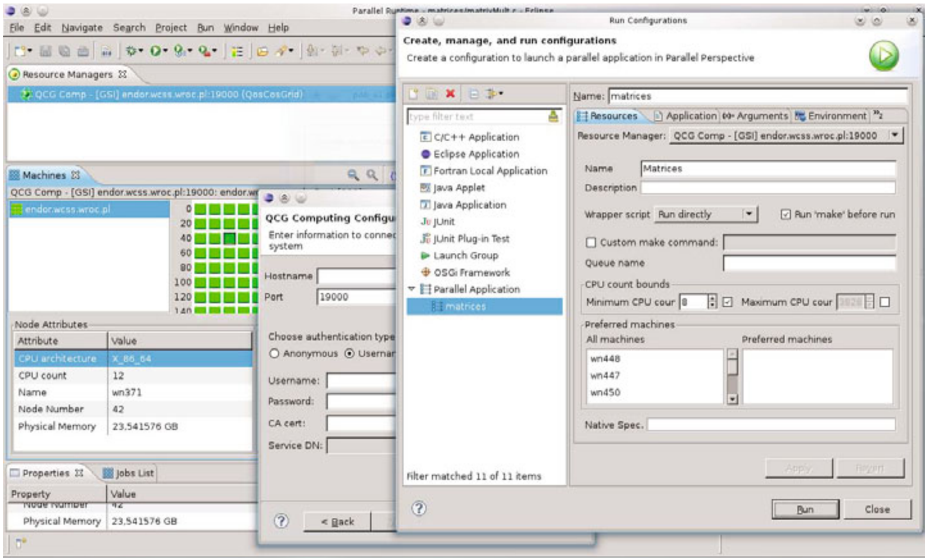
The QosCosGrid infrastructure may provide access to a set of so-called wrapper scripts. These allow for easy integration of QCG with external tools. A wrapper script is a shell script responsible for proper launching of the given application. One of the examples is a script for running MPI applications, which sets up required paths to the MPI flavor selected from the available ones, starts the build process (if a user chooses to do so) and uses `mpiexec` or equivalent to spawn the processes across the allocated machines.

This approach also increases user's comfort – one does not need to take into consideration the local paths or environment variables which need to be setup on given system. Moreover, user can always omit the wrapper script and submit their own script directly to the resource.

5.7 Debugging

SDM, the Scalable Debugging Manager, is a debugger middleware integrated with PTP. The SDM debugger uses GDB [2] (GNU debugger) as the backend. Once user selects to start a debugging session, QCG plug-in submits a job with proper environment variables set, indicating that this should be a debugging session and pointing at a file where information about the SDM ports and process location will be written.

In case of debugging it is mandatory to use a debugging-enabled wrapper script, or to write own startup script delivering required information. The QCG


Resource Managers: QCG Comp - [GSI] endor.wcss.wroc.pl:19000 (QosCosGrid)

Machines: QCG Comp - [GSI] endor.wcss.wroc.pl:19000: endor.wcss.wroc.pl

Node Attributes

Attribute	Value
CPU architecture	X_86_64
CPU count	12
Name	wn371
Node Number	42
Physical Memory	23.541576 GB

Properties

Property	Value
Node Number	42
Physical Memory	23.541576 GB

QCG Computing Configu
Enter information to connec
system

Hostname:
Port: 19000
Choose authentication type: ( ) Anonymous (•) Usernar
Username:
Password:
CA cert:
Service DN:

Run Configurations
Create, manage, and run configurations
Create a configuration to launch a parallel application in Parallel Perspective

Name: matrices
Resource Manager: QCG Comp - [GSI] endor.wcss.wroc.pl:19000
Name: Matrices
Description:
Wrapper script: Run directly
[✓] Run 'make' before run
[ ] Custom make command:
Queue name:
CPU count bounds: Minimum CPU cour 8 [✓] Maximum CPU cour [ ]
Preferred machines: All machines: wn448, wn447, wn450; Preferred machines:
Native Spec.:

Filter matched 11 of 11 items

Fig. 3. QCG plug-in in action

plug-in must receive information concerning location of every single process in order to be able to contact the SDM instances connected to it. In most cases, in order to access the execution hosts, port tunneling must be available. Currently, this is achievable only via external Remote Services implementations, such as RSE or Remote Tools.

5.8 Tasks Monitoring

Eclipse PTP provides generic monitoring routines. The QCG plug-in implements all the required functions, thus allowing the user to see the job state, cancel the activity and view the standard output and error streams. Currently, the QCG plug-in uses polling mechanism to get job statuses. In the next version of the plugin it is planned to exploit XMPP notification capabilities of the QCG stack.

Due to Eclipse PTP limitations, only the jobs started by Eclipse PTP since the last Eclipse IDE start-up are monitored.

6 Conclusions

The QosCosGrid infrastructure has been successfully integrated with Eclipse Parallel Tools Platform. Therefore, all research groups which develop their own code can fully benefit from Eclipse PTP plug-in in order to accelerate and simplify the development process of their applications. Finally, we also plan to offer tutorials on using PTP as a part of the PL-Grid training activities.

References

1. Kurowski, K., de Back, W., Dubitzky, W., Gulyás, L., Kampis, G., Mamonski, M., Szemes, G., Swain, M.: Complex System Simulations with QosCosGrid. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part I. LNCS, vol. 5544, pp. 387–396. Springer, Heidelberg (2009)
2. Matloff, N., Salzman, P.J.: The Art of Debugging with GDB, DDD, and Eclipse. No Starch Press, San Francisco (2008)
3. Shende, S.S., Malony, A.D.: The tau parallel performance system. International Journal of High Performance Computing Applications 20(2), 287 (2006)
4. Troger, P., Rajic, H., Haas, A., Domagalski, P.: Standardization of an API for Distributed Resource Management Systems. In: Seventh IEEE International Symposium on Cluster Computing and the Grid, CCGRID 2007, pp. 619–626. IEEE (2007)
5. Eclipse Parallel Tools Platform, http://www.eclipse.org/ptp/
6. Open Grid Forum HPC Basic Profile, Version 1.0, http://ogf.org/Public_Comment_Docs/Documents/Feb-2007/HPC_Basic_Profile_v1.0.pdf
7. PTP QCG plug-in user manual, http://www.qoscosgrid.org/trac/qcg-ptp/
8. QosCosGrid – distributed computing infrastructure middleware, http://www.qoscosgrid.org/
9. Bosak, B., Komasa, J., Kopta, P., Kurowski, P., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)
10. The rsync protocol, http://rsync.samba.org/tech_report/tech_report.html, http://rsync.samba.org/

User-Friendly Frameworks for Accessing Computational Resources

Bartek Palak, Paweł Wolniewicz, Marcin Płóciennik, Michał Owsiak, and Tomasz Żok

Poznań Supercomputing and Networking Center, ul. Noskowskiego 10, 61-704 Poznań
{bartek,pawel.wolniewicz,marcinp,michalo,tzok}@man.poznan.pl

Abstract. The grid technology, even though it gives great computational power and greatly improves the manner of exploitation of resources, also has its disadvantages. Grid systems, due to their distribution and heterogeneity, are very complex and hard to access and oversee. Thus, one of the research fields designs tools and technologies that give easy, secure and consistent access to grid applications and resources as well as seamless interoperation between various computing environments. The lack of user-friendly tools is especially annoying in cases when users need to access computing resources of different grid infrastructures. Similar problems are experienced by grid application developers who should focus on applications themselves and not on their interoperation with different grid middleware. In this article we describe the concept of an abstract grid model and its implementation in two user-friendly frameworks – Migrating Desktop and g-Eclipse. They both are intuitive graphical environments that provide: easy access to heterogeneous resources and seamless interoperation of underlying middleware solutions. Although the two products provide similar functionalities they are complementary to each other and target different user groups. The method of integration of scientific applications with both frameworks was also presented.

Keywords: grid, graphical user interface, interoperation, gLite, UNICORE.

1 Introduction

To use modern computing applications scientists often need access to computing resources, that are easily available and easily accessible. Owing to several grid initiatives, which provided efficient distributed computing infrastructures over the last decade, a significant improvement of computing power availability could be noticed. In recent years, grids have emerged as wide-scale distributed infrastructures that support the sharing of geographically distributed, heterogeneous computing and storage resources [1].

Many grid projects (such as EGEE, BalticGrid, DEISA, etc.) demonstrated the benefit of a general infrastructure for scientific and commercial applications.

However, the complexity of grid infrastructures is often discouraging to application developers and impedes the use of grid technologies in scientific application domains. Unfortunately, in many cases accessibility of the resources, understood as the ease of use experienced by users, is still not at a satisfactory level. Understanding the behaviour of grid resources is difficult and the learning curve for newcomers is too steep. It often discourages users who are non-experts in the technology and systems being used. Additionally the diversity of environments, based on various concepts and architectures along with their complexity, implies that even deeper knowledge is required to access the resources. Unfortunately, even though there is a trend towards interoperable, service-oriented implementations of grid-services, currently different grid middleware systems are in use. The most popular and widely distributed middleware systems are gLite [2], Globus Toolkit [7] and UNICORE [3,15]. While all these middleware systems offer basic services to interact with the underlying grid infrastructure, each follows a slightly different approach. Incompatibility of middleware implementations is especially annoying in cases when the nature of the problem being solved forces the user to use computing resources of various kinds.

The issues described above clearly show the need for high-level, user-friendly environments that provide: intuitive access to heterogeneous resources and seamless interoperation of underlying middleware solutions. More user-friendly and intuitive tools and user interfaces are needed, in order to make the look-and-feel of grid infrastructures similar to that of existing computer desktop systems which people are already familiar with. In this article we describe two user-friendly client platforms for users, which were developed and deployed within the PL-Grid project – Migrating Desktop [4] and g-Eclipse [5]. They both allow for interoperation with various grid or cloud infrastructures and provide user friendly, transparent access to different middlewares, although their development was driven by different requirements. The two products provide similar functionalities and are complementary with each other because they target different user groups.

2 Related Work

Among the commercial or educational products, which can be used for accessing the grid and interactions between underlying environments, one can distinguish three most popular groups: command line interfaces, portals, and advanced graphical tools. Command line interfaces (CLI) provide only the simplest text interface. They are available in all grid systems (e.g. ARC, UNICORE, gLite) but are used mainly by very experienced users (e.g. system administrators, application developers, etc). This kind of interaction with computing environment requires very deep knowledge of the system being used, so their usage could be very difficult for beginners. Portals and advanced graphical tools are much more popular among the users as a way of accessing grid remote resources. They offer intuitive and easy manner of interaction with user by providing services such as user profile management, information services, remote job submission, job tracking, file transfer, authentication and authorisation, composition of workflow between tasks etc. Advanced graphical tools have usually more complex, flexible and extensible characteristics compared to portals that are usually limited to support only one type of middleware. Rich clients targeted to Windows, Linux/Unix and MacOS/X are in most cases Java-based, to ensure platform independency.

NGS Job Submission Portal and UCLA Grid Portal [11] are good examples of grid dedicated portals. The NGS Job Submission Portal [10] can be used to access and interact with the HPC and data resources available on the NGS. Functionality available to users include: support for submission and monitoring of job computing as well as data access and transfer around the Compute and Data Grid. The UCLA Grid Portal provides a single web interface to computational clusters of UCLA Grid, and other-including clusters on the TeraGrid. The features offered cover: resource discovery, job handling, file management and results visualisation.

Vine Toolkit [14] has been successfully used as a core web platform for various Science Gateways. Vine Toolkit is a modular, extensible and easy-to-use tool as well as high-level Application Programming Interface for various applications, visualisation components and building blocks. It allows for interoperability between many different HPC and grid technologies on the service layer. Using Vine Toolkit it is possible to build a portal upon the different HPC technologies working together to deliver a complete solution to the users.

P-GRADE [12] offers workflow construction and execution mechanisms for grid including execution and performance visualisation, advanced security features, access to information systems and resource brokers, and multi-grid support. It is de facto "a hybrid" of two different user interfaces: the Workflow Manager runs as a portlet on the P-GRADE Portal server and the Workflow Editor runs as a separate application on the desktop downloaded from the portal. The P-GRADE Portal hides the low level details of grid systems with high-level, user-friendly interfaces that can be easily integrated with various middleware. It offers following services: definition of grid environments, creation and modification of workflow applications, management of grid certificates, controlling and execution of workflow applications on grid resources and monitoring and visualisation of workflows and their component jobs.

Another popular example of advanced graphical tools is UNICORE Rich Client (URC) that provides users with a graphical representation and management of UNICORE controlled resources, services or files. This, Eclipse based, graphical client for UNICORE 6 enables user to build, submit and monitor jobs (which can be scheduled also as workflows that combine several applications), as well as allows them for integrated data and storage management. For better integration of grid applications URC uses the concept of GridBeans – small software packages that provide tailored graphical user interfaces for scientific applications available on the grid.

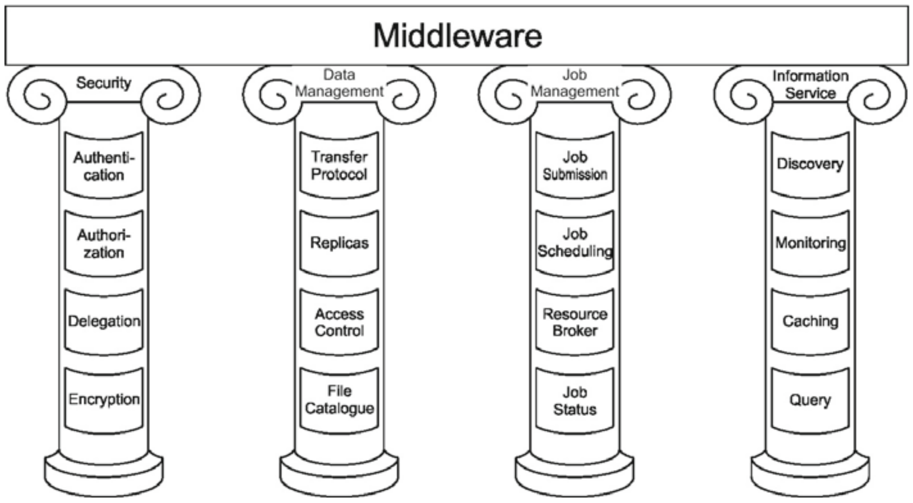


Fig. 1. The four pillars of modern grid middleware

3 Description of the Solution

The computing and storage resources in grid are connected by means of middleware in order to form a virtual distributed computing centre. Apart from the ongoing standardisation efforts, this middleware still follows different concepts and therefore builds grid infrastructures with diverging architectures. Nevertheless one can find a common subset of components which each middleware has to provide (see Fig. 1). The following list gives an overview of these components:

Authentication/Authorization Framework. As grid infrastructures consist of resources that are not parts of local administrative domains, grid middleware has to provide security frameworks to protect against intruders and faked resources. Therefore communication channels may be encrypted, users have to be authenticated and access rights have to be managed.

Data Management. One of the main use cases of a grid infrastructure is distributed data management. Users have to be able to store their data on the infrastructure without worrying about the exact location, backups/replicas and accessibility or reliability of the system. Therefore middleware provides at least some kind of secure transport protocol but also higher level services, for instance file catalogues, which are becoming more and more popular.

Job Management. The most important use case covered by grid middleware is management of the user's jobs computing. In heterogeneous and distributed computing environments this is of course not as easy as running an executable on a local machine. Incoming jobs have to be distributed on the available resources, most often according to specific requirements such as necessary libraries, OS versions or available memory. Furthermore, different jobs may have different priorities. Therefore the job management parts of any middleware in connection with data management are normally the most challenging ones.

Information Service. Usually an ordinary user needs not be aware of the inner structure of a grid. Nevertheless, at least for operators or administrators, it is essential to have access to the current state of the infrastructure, i.e. which services are currently available and what their status is, or which jobs are scheduled or running. Such information can most often be retrieved by querying an information service or cache. Usually the information services are bound to a Virtual Organisation.

We propose that user-friendly environments act as a bridge between the middleware stacks to ensure seamless access to various underlying middleware in terms of job handling and file management. They are structured according to the above list of components. All of the component modules have the same two-layer structure: the abstraction layer provides a generalised model of grid services and grid resources, the implementation layer implements the functionality of the specific grid middleware by a number of middleware plug-ins. Wherever possible, the implementation layer makes use of common standards, such as of those defined by the OGF [6].

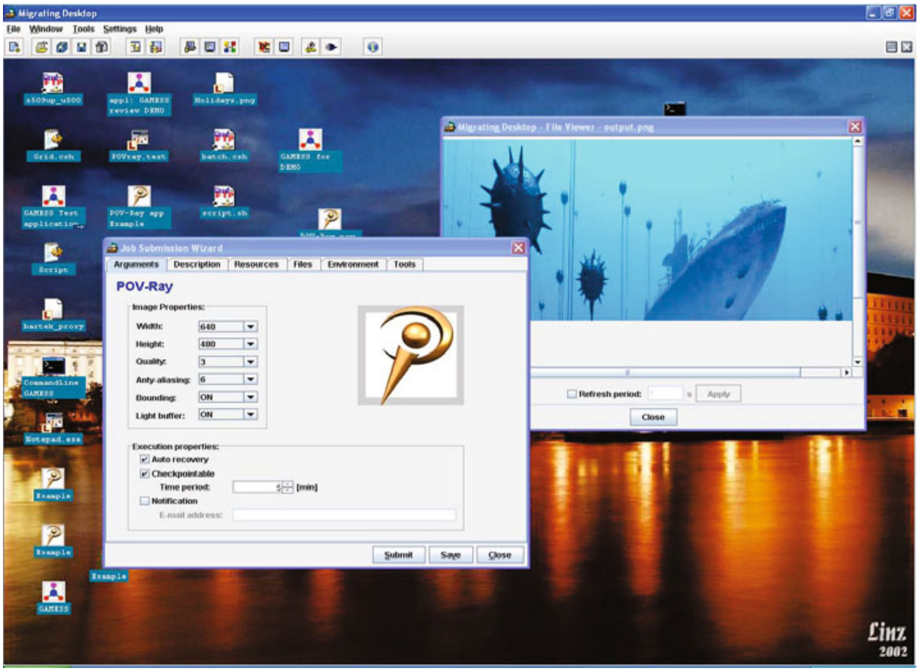
4 Implementation

There are two graphical frameworks available for accessing PL-Grid resources. Both implement the structure described in previous chapter and implement two layer architecture. Although they provide similar features, they are dedicated to different groups of users and can be used in different conditions.

4.1 Migrating Desktop

The Migrating Desktop Platform is based on the client-server paradigm and consists of a Java Web Start application – rich graphical client and server intermediating with grid infrastructures. MD simultaneously hides the complexity of the computing environment behind the advanced graphical user interface, and provides the users with a unified view of the infrastructures used at the interoperation level. It is a powerful and flexible user interface providing transparent user work environment and easy access to the resources. It allows the user to run applications and tools, manage data files, and store personal settings independently of the location or the terminal type.

Open architecture of Migrating Desktop and mechanism of well defined plug-ins and interfaces makes interoperations of various computing architectures available in terms of data handling, job submission and monitoring. The platform offers a framework that can be easily extended on the basis of a set of well-defined plug-ins used to access data, define job parameters, pre-process job parameters, and visualise the job results. Thus the key feature of the Migrating Desktop is



Job Submission Wizard

Arguments | Description | Resources | Files | Environment | Tools

POV-Ray

Image Properties:
- Width: 640
- Height: 480
- Quality: 3
- Anty-aliasing: 6
- Bounding: ON
- Light buffer: ON

Execution properties:
- [✓] Auto recovery
- [✓] Checkpointable
  - Time period: 5 [min]
- [ ] Notification
  - E-mail address:

Submit | Save | Close

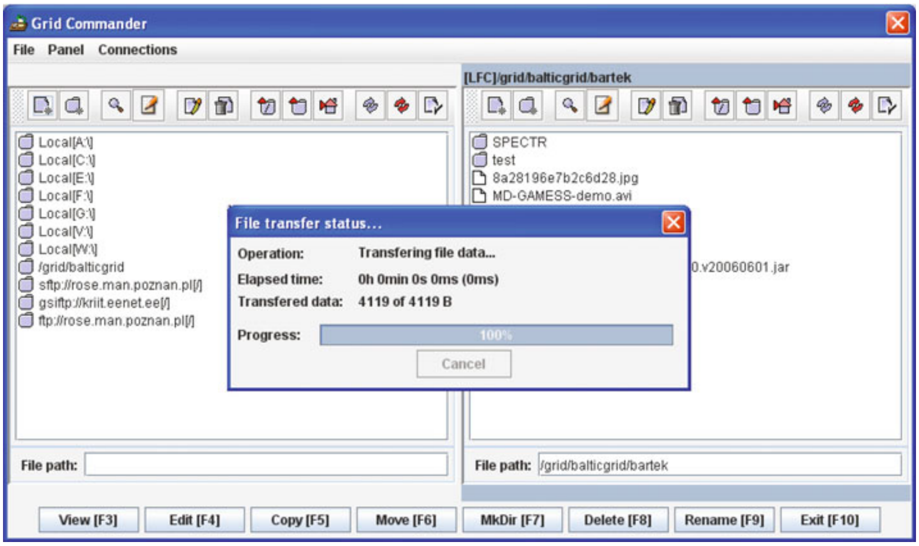
Fig. 2. Screenshot of Migrating Desktop – job defining and visualisation

a possibility of adding various tools and applications, and supporting different visualisation formats easily. Other Migrating Desktop features include:

Single sign-on mechanism. Authentication and authorisation based of X509 certificates schema allows once authenticated users to access resources independently of the underlying infrastructure.

Capability of files exchange among different kinds of storage. Migrating Desktop currently supports several file systems: local, FTP, GridFTP, SRMv2.2, LFN. The framework offers an abstract layer with well-defined interfaces that intermediate while data is exchanged between systems, so the user could have no knowledge of the characteristics of storage parts. Storage, seen as an abstract file systems, is managed by Grid Commander – a graphical tool based on the concept of well-known applications (like e.g. Total Commander) dedicated to file management.

Job submission to various infrastructures using one tool. Migrating Desktop Platform handles the whole job's life-cycle from job defining and submission, up to obtaining results and visualisation of the job outcome. Provided mechanism (based on the idea of plug-ins) allows easy integration of applications within Migrating Desktop and submission of jobs to computing resources of various grid infrastructures.



[LFC]/grid/balticgrid/bartek

Local[A:\]
Local[C:\]
Local[E:\]
Local[F:\]
Local[G:\]
Local[V:\]
Local[W:\]
/grid/balticgrid
sftp://rose.man.poznan.pl[/]
gsiftp://kriit.eenet.ee[/]
ftp://rose.man.poznan.pl[/]

SPECTR
test
8a28196e7b2c6d28.jpg
MD-GAMESS-demo.avi
0.v20060601.jar

File transfer status...

Operation: Transfering file data...
Elapsed time: 0h 0min 0s 0ms (0ms)
Transfered data: 4119 of 4119 B
Progress: 100%

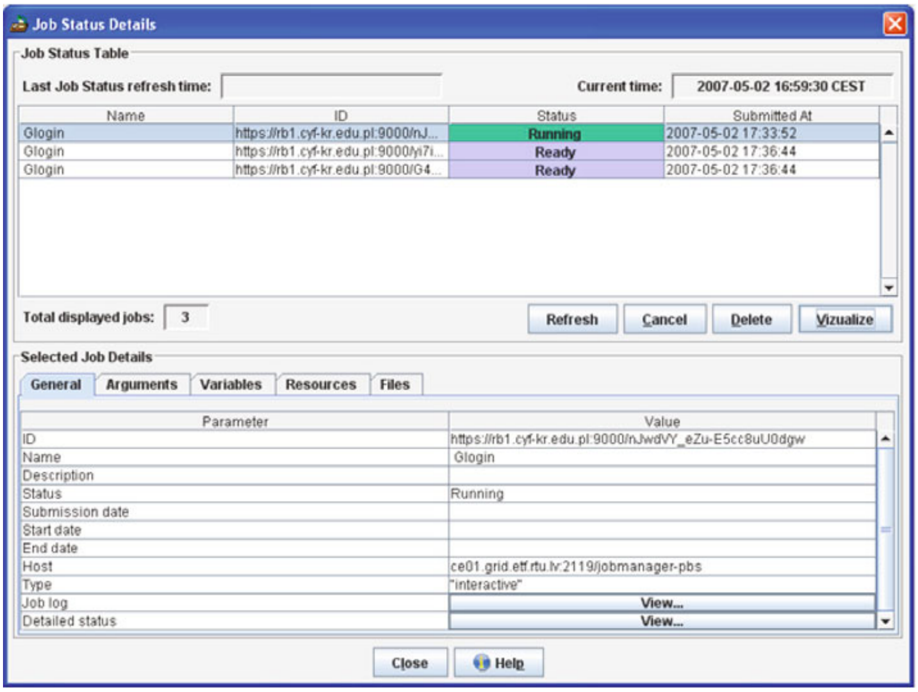
File path:
File path: /grid/balticgrid/bartek

Fig. 3. Migrating Desktop – file transfer between storages of various kind

Simultaneous monitoring of jobs running on different infrastructures. Migrating Desktop features a possibility of obtaining data from heterogeneous mechanisms of job tracking of different middleware, so the status of jobs submitted by the user can be monitored using one dialog.

Migrating Desktop strictly cooperates with the Roaming Access Server (RAS), which intermediates between various grid middleware and applications. The RAS offers a well-defined set of web-services that can be used as an interface for accessing HPC systems and services (based on various technologies) in a common, standardized way. The Roaming Access Server consists of several independent modules responsible for job submission, job monitoring, user profile management, data management, authorisation, and application information management. Using features listed above MD provides users with a tool that makes grid usage much easier and more intuitive, which is essential for encouraging potential beneficiaries to reap profits from using grid infrastructure for compute- and data-intensive applications.

Migrating Desktop was created in the EU CrossGrid Project [8] and developed in the EU Interactive European Grid Project [9], where Migrating Desktop functionality for handling interactive grid applications (i.e. grid applications characterised by the interaction with a person in a processing loop) was implemented. It also proved its usefulness in everyday work of BalticGrid project users communities chosen as a common point for accessing and managing the project applications, tools, resources and services. Currently it is developed within the PL-Grid project.



Job Status Table

Last Job Status refresh time:

Current time: 2007-05-02 16:59:30 CEST

Name	ID	Status	Submitted At
Glogin	https://rb1.cyf-kr.edu.pl:9000/nJ...	Running	2007-05-02 17:33:52
Glogin	https://rb1.cyf-kr.edu.pl:9000/yi7i...	Ready	2007-05-02 17:36:44
Glogin	https://rb1.cyf-kr.edu.pl:9000/G4...	Ready	2007-05-02 17:36:44

Total displayed jobs: 3

Selected Job Details

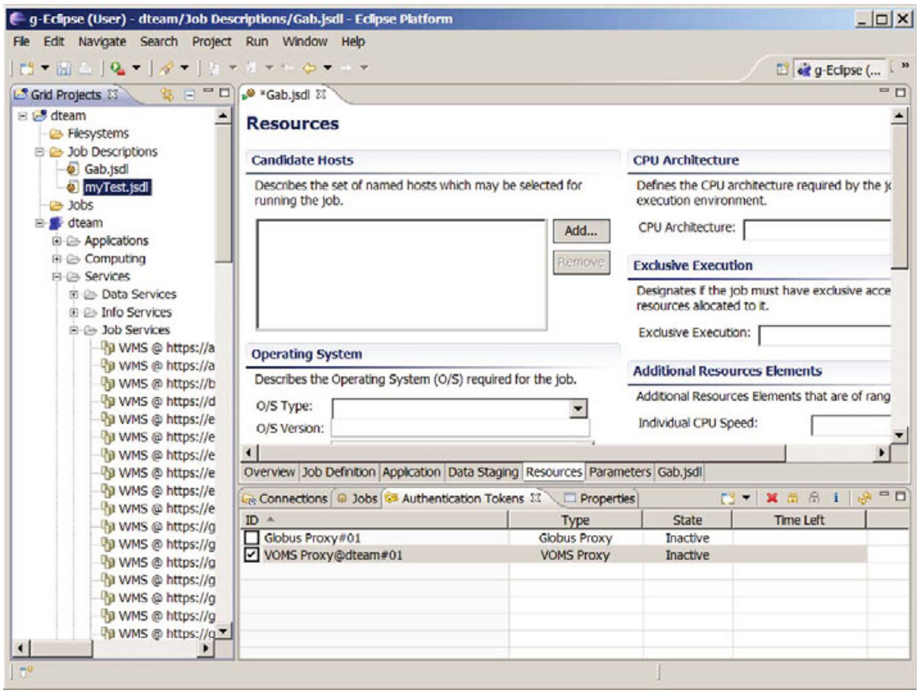
Parameter	Value
ID	https://rb1.cyf-kr.edu.pl:9000/nJwdVY_eZu-E5cc8uU0dgw
Name	Glogin
Description	
Status	Running
Submission date	
Start date	
End date	
Host	ce01.grid.etf.rtu.lv:2119/jobmanager-pbs
Type	"interactive"
Job log	View...
Detailed status	View...

Fig. 4. Migrating Desktop – simultaneous monitoring of jobs running on various grid infrastructures

4.2 g-Eclipse

g-Eclipse is a standalone application – an integrated workbench framework based on well-known Eclipse platform. The g-Eclipse platform is developed as a general framework that can be used by grid users, grid developers and grid operators. The software developed in the g-Eclipse project consists of "core grid plug-ins" for the Eclipse platform. These enable and standardise the access of grid infrastructures from within Eclipse, independent of the grid middleware used. The intuitive interface hides the complexity of using grid services from the end user or application developer, which results in a low entry barrier especially for users and developers new to grid technology. The g-Eclipse platform can be used as a rich client application with user friendly interface to access grid resources, but can also be used as a base for writing customised grid applications using the g-Eclipse model and the g-Eclipse common grid library.

The graphical user interface follows the Eclipse UI guidelines. The intuitive interface hides the complexity of using grid services from the end user or application developer, which results in a low entry barrier especially for users and developers new to grid technology. The graphical front-end of Eclipse is organised with the concepts of views, editors, wizards, preference pages, etc. These components provide the basic functionality to integrate new GUI elements into



Describes the set of named hosts which may be selected for running the job.

Defines the CPU architecture required by the j execution environment.

Designates if the job must have exclusive acce resources allocated to it.

Describes the Operating System (O/S) required for the job.

Additional Resources Elements that are of rang

ID	Type	State	Time Left
☐ Globus Proxy#01	Globus Proxy	Inactive	
☑ VOMS Proxy@dteam#01	VOMS Proxy	Inactive	

Fig. 5. Example layout of g-Eclipse components

the framework. These basic elements are grouped in so-called perspectives. Perspectives determine the visible actions and views within the workbench, but go well beyond this by providing mechanisms for task oriented interaction with resources in the Eclipse Platform. Users can rearrange their workbench and therefore customise it to their needs and habits with the help of these components. An example screenshot from g-Eclipse is shown in Fig. 5.

The g-Eclipse Framework was designed as an open platform for a wide range of tools and the initial contribution was an integrated development environment for Java. The central point of the Eclipse architecture and framework is its plug-in architecture, a component-based software architecture that leads to a clear and modular design. This is achieved by the underlying OSGi framework that defines the dependencies between different plug-ins, and how and when additional plug-ins are loaded. In addition, the Eclipse framework relies on the mechanisms of extension points and extensions. An extension point is a definition of how to enhance an existing functionality. This way of building software components leads to an extensible architecture with well-defined interfaces.

The g-Eclipse framework is a general grid workbench tools that can be extended for many different grid middleware (such as gLite, UNICORE, Globus Toolkit), and comes with exemplary support for the QCG [16] and GRIA [13] middleware. Furthermore an adapter to Amazon's Web services (i.e. S3 and EC2) is available. As a first outcome of this development, Amazon's Simple Storage

Service can be easily accessed from within g-Eclipse and instances in Amazon's Elastic Compute Cloud can be controlled. With this step, g-Eclipse introduces itself to clouds and is no longer only middleware independent, but also grid independent. The important thing for application developers is that they do not need to care about the specific grid middleware support. The same application can work with different grids or infrastructures without any changes.

5 Exploitation

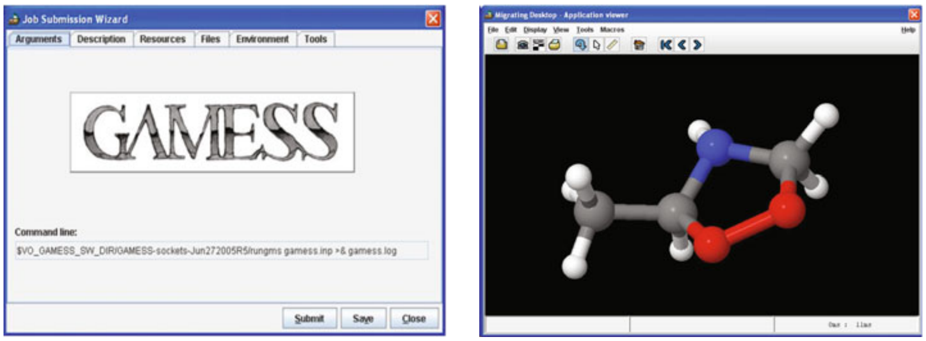
Migrating Desktop and g-Eclipse can be used as a general purpose tool for accessing grid but their power is also in the ability to adapt to a specific application needs. In the following sections the methods of application integration are explained and exemplary integrations of Gamess and Gaussian applications are presented. GAMESS and Gaussian applications were chosen to prove correctness of integration mechanisms and procedures as both applications are very popular within a quantum chemistry community which is one of the main grid users group.

The Migrating Desktop Platform and g-Eclipse were successfully deployed on the PL-Grid infrastructure. The deployment procedure included evaluation of products features, thorough examination of security vulnerabilities as well as general tests. Now, both tools are used by PL-Grid users as a common point for accessing and managing the project applications, tools, resources and services.

5.1 Integrating Application with Migrating Desktop

Applications whose requirements are accomplished by Migrating Desktop functionality can be integrated without much effort. In other cases application developers have to add an extra functionality to fulfill application's specific needs using concept of plug-ins as integration points between the Migrating Desktop and applications. Depending on the functionality that the developer would like to add to the Migrating Desktop framework, he/she has to choose what kind of plug-in has to be implemented. The developer can choose from the types described in this document:

- Job input plug-in – for defining specific job input parameters;
- Job process plug-in – if any additional activities should be performed before submitting a job (e.g. job parameters have to be optimised before submission);
- File viewer plug-in – if applications produce a non-standard format files that are not handled by the Migrating Desktop and that have to be presented in a user-friendly graphical format;
- Application viewer plug-in – for visualisation of job results that are produced as a set of files or that have to be pre-processed (e.g. any animations, visualisation of meteorological data such as weather forecast map, etc.);
- Tool plug-in – for adding any Java tools to the Migrating Desktop framework.



Command line:

```
$VO_GAMESS_SW_DIR/GAMESS-sockets-Jun272005R5/rungms gamess.inp >& gamess.log
```

Fig. 6. GAMESS application – job submission plug-in (left) and job visualization plug-in (right)

Integration of GAMESS application within Migrating Desktop Framework includes:

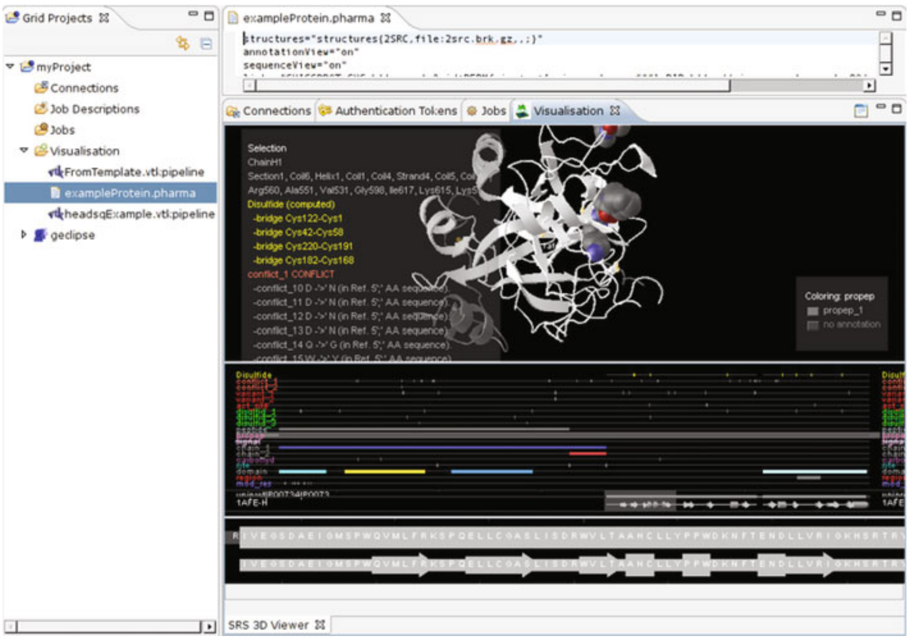
– implementation of specific plug-ins for job defining to make job submission easier and more intuitive for the user (see Fig. 6),
– adaptation of the open-source JMol application as a Migrating Desktop plug-in for visualization of application results (see Fig. 6),
– preparation and publishing a user guide, describing in details the procedure of defining, monitoring and results visualising of GAMESS application.

5.2 Integrating Application with g-Eclipse

g-Eclipse can also be used as a base for users' applications which should access grid resources. The easiest way is to provide an application description which defines application parameters. Such description is then automatically converted to a form of graphical dialog containing fields for entering application parameters. With the help of the g-Eclipse framework an application developer or integrator can prepare applications, both simple or workflow, and share them with the application users group.

A more advanced way of integrating applications can be achieved by providing more advanced plug-ins such as input file editor or result visualisation. g-Eclipse provides visualisation functionality that can be used by application plug-ins. For some applications also submission support plug-in is necessary to prepare a proper job description and input scripts. Plug-ins can also register additional job description format and supporting editors (textual or graphical ones) which can submit and handle specific application input file format. By embedding applications into g-Eclipse developers need not provide client interface or grid resources management. All elements, menus and actions to submit a job, manage a job, update status or transfer files are provided by the g-Eclipse framework. An example integration with pharmaceutical integration is shown in Fig. 7.

The advanced method of integration is to prepare custom applications. By extensively using the Eclipse extension mechanism in combination with



```
structures="structures(2SRC,file:2src.brk.gz,.;)"
annotationView="on"
sequenceView="on"
```

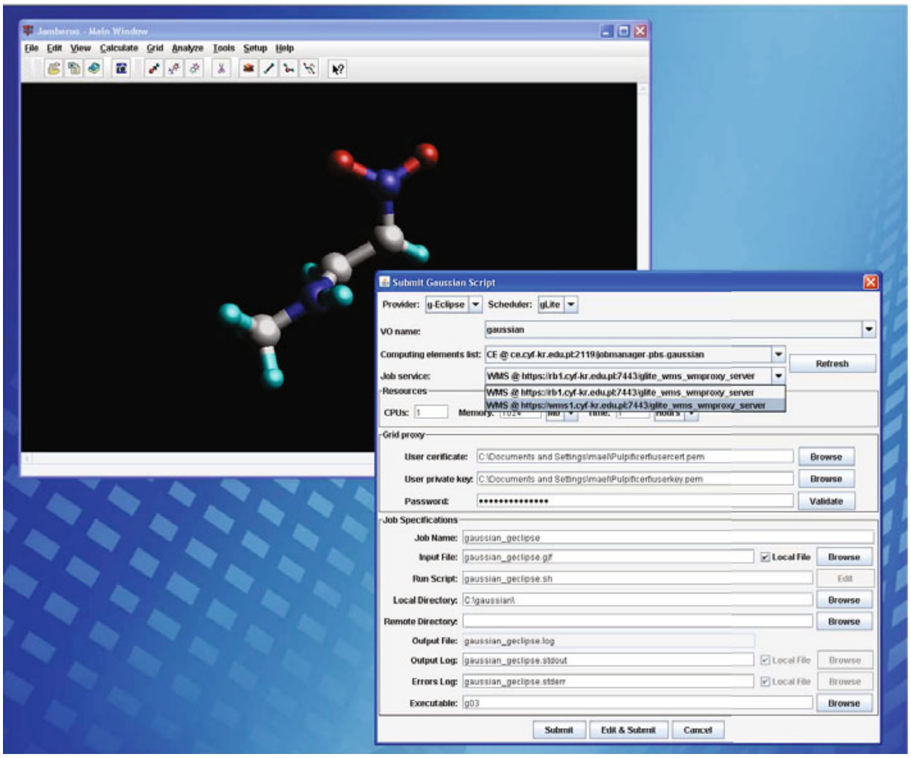
Selection
ChainH1
Section1, Coil6, Helix1, Coil1, Coil4, Strand4, Coil5, Co
Arg560, Ala551, Val531, Gly598, Ile617, Lys615, LysS
Disulfide (computed)
-bridge Cys122-Cys1
-bridge Cys42-Cys58
-bridge Cys220-Cys191
-bridge Cys182-Cys168
conflict_1 CONFLICT
-conflict_10 D ->' N (in Ref. 5;' AA sequence).
-conflict_11 D ->' N (in Ref. 5;' AA sequence).
-conflict_12 D ->' N (in Ref. 5;' AA sequence).
-conflict_13 D ->' N (in Ref. 5;' AA sequence).
-conflict_14 Q ->' G (in Ref. 5;' AA sequence).
-conflict_15 W ->' Y (in Ref. 5;' AA sequence).

Fig. 7. Pharmaceutical application integrated into g-Eclipse

object-oriented design patterns, the framework can be easily extended by application-specific implementation. The application is not plugged into g-Eclipse, but rather built on top of it. It follows the approach of the Rich Client Platform (RCP) of Eclipse. Dedicated applications can be built by using parts of g-Eclipse as a common library to handle resources and grid access. With such an approach existing applications can be gridified. As the g-Eclipse core model is grid middleware independent, the same application can access resources from various grid middleware. Example integration with JMolEditor is shown in Fig. 8. The quantum chemistry task prepared in JMolEditor can be easily submitted to PL-Grid resources for Gaussian computation.

6 Summary

The Migrating Desktop Platform and g-Eclipse Platform interoperation mechanisms described above provide the user with unified and intuitive access to different kinds of computing infrastructures. This feature allows users to forget about the complexity of underlying environments and focus on their research without the need of understanding grid technologies. User-friendly and intuitive frameworks such as g-Eclipse and Migrating Desktop make the look-and-feel of grid infrastructures similar to that of existing computer desktop systems



Provider: g-Eclipse Scheduler: gLite
VO name: gaussian
Computing elements list: CE @ ce.cyf-kr.edu.pl:2119/jobmanager-pbs-gaussian
Job service: WMS @ https://rb1.cyf-kr.edu.pl:7443/glite_wms_wmproxy_server
WMS @ https://rb1.cyf-kr.edu.pl:7443/glite_wms_wmproxy_server
WMS @ https://wms1.cyf-kr.edu.pl:7443/glite_wms_wmproxy_server
Resources
CPUs: 1 Memory: 1024 MB Time: 1 hours
Grid proxy
User certificate: C:\Documents and Settings\mael\Pulpit\cert\usercert.pem
User private key: C:\Documents and Settings\mael\Pulpit\cert\userkey.pem
Password: ••••••••••••••
Job Specifications
Job Name: gaussian_geclipse
Input File: gaussian_geclipse.gjf ✓ Local File
Run Script: gaussian_geclipse.sh
Local Directory: C:\gaussian\
Remote Directory:
Output File: gaussian_geclipse.log
Output Log: gaussian_geclipse.stdout ✓ Local File
Errors Log: gaussian_geclipse.stderr ✓ Local File
Executable: g03

Fig. 8. JMolEditor enhanced with grid capabilities

which people are already familiar with. The ease of use of the resources independently on their type in an obvious way could attract new users to benefit from the newest computing technologies.

Diversity and complexity of computing environments very often discourages users who could benefit from usage of computing resources. Intuitive interface, open architecture and possibility of interoperation between various middlewares makes the described frameworks valuable tools, that can be used both by inexperienced users or skilled application developers, supporting their work with various grid infrastructures. The Migrating Desktop and g-Eclipse were successfully deployed in the PL-Grid project. The frameworks are not dedicated to any specific user community, and can be used for different user needs, different infrastructures, Virtual Organisations and applications. Example integration was shown for fusion, chemical and pharmaceutical applications. g-Eclipse and Migrating Desktop are open source projects with user communities gathered around it, which allows for sustainable development.

References

1. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) NPC 2005. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005)
2. Laure, E., et al.: Programming the Grid with gLite. Computational Methods in Science and Technology 12(1), 33–45 (2006)
3. Romberg, M.: The UNICORE Architecture: Seamless Access to Distributed Resources. In: Proceedings of the 8th IEEE International Symposium on High Performace Distributed Computing (HPDC), pp. 287–293 (1999)
4. Kupczyk, M., Lichwała, R., Meyer, N., Palak, B., Płóciennik, M., Stroiński, M., Wolniewicz, P.: The Migrating Desktop as a GUI Framework for the "Applications on Demand" Concept. In: Bubak, M., van Albada, G.D., Sloot, P.M.A., Dongarra, J. (eds.) ICCS 2004, Part I. LNCS, vol. 3036, pp. 91–98. Springer, Heidelberg (2004)
5. Kornmayer, H., Stümpert, M., Gjermundrød, H., Wolniewicz, P.: g-Eclipse – A Contextualised Framework for Grid Users, Grid Resource Providers and Grid Application Developers. In: Bubak, M., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2008, Part III. LNCS, vol. 5103, pp. 399–408. Springer, Heidelberg (2008)
6. Open Grid Forum, `http://www.gridforum.org/`
7. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) NPC 2005. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005)
8. Crossgrid project, `http://www.eu-crossgrid.org/`
9. IntEuGrid project, `http://www.interactive-grid.eu/`
10. UK National Grid Service Job Portal, `https://portal.ngs.ac.uk/`
11. UCLA Grid Portal, `https://grid.ucla.edu`
12. Kacsuk, P., Sipos, G.: Multi-Grid, Multi-User Workflows in the P-GRADE Portal. Journal of Grid Computing 3(3-4), 221–238 (2005)
13. Surridge, M., Taylor, S., De Roure, D., Zaluska, E.: Experiences with GRIA – Industrial Applications on a Web Services Grid. In: Proceedings of the First International Conference on e-Science and Grid Computing, pp. 98–105 (2005)
14. Dziubecki, P., Grabowski, P., Krysiński, M., Kuczyński, T., Kurowski, K., Piontek, T., Szejnfeld, D.: Online Web-based Science Gateway for Nanotechnology Research. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 205–216. Springer, Heidelberg (2012)
15. Benedyczak, K., Stolarek, M., Rowicki, R., Kluszczyński, R., Borcz, M., Marczak, G., Filocha, M., Bała, P.: Seamless Access to the PL-Grid e-Infrastructure Using UNICORE Middleware. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 56–72. Springer, Heidelberg (2012)
16. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)

Online Web-Based Science Gateway for Nanotechnology Research

Piotr Dziubecki, Piotr Grabowski, Michał Krysiński, Tomasz Kuczyński, Krzysztof Kurowski, Tomasz Piontek, and Dawid Szejnfeld

Poznań Supercomputing and Networking Center, Poznań, Poland
{deepres,piotrg,mich,docentt,krzysztof.kurowski,
piontek,dejw}@man.poznan.pl
http://www.man.poznan.pl

Abstract. The main objective of Science Gateways is to give users remote access to supercomputers and large-scale computing environments in an interactive, web-based and graphical manner. We present a tool, called Vine Toolkit, that has been successfully used as a core web platform for various Science Gateways. Vine Toolkit is a modular, extensible and easy-to-use tool as well as a high-level Application Programming Interface (API) for various applications, visualization components and building blocks. In a nutshell, it allows interoperability between many different HPC and grid technologies within the service layer. As a result, Vine Toolkit provides an ability to build a portal upon different HPC technologies working together to deliver a complete solution to the users. In this article, we briefly describe our most complex and feature-rich project – the Nanotechnology Gateway, as well as a set of tools relevant to advanced scientific portals, development of which was driven by various requirements defined by scientists and gathered in scope of the PL-Grid project.

Keywords: Science Gateway, Web2.0, ABINIT, Vine Toolkit, Liferay, Java, Adobe Flex, Material Science, Nanotechnology.

1 Introduction

Advanced web-based graphic- and multimedia-oriented user interfaces (GUIs) designed for scientists and engineers could change the way their users collaborate, share computing experiments and data, and work together to solve day-to-day problems. Moreover, the future science and engineering gateways influence not only the way the users access their data, but also how they control and monitor their demanding computing simulations. In order to allow users to communicate remotely with supercomputers and large-scale computing environments in a more interactive and graphical manner, we present a tool, called Vine Toolkit, that has been already successfully used as a core web platform for various Science Gateways [1], [2], [3]. Vine Toolkit is a modular, extensible and easy-to-use Java-based tool as well as high-level Application Programming Interface (API) for various applications, visualization components and building blocks. In this way, it allows interoperability between many different HPC and grid technologies. Below, we present a list of modules and plugins currently provided by Vine Toolkit as well as standards supported by this tool:

– job submission and monitoring: QosCosGrid 2.0 [4], [5], UNICORE 6 [6], gLite3 [7], GRIA 5.3 [8], Globus Toolkit 4.0.x, 4.2.1 [9],
– data access and management: IRODS [10], Storage Resource Broker, Storage Resource Manager [11], OGSA-DAI 2.2 [12],
– supported standards and services: BES [13], JSDL [14], RUS [15], Active Directory [16].

Moreover, Vine Toolkit supports also Adobe FLEX and BlazeDS technologies, allowing developers to create rich and advanced web applications similar to many stand-alone Graphical User Interfaces. Additionally, the tool has been integrated with well-known open source web frameworks such as Liferay and Gridsphere [17]. In this chapter, we briefly describe new technological solutions relevant to advanced scientific and engineering portals. Their development was driven by various requirements defined by scientists and gathered in scope of the PL-Grid project.

The rest of this chapter is organized as follows. In Section 2, related work in area of Science Gateways and Science Gateway frameworks is presented. Section 3 briefly describes our main motivations. Then, in Section 4, there are short descriptions of Scientific Applications developed with use of the described solution. Finally, Section 5 contains information about the planned future work.

2 Related Work

Currently, there are several approaches to implementing the Science Gateway concept. There is a group of frameworks facilitating design and creation of Science Gateways as well as already running portals ready to be used by the scientists. Among available tools that help users to create advanced science gateways, P-GRADE is a good example of a parallel application development system for Grid and clusters [18]. It uses Globus, Condor-G, ARC, BOINC and MPICH-G2 as grid-aware middleware to conduct computations. EnginFrame is another good example of a web-based front-end for simple job submission, tracking and integrated data management for HPC applications and other services [19]. EnginFrame can be easily plugged to several different schedulers or grid middlewares like: Platform LSF, Sun Grid Engine, PBS, or gLite. A slightly different approach to an API that provides basic functionality required to build distributed applications is presented by SAGA [20]. It focuses on delivering a set of programming interfaces covering the functionality of an HPC-aware application. Unfortunately it does not provide a GUI support, needed to create an easy-to-use Science Gateway (see the Vine Toolkit and SAGA comparison Table 1).

Fortunately, apart from development environments, there are also several sites offering direct access to specialist applications for scientists. A good example

Table 1. SAGA vs Vine Toolkit comparison

Middleware	**Vine Toolkit**	**SAGA – Java adaptors**
gLite 3 – Cream	Yes	Yes – JSAGA
gLite 3 – WMS	Yes	Yes – JSAGA
gLite 3 – JDL	Yes	under development – JSAGA
Globus Toolkit	Yes (4.0.x, 4.2.1)	Yes – JSAGA/JavaGAT
Globus Toolkit – MyProxy	Yes	Yes – JSAGA
Globus Toolkit – gsiftp	Yes	Yes – JSAGA
Globus Toolkit – WS-GRAM	Yes	Yes – JSAGA
BES	Yes	Yes – JSAGA
JSDL	Yes	Yes – JSAGA
GRIA	Yes (5.3)	No
UNICORE 6	Yes	Yes – JSAGA
Active Directory	Yes	No
Java Keystore	Yes	Yes – JSAGA
X509 Certificates	Yes	Yes – JSAGA
Storage Resource Manager	Yes	Yes – JSAGA
Storage Resource Broker	Yes	Yes – JSAGA
(S)FTP, SSH, HTTP(S), ZIP	Partly (http, SSH applet)	Yes – JSAGA/JavaGAT
local data management	Yes	Yes – JSAGA
WebDav	Yes	No
VOMS	Yes	Yes – JSAGA
iRODS	Yes	Yes – JSAGA
NAREGI (Super Scheduler)	NO	Yes – JSAGA
OGSA-DAI	Yes (2.2)	No
RUS	Yes	No
QosCosGrid	Yes	No
GRMS,GRMS3	Yes	No

of such well-established Science Gateway is `nanohub.org` [21]. Its main purpose is to deliver tools and materials, as well as to help with education, research and collaboration in nanotechnology. According to statistics provided by `nanohub.org`, they have over 10,000 simulation users and over 50,000 interactive users. This is a result of a wide range of nanotechnology applications and simulations/visualizations available within the portal. Regarding technical details, the `nanohub.org` web interface serves as a proxy between a remotely installed application and the end user. In order to achieve that, a Java applet with VNC plugin is used to connect the user to the remote GUI of the desired application.

Another example of a collaborative environment where scientists can safely publish their workflows and experiment plans, share them with groups and find those published by other users, is `myExperiment.org` [22]. In this approach, workflows, other digital objects and bundles (called Packs) can be swapped, sorted and searched like photos or videos on the Web. Unlike Facebook or MySpace, myExperiment fully understands the needs of the researchers; making it really easy for the next generation of scientists to contribute to a pool of scientific methods, build communities and form relationships. In this way, it reduces the time needed to experiment and to share expertise, as well as helps to avoid reinvention. At last, we should not forget about similar projects – GridSpace [32] and InSilicoLab [31], that have been developed in the frame of PL-Grid.

The aforementioned examples differ from each other in implementation and visual form, still all of them were designed to solve specific scientific problems.

3 Motivation

Since its planning stage, one of the main goals of the Science Gateway for nanotechnology was to combine experimental nano-scale measurements, real data analysis and verification together with advanced modeling and numerical simulations on the PL-Grid computing resources. According to our analysis, based on the user feedback from the "Vine-users" mailing list [30] and surveys conducted at the `vinetoolkit.org` website, many researchers need efficient and easy-to-use interfaces to their scientific applications. However, they do not have a need for understanding of the underlying middleware services and IT systems. Therefore, we decided to develop a user-friendly, domain-focused Science Gateway that allows scientists to solve, run and control nanotechnology simulations in an intuitive way. The proposed solution is a web-based collaborative platform, that takes advantage of Vine Toolkit and Liferay, along with web-based tools integrated with QosCosGrid middleware [4], [5]. Additionally, scientific applications like ABINIT and Quantum Espresso are supported. The basic mode of the gateway use covers many functionalities, such as: preparation of input data, submission of jobs, monitoring and controlling simulations, post-processing and analysis of their outcomes, data storage and archiving [23]. In advanced mode, users can benefit from additional actions, like data migration, conversion, post-processing and visualization.

4 Scientific Applications

The most advanced part of our Science Gateway for nanotechnology research consists of two web-based interfaces for scientific applications: ABINIT and Quantum Espresso [24], [25].

The ABINIT simulation software package allows users to solve problems like finding the total energy, charge density and electronic structure of systems made of electrons and nuclei within Density Functional Theory (DFT), using pseudopotentials and planewave basis. ABINIT also includes options to optimize a geometry according to the DFT forces and stresses, and to perform molecular dynamics simulations using these forces, as well as to generate dynamical matrices, Born effective charges, and dielectric tensors. Excited states can be computed within Time-Dependent Density Functional Theory (for molecules) or within Many-Body Perturbation Theory. However, ABINIT itself defines several requirements towards the user:

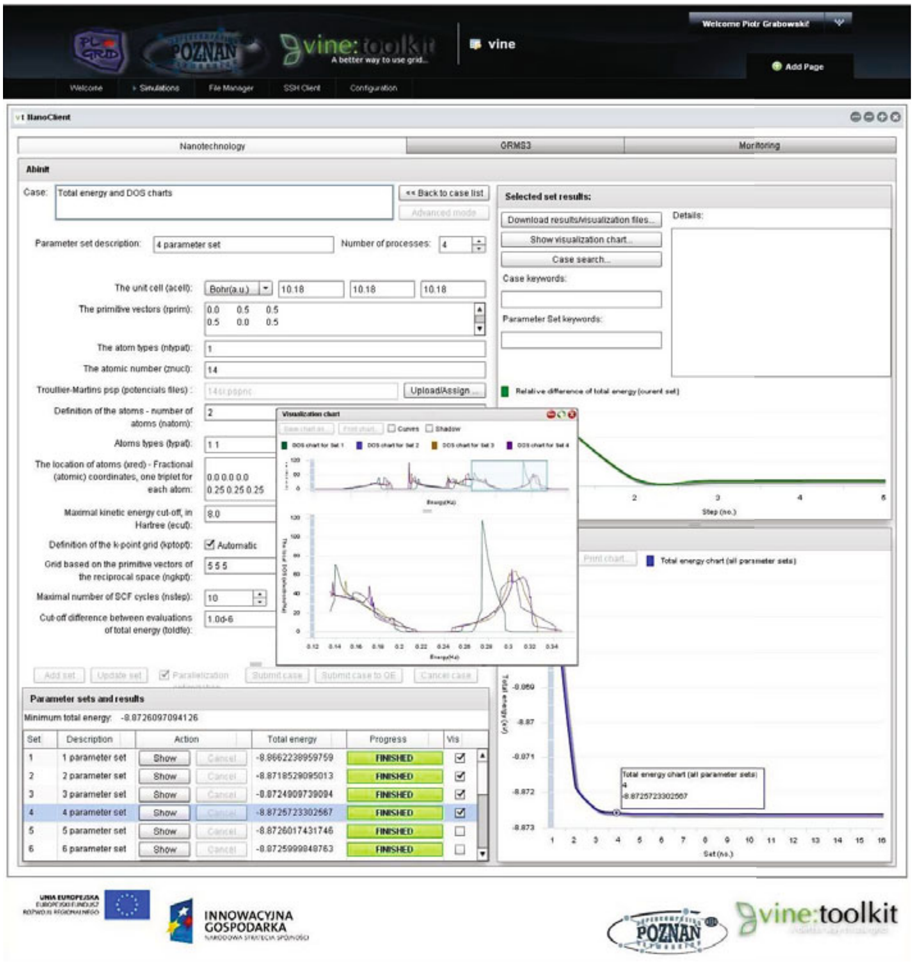
- the user needs to know how to translate domain-specific problem to an ABINIT task;

- the command-line tool requires from the user experience with ABINIT in/out data structures;
- in case of parallel or large scale deployment, the user has to be familiar with many complex IT and HPC technologies.

In order to hide this complexity, we successfully developed a web-based collaborative interface enabling access to ABINIT, with use of Vine Toolkit and Adobe Flex [26]. Consequently, we are able to support fully transparent web-based access to sequential and parallel ABINIT executions deployed on computing clusters within the PL-Grid infrastructure. The list below contains the main features of our web interface:

- basic and advanced modes provide support both for experts and beginners;
- instead of using SSH, file transfer protocols, or other tools, the users can manage even complex simulation and data operations using web browsers;
- several tools facilitating the process of data visualization and analysis.

In addition, to meet specific users requirements, we developed web-based tools which provide means to analyze input/output parameters in a form-like panel – for ABINIT and other DFT code, e.g. Quantum Espresso. In this way, some of the parameters that were not reflected in the parameter input form can be easily added using a new rich editor of the interface advanced mode. One should note, that the inputs required for ABINIT job submission are pseudopotentials files. Using another web-based tool – Vine Toolkit File Manager – the users can easily access, copy and assign appropriate files stored on remote machines. What is even more important, the user is able to visualize data in the Science Gateway immediately after the simulation is completed. The calculated functions of the total density of electronic states (DOS) can be easily displayed as visualization results. It is also possible to view multiple series from different simulations on the same chart. The chart can be dynamically cropped and zoomed. What is more, special values like the Fermi energy can be marked on the chart. The example layout of the basic mode for calculation of total energy using ABINIT is presented in Fig. 1. The example case consists of 16 parallel executions of ABINIT using MPI processes on four cores. Fig. 1 shows also sample charts generated after ABINIT simulations. Moreover, in case of band structure calculations all required post-processing procedures are realized in the background, and only a final picture containing band structure chart is shown to the users. An additional, useful component is the Nano Structure Builder graphical tool, which allows to create and edit 3D crystallographic super cells and to generate ABINIT-compatible descriptions. Therefore, the users are able to prepare the whole simulation within the Science Gateway, without the need to use third party software. Nano Structure Builder is a web-based application that provides functionalities available in both commercial and open source tools, such as XCrysDen or GoVasp [27], [28]. XCrysDen is an open source application that enables visualizations of crystalline and molecular structures. GoVasp is a commercial software providing, i.a., a structure builder – a graphical tool that enables creation and manipulation of solid state structures. Currently, Nano



Case: Total energy and DOS charts

Parameter set description: 4 parameter set
Number of processes: 4

The unit cell (acell): Bohr(a.u.) 10.18 10.18 10.18

The primitive vectors (rprim):
0.0 0.5 0.5
0.5 0.0 0.5

The atom types (ntypat): 1

The atomic number (znucl): 14

Troullier-Martins psp (potencials files) : 14si.pspnc

Definition of the atoms - number of atoms (natom): 2

Atoms types (typat): 1 1

The location of atoms (xred) - Fractional (atomic) coordinates, one triplet for each atom:
0.0 0.0 0.0
0.25 0.25 0.25

Maximal kinetic energy cut-off, in Hartree (ecut): 8.0

Definition of the k-point grid (kptopt): ✓ Automatic

Grid based on the primitive vectors of the reciprocal space (ngkpt): 5 5 5

Maximal number of SCF cycles (nstep): 10

Cut-off difference between evaluations of total energy (toldfe): 1.0d-6

Parameter sets and results

Minimum total energy: -8.8726097094126

Set	Description	Action		Total energy	Progress	Vis
1	1 parameter set	Show	Cancel	-8.8662238959759	FINISHED	✓
2	2 parameter set	Show	Cancel	-8.8718529095013	FINISHED	✓
3	3 parameter set	Show	Cancel	-8.8724909739094	FINISHED	✓
4	4 parameter set	Show	Cancel	-8.8725723302567	FINISHED	✓
5	5 parameter set	Show	Cancel	-8.8726017431746	FINISHED	
6	6 parameter set	Show	Cancel	-8.8725999848763	FINISHED	

Total energy chart (all parameter sets)
4
-8.8725723302567

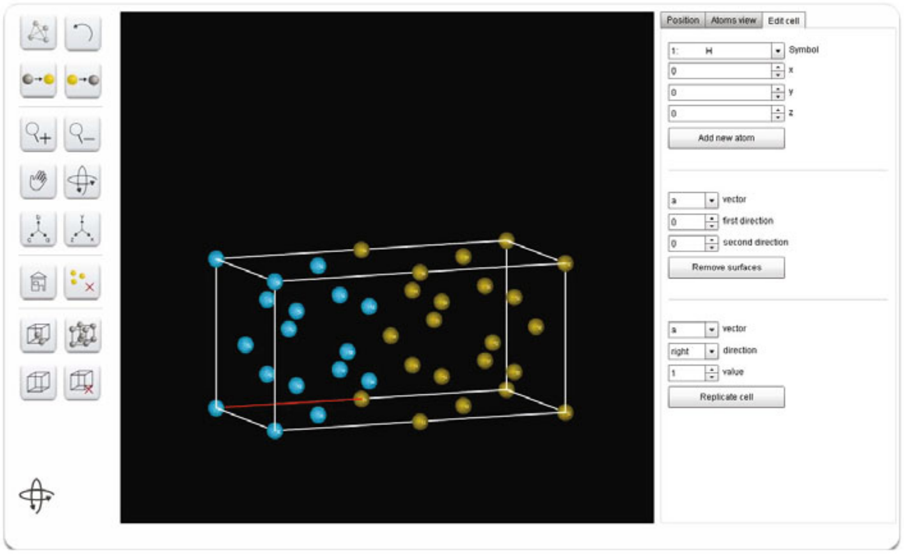
Fig. 1. Web-based interface for ABINIT simulation with DOS charts

Structure Builder does not cover all functionalities of the aforementioned applications; however, it enables ab inito creation of crystal structures, their edition and manipulation.

Nano Structure Builder has been divided into two separate web-based components: NanoBuilder and NanoEditor. The first generates geometry of crystallographic coordinates of at least one of the 230 symbols of symmetry group in the Hermann-Mauguin notation and lattice vectors and angles. The generated results are passed to NanoEditor. NanoEditor exploits the received coordinates of all atoms in the cell, in accordance with established symmetry, to generate graphical representation of the cell. The cell can be manually edited and modified with the provided tools. Moreover, the results obtained in that process can be saved as a file, and, thus, further used as an input to ABINIT.

NanoBuilder was created using the Adobe FLEX technology, wrapping the Spacegroup software. Spacegroup is a command line tool, which produces a crystal geometry; however, it can be used only under Unix systems. NanoBuilder gives its users an opportunity to use the tool within a web browser: defining basic information (space group by its Hermann-Mauguin symbol, lattice vectors lengths, lattice angles, number of unit cells in each direction and whether a unit cell should be found), adding atoms (by selecting them from a displayed periodic table) and defining position of each of the atoms. All these information is, invisibly to the user, passed to the Spacegroup software, which returns results as an output file.

NanoEditor is a graphical tool, which was also created in FLEX technology. It enables displaying a crystal cell in three dimensions. To achieve such a functionality, the Papervision3D library – an open source real-time 3D engine for the Flash platform – was used. The main purposes of using NanoEditor are: creation, edition and manipulation of super-cells. First, the user loads results obtained in NanoBuilder by selecting a proper file in a File Browser. When the file is loaded, its graphical representation is displayed in the middle panel of the component (Fig. 2). NanoEditor provides a whole set of tools to navigate and manipulate the model. The user can freely rotate it in any direction, or shift it, if the model does not fit entirely in the window. There is also a possibility to zoom in and zoom out the model, which is helpful when viewing the details of the cell. A very helpful feature of the tool is the ability to use two coordinate systems: Cartesian and cell coordinate system. Such distinction is needed because of the different functions available in the application. The first can be used when a new atom is added (coordinates need to be defined along x, y and z axis). The latter, in turn,



Symbol: 1: H
x: 0
y: 0
z: 0

vector: a
first direction: 0
second direction: 0

vector: a
direction: right
value: 1

Fig. 2. Model of a unit cell displayed in a 3 dimensions

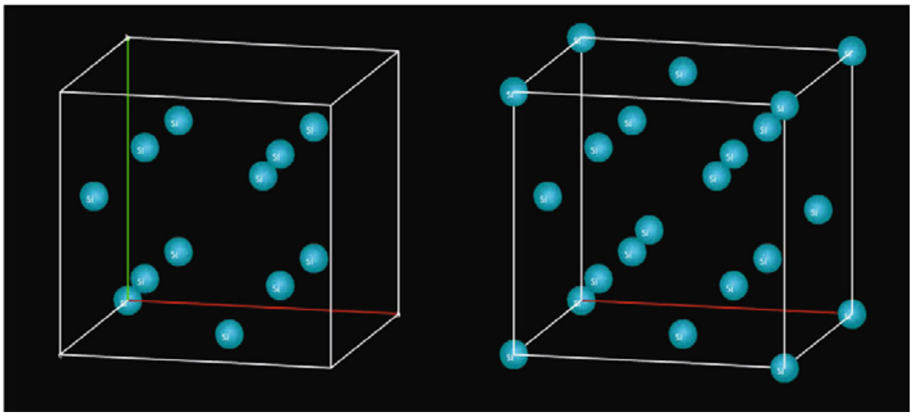


Fig. 3. Difference between primitive cell (on the left side) and the unit cell (on the right side)

helps to recognize directions of the cell when, for instance, some surface needs to be removed (surface is defined along a, b or c axis). The user can display any of the coordinate system or hide both of them.

There is also a possibility to show (and hide) the border of a crystal cell, which is helpful when identifying a crystal system. Moreover, the user is able to switch between two display modes of the unit cell. By design, a crystal cell is displayed as a primitive cell (translational asymmetric unit). But there is an additional option – displaying a nicely-cut unit cell (shortly unit cell). The difference between both cell views is demonstrated in Fig. 3. The unit cell displays additional atoms in places where multiplied cells are connected after the cell multiplication process.

A crucial functionality of NanoEditor is the edition of a crystal cell. The tool enables selection of a single atom or group of atoms, which can be further altered. It is possible to remove selected atoms and to change their positions by shifting them by, provided in Angstroms, values along each axis. To check the current position of an atom, the user can display it after selecting a proper atom. What is more, the user can display the distance between any pair of atoms. In addition, the user can add a new atom at every position. NanoEditor enables also replicating a whole cell a specified number of times along selected lattice vector. It can be done in both primitive and unit display modes. After achieving a desired form of a cell, the user can define vacuums. It is done by removing proper surfaces of the crystal cell. All the actions mentioned above, except for defining the vacuum, can be undone. It allows the users to make alterations without worrying about irreparable consequences. Such functionality was provided on account of the fact that users are prone to make some errors and like to have an ability to go back.

The last, but not least, important functionality of NanoEditor is its ability to display model parameters in ABINIT file format. Such parameters include: number of atoms, lattice vectors, cell angles, atom types and many more. After copying such output, the user can paste it as an input to another nano product

– Nano-Science Gateway. Thanks to that, the obtained model can be further processed and used.

The presented editor has become a really useful application for nano scientists who tested its prototype. They were able to design a nano structure without a need to have access to very extensive specialist software packages, like GoVasp. The source code is distributed as open source, therefore, it could be further developed in the future. It is also a great example of how we can use Vine mechanisms to prepare advanced graphical tools for end users.

In order to show how the described components are placed within a portal, we present the following diagram (Fig. 4). It depicts the elements of the portal, underlying services and dependencies between them.

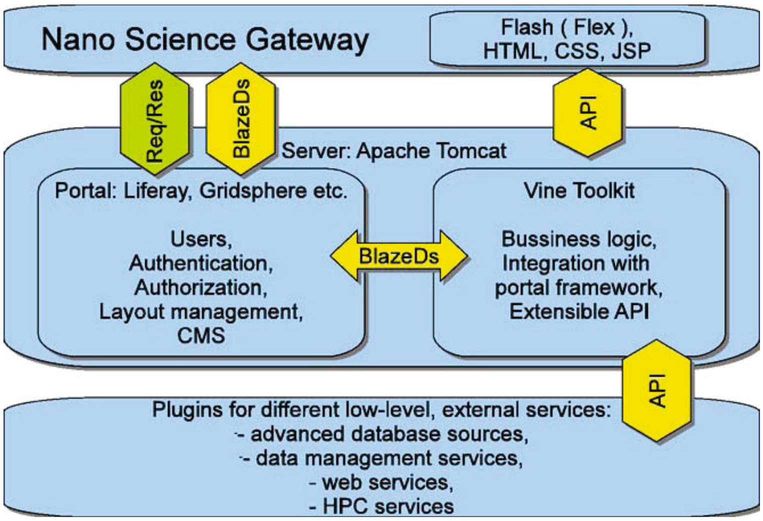
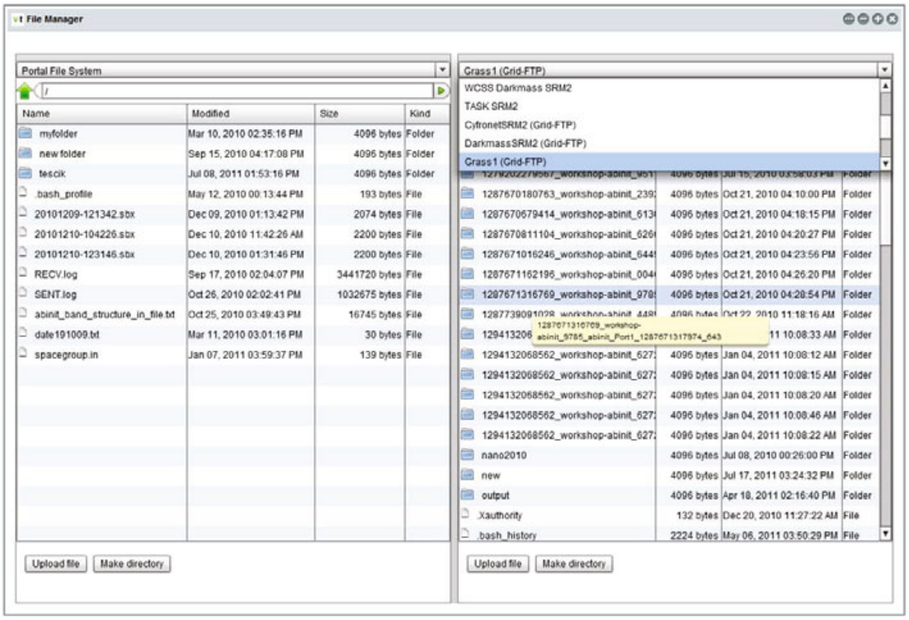

Nano Science Gateway
Flash (Flex), HTML, CSS, JSP
Req/Res
BlazeDs
API
Server: Apache Tomcat
Portal: Liferay, Gridsphere etc.
Users, Authentication, Authorization, Layout management, CMS
BlazeDs
Vine Toolkit
Bussiness logic, Integration with portal framework, Extensible API
API
Plugins for different low-level, external services:
- advanced database sources,
- data management services,
- web services,
- HPC services

Fig. 4. Science Gateway deployment diagram

The Science Gateway for Nanotechnologists consists of two kinds of components available to a logged-in user: standard Vine Toolkit components for managing user profile, file management and SSH client; and specialized components designed for Nanotechnology environments. Both component groups are designed according to specific rules of grid portal design, based on the work conducted within the BEinGrid project [29]. One of the most important portal design rule is incorporating Single Sign-On mechanisms. In our Science Gateway it is fully supported, and based on Vine Toolkit – the user has to authenticate himself/herself only once, during the login procedure; then, all the credentials needed for accessing external services are automatically used by the portal on the user's account. It means, that even if the user needs to access a remote host with the Portal SSH Client (the researchers strongly indicated such a need in a survey conducted by the PL-Grid project), the process of authentication is performed entirely by the portal – the only thing the user has to do, is to choose appropriate resource. The same applies to the Science Gateway File Manager application



Portal File System

/

Name	Modified	Size	Kind
myfolder	Mar 10, 2010 02:35:16 PM	4096 bytes	Folder
new folder	Sep 15, 2010 04:17:08 PM	4096 bytes	Folder
tescik	Jul 08, 2011 01:53:16 PM	4096 bytes	Folder
.bash_profile	May 12, 2010 00:13:44 PM	193 bytes	File
20101209-121342.sbx	Dec 09, 2010 01:13:42 PM	2074 bytes	File
20101210-104226.sbx	Dec 10, 2010 11:42:26 AM	2200 bytes	File
20101210-123146.sbx	Dec 10, 2010 01:31:46 PM	2200 bytes	File
RECV.log	Sep 17, 2010 02:04:07 PM	3441720 bytes	File
SENT.log	Oct 26, 2010 02:02:41 PM	1032675 bytes	File
abinit_band_structure_in_file.txt	Oct 25, 2010 03:49:43 PM	16745 bytes	File
date191009.txt	Mar 11, 2010 03:01:16 PM	30 bytes	File
spacegroup.in	Jan 07, 2011 03:59:37 PM	139 bytes	File

Grass1 (Grid-FTP)

WCSS Darkmass SRM2
TASK SRM2
CyfronetSRM2 (Grid-FTP)
DarkmassSRM2 (Grid-FTP)
Grass1 (Grid-FTP)

1279202279567_workshop-abinit_951	4096 bytes	Jul 15, 2010 03:58:03 PM	Folder
1287670180763_workshop-abinit_239	4096 bytes	Oct 21, 2010 04:10:00 PM	Folder
1287670679414_workshop-abinit_613	4096 bytes	Oct 21, 2010 04:18:15 PM	Folder
1287670811104_workshop-abinit_626	4096 bytes	Oct 21, 2010 04:20:27 PM	Folder
1287671016246_workshop-abinit_644	4096 bytes	Oct 21, 2010 04:23:56 PM	Folder
1287671162196_workshop-abinit_004	4096 bytes	Oct 21, 2010 04:26:20 PM	Folder
1287671316769_workshop-abinit_978	4096 bytes	Oct 21, 2010 04:28:54 PM	Folder
1287739091028_workshop-abinit_448	4096 bytes	Oct 22, 2010 11:18:16 AM	Folder
129413206		11 10:08:33 AM	Folder
1294132068562_workshop-abinit_627	4096 bytes	Jan 04, 2011 10:08:12 AM	Folder
1294132068562_workshop-abinit_627	4096 bytes	Jan 04, 2011 10:08:15 AM	Folder
1294132068562_workshop-abinit_627	4096 bytes	Jan 04, 2011 10:08:20 AM	Folder
1294132068562_workshop-abinit_627	4096 bytes	Jan 04, 2011 10:08:46 AM	Folder
1294132068562_workshop-abinit_627	4096 bytes	Jan 04, 2011 10:08:22 AM	Folder
nano2010	4096 bytes	Jul 08, 2010 00:26:00 PM	Folder
new	4096 bytes	Jul 17, 2011 03:24:32 PM	Folder
output	4096 bytes	Apr 18, 2011 02:16:40 PM	Folder
.Xauthority	132 bytes	Dec 20, 2010 11:27:22 AM	File
.bash_history	2224 bytes	May 06, 2011 03:50:29 PM	File

1287671316769_workshop-abinit_9785_abinit_Port1_1287671317974_643

Fig. 5. Vine Toolkit File Manager with Single Sign-On

(Fig. 5). For ease-of-use, it consists of two panels (each for one storage resource). The user can make all standard file operations (e.g. copy files between panels with drag and drop interface, delete, create folders, change names, download or upload files) without the need to authenticate again.

5 Future Work

The future work on the Vine Toolkit-based Science Gateways are divided into two parts. The first are various new extensions to the existing Science Gateways, including support for other applications and packages, like NAMD and MOPAC. The second, based on the end-users needs collected within the PL-Grid project, is adding several new applications to the Science Gateways, e.g. NWChem or BLAST/CLUSTALW2. Therefore, our Science Gateways will be able to support all major applications used within large-scale parallel simulations in other science domains. In this way, rendering scientists able to deal with advanced calculations using web spaces that are easy to use and share.

References

1. Russell, M., Dziubecki, P., Grabowski, P., Krysiński, M., Kuczyński, T., Szjenfeld, D., Tarnawczyk, D., Wolniewicz, G., Nabrzyski, J.: The Vine Toolkit: A Java Framework for Developing Grid Applications. In: Wyrzykowski, R., Dongarra, J., Karczewski, K., Wasniewski, J. (eds.) PPAM 2007. LNCS, vol. 4967, pp. 331–340. Springer, Heidelberg (2008)

2. Szejnfeld, D., Dziubecki, P., Kopta, P., Krysinski, M., Kuczynski, T., Kurowski, K., Ludwiczak, B., Piontek, T., Tarnawczyk, D., Wolniewicz, M., Domagalski, P., Nabrzyski, J., Witkowski, K.: Vine Toolkit – Towards Portal Based Production Solutions for Scientific and Engineering Communities with Grid-Enabled Resources Support. Scalable Computing: Practice and Experience 11(2), 161–172 (2011)
3. Dziubecki, P., Grabowski, P., Krysinski, M., Kuczynski, T., Kurowski, K., Szejnfeld, D.: Nano-Science Gateway development with Vine Toolkit and Adobe Flex. In: Barbera, R., Andronico, G., La Rocca, G. (eds.) Proceedings of the International Workshop on Science Gateways (IWSG 2010). Consorzio COMETA, Catania (2010)
4. Agullo, E., Coti, C., Herault, T., Langou, J., Peyronnet, S., Rezmerita, A., Cappello, F., Dongarra, J.: QCG-OMPI: MPI applications on grids. Future Generation Computer Systems 27(4), 357–369 (2011) ISSN 0167-739X, doi:10.1016/j.future.2010.11.015
5. Kurowski, K., de Back, W., Dubitzky, W., Gulyás, L., Kampis, G., Mamonski, M., Szemes, G., Swain, M.: Complex System Simulations with QosCosGrid. In: Allen, G., Nabrzyski, J., Seidel, E., van Albada, G.D., Dongarra, J., Sloot, P.M.A. (eds.) ICCS 2009, Part I. LNCS, vol. 5544, pp. 387–396. Springer, Heidelberg (2009), doi:10.1007/978-3-642-01970-8_38
6. Streit, A., Bala, P., Beck-Ratzka, A., Benedyczak, K., Bergmann, S., Breu, R., Daivandy, J. M., Demuth, B., Eifer, A., Giesler, A., Hagemeier, B., Holl, S., Huber, V., Lamla, N., Mallmann, D., Memon, A. S., Memon, M. S., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Schlauch, T., Schreiber, A., Soddemann, T., Ziegler, W.: UNICORE 6 – Recent and Future Advancements. JUEL-4319 (2010) ISSN 0944-2952
7. gLite 3, `http://glite.cern.ch/introduction`
8. GRIA 5.3, `http://www.gria.org/about-gria/overview`
9. Foster, I.: Globus Toolkit Version 4: Software for Service-Oriented Systems. In: Jin, H., Reed, D., Jiang, W. (eds.) NPC 2005. LNCS, vol. 3779, pp. 2–13. Springer, Heidelberg (2005)
10. Moor, R.: Towards a Theory of Digital Preservation. The International Journal of Digital Curation 3(1) (2008)
11. Rajasekar, A., Wan, M., Moore, R., Schroeder, W., Kremenek, G., Jagatheesan, A., Cowart, C., Zhu, B., Chen, S., Olschanowsky, R.: Storage Resource Broker – Managing Distributed Data in a Grid. Computer Society of India Journal, Special Issue on SAN 33(4), 42–54 (2003)
12. Antonioletti, M., Atkinson, M.P., Baxter, R., Borley, A., Chue Hong, N.P., Collins, B., Hardman, N., Hume, A., Knox, A., Jackson, M., Krause, A., Laws, S., Magowan, J., Paton, N.W., Pearson, D., Sugden, T., Watson, P., Westhead, M.: The Design and Implementation of Grid Database Services in OGSA-DAI. Concurrency and Computation: Practice and Experience 17(2-4), 357–376 (2005)
13. Foster, I., Grimshaw, A., Lane, P., Lee, W., Morgan, M., Newhouse, S., Pickles, S., Pulsipher, D., Smith, C., Theimer, M.: Open Grid Services Architecture Basic Execution Service Version 1.0 GFD-R.108 (2008), `http://www.ggf.org/documents/GFD.108.pdf`
14. Anjomshoaa, A., Brisard, F., Drescher, M., Fellows, D., Ly, A., McGough, S., Pulsipher, D., Savva, A.: Job Submission Description Language (JSDL) Specification, Version 1.0. GFD-R.056 (2005), `http://www.ggf.org/documents/GFD.56.pdf`
15. OGSA Resource Usage Service, `http://forge.ggf.org/sf/projects/rus-wg`

16. Microsoft's Active Directory, http://www.microsoft.com/windowsserver2008/en/us/active-directory.aspx
17. Oleksiak, A., Tullo, A., Graham, P., Kuczynski, T., Nabrzyski, J., Szejnfeld, D., Sloan, T.: HPC-Europa single point of access as a framework for building science gateways: Research Articles. Concurrency and Computation: Practice & Experience 19(6), 851–866 (2007)
18. P-GRADE, http://www.p-grade.hu/
19. EnginFrame, http://www.nice-software.com/web/nice/products/enginframe
20. SAGA, http://saga.cct.lsu.edu/
21. Klimeck, G., McLennan, M., Brophy, S.P., Adams III, G.B., Lundstrom, M.S.: nanoHUB.org: Advancing Education and Research in Nanotechnology. Computing in Science and Engineering 10(5), 17–23 (2008), doi:10.1109/MCSE.2008.120
22. myExperiment, http://www.myexperiment.org/
23. Guim, F., Rodero, I., Corbalan, J., Labarta, J., Oleksiak, A., Kuczynski, T., Szejnfeld, D., Nabrzyski, J.: Uniform job monitoring in the HPC-Europa project: data model, API and services. International Journal of Web and Grid Services 3(3), 333–353 (2007)
24. Gonze, X., Amadon, B., Anglade, P.-M., Beuken, J.-M., Bottin, F., Boulanger, P., Bruneval, F., Caliste, D., Caracas, R., Cote, M., Deutsch, T., Genovese, L., Ghosez, P., Giantomassi, M., Goedecker, S., Hamann, D.R., Hermet, P., Jollet, F., Jomard, G., Leroux, S., Mancini, M., Mazevet, S., Oliveira, M.J.T., Onida, G., Pouillon, Y., Rangel, T., Rignanese, G.-M., Sangalli, D., Shaltaf, R., Torrent, M., Verstraete, M.J., Zerah, G., Zwanziger, J.W.: ABINIT: First-principles approach of materials and nanosystem properties. Computer Phys. Commun. 180, 2582–2615 (2009)
25. Gonze, X., Rignanese, G.-M., Verstraete, M., Beuken, J.-M., Pouillon, Y., Caracas, R., Jollet, F., Torrent, M., Zerah, G., Mikami, M., Ghosez, P., Veithen, M., Raty, J.-Y., Olevano, V., Bruneval, F., Reining, L., Godby, R., Onida, G., Hamann, D.R., Allan, D.C.: A brief introduction to the ABINIT software package. Zeit. Kristallogr. 220, 558–562 (2005)
26. Adobe, Flex/BlazeDs, http://www.adobe.com/products/flex/
27. Kokalj, A.: Computer graphics and graphical user interfaces as tools in simulations of matter at the atomic scale. Comp. Mat. Sci. 28, 155–168 (2003)
28. GoVASP, http://www.govasp.com/
29. Karanastasis, E., Varvarigou, T., Grabowski, P.: Portals for Service Oriented Infrastructures. In: Dimitrakos, T., Martrat, J., Wesner, S. (eds.) Service Oriented Infrastructures and Cloud Service Platforms for the Enterprise, pp. 159–177. Springer, Heidelberg (2010) ISBN 978-3-642-04085-6
30. Vine-users mailing list, http://gforge.man.poznan.pl/mailman/listinfo/vine-users
31. Kocot, J., Szepieniec, T., Noga, K., Harężlak, D., Sterzel, M.: InSilicoLab – Managing Complexity of Chemistry Computations. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 265–275. Springer, Heidelberg (2012)
32. Ciepiela, E., Nowakowski, P., Kocot, J., Harężlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing Entire Lifecycles of e-Science Applications in the GridSpace2 Virtual Laboratory – from Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)

Scripting Language Extensions Offered by the GridSpace Experiment Platform

Daniel Harężlak[1], Marek Kasztelnik[1], Eryk Ciepiela[1], and Marian Bubak[1,2]

[1] AGH University of Science and Technology, ACC Cyfronet AGH,
ul. Nawojki 11, 30-950 Kraków, Poland
{d.harezlak,m.kasztelnik,e.ciepiela}@cyfronet.pl

[2] AGH University of Science and Technology, Faculty of Electrical Engineering,
Automatics, Computer Science and Electronics, Department of Computer Science,
al. Mickiewicza 30, 30-059 Kraków, Poland
bubak@agh.edu.pl

Abstract. Many existing problem solving environments provide scientists with convenient methods for building scientific applications over distributed computational and storage resources. In many cases a basic set of features of such environments is sufficient to conduct a complete experiment flow. However, complex cases often require extensions supporting an external piece of software or a communication standard not integrated beforehand. Most environments deal with such cases by providing an extension facility and letting third parties add required features. The GridSpace environment also includes several mechanisms for extending its own functionality and here we describe how this can be accomplished. We focus on extensions already implemented such as local job submission and scripting language repositories, as well as on a GUI extension point which can be used to add custom graphical user interfaces to GridSpace experiments independently of their release process.

Keywords: virtual laboratories, computational science, scientific computing, extensions, rich Internet applications.

1 Introduction

GridSpace [1] is a web-based script integration platform which allows researchers to contribute their code to a self-contained experiment or a set of experiments. It was developed within the PL-Grid project [7]. For experiment creation, execution and management a graphical user interface implemented as a web application is provided – the GridSpace Experiment Workbench (EW for short). GridSpace experiments can be shared within collaborative research communities. The scripting nature of experiments gives authors great flexibility in applying the run-observe-modify-rerun cycle. In order to better exploit this exploratory programming model, GridSpace integrates several mechanisms that decrease the amount of boilerplate code and enable users to focus on relevant research issues. These mechanism are called GridSpace extensions and are the main subject of the following sections. The first extension, called Gem Manager, takes care of managing various libraries available for different scripting languages (e.g. Ruby gems or Python eggs). Apart from adding required libraries to the user's execution environment the Gem Manager attaches dependency information to experiments so as to prepare similar conditions for future runs (even if requested by different users). Scientific research often requires high-performance computing resources provided by clusters, access to which is possible through a PBS [11] implementation. GridSpace provides an extension that offers access to this API from within a scripting language of the user's choice. Addressing the common requirement of being able to present intermediate experiment results, GridSpace provides a WebGUI extension which allows authors to easily integrate RIA (Rich Internet Applications) with their experiments. For convenience, one RIA implementation is available through GridSpace itself and can be used for simple use cases. In the following sections we explain how these extensions fit in the general GridSpace architecture and discuss their internal workings.

2 Related Work

Open-source scripting languages such as Ruby or Python have extensive user communities, delivering many extensions to existing platforms in various domains. This delivery process has been formalized through Gems [9] and Eggs [10] respectively, making the process of extending local language interpreters easy for users. Still, the process bases on console commands which, from the perspective of a graphical user interface, is somewhat inconvenient. To make the process of managing language extensions even simpler, a graphical widget embedded in the Experiment Workbench interface needs to be provided as one of the extensions to the GridSpace platform.

Access to computational resources can be obtained by using one of many available batch systems (e.g. Oracle Grid Engine [8], Condor [4], TORQUE [13], etc.) Being able to access such platforms in the context of a scientific experiment development environment such as GridSpace is critical. Thus, the PBS extension was introduced. In the area of job submission attempts have been made to expose various client libraries and as a result the DRMAA [3] specification was produced. However, during the implementation phase no reliable release of a compliant library was available.

Web-based rich graphical user interfaces are becoming a standard. There are many tools offering out-of-the-box widgets for building fancy-looking application views, among them GWT [14], Adobe Flex [15], JavaFX [16] and Echo3 [17]. To make the GridSpace platform independent of such technologies the WebGUI protocol was introduced, making the ability to operate on REST-like [6] services the only requirement of integration. This approach enables experiment developers to use any web framework to implement their presentation layer.

One example of providing an extension-aware framework is Taverna [2]. It exposes an SPI (Service Provider Interface) with a set of Java interfaces which can be extended in order to add extra functionality to the Taverna framework.

In GridSpace the extension process assumes a more loose approach and allows for any modifications from within the given script interpreter. Because GridSpace supports many scripting languages it is the responsibility of the extension provider to ensure different ports of the extension for different languages.

Another example of a plugin-ready platform used for providing an abstraction layer over Grid jobs is GridBeans [5]. By extending existing Java classes developers may build custom models and user interfaces for Grid computations. Implementing portal-based GUIs is also possible in the same manner.

3 Description of the Solution

In Fig. 1 a general overview of GridSpace extensions is presented. It depicts a layered architecture with underlying resources in the bottom layer, a central containing the Execution Host where user experiments are executed (also called the Experiment Host) and the topmost presentation layer which embeds extension views, also called widgets. Each extension makes a vertical cut in this diagram and is described in detail below.

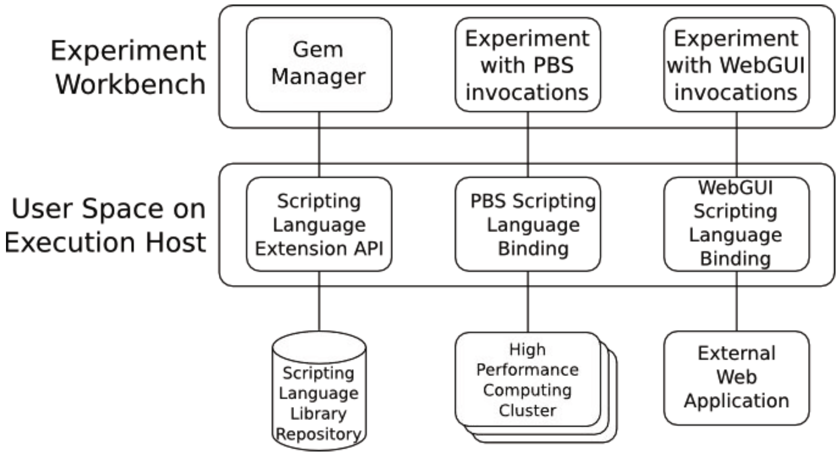

Experiment Workbench
Gem Manager
Experiment with PBS invocations
Experiment with WebGUI invocations
User Space on Execution Host
Scripting Language Extension API
PBS Scripting Language Binding
WebGUI Scripting Language Binding
Scripting Language Library Repository
High Performance Computing Cluster
External Web Application

Fig. 1. Architectural dependencies of GridSpace extensions

The Gem Manager uses external library repositories specific for a given scripting language. For Ruby and Python, gem and egg public repositories are used which gives GridSpace users access to hundreds of language extensions. In GridSpace the extension is implemented in such a way that integrating new repositories is a matter of implementing the interface presented in Fig. 2. Following the pattern established by the Gem Manager extension, each scripting language library is called a gem.

The interface contains two methods which allow users to search for installed gems (`searchLocal`) and for gems present in remote repositories (`search`). Using regular expressions should also be possible. Additional information about

```
public interface GemRepo {
  List<GemInfo> searchLocal(String searchPattern) throws SearchException;

  List<GemInfo> search(String searchPattern) throws SearchException;

  String getInfo(String gemName, String gemVersion)
                                             throws SearchException;

  String install(String gemName, String gemVersion,
      boolean installDocumentation) throws ManagementException;

  String uninstall(String gemName, String gemVersion)
      throws ManagementException;
}
```

Fig. 2. Main interface of the Gem Manager extension

particular gems can be retrieved using the `getInfo` method, by inputting the name and version of the requested gem (both of which can be obtained from the `GemInfo` bean). The `install` and `uninstall` methods are used to manage the lifecycle of gems. New gems are installed in the user space on the Experiment Host according to the conventions of a particular scripting language repository implementation. The main advantage of this approach is that the user does not have to apply for required extensions to site administrators, which usually takes a considerable amount of time. In the presentation layer Gem Manager exposes the described functionality as a widget, enabling users to manage their gems through a user-friendly point-and-click interface.

High Performance Computing clusters can be utilized to execute computationally-intensive parts of user experiments. The process of obtaining the necessary credentials to access such clusters is the user's responsibility. In the PL-Grid project this is very straightforward and requires users to fill out a simple web form. Once this step is complete, over 5000 CPUs and 2500 TB of storage are made available to users and can be interfaced by GridSpace through its PBS extension. As the main approach of the GridSpace platform is to enable users to develop software in a programming language of their choice, it becomes necessary to maintain a common code base for the PBS extension written in C, with several bindings to high-level languages. Following a careful study of user requirements it has become evident that the SWIG [12] wrapper will be necessary to support most use case scenarios. This is why our codebase is implemented in C and directly binds to PBS Torque [13] libraries. The API exposed by the extension (and made available to any supported language) is presented in Fig. 3.

The PBS API is relatively rich and allows for full control over job submission process. The first action required to handle job management is setting up the execution environment for the user process. For this purpose three methods are available. The `pbs_env_init` method is used to read Experiment Workbench configuration and set proper environmental variables according to policies on a given head node. The `pbs_init` and `pbs_add_env` methods can be applied

```
/* environment preparation */
void pbs_env_init();
void pbs_init(char* server, int debug_flag);
void pbs_add_env(struct pbs_job* job);

/* job preparation */
struct pbs_job* pbs_job_init(char* job_script);
void pbs_set_resources(struct pbs_job* job, char* resources);
void pbs_stagein(struct pbs_job* job, char* path);
void pbs_stageout(struct pbs_job* job, char* path);
char* pbs_get_job_name(struct pbs_job* job);

/* job life-cycle management */
char* pbs_job_submit(struct pbs_job* job, char* queue_name);
char* pbs_job_submit_async(struct pbs_job* job, char* queue_name);
char* pbs_get_job_id(struct pbs_job* job);
char* pbs_job_status(struct pbs_job* job);
char* pbs_get_output_path(struct pbs_job* job);
char* pbs_get_error_path(struct pbs_job* job);
void pbs_job_delete(struct pbs_job* job);
```

Fig. 3. PBS API exposed in scripting languages used by users to access cluster resources

to set up the environment manually and should be used by more experienced users. It is important to remember that invoking `pbs_env_init` is mandatory at the beginning of the job management process, as this method sets up several variables which are used by subsequent calls of the library's methods. Once the environment is set up, the actual jobs may be prepared by using the second set of methods presented in Fig. 3. The library is designed to use a C structure in order to represent a single job. This structure can be obtained by calling the `pbs_job_init` method which presets some basic values. Before the job is submitted its description can be refined by passing the job structure to other job preparation methods. In this way additional job requirements such as wall-clock time or number of nodes can be specified by using the `pbs_set_resources` method. Stage-in and stage-out files can be set by using `pbs_stagein` and `pbs_stageout` methods. The final preparation method (`pbs_get_job_name`) retrieves the job's name which is generated by the library during job initialization (done by a call to the `pbs_job_init` method).

The PBS library also exposes a set of methods which enable experiment authors to control the job lifecycle. Job submission can be performed in two modes. The synchronous mode (using the `pbs_job_submit` method) offers the advantage of blocking the user process until a job finishes without any additional job status checking (the method returns the job identifier). This is mostly useful for experiments which involve single job runs or when a set of jobs needs to be executed sequentially. The asynchronous mode (the `pbs_job_submit_async` method) requires more advanced lifecycle handling by using the `pbs_job_status` method for job status monitoring. This is usually done in a loop which periodically checks the status of the job. In either mode, job identifiers can be retrieved by calling the `pbs_get_job_id` method while standard output and error file locations can be obtained by using `pbs_get_output_path` and `pbs_get_error_path` methods respectively. The `pbs_delete_job` method removes a job from the cluster.

The final GridSpace extension allows users to enrich experiments with external web applications offering a way to present results or request user input during the course of the experiment process. This mechanism can also be applied to steer experiment execution with a human in a computing loop. The name of the extension is WebGUI and, just like in the case of the PBS extension, a C codebase is maintained and ported to different scripting languages by using SWIG wrapping mechanisms. From the perspective of the experiment code WebGUI is a single call. In Ruby it assumes the following form:

```
result = webgui.process_query("https://external.web.app/url",\
  "{webguiDocType: ‘request’,\
  label: ‘User data’,\
  data: [{name: ‘gender’, label:
    ‘Gender’, pref: ‘radio’, options:[\
  {label: ‘Male’, value: ‘male’},
    {label: ‘Female’, value: ‘female’}],
    value: ‘female’},\
  {name: ‘age’, label: ‘Age’, pref: ‘text’},\
  {name: ‘cv’, label: ‘CV’, pref: ‘richTextArea’}]}")
```

The example above requests gender (choice presented with radio buttons), age (single-line text input) and CV (rich text editor) information from the user. The `process_query` method takes two parameters: a web application endpoint and a JSON (JavaScript Object Notation) query definition. In this example the query is valid for a reference implementation of the external web application which ships together with GridSpace. In more general cases, the query value can be an arbitrary string. Once the external web application is done processing the request it should return a value to the experiment process which then exposes the results through the `result` variable. Below we present a detailed description of the protocol used by the WebGUI library to exchange information with the custom web application.

A complete sequence diagram depicting integration of an external application with the WebGUI extension is presented in Fig. 4. The following entities are part of the communication process:

- **WebGUI Processor (Browser):** This component is part of the WebGUI core implementation. It is implemented as a JavaScript library and executed in the user’s browser when a WebGUI request is generated. It is also responsible for embedding the interface served by the external web application inside EW.
- **WebGUI Processor (EW Server):** Server-side implementation of the WebGUI extensions used to coordinate message exchange between the Script Interpreter and the External Web Application. It is part of the Experiment Workbench.

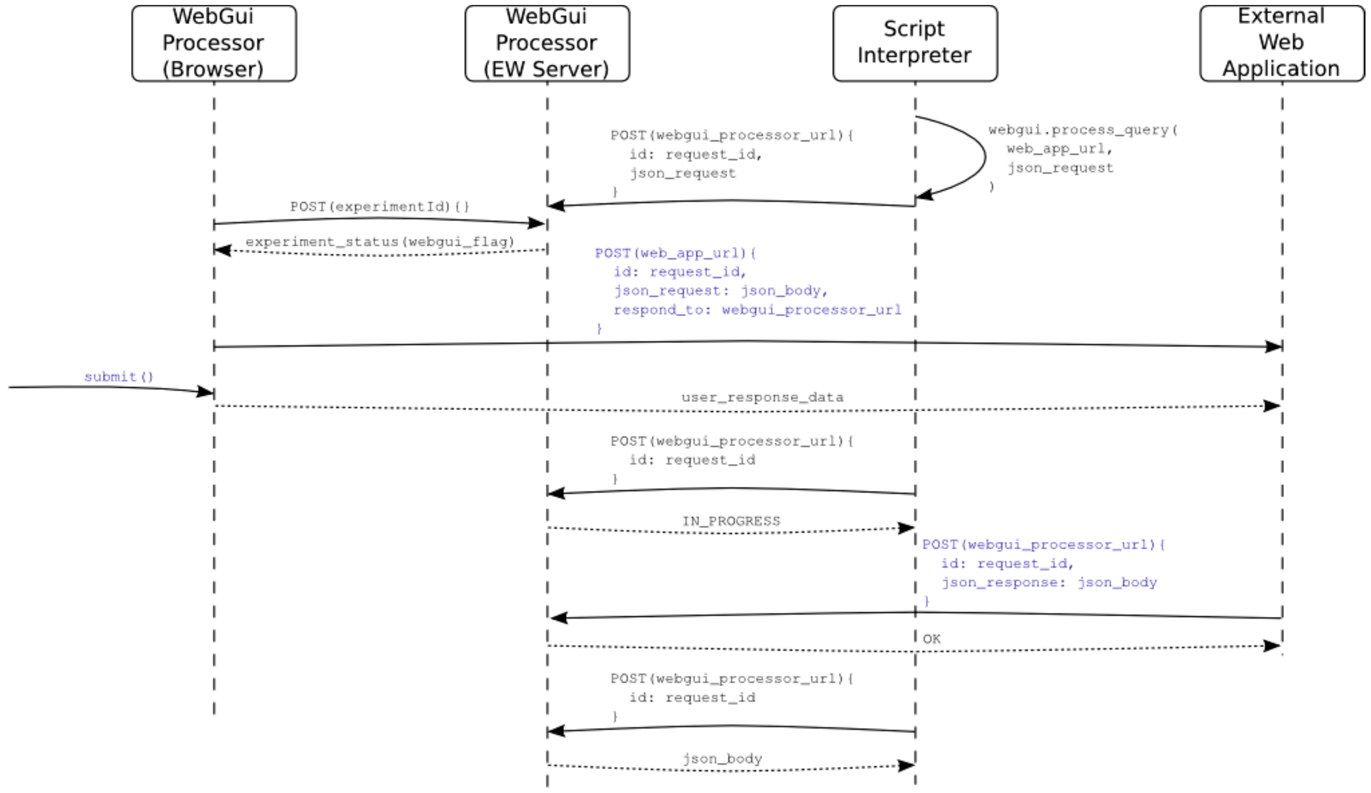

WebGui Processor (Browser)
WebGui Processor (EW Server)
Script Interpreter
External Web Application
webgui.process_query(web_app_url, json_request)
POST(webgui_processor_url){ id: request_id, json_request }
POST(experimentId){}
experiment_status(webgui_flag)
POST(web_app_url){ id: request_id, json_request: json_body, respond_to: webgui_processor_url }
submit()
user_response_data
POST(webgui_processor_url){ id: request_id }
IN_PROGRESS
POST(webgui_processor_url){ id: request_id, json_response: json_body }
OK
POST(webgui_processor_url){ id: request_id }
json_body

Fig. 4. WebGUI sequence diagram presenting the interaction schema between GridSpace and external web applications

– **Script Interpreter**: Represents the process which calls the WebGUI extension and expects user input in return. This can be any scripting language interpreter supported by EW and it is run on the experiment host machine.
– **External Web Application**: Implements the WebGUI protocol to add custom web views to EW experiments. It has to be publicly available in order for the communication to work.

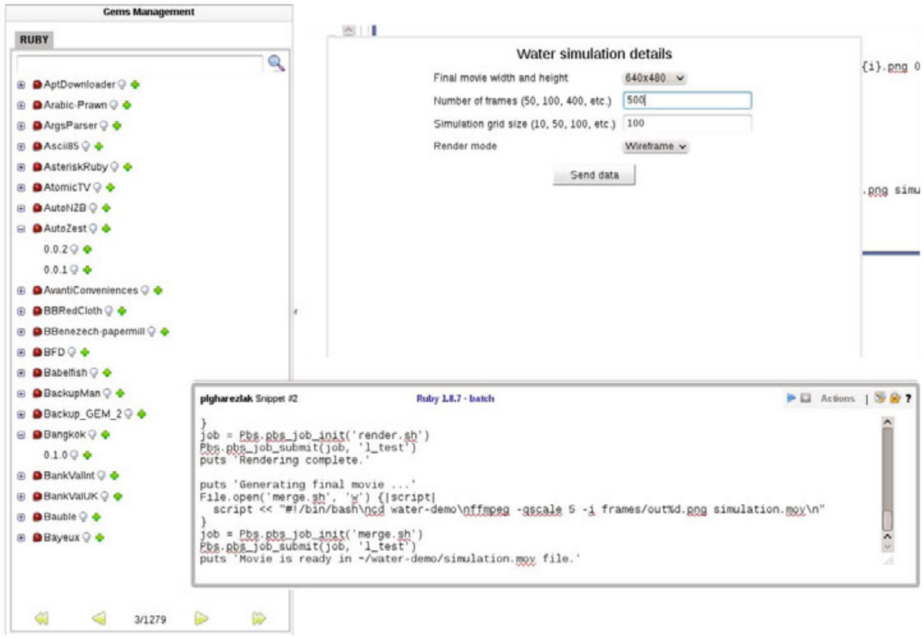
The WebGUI protocol sequence begins with a call to the `process_query` method which takes two parameters. The first is the location of the WebGUI processing endpoint of the external web application. The second is the body of the request which should be a JSON object. After the call, the Script Interpreter contacts the WebGUI Processor server through a POST request and registers a WebGUI request. Starting from this moment the call to the `process_query` blocks the interpreter's current thread and, by using an auxiliary thread, it periodically polls the WebGUI processor for a response (using a POST request). The whole conversation is assigned a unique id generated by the WebGUI library when the `process_query` method is initially called.

Once the WebGUI Processor server receives a render request it changes the experiment status accordingly. The next time this status is updated by the user's browser the WebGUI client code (JavaScript-based) dispatches a POST request to the external web application and renders the served view in a frame integrated with EW. The web application is assigned an identifier (`request_id`), a JSON request body (`json_request`) and a location (`respond_to`) which should be used to return the response. After the user submits data to the web application a response is generated and sent back to the WebGUI Processor server containing the conversation id and the response itself. In the meantime, the script interpreter is continuously asking for user response and receiving an `IN_PROGRESS` message. When the server records a response from the external web application it sends it back to the script interpreter according to the matching conversation id. This concludes the message exchange and the resulting JSON value is returned as the `process_query`'s method return value.

4 Results

The basic idea behind the Gem Manager is to overlay the process of extending a given scripting language with third-party libraries by a universal web interface which enables users to manage the process from within the GridSpace experiment editor. In this way the experiment author – by way of a few mouse clicks – can include the required library in the experiment and carry on with the research task. In Fig. 5 a sample view of the manager is placed on the left, with a list of Ruby gems. Each gem can be installed or removed with a single mouse click. Search capability is implemented through a single text input. For large search results a pagination mechanism is in place.

To enable PBS features in the scripting languages supported by GridSpace the SWIG [12] tool was used to produce corresponding bindings. The basic PBS



Water simulation details

Final movie width and height	640x480
Number of frames (50, 100, 400, etc.)	500
Simulation grid size (10, 50, 100, etc.)	100
Render mode	Wireframe

```
}
job = Pbs.pbs_job_init('render.sh')
Pbs.pbs_job_submit(job, 'l_test')
puts 'Rendering complete.'

puts 'Generating final movie ...'
File.open('merge.sh', 'w') {|script|
  script << "#!/bin/bash\ncd water-demo\nffmpeg -qscale 5 -i frames/out%d.png simulation.mov\n"
}
job = Pbs.pbs_job_init('merge.sh')
Pbs.pbs_job_submit(job, 'l_test')
puts 'Movie is ready in ~/water-demo/simulation.mov file.'
```

Fig. 5. Screenshots of GridSpace extensions as they are seen in the EW presentation layer

implementation was written in C and linked directly with the PBS API offered by TORQUE [13]. Currently, Python and Ruby ports are being tested. An example of using the PBS extension is presented in the code snippet situated in the bottom-right part of Fig. 5. The snippet contains a job preparation routine and submission is carried out in the synchronous mode, which will block the whole process until the job finishes. One advantage of this approach is that the job code can be generated on the basis of previous experiment results and requirements (e.g. dynamic values of input and output file paths).

The WebGUI mechanism can be used in more complex experiment implementations where intermediate results can determine the future experiment execution path, or where user input is required. Any web application available through an URL and conforming with a simple protocol offered by the extension can be used to enrich an experiment. In Fig. 5 the top-right part depicts a sample WebGUI window whose content was generated by an external web application. In this case it is a simple web form asking the user for water simulation input parameters.

5 Conclusions and Future Work

Future work will focus on improving the functionality of the presented extensions. The Gem Manager will be extended to support other script languages depending on the availability of a mechanism similar to Ruby gems. The experiment gem dependency feature is still experimental and requires further testing. In addition, the manager should provide support for recreating the complete user environment on different experiment hosts, including all gems installed on the original host.

The PBS API needs to be extended with some noncritical job submission options (which can be added according to user requests). Moreover, ports to other scripting languages are foreseen. As the DRMAA implementation for PBS has reached maturity, this standard interface might eventually substitute the native PBS API.

Regarding the WebGUI extension, the reference RIA implementation will be extended with file upload and graphics presentation support.

References

1. Ciepiela, E., Harężlak, D., Kocot, J., Bartyński, T., Kasztelnik, M., Nowakowski, P., Gubała, T., Malawski, M., Bubak, M.: Exploratory programming in the virtual laboratory. In: Proceedings of the International Multiconference on Computer Science and Information Technology, Wisla, Poland, pp. 621–628 (2010)
2. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M., Li, P., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34(Web Server issue), 729–732 (2006)
3. Tröger, P., Rajic, H., Haas, A., Domagalski, P.: Standardization of an APIA for Distributed Resource Management Systems. In: Proceedings of the Seventh IEEE International Symposium on Cluster Computing and the Grid (CCGrid 2007), Rio de Janeiro, Brazil, pp. 619–626 (May 2007)
4. Thain, D., Tannenbaum, T., Livny, M.: Distributed computing in practice: the Condor experience. Concurrency – Practice and Experience 17(2-4), 323–356 (2005)
5. Ratering, R., Lukichev, A., Riedel, M., Mallmann, D., Vanni, A., Cacciari, C., Lanzarini, S., Benedyczak, K., Borcz, M., Kluszczynski, R., Bala, P., Ohme, G.: GridBeans: Support e-Science and Grid Applications. In: Proceedings of the Second IEEE International Conference on e-Science and Grid Computing, p. 45. IEEE Computer Society (2006)
6. Fielding, R., Taylor, R.: Principled Design of the Modern Web Architecture. ACM Trans. Internet Technol. 2(2), 115–150 (2002)
7. Kitowski, J., Turała, M., Wiatr, K., Dutka, Ł.: PL-Grid: Foundations and Perspectives of National Computing Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 1–14. Springer, Heidelberg (2012)
8. Oracle Grid Engine, http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html
9. Community RubyGem Host, http://rubygems.org
10. Python Eggs, http://www.python-eggs.org
11. Portable Batch System: External Reference Specification, http://teal.gmu.edu/lucite/manuals/PBSPro5.0/pbs_ers.pdf
12. Simplified Wrapper and Interface Generator, http://www.swig.org

13. TORQUE Resource Manager,
http://www.clusterresources.com/products/torque-resource-manager.php
14. Google Web Toolkit, http://code.google.com/webtoolkit
15. Adobe Flex, http://www.adobe.com/pl/products/flex
16. JavaFX, http://javafx.com
17. Echo3 Framework, http://echo.nextapp.com/site/echo3

Managing Entire Lifecycles of e-Science Applications in the GridSpace2 Virtual Laboratory – From Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure

Eryk Ciepiela[1], Piotr Nowakowski[1], Joanna Kocot[1], Daniel Harężlak[1], Tomasz Gubała[1,3], Jan Meizner[1], Marek Kasztelnik[1], Tomasz Bartyński[1], Maciej Malawski[2], and Marian Bubak[2,3]

[1] AGH University of Science and Technology, ACC Cyfronet AGH, Kraków, Poland
e.ciepiela@cyfronet.pl

[2] AGH University of Science and Technology, Institute of Computer Science AGH, Kraków, Poland

[3] University of Amsterdam, Informatics Institute, The Netherlands

Abstract. The GridSpace2 environment, developed in the scope of the PL-Grid Polish National Grid Initiative, constitutes a comprehensive platform which supports e-science applications throughout their entire lifecycle. Application development may involve multiple phases, including writing, prototyping, testing and composing the application. Once the application attains maturity it becomes operable and capable of being executed, although it may still be subject to further development – including actions such as sharing with collaborating researchers or making results publicly available with the use of dedicated publishing interfaces. This paper describes each of these phases in detail, showing how the GridSpace2 platform can assist the developers and publishers of computational experiments.

Keywords: virtual laboratories, computational science, application development, collaborative research.

1 Introduction

The GridSpace2 environment, developed in the scope of the PL-Grid Polish National Grid Initiative [4], is a comprehensive platform which supports all phases of developing and sharing computational science experiments. The motivation behind the development of GridSpace2 was to assist computational scientists from various domains of e-science in migrating their experimental activities from home and office computers to the wider world of Grid and Cloud computing. Thus, GridSpace2 enables scientists to take advantage of large-scale distributed computing and storage architectures without forcing them to abandon their traditional experimentation habits.

The development of GridSpace2 was heavily influenced by discussions and consultations held with representatives of end-user communities (i.e. computational scientists) [1]. Prospective research domains included – among others – biochemistry, molecular dynamics, genomics and material studies. The resulting platform is based on the requirements gathered from user groups as well as on our own experience with developing custom solutions for e-science.

The structure of this chapter is as follows: section 2 presents the state of the art in problem-solving environments and distributed research systems, citing some related work. In section 3 we present the approach to supporting the computational research process at each of its stages. Section 4 details the outcome of our development and research effort while the final section contains conclusions and presents future work prospects.

2 Related Work

Over the years there have been many attempts at developing problem-solving environments and virtual laboratories [11]. Most such environments are constructed on top of workflow management systems [12]. While superficially attractive in that they do not require much in the way of computing skills or IT expertise, they also carry important drawbacks, such as the limited expressiveness of the programming model and lack of mechanisms for integration of computing resources. Moreover, they do not translate gracefully into the programming and development model employed by many computational scientists, where small applications and special-purpose pieces of code are engineered "in-house" to perform specific, one-off actions related to the research domain being studied.

The LEAD project [14] is an example of a Virtual Laboratory for weather applications. The LEAD environment features a portal with user interfaces (UI) and a set of dedicated, distributed grid resources (both computational and data-related) that are available through those UIs. The underlying workflow system combines resources to define task-specific processing. The system is provided for weather analysts.

The popular MyGrid [15] software package enables researchers (such as bioinformatics experts) to employ the Taverna system to compose complex experiment workflows from standalone building blocks. A library of those basic elements is available, supporting many different experiments. Processing schemes are executed by Taverna (the workflow engine), yielding results. The collaborative aspect of this solution is being investigated in the MyExperiment project [16].

Another interesting example of an experimentation space is the Kepler [17] system. Kepler provides a tool to construct application workflows (which may include computational experiments) out of local and remote components. Once prepared, workflows may be executed under a specific regime with the use of an underlying enactment engine.

On the publishing side, linking articles with primary data is pursued by the Utopia Documents project [23], which attempts to turn static data into live interactive content, such as curated database entries, molecular structures, sequence and alignment data and plots (however, it lacks on-demand execution capabilities). Research on new ways of utilizing Internet and social networks for scientific publications has also been pursued in EU-funded projects like SciX [19], PEER [20] and Liquidpub [21], gaining attention from collaborations of professionals such as the Concept Web Alliance [22]. An interesting example of how Web 2.0 technologies may be leveraged in collaborative scientific endeavors can be found in [13]. However, while these projects aim at addressing the problems of linking and referencing scientific data, publications, services and even executable workflows, they do not provide for a unified infrastructure which can link together the computing and data e-infrastructure, execution engines and the resulting executable paper, which is the ultimate objective of GridSpace2.

3 Description of the Solution

GridSpace2 [3] aims to support development of e-science applications, which we call virtual experiments. In order to present how GridSpace2 fulfills this goal, a set of basic concepts should be defined.

First of all, it should be stressed that instead of a drag-and-drop type of environment (such as those discussed in Section 2) GridSpace2 relies on the developers describing their experiments in terms of a set of code *snippets*, each written in a specific programming language. The notion of snippets is central to the concept of GridSpace2 and underpins all the functionality of the presented solution. We feel this to be a natural evolution of the way in which computational researchers go about their activities. This view is further strengthened by numerous consultations and joint design sessions held with representatives of various domains of e-science. A schematic view of the experimentation process, as supported by GridSpace2, is shown in Fig. 1.

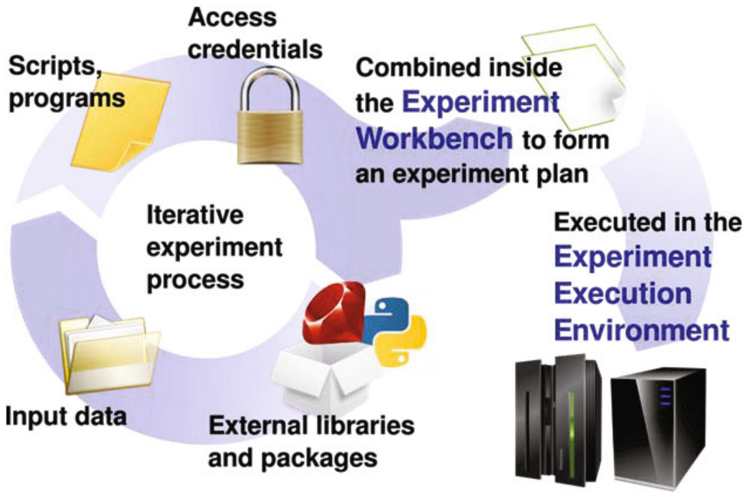

Access credentials
Scripts, programs
Combined inside the Experiment Workbench to form an experiment plan
Iterative experiment process
Executed in the Experiment Execution Environment
Input data
External libraries and packages

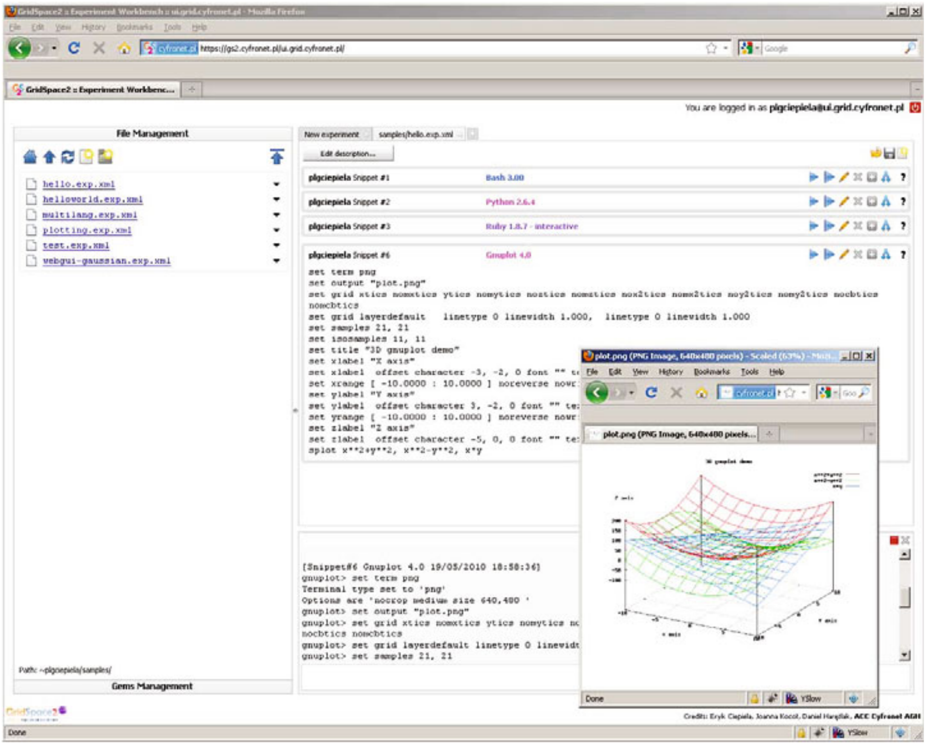
Fig. 1. The GridSpace2 Experimentation Cycle

In order to describe how an experiment is born, developed, shared and ultimately published and reused, we will describe the experimentation process on a phase-by-phase basis. Let us therefore start at the beginning.

3.1 Creating a Virtual Experiment

GridSpace2 provides an environment which supports the developer in creating and testing virtual experiments. This environment is called the Experiment Workbench (see Fig. 2). Protected by standard login mechanisms, the Workbench provides each participating developer with his/her own research space on a dedicated server called the *Experiment Host* (EH for short). The Experiment Host is a place where experiments can be executed in the context of an individual user's account. It is also a place where experiment input files can be stored and results preserved for later use. If necessary, the Experiment Host may delegate execution of computations to underlying HPC resources using mechanisms such as PBS queues.

Once the user logs in to the Workbench, it is possible to begin writing code snippets which constitute the virtual experiment. As already mentioned,



```
set term png
set output "plot.png"
set grid xtics nomxtics ytics nomytics noztics nomztics nox2tics nomx2tics noy2tics nomy2tics nocbtics
nomcbtics
set grid layerdefault   linetype 0 linewidth 1.000,  linetype 0 linewidth 1.000
set samples 21, 21
set isosamples 11, 11
set title "3D gnuplot demo"
set xlabel "X axis"
set xlabel  offset character -3, -2, 0 font "" te
set xrange [ -10.0000 : 10.0000 ] noreverse nowr
set ylabel "Y axis"
set ylabel  offset character 3, -2, 0 font "" te
set yrange [ -10.0000 : 10.0000 ] noreverse nowr
set zlabel "Z axis"
set zlabel  offset character -5, 0, 0 font "" te
splot x**2+y**2, x**2-y**2, x*y
```

```
[Snippet#6 Gnuplot 4.0 19/05/2010 18:58:36]
gnuplot> set term png
Terminal type set to 'png'
Options are 'nocrop medium size 640,480 '
gnuplot> set output "plot.png"
gnuplot> set grid xtics nomxtics ytics nomytics nc
nocbtics nomcbtics
gnuplot> set grid layerdefault linetype 0 linewidt
gnuplot> set samples 21, 21
```

Fig. 2. The GridSpace2 Experiment Workbench

GridSpace2 enables iterative development and execution of code, which can be either built from the ground up or imported from an external source (such as a custom IDE). While the Workbench presents a Web-based platform on which to develop arbitrary code, an experiment may also be prepared using external tools and wrapped with GridSpace2 mechanisms when ready, simply by copying and pasting code into the Workbench.

3.2 Developing Experiments

Experiments are developed by writing code and arranging it into snippets. Each of the snippets which together constitute a virtual experiment may utilize a different programming environment and use external tools and libraries. Any tool which exposes a command-line interface may be set up as a GridSpace2 interpreter – this includes most general-purpose scripting languages, such as Ruby, Python and Perl, but also more specialized tools, including math packages (Matlab, Mathematica), dedicated computational science libraries (Gaussian, PackMol, etc.) and purpose-specific notations (e.g. for drawing plots). The set of languages and notations supported is not constrained and new languages can be easily plugged in on a configuration level. We find that e-science applications mostly emerge in an exploratory and iterative manner; therefore we enable and promote coding in interactive write-run loops which can significantly improve the productivity of scientific programmers [2].

Of note is also the fact that GridSpace2 does not alter the syntax of programming languages used to develop experiments – thus, any properly written Ruby/Python/Perl program is automatically a valid GridSpace2 experiment which can be managed and further developed with the use of GridSpace2 tools.

As with any IDE, the Workbench enables the developer to execute the experiment in a debug mode (using EH resources) as well as store/retrieve existing experiments. Experiments can be executed in their entirety as well as on a per-snippet basis. An interesting feature of the Experiment Workbench is the ability to execute all snippets up to and including the one being currently edited – this facilitates exploratory development of experiments.

3.3 Sharing Experiments

When a save request is issued, an inbuilt serializer saves the entire contents of the experiment to an XML file in the user's home directory – such files are entirely self-contained and can be shared with collaborating developers. Sharing an experiment is as simple as sending the serialized file to an interested party, or accessing it in the developer's home directory (which can be facilitated by GridSpace2 with the use of standard file access mechanisms employed by the Experiment Host). Collaborative development is thus possible and each collaborator may have access to his/her personal copy of the experiment file. Furthermore, experiments can be duplicated for testing or branching purposes, and individual snippets can be extracted with the use of GridSpace2 tools.

GridSpace2 also provides the ability to expose an experiment as a website (this is the so-called run mode), without the need to access the Experiment Workbench. In the run mode, experiments resemble standalone web applications and can be enriched by additional content, including descriptions, comments, etc.

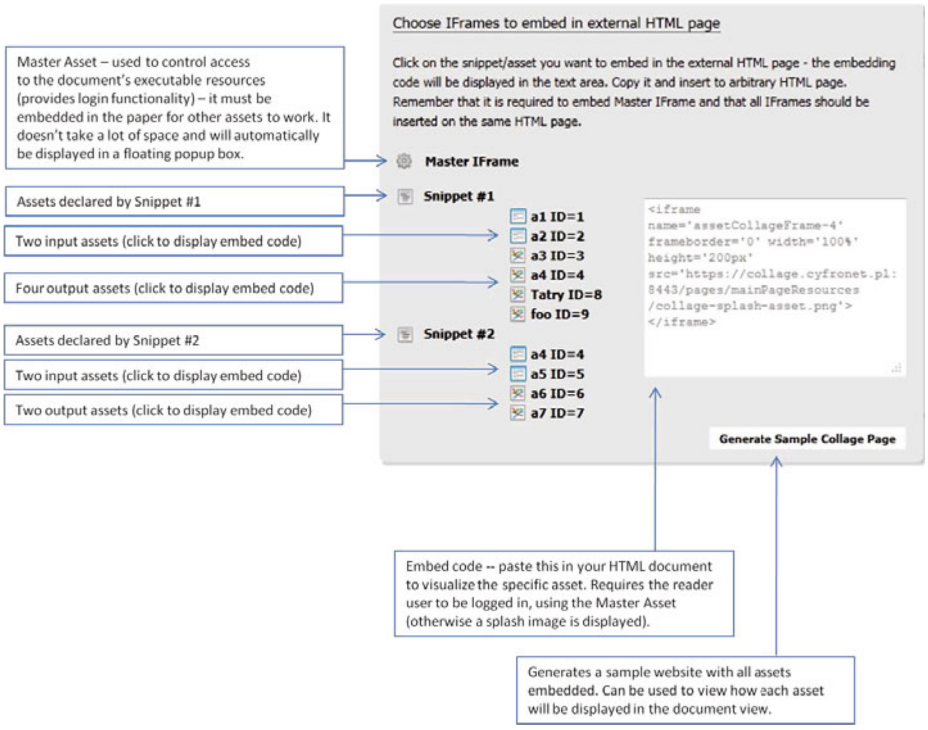
3.4 Publishing Experiment Results

GridSpace2 includes a novel mechanism for publishing experiment results, called Collage. By using Collage the results of computational experiments conducted with the help of GridSpace2 can be seamlessly embedded in external research publications, thus satisfying the concept of the so-called executable paper [13]. Collage works by introducing a mechanism called *Assets*, each of which presents a visual interface to the end user who browses a research publication (typically in the form of a diagram, graph, figure, etc.) and is capable of requesting the underlying infrastructure to repeat a given part of the experiment or redisplay its results depending on the parameters specified by the reader. Initially this interface assumes a predefined default state, as dictated by the experiment; however it may change following execution (at the reader's request). While the executable paper is being loaded each asset refers to a predetermined piece of data provided by the Collage server (which – if needed – can forward execution requests to underlying computing and storage resources). As results arrive, placeholders are replaced in the paper view with actual results of computations.

The authors of the executable paper are thus provided with two distinct user interfaces: a Web environment (the GridSpace2 Experiment Workbench) where they can code their experiments and determine the extent to which input data can be manipulated by the end user, as well as a separate interface which enables them to develop the actual executable paper as a Web document with embedded assets (this is called the Authoring UI). Since generating results on the fly might prove unfeasible, assets can also access local storage (flat files and databases) where results of previous runs are cached. A sample asset management window is shown in Fig. 3.

Three types of assets are available in the infrastructure. All assets belonging to a given type share a common rendering widget by which they are represented in the executable paper (which can be any environment capable of rendering webpages). They also share a set of common actions which can be performed on their content. The asset types foreseen in Collage are:

– **input assets** – the goal of this asset is to visualize an input form in the executable paper view. The form can be used by the user to feed input data into the running experiment. This type of asset directly implements the interactivity aspects of the executable paper as the users are able to browse large result spaces with the aid of input forms. Upon submission of an input form, the GridSpace2 server receives the input data and may further apply it in the course of processing the experiment, possibly generating further assets. An example of this functionality would be an interactive graph, which can be manipulated by the user through an input form. Each time the input form



Choose IFrames to embed in external HTML page

Click on the snippet/asset you want to embed in the external HTML page - the embedding code will be displayed in the text area. Copy it and insert to arbitrary HTML page. Remember that it is required to embed Master IFrame and that all IFrames should be inserted on the same HTML page.

Master Asset – used to control access to the document's executable resources (provides login functionality) – it must be embedded in the paper for other assets to work. It doesn't take a lot of space and will automatically be displayed in a floating popup box.

Embed code -- paste this in your HTML document to visualize the specific asset. Requires the reader user to be logged in, using the Master Asset (otherwise a splash image is displayed).

Generates a sample website with all assets embedded. Can be used to view how each asset will be displayed in the document view.

```
<iframe
name='assetCollageFrame-4'
frameborder='0' width='100%'
height='200px'
src='https://collage.cyfronet.pl:
8443/pages/mainPageResources
/collage-splash-asset.png'>
</iframe>
```

Master IFrame
Snippet #1
a1 ID=1
a2 ID=2
a3 ID=3
a4 ID=4
Tatry ID=8
foo ID=9
Snippet #2
a4 ID=4
a5 ID=5
a6 ID=6
a7 ID=7
Generate Sample Collage Page
Assets declared by Snippet #1
Two input assets (click to display embed code)
Four output assets (click to display embed code)
Assets declared by Snippet #2
Two input assets (click to display embed code)
Two output assets (click to display embed code)

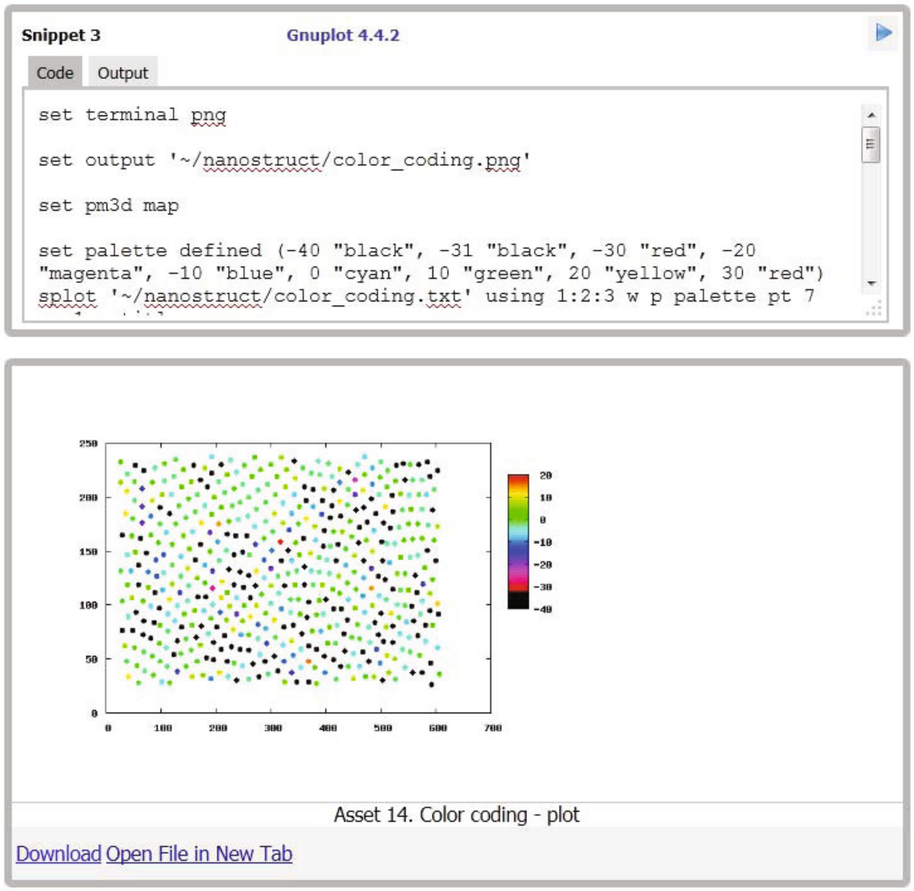
Fig. 3. Asset management window (with descriptions of available actions)

is filled out and submitted, the server reruns the required computations and generates a fresh graph (which is rendered as an output asset – see below). If computations are too complex to be performed in real time, the results may instead be read from data sources (databases and/or flat files) according to the input specified by the reader. Note that this type of asset can also be used to upload data files to the infrastructure for processing;

– **output assets** – the purpose of this type of asset is to render an experiment result which can be directly visualized in the research paper. Typically, this would be a figure, diagram or chart, although the environment itself does not impose restrictions on the nature of the visualization. In the case of inherently static visualizations (such as images) either the Publisher server or the client browser may issue periodical requests to the GridSpace2 server, to determine whether the payload of a given visualization has changed (which may often be the case – for instance as partial results are returned by the computing backend). Should this be the case, the contents of the visualization asset are automatically updated in the client browser. Furthermore, the GridSpace2 server can also detect that a given visualization is not yet available (e.g. if the underlying computations are still in progress) and notify the Publisher server (or client browser) so that an appropriate message may be displayed as the client awaits results;

– **code snippets** – these assets embed an editable view of the code which enacts a specific computation and may be used to generate additional assets. The purpose of this type of asset is to enable the reader to exercise more in-depth control over the experiment execution process, and also to review the inner workings of the executable paper for the purposes of validation and/or reuse. Subject to the author's control, the code of experiment snippets may be stored and the experiment reenacted.

A sample view of how GridSpace2 assets appear in the resulting publication can be seen in Fig. 4. Together, the three asset types mentioned above cover the full spectrum of interactivity as it pertains to executable papers. The authors can prepare and deploy assets (including executable code) while the viewers can interact with them by means of input forms. Results are displayed via visualization assets and can be refreshed by the server as more



```
set terminal png

set output '~/nanostruct/color_coding.png'

set pm3d map

set palette defined (-40 "black", -31 "black", -30 "red", -20
"magenta", -10 "blue", 0 "cyan", 10 "green", 20 "yellow", 30 "red")
splot '~/nanostruct/color_coding.txt' using 1:2:3 w p palette pt 7
```

Fig. 4. A Snippet Asset and a dependent Output Asset as visualized by the Publisher server in the body of an executable publication, developed with GridSpace2

output data becomes available. The system may also visualize external (non-Collage) assets referenced by static URLs.

3.5 Other Features of the Environment

GridSpace2 contains many features expected of a full-fledged IDE. While not specifically discussed here, they include tools for merging/managing snippets and file operations. There is also a dedicated wallet mechanism which stores sensitive data (such as passwords/security keys) in a secure manner, preventing them from being inadvertently exposed to other users of the framework (including collaborators).

In addition to the experiments themselves, the GridSpace2 environment also handles management of gems, which are dedicated interpreter libraries facilitating domain-specific operations on input data. Gems can be either local (invoked by importing libraries into the interpreter context) or remote (backended by external Web services). A more detailed description of GridSpace2, including end-user tutorials and sample experiments, can be found in [3].

4 Deployment

In order to ensure pervasive accessibility to the capabilities of GridSpace2 it is made available to the users through the single entry-point of a web portal, as depicted in Fig. 5. With this approach writing, running and managing applications take place in a consistent environment. Thus, a scientific experiment becomes as easily accessible as a common web application. By mapping resources (experiments and results they produce) to URLs we can share, reference or catalogue them like any other web resource. Experiments exposed through URLs can be displayed as standalone Web pages with rich content, manuals, blog-like comments and, last but not least, the ability to reenact computations with a single click. Mindful of confidentiality concerns and authorization requirements, we provide the ability to restrict access to experiments and their results to specific user communities. Behind the scenes, the GridSpace2 Web portal interfaces large-scale computational resources available to scientific applications. Depending on the scope of computations and required interactivity the application can be dispatched to a computing cluster, a single node or even to the user's own browser where it may operate as an applet.

Over time we are deploying the GridSpace2 environment in support of numerous domains of e-science, assisting researchers in developing virtual experiments and publishing their results. While a detailed list of GridSpace2 applications and experiments is out of scope of this chapter, it should be mentioned that we are actively involved in the following studies (where relevant the associated publications are also cited):

– Similarity-based detection of RNA subsections [25],
– Parsing computational chemistry results with cclib,

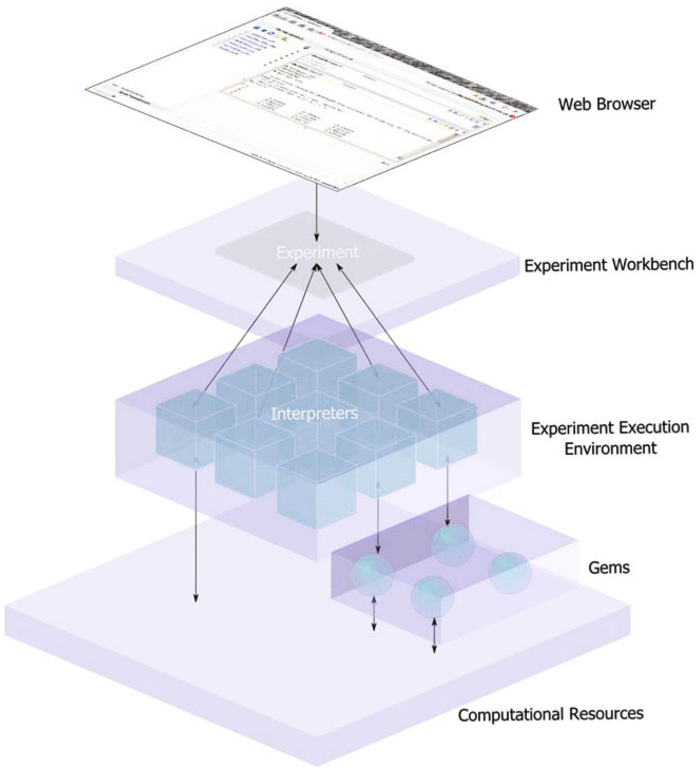


Fig. 5. The layered structure of GridSpace2, with end-user interfaces served by a Web browser and experiments executed on the underlying resources by the Experiment Execution Environment (Experiment Host)

– Analysis of compounds from the Protein Data Bank [24] with the CastP library,
– Applicability studies of the "Fuzzy Oil Drop" model to determining the secondary and tertiary structure of native proteins [26],
– Implementation of algorithms of quantitative analysis of the grain morphology in self-assembled hexagonal lattices [27].

It should be noted that the Collage framework, itself part of GridSpace2, won first prize in the Executable Paper Grand Challenge organized by Elsevier (an internationally renowned publishing house) in 2011. More details on this success can be found in [18].

5 Conclusions and Future Work

GridSpace2 aims to support e-scientists in their daily work by providing a comprehensive web-accessible solution where the entire research process can be carried out. The solution is deployed and operable on the PL-Grid e-infrastructure.

Future prospects for GridSpace2 include adjustments to the platform to make it suitable for various kinds of computations, e.g. multiscale applications within the Mapper project [5], and further integration with specific authoring environments for executable publications. Moreover, we are working on integrating GridSpace2 with a custom experiment provenance mechanism, tracking the development history of each experiment.

References

1. List of scientific publications associated with GridSpace, http://dice.cyfronet.pl/publications/
2. Ciepiela, E., Harężlak, D., Kocot, J., Bartyński, T., Kasztelnik, M., Nowakowski, P., Gubała, T., Malawski, M., Bubak, M.: Exploratory Programming in the Virtual Laboratory. In: Proceedings of the International Multiconference on Computer Science and Information Technology, Wisla, Poland, pp. 621–628 (2010)
3. GridSpace2 technology homepage, http://dice.cyfronet.pl/gridspace
4. The PL-Grid project, http://www.plgrid.pl/en
5. The Mapper project, http://mapper-project.eu
6. Triana homepage, http://www.trianacode.org
7. Kepler project homepage, http://kepler-project.org
8. Taverna homepage, http://taverna.sourceforge.net
9. Gunarathne, T., Herath, C., Chinthaka, E., Marru, S.: Experience with Adapting a WS-BPEL Runtime for e-Science Workflows. In: Proceedings of the 5th Grid Computing Environments Workshop, pp. 1–10. ACM (2009)
10. The ViroLab Project, http://virolab.cyfronet.pl
11. Rycerz, K., Bubak, M., Sloot, P.M.A., Getov, V.: Problem-Solving Environment for Distributed Interactive Simulations. In: Gorlatch, S., Bubak, M., Priol, T. (eds.) Achievements in European Research on Grid Systems, CoreGRID Integration Workshop 2006 (Selected Papers), pp. 55–66. Springer, Heidelberg (2008) ISBN-13: 978-0-387-72811-7
12. Gil, Y., Deelman, E., Ellisman, M., Fahringer, T., Fox, G., Gannon, D., Goble, C., Livny, M., Moreau, L., Myers, J.: Examining the Challenges of Scientific Workflows. IEEE Computer 40(12), 24–32 (2007)
13. Allen, G., Löffler, F., Radke, T., Schnetter, E., Seidel, E.: Integrating Web 2.0 Technologies with Scientific Simulation Codes for Real-time Collaboration. Cluster, 1–10 (2009)
14. Droegemeier, K.K., et al.: Service-oriented Environments in Research and Education for Dynamically Interacting with Mesoscale Weather. IEEE Computing in Science and Engineering (2005)
15. Stevens, R., et al.: Exploring Williams-Beuren Syndrome using MyGrid. Bioinformatics 1(20), 303–310 (2004)
16. The myExperiment, http://myexperiment.org
17. Altintas, I., Jaeger, E., Lin, K., Ludaescher, B., Memon, A.: A Web Service Composition and Deployment Framework for Scientific Workflows. In: ICWS, p. 814 (2004)
18. The Elsevier Executable Paper Grand Challenge, http://www.executablepapers.com
19. The SCiX project, http://www.scix.net
20. The PEER project, http://www.peerproject.eu

21. The LiquidPub project, http://liquidpub.org
22. The Concept Web Alliance homepage, http://conceptweblog.wordpress.com/declaration
23. Attwood, T.K., Kell, D.B., McDermott, P., Marsh, J., Pettifer, S.R., Thorne, D.: Utopia Documents: Linking Scholarly Literature with Research Data. In: Proceedings of 9th European Conference on Computational Biology, Ghent, Belgium (September 2010)
24. The Worldwide Protein Data Bank homepage, http://www.wwpdb.org
25. Gubała, T., Prymula, K., Nowakowski, P., Bubak, M.: Semantic Integration for Model-based Life Science Applications. Paper submitted to IEEE Internet Computing; evaluation pending
26. Jadczyk, T., Bubak, M., Roterman, I.: Is the "Fuzzy Oil Drop" Model of General Character? Poster presentation at KUKDM in Zakopane, Poland (2011)
27. Ciepiela, E., Zaraska, L., Sulka, G.D.: Implementation of Algorithms of Quantitative Analysis of the Grain Morphology in Self-Assembled Hexagonal Lattices According to Hillebrand Method, GridSpace2/Collage executable paper http://collage.cyfronet.pl/hillebrand-grains/

GridSpace2 Virtual Laboratory Case Study: Implementation of Algorithms for Quantitative Analysis of Grain Morphology in Self-assembled Hexagonal Lattices According to the Hillebrand Method

Eryk Ciepiela[1], Leszek Zaraska[2], and Grzegorz D. Sulka[2]

[1] AGH University of Science and Technology, ACC Cyfronet AGH,
ul. Nawojki 11, 30-950 Kraków, Poland
eryk.ciepiela@cyfronet.pl

[2] Jagiellonian University, Faculty of Chemistry,
Dept. of Physical Chemistry and Electrochemistry,
Ingardena 3, 30-060 Kraków, Poland
zaraska@chemia.uj.edu.pl

Abstract. This work presents the implementation of a method, originally proposed by Hillebrand et al. [1], of quantitative analysis of the grain morphology in self-assembled hexagonal lattices. This method can be effectively used for investigation of structural features as well as regular hexagonal arrangement of nanoporous alumina layers formed on the metal surface during the self-organized anodization process. The method has been implemented as a *virtual experiment* in the GridSpace2 Virtual Laboratory [15] which is a scientific computing platform developed in the scope of the PL-Grid [9] project. The experiment is a GridSpace2 pilot and therefore made available to the wider community of PL-Grid users. It is both editable and executable through a web portal offered by the GridSpace2 Experiment Workbench [17], dedicated to PL-Grid users. Moreover, since all GridSpace2 experiments are embeddable on arbitrary web sites owing to the *Collage* [16] feature, the final version of the experiment has been published as an *executable publication* [18] with execution rights granted to all PL-Grid users.

Keywords: virtual laboratories, computational science, scientific computing, nanochemistry, self-assembled hexagonal lattices.

1 Introduction

Recently, nanomaterials have become the subject of great scientific interest due to their unique chemical and physical properties, making them useful for a variety of promising applications. Therefore, considerable attention is focused on developing new, simple, and non-expensive methods of nanofabrication. Among them, simple anodization – *anodic oxidation* – has been successively employed for fabrication of nanoporous oxide layers on the surface of different metals, such as aluminum, titanium, zirconium, tungsten etc. Nanoporous anodic alumina (AAO) layers can be easily obtained by self-organized anodizing of aluminum foils in acidic electrolytes [2]. Under certain conditions (anodizing potential, type and concentration of electrolyte, temperature) the process results in formation of hexagonal cells with uniform nanopores at their centres. Following suitable preparation, the nanoporous oxide layers can be employed as templates for fabrication of other nanomaterials, e.g. nanowires [3]. An example of a nanoporous alumina membrane, together with *Sn* nanowires obtained by electrodeposition of metal inside nanopores, is shown in Fig. 1 (A) and (B) respectively [4].

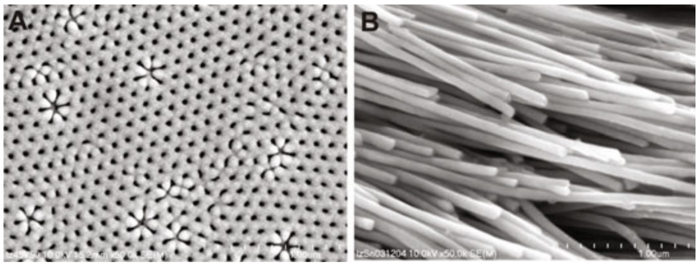


Fig. 1. FE-SEM images of AAO membrane (A) together with Sn nanowires (B) obtained by electrodeposition of metal inside the pores of the alumina template. The oxide membrane was chemically removed following electrodeposition.

It is noteworthy that all structural features of nanoporous alumina layers such as pore diameter, interpore distance and, especially, the degree of pore order depend strongly on anodizing conditions [2]. Bearing this in mind, a detailed optimization of all parameters of the procedure [5] is of great importance. What is more, from the nanotechnology point of view, avoiding defects in ordered nanostructures during their growth is of crucial significance. Therefore, there is a real need for an effective, versatile and easy-to-use software tool that enables rapid analysis of the degree of hexagonal pore order as well as structural features of nanoporous oxide layers.

This chapter focuses on the implementation of algorithms measuring the above mentioned characteristics.

2 Related Work

In most cases, the morphology of porous alumina films is investigated by field emission scanning electron microscope. Up until now, the characteristic parameters such as pore diameter and interpore distance were evaluated directly from FE-SEM images or by using WSxM [6], and ImageJ [19] software that often required several steps (e.g. Fast Fourier transform calculation) and time-consuming procedures.

Still, some brief and qualitative information regarding the regularity of pore arrangement can be derived from two-dimensional FFT images. For a perfect triangular lattice with a hexagonal pore arrangement, an FFT pattern consists of six distinct spots along the edges of a hexagon. When the order of the lattice is disturbed, a ring shape or even disc shape form appears in the first Bragg reflection of the 2D FFT power spectrum [4,7,8]. Examples of near-ideal hexagonal and non-ideal structures, together with corresponding FFT images, are shown in 2 (A) and (B) respectively.

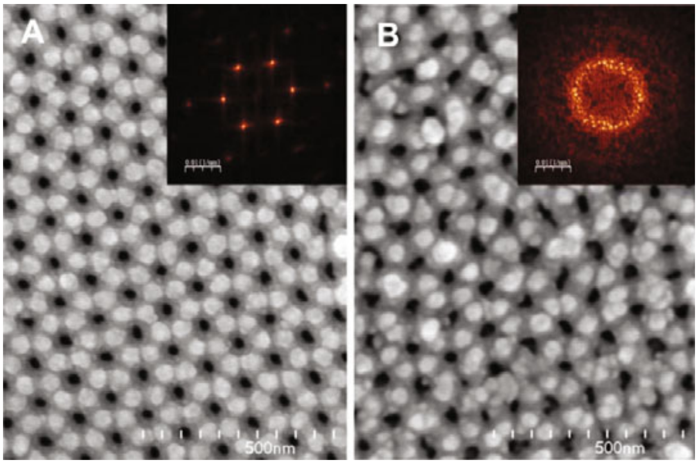


Fig. 2. FE-SEM images of anodic alumina layers with ideal (A) and non-ideal (B) hexagonal arrangement together with corresponding 2D FFT images

In order to acquire deeper insight into the regularity of the pore arrangement, a regularity ratio can be calculated on the basis of the intensity profiles of FFT images. Unfortunately, the FFT pattern strongly depends on the SEM image magnification and even on the quality of the picture. Hence, the resulting values are only useful for qualitative comparison of the degree of pore order for two or more similar nanoporous layers [4,7,8].

The authors have concluded that the method proposed by Hillebrand et al. [1] can give some quantitative information about sample morphology, including the number and size of hexagonally arranged grains.

The method consists of several steps. Analysis is based on FE-SEM images and the centre of mass coordinates for each pore. The grain structure can be characterized in two ways. First, we can obtain a qualitative visual description of the overall grain arrangement by assigning colors to pores depending to the relative angular orientation of their basic pattern and then indicating any transition between grains as a change in the colors attributed to individual pores. This enables us to observe e.g. border pores, not incorporated into any grain.

In the second approach, a set of interconnected triangles is plotted for all neighboring pores. Such triangles are then incorporated or removed from grains based on deviations in side lengths and internal angles (compared with an equilateral triangle). In this case the transition between grains is associated with deviations from the ideal lattice. Individual grains can be easily identified by using a spreading algorithm with tolerance parameters. The authors propose to treat the size of the largest grain identified as an indicator of the overall grain structure quality. In addition, the distribution function of distances between pore pairs (PDF) and the angular distribution function (ADF) can be easily calculated, yielding some additional information about quality of the hexagonal structure.

The described method is not currently supported by any available implementation enabling investigations into the morphology of nanoporous layers in a fast and easy way in order to support chemists in their daily work. In terms of software accessibility and usability, existing solutions fall short of some of the researchers' expectations. First, they require users to perform a fair amount of manual, tedious and repetitive tasks using a graphical user interface (GUI). When hundreds of SEM images are to be processed, automation would prove highly beneficial for end users. Moreover, the community of chemists often has trouble accessing the latest software as existing chemical applications are rather monolithic, standalone and often written from scratch. The method discussed here was first published three years ago but still needs to be supplemented by a robust publicly-available implementation. What is more, the authors haven't found any reusable components based on standardized interfaces which could be applied in this scope.

Another problem is that the user community suffers from lack of appropriate software published in accordance with the software-as-a-service (SaaS) paradigm. Most investigators continue to develop and use proprietary software that doesn't contribute to the service-oriented architecture (SOA).

Last, but not least, accessibility remains an issue. Existing solutions are mostly platform-dependent desktop applications, requiring installation on end-user workstations and incurring significant costs related to maintenance, updating and other technical support actions. This also inhibits wide adoption of such solutions.

In order to mitigate the disadvantages listed above, the authors have formed a multidisciplinary research team, combining the skills of chemistry experts and scientific IT developers to jointly pursue elegant solutions which address all the aforementioned issues.

3 Description of the Solution

The proposed solution is based on the GridSpace2 Virtual Laboratory [15] and the underlying computing infrastructure offered by the PL-Grid [9] project, including: hardware infrastructure, software, intellectual property rights (such as licenses) and qualified user support teams.

Although algorithms described above do not require massive computational resources or access to high-performance grid fabric, offering convenient and transparent access to external computing infrastructures and software remains a challenge. What is more, workflow composition capabilities are still needed as they allow for more effective, automatic execution of entire processing cycles, starting with raw input data and ending with final results expected by scientists. (It should be noted that the GridSpace2 platform also supports resource-consuming computations harnessing the power of grids.) The interpreter suite available in GridSpace2 includes software which can run in parallel on computing clusters through PBS queues or the QosCosGrid [10] [14] grid meta-scheduler. This reduces the steepness of the learning curve associated with utilizing high-performance computing infrastructures, enabling scientists to focus on productive research.

GridSpace2 is based on the assumption that specific parts of the reasearch process are susceptible to automation with the proper software. However, the overarching research process (treated as a workflow) still needs a researcher in the loop. The idea behind the GridSpace2 Virtual Laboratory is to support execution of scientific workflow applications in an interactive and explorative [12] manner, on behalf of researchers who:

1. have insight into detailed workflow status, intermediary and partial results,
2. influence the course of the workflow through manual adjustments of code, parameters and input data, and by deciding which workflow steps need to be performed at runtime,
3. refine and re-enact the workflow in a loop until valid results are obtained.

Among the many features of GridSpace2 (most of which are arguably out of scope of this work [11]) some basic concepts worth mentioning are presented below.

First, all GridSpace2 capabilities are accessible through a single Web-based entry point – namely, the Experiment Workbench, hosted for PL-Grid users at [17]. This is where all PL-Grid users can create, execute and manage computer programs in the form of scientific workflows called *virtual experiments*. Experiments, along with results obtained through their execution, are straightforwardly yet securely accessible through the HTTP(s) protocol, which means that a web is the only piece of software required on the user's workstation. This approach matches the architecture and ecosystem of the World Wide Web, improves accessibility, and makes our solution easy to adopt.

Underneath the Experiment Workbench a number of remote hosts, called *Experiment Hosts*, are operated and managed. Together, they constitute a computing backend where experiment execution actually takes place. These hosts can represent standalone machines, head hosts of computing clusters or grid meta-schedulers. They can be made available by arbitrary providers, however, in our particular case we rely on the PL-Grid resource, or, to be more exact, on the *Zeus* cluster operated by the Academic Computer Centre Cyfronet AGH.

The *Collage Authoring Environment* [13] offers an alternative way of customizing and executing experiments that are made available to the public or to restricted user groups. This feature may appeal to scientific communities interested in providing mashed-up content-rich websites (e.g. blogs, wikis, content management systems, social networking services, etc.) with executable *Collage assets* served by the Experiment Workbench. Collage enables execution of experiments from arbitrary websites, using remote computing resources to process input data provided by the visitor to a web page. It doesn't incur any technical overhead for site administrators as embedding assets on a web page is as straightforward as embedding Google Maps widgets or YouTube videos.

With GridSpace2, developing virtual experiments is a simple task, as the experiment is defined in terms of a sequence of steps, each written in an arbitrary scripting language, domain-specific program or custom tool. This leverages the expressiveness and abstraction of programming while enabling programmers to take advantage of languages designed for particular tasks instead of having to cope with general-purpose languages. It also promotes harnessing the tools already provided by the computing infrastructure as well as automating respective tasks with the use of dedicated software suites – e.g. generating plots with GnuPlot [23] or performing mathematical calculations with Mathematica [21].

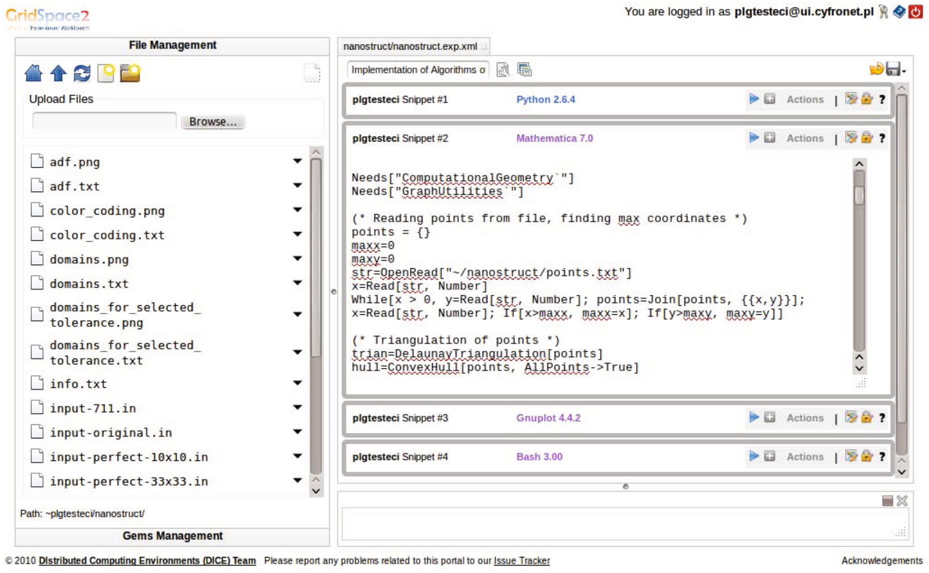
Having the above in mind, the authors have addressed the presented problem by creating a virtual experiment. The experiment was implemented by using the web interface provided by the GridSpace2 Experiment Workbench, with *Zeus* cluster picked as the Experiment Host.

As mentioned above, a GridSpace2 virtual experiment is defined as a sequence of steps, each written in an arbitrary scripting, domain-specific or purpose-specific language (which we call an *interpreter*). Accordingly, the first step to perform was to identify the individual tasks involved in the workflow that can be handled by existing languages and tools. This process yielded the following list of steps, each assigned to a specific interpreter chosen from among those installed on the *Zeus* cluster:

1. Analysis of the SEM lattice image: ImageJ [19].
2. Rejecting irrelevant data, extracting geometric properties of pores: Python [20] script.
3. Calculating the regularity measures of the pore structure: Mathematica [21] script.
4. Visualization of the measures in plots and diagrams: GnuPlot [23] script.
5. Choosing a subset of interesting results: manual task; the tolerance value serves as an input parameter for the subsequent script.
6. Extracting data for the subset chosen: Bash [22] script.
7. Thorough visualization of data for the subset, generating plots and diagrams: GnuPlot [23] script.

Each step of the above defined workflow maps to exactly one snippet in the virtual experiment. Individual snippets were implemented by using languages supported by each interpreter.

Coding was carried out in an exploratory way, by writing, executing, refining and reexecuting code with the use of the Experiment Workbench. Fig. 3 shows one of the snippets of the target experiment being edited in the Experiment Workbench.



```
Needs["ComputationalGeometry`"]
Needs["GraphUtilities`"]

(* Reading points from file, finding max coordinates *)
points = {}
maxx=0
maxy=0
str=OpenRead["~/nanostruct/points.txt"]
x=Read[str, Number]
While[x > 0, y=Read[str, Number]; points=Join[points, {{x,y}}];
x=Read[str, Number]; If[x>maxx, maxx=x]; If[y>maxy, maxy=y]]

(* Triangulation of points *)
trian=DelaunayTriangulation[points]
hull=ConvexHull[points, AllPoints->True]
```

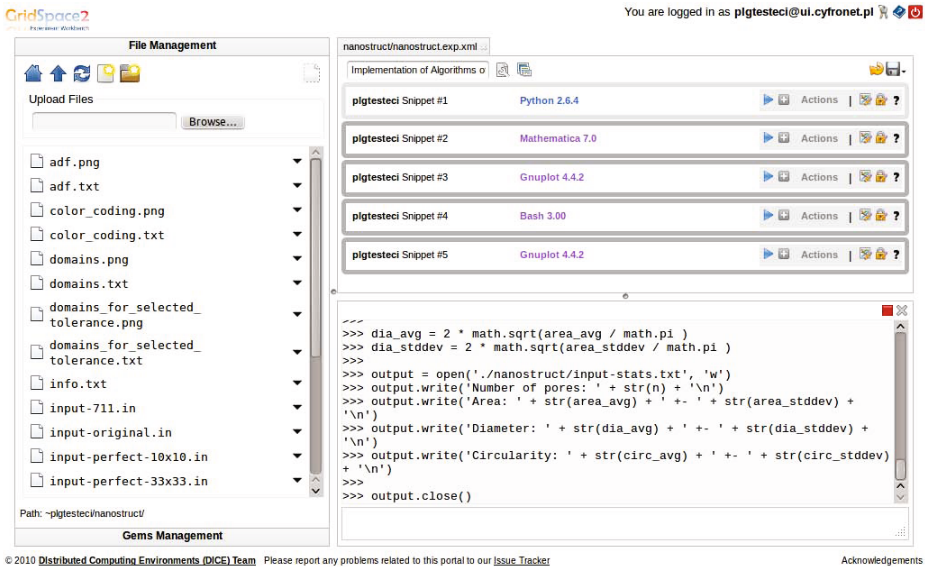
Fig. 3. Developing an experiment in Experiment Workbench: the left-hand panel hosts the File Management window, used for accessing and managing files on the Experiment Host, while the right-hand panel contains a selection of snippets, one of which has been opened for editing

4 Results

Following several iterations carried out in an exploratory way the final version of the experiment was produced. Fig. 4 shows the experiment being executed in the Experiment Workbench. The Workbench provides indication of which snippets have already run, which one is currently running and what output has been created. The resulting files can be viewed and managed in the File Management window.

The Experiment Workbench allows developers to export the experiment as a set of Collage *assets*, ready to paste in arbitrary websites. Taking advantage of this feature, the experiment was also published in the form of an executable publication [18]. For this purpose the WordPress [24] blog platform was installed with the target experiment posted as a blog entry. The header fragment of this entry is depicted in Fig. 5. Sample assets of the target executable paper are collected in Fig. 6, respectively: sample input – SEM image of the pores (A), file containing pore information (B), sample snippet – geometric calculations in Mathematica [21] (C), sample output – calculated pore statistics (D), sample output – color coding of pores (E), sample output – quantitative features of pore grains plotted (F).

The presented experiment will be extensively used for investigation of pore order and grain morphology in nanoporous oxide layers formed by anodization of aluminum and titanium – a subject of intensive research at the Jagiellonian


GridSpace2
You are logged in as plgtesteci@ui.cyfronet.pl
File Management
Upload Files
Browse...
nanostruct/nanostruct.exp.xml
Implementation of Algorithms o
Gems Management
© 2010 Distributed Computing Environments (DICE) Team Please report any problems related to this portal to our Issue Tracker
Acknowledgements

```
adf.png
adf.txt
color_coding.png
color_coding.txt
domains.png
domains.txt
domains_for_selected_
tolerance.png
domains_for_selected_
tolerance.txt
info.txt
input-711.in
input-original.in
input-perfect-10x10.in
input-perfect-33x33.in
```

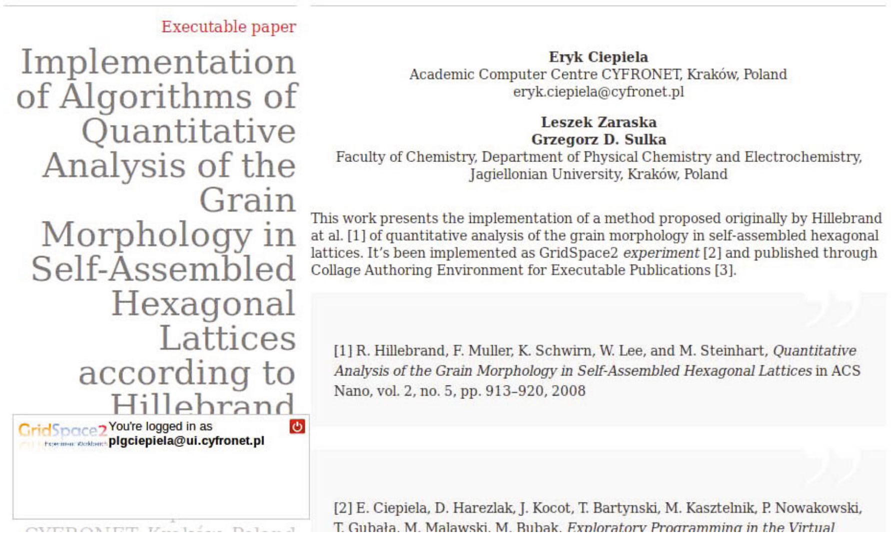
Path: ~plgtesteci/nanostruct/

plgtesteci Snippet #1	Python 2.6.4	Actions
plgtesteci Snippet #2	Mathematica 7.0	Actions
plgtesteci Snippet #3	Gnuplot 4.4.2	Actions
plgtesteci Snippet #4	Bash 3.00	Actions
plgtesteci Snippet #5	Gnuplot 4.4.2	Actions

```
>>> dia_avg = 2 * math.sqrt(area_avg / math.pi )
>>> dia_stddev = 2 * math.sqrt(area_stddev / math.pi )
>>>
>>> output = open('./nanostruct/input-stats.txt', 'w')
>>> output.write('Number of pores: ' + str(n) + '\n')
>>> output.write('Area: ' + str(area_avg) + ' +- ' + str(area_stddev) +
'\n')
>>> output.write('Diameter: ' + str(dia_avg) + ' +- ' + str(dia_stddev) +
'\n')
>>> output.write('Circularity: ' + str(circ_avg) + ' +- ' + str(circ_stddev)
+ '\n')
>>>
>>> output.close()
```

Fig. 4. Executing an experiment in the Experiment Workbench: the top right-hand panel displays the standard output of a snippet being executed. Resulting files are accessible through the left-hand side File Management panel.

University Electrochemistry Group since 2002. It should be mentioned that the experiment will also be used for analysis of structural features of nanostructures



Executable paper

Implementation of Algorithms of Quantitative Analysis of the Grain Morphology in Self-Assembled Hexagonal Lattices according to Hillebrand

Eryk Ciepiela
Academic Computer Centre CYFRONET, Kraków, Poland
eryk.ciepiela@cyfronet.pl

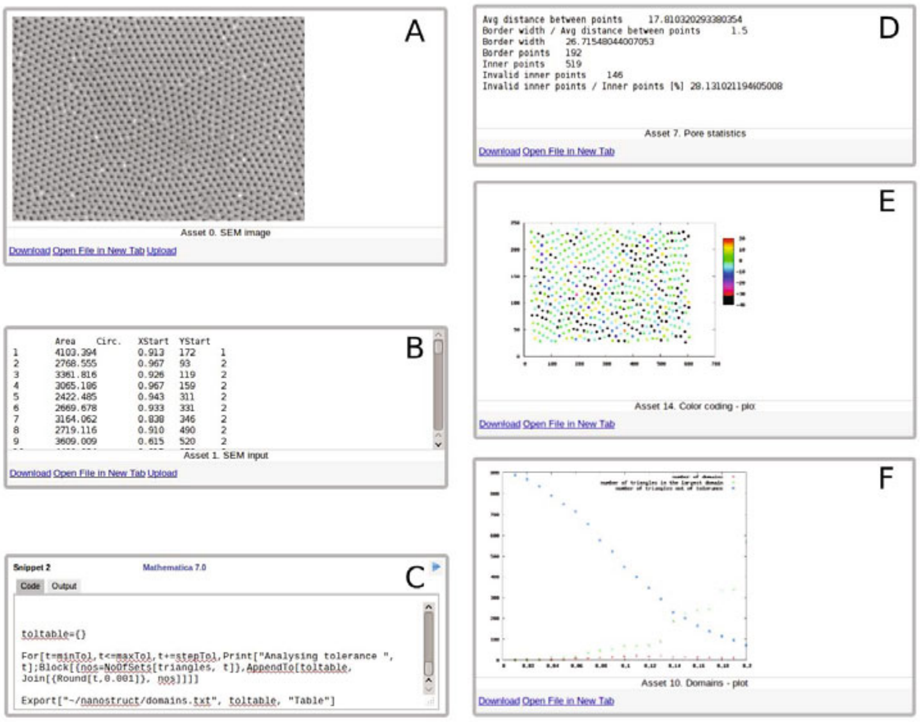
Leszek Zaraska
Grzegorz D. Sulka
Faculty of Chemistry, Department of Physical Chemistry and Electrochemistry, Jagiellonian University, Kraków, Poland

This work presents the implementation of a method proposed originally by Hillebrand at al. [1] of quantitative analysis of the grain morphology in self-assembled hexagonal lattices. It's been implemented as GridSpace2 *experiment* [2] and published through Collage Authoring Environment for Executable Publications [3].

[1] R. Hillebrand, F. Muller, K. Schwirn, W. Lee, and M. Steinhart, *Quantitative Analysis of the Grain Morphology in Self-Assembled Hexagonal Lattices* in ACS Nano, vol. 2, no. 5, pp. 913–920, 2008

[2] E. Ciepiela, D. Harezlak, J. Kocot, T. Bartynski, M. Kasztelnik, P. Nowakowski, T. Gubała, M. Malawski, M. Bubak, *Exploratory Programming in the Virtual*

Fig. 5. Header of the target executable paper published as a blog entry on the WordPress [24] blog platform



```
        Area      Circ.    XStart  YStart
1       4103.394           0.913   172     1
2       2768.555           0.967   93      2
3       3361.816           0.926   119     2
4       3065.186           0.967   159     2
5       2422.485           0.943   311     2
6       2669.678           0.933   331     2
7       3164.062           0.838   346     2
8       2719.116           0.910   490     2
9       3609.009           0.615   520     2
```

```
toltable={}

For[t=minTol,t<=maxTol,t+=stepTol,Print["Analysing tolerance ",
t];Block[{nos=NoOfSets[triangles, t]},AppendTo[toltable,
Join[{Round[t,0.001]}, nos]]]]

Export["~/nanostruct/domains.txt", toltable, "Table"]
```

```
Avg distance between points     17.810320293380354
Border width / Avg distance between points      1.5
Border width    26.71548044007053
Border points   192
Inner points    519
Invalid inner points    146
Invalid inner points / Inner points [%] 28.131021194605008
```

Asset 7. Pore statistics
Download Open File in New Tab
E
Asset 14. Color coding - plot
Download Open File in New Tab
F
Asset 10. Domains - plot
Download Open File in New Tab

Fig. 6. Sample assets of the target executable paper: sample input – SEM image of the pores (A), file containing pore information (B), sample snippet – geometric calculations in Mathematica [21] (C), sample output – calculated pore statistics (D), sample output – color coding of pores (E), sample output – quantitative features of pore grains plotted (F)

such as pore diameter, interpore distance, average circularity of pores, percentage of defective pores etc. The main advantage of the proposed solution is that all the parameters required for complex sample analysis can be obtained promptly without any additional calculations, time-consuming procedures, etc. The experiment is currently used for analysis of structural features of anodic alumina layers formed by anodization of aluminum in oxalic acid as well as anodic titania layers fabricated by anodization of titanium in ethylene glycol and glycerin based electrolytes. Several papers based on results obtained with the presented tool will be published soon.

5 Conclusions and Future Work

The authors first outlined the domain of self-assembled hexagonal nanoscale lattices and noted the significance of quantitative investigation of their features. This was followed by the problem statement: providing a convenient, easy to adopt and highly accessible software service implementing the Hillebrand et al.

method [1]. While designating a suitable solution both functional and nonfunctional requirements were taken into account. Although the required functionality could be achieved in many ways, the most effective (in the authors' opinion) approach to addressing the presented nonfunctional requirements was by using the PL-Grid infrastructure, or, more specifically, the GridSpace2 Virtual Laboratory managing the operation of cluster-based computational resources. Among the issues which determined the choice of GridSpace2 were the following:

1. Rapid experiment development enabled by the feature-rich platform.
2. No costs (in terms of power, hardware, software or intellectual property rights) to be borne by end users.
3. Qualified IT staff ensuring the operability and availability of the environment.
4. Excellent accessibility owing to the use of web technologies.

The result of this work is a full-fledged virtual experiment available for PL-Grid users through the Experiment Workbench [17] portal, as an executable publication [18] or as a collection of assets embeddable on arbitrary external websites.

Regarding methodology, the development of the GridSpace2 platform and the pilot experiment were conducted in a tight feedback loop. Use cases such as the one discussed in this work were the rationale behind several new features added to the GridSpace2 roadmap. Both the chemists and the PL-Grid development team benefited from this collaboration – on the one hand a useful domain-specific research tool has been created while on the other hand GridSpace2 was further adapted to the requirements of Polish computational scientists.

References

1. Hillebrand, R., Muller, F., Schwirn, K., Lee, W., Steinhart, M.: Quantitative Analysis of the Grain Morphology in Self-Assembled Hexagonal Lattices. ACS Nano 2(5), 913–920 (2008)
2. Sulka, G.D.: Highly ordered anodic porous alumina formation by self-organised anodising and template-assisted fabrication of nanostructured materials. In: Eftekhari, A. (ed.) Nanostructured Materials in Electrochemistry, pp. 1–116. Wiley-VCH (2008)
3. Sulka, G.D., Zaraska, L., Stępniowski, W.J.: Anodic porous alumina as a template for nanofabrication. In: Nalwa, H.S. (ed.) Encyclopedia of Nanoscience and Nanotechnology, vol. 11, pp. 261–349. American Scientific Publishers (2011)
4. Sulka, G.D., Brzózka, A., Zaraska, L., Jaskuła, M.: Through-hole membranes of nanoporous alumina formed by anodizing in oxalic acid and their applications in fabrication of nanowire arrays. Electrochimica Acta 55, 4368–4376 (2010)
5. Zaraska, L., Sulka, G.D., Jaskuła, M.: Anodic alumina membranes with defined pore diameters ant thicknesses obtained by adjusting the anodizing duration and pore opening/widening time. Journal of Solid State Electrochemistry 15, 2427–2436 (2011)

6. Horcas, I., Fernández, R., Gómez-Rodríguez, J.M., Colchero, J., Gómez-Herrero, J., Baro, A.M.: Rev. Sci. Instrum. 78, 13705 (2007)
7. Sulka, G.D., Jaskuła, M.: Defect analysis in self-ordered nanopore arrays formed by anodization of aluminium at various temperatures. J. Nanosci. Nanotechnol. 6(12), 3803–3811 (2006)
8. Zaraska, L., Sulka, G.D., Szeremeta, J., Jaskuła, M.: Porous anodic alumina formed by anodization of aluminum alloy (AA1050) and high purity aluminum. Electrochimica Acta 55(14), 4377–4386 (2010)
9. Kitowski, J., Turała, M., Wiatr, K., Dutka, Ł.: PL-Grid: Foundations and Perspectives of National Computing Infrastructure. In: PL-Grid 2011. LNCS, vol. 7136, pp. 1–14. Springer, Heidelberg (2012)
10. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)
11. Ciepiela, E., Nowakowski, P., Kocot, J., Harężlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing entire lifecycles of e-science applications in the gridSpace2 virtual laboratory – from motivation through idea to operable web-accessible environment built on top of PL-grid e-infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)
12. Ciepiela, E., Harężlak, D., Kocot, J., Bartyński, T., Kasztelnik, M., Nowakowski, P., Gubała, T., Malawski, M., Bubak, M.: Exploratory programming in the virtual laboratory. In: Proceedings of the International Multiconference on Computer Science and Information Technology, Wisla, Poland, pp. 621–628 (2010)
13. Nowakowski, P., Ciepiela, E., Harężlak, D., Kocot, J., Kasztelnik, M., Bartyński, T., Meizner, J., Dyk, G., Malawski, M.: The Collage Authoring Environment. In: Proceedings of the International Conference on Computational Science, ICCS 2011. Procedia Computer Science, vol. 4, pp. 608–617 (2011), http://www.sciencedirect.com/science/article/pii/S1877050911001220
14. Kurowski, K., Piontek, T., Kopta, P., Mamoński, M., Bosak, B.: Parallel Large Scale Simulations in the PL-Grid Environment. Computational Methods in Science and Technology, Special Issue, 47–56 (2010)
15. GridSpace technology homepage, http://dice.cyfronet.pl/gridspace
16. Collage Authoring Environment for executable publications, http://collage.cyfronet.pl
17. GridSpace2 Experiment Workbench – portal for PL-Grid users, https://gs2.plgrid.pl
18. Ciepiela, E., Zaraska, L., Sulka, G.D.: Implementation of Algorithms of Quantitative Analysis of the Grain Morphology. In: Self-Assembled Hexagonal Lattices according to Hillebrand method, executable publication hosted on Collage Authoring Environment demo instance in Academic Computer Centre Cyfronet AGH, http://gs2.cyfronet.pl/epapers/hillebrand-grains/
19. ImageJ, National Institute of Mental Health, Bethesda, Maryland, USA, http://rsb.info.nih.gov/ij

20. Python programming language, http://www.python.org/
21. Methematica software, http://www.wolfram.com/mathematica/
22. Python programming language, http://www.python.org/
23. Gnuplot – portable command-line driven graphing utility,
http://www.gnuplot.info/
24. WordPress, Blog Tool and Publishing Platform, http://wordpress.org/

Examining Protein Folding Process Simulation and Searching for Common Structure Motifs in a Protein Family as Experiments in the GridSpace2 Virtual Laboratory

Tomasz Jadczyk[1], Maciej Malawski[2], Marian Bubak[2,3], and Irena Roterman[4]

[1] AGH University of Science and Technology, ACC Cyfronet AGH, Kraków, Poland
jadczyk@agh.edu.pl
[2] AGH University of Science and Technology, Department of Computer Science, Kraków, Poland
[3] University of Amsterdam, Informatics Institute, The Netherlands
[4] Jagiellonian University, Medical College, Department of Bioinformatics and Telemedicine, Kraków, Poland

Abstract. This paper presents two *in-silico* experiments from the field of bioinformatics. The first experiment covers the popular problem of protein folding process simulation and investigates the correctness of the "Fuzzy Oil Drop" model (FOD) [3], on over 60 thousands of proteins deposited in Protein Data Bank [18]. The FOD model assumes the hydrophobicity distribution in proteins to be accordant with the 3D Gauss function differentiating the hydrophobicity density from the highest in the center of the molecule, to zero level on the surface. The second experiment focuses on performing comparison of proteins that belong to the same family. Examination of proteins alignment at three different levels of protein description may lead to identifying a conservative area in protein family, which is responsible for the protein function. It also creates a possibility of determining a ligand binding site for protein, which is a key issue in drug design. Both experiments were realized as *virtual experiments* in the GridSpace2 Virtual Laboratory [13] Experiment Workbench [16] and were executed on Zeus cluster provided by PL-Grid.

Keywords: virtual laboratory, in-silico experiment, bioinformatics, protein folding, fuzzy oil drop model, protein function, protein comparison.

1 Introduction

Bioinformatics is a relatively new field of science that has rapidly advanced in recent years. It is the application of computer science and information technology in the field of biology and medicine. Bioinformatics was first focused on creating biological databases and web-based front ends available to any user interested in searching and modifying biological data [1]. When the data availability has reached a satisfying level, the next stage of development in the field of bioinformatics approached. It was focused on providing and developing algorithms purposed to development of computing methods of analysing the structure, function and evolution of genes, proteins and genomes, as well as methods used in management and analysis of biological information data. Such data is gathered in the process of genomics studies and research conducted with the use of high-throughput experimental techniques [2]. Recently, rapid development of bioinformatics created a great opportunity to increase the understanding of biological processes, which may lead to development in many areas, important for humans, such as personalised medicine, gene therapy or computer-aided drug design. Among the variety of research areas, such as sequence alignment and analysis, genome annotation, evolutionary biology, phylogenetic analysis or analysis of protein or gene expression, we have focused on one of the most important fields – protein structure prediction and development of methods for quality testing of these algorithms.

Protein folding is the process by which a polypeptide folds into its characteristic and functional three-dimensional structure. The native conformation of a protein is determined by the amino-acid sequence (also called primary structure). Proteins are built up within cells, during the synthesis process. The correct three-dimensional structure is essential for the functioning of the process. The exact mechanism of protein folding is still unknown despite of long lasting research in biochemistry and bioinformatics. The goal expressed as: "Prediction of protein structure for known amino acid sequence" is suggested to be changed to the form: "Protein folding process simulation". The precise simulation of the process makes the computer models close to the real process observed in the experiments, suggesting more than one mechanism of folding and multi-intermediate character of the process.

The protein molecule was suggested by Kauzmann to represent the form of an oil drop [9]. This model, which is treated as a discrete one, was extended to the form of "fuzzy oil drop" taking the 3D Gauss function to describe the idealized hydrophobicity density distribution [3]. The "Fuzzy Oil Drop" model assumes the folding process as directed by water environment in form of introduction of external force field of hydrophobic character. The highest hydrophobicity concentration is expected in the center of the protein body, with the decrease of its values toward the surface where the hydrophobicity is expected to be close to zero. Before the model can be applied in protein folding simulation, the accordance of the assumed model with the real protein structure needs to be checked, using the proteins with known structure deposited in Protein Data Bank [18]. The goal of the *first experiment* described in this paper was to measure the accordance of the assumed model with the protein structure using elements of information theory.

The second part of the research takes into account a complex problem of protein comparison [4]. The problem of similarity search is closely related to all methods predicting protein structure and protein function based on the known amino acid sequence. The structures and functions of many important proteins in biomedicine and drug discovery are derived using the sequence alignment technique as the first step [12].

The proposed model for the search for conservative areas in a protein family was introduced on the basis of geometric characteristics of the polypeptide backbone conformation [7,8] and information theory (balancing the amount of information carried by an amino acid and the amount of information necessary to predict the conformation of that amino acid) [6].

Common structure motifs and conservative areas in a protein family may be found by performing comparison on three levels of protein description: amino acid sequences, secondary structures and 3D structures. Although the protein structure and the amino acid sequence are known, the structural codes representing conformations limited to the ellipse path can be reached by transformation of the observed Φ, Ψ angles to the ellipse path-limited conformational subspace on the Ramachandran map (dihedral angles obtained according to the shortest-distance criterion) [5]. Seven different structural classes have been distinguished on the basis of the limited conformational subspace. Some of the structural codes represent well-known secondary structures: $C-$ right-handed α-helix, $G-$ left-handed α-helix, E and F – various forms of β-structures. The others (A, B, D) are different forms of random coil. The goal of the second experiment is to align amino acids and structural codes by Multiple Sequence Alignment algorithm, which uses a two stage heuristic search. Firstly a phylogenetic tree that represents relationships between sequences is constructed and secondly, the sequences are added sequentially to alignment on the basis of this tree. To determine proper alignment the correct scoring matrix and gap penalties function are required. Although the algorithms for structures alignment work on different input data (atoms positions), the result is often reduced to aligned amino acid sequences, which are based on created superpositions of protein atoms.

2 Related Work

The complex nature of many biological processes enforces workflow structure of the bioinformatics experiments. Large scale processes could be divided into several smaller tasks, that are suitable to further use in other types of computations. Complexity of many processes and computations required to reflect the nature of known molecular interactions is another reason to use supercomputers. Many biological experiments should be conducted more than once, with different datasets and parameter compositions generating a "parameter study" series of experiments. This situation creates a background for creation of *in-silico* experiments. From the variety of choices, the Taverna project [17] is a Workflow Management System suited to create advanced workflows. It allows for designing and executing workflows from existing public services, as well as user-defined ones. The main advantage of GridSpace2 (GS2) Virtual Laboratory [13] is that the user can easily run their own algorithms on Grid resources. The GS2 Virtual Laboratory also aids users in composition of advanced workflows as well as scripts containing necessary tools (algorithms).

Previous research on accuracy of protein folding simulation models has focused on comparing the results of simulation with structure of the real protein, which has been analysed with X-Ray crystalography or Nuclear Magnetic Resonance (NMR) spectroscopy. This research does not take into account hydrophobicity within protein and was not conducted on large data sets, which is necessary to ensure the correctness of the model. It is the main part of this work, however we still need to use proteins with known structures.

In the field of Multiple Sequence Alignment (MSA) a variety of algorithms and improvements to the standard technique based on modifications in scoring matrix and gap penalties functions have been developed. In Multiple Structure Alignment there are available many programs for solving these issues. One of the most popular solutions has been implemented in *ClustalX* software (also available as a Web service – *ClustalW* and its new version – *Clustal Omega*). The other programs working in the MSA domain are e.g. *Muscle*, *T-Coffee*, *K-Align*. The multiple structural alignment is a much more complicated issue, where we can find a greater variety of algorithms. We have selected *Mammoth-mult* [20], which stands for MAtching Molecular Models Obtained from THeory. MAMMOTH based structure alignment methods decompose the protein structure into short peptides which are compared with the same length peptides of another protein. Similarity score between the two polypeptides is then calculated using a unit-vector RMS (URMS) method. Multiple structural alignments calculated with *MAMMOTH-mult* produce structurally implied sequence alignments. Other important programs are *Dali* (distance alignment matrix method, breaks the input structures into hexapeptide fragments and calculates a distance matrix by evaluating the contact patterns between successive fragments), *Combinatorial extension CE* (similar to *Dali* in that it also breaks each structure into a series of fragments which it then attempts to reassemble into a complete alignment), *RAPIDO* (Rapid Alignment of Proteins In terms of DOmains). However, presented software is able to perform only sequence or structural alignments. We will use these programs in a standard manner and combine the results in order to find conservative areas in protein families.

3 Description of the Solution

Solutions to bioinformatics problems are often well suited to be realized as *in-silico* experiments in virtual laboratory. Complicated workflows entail combination of many programs to pursue meaningful results regarding biological processes. Sufficient computational resources are often the second important requirement. The first problem – checking the accordance of protein folding model with proteins deposited in PDB – requires the experiment to be conducted on over 60 thousands of PDB files containing the protein sequence information. After successful execution the results should be summarized. The second problem – protein comparison – requires the use of different programs for extracting structural codes from 3D structures, by aligning sequences and structural codes, as well as 3D structures, and computing conformation scores and generating charts.

Many of the bioinformatical problems have similar structure to these presented above.

The GridSpace2 Virtual Laboratory was designed to facilitate the usage of scientific tools as sequential parts of experiments, which are written in one of the popular scripting languages. The idea of GridSpace2 Virtual Laboratory is also to provide access to computational resources of the PL-Grid Project, in this particular case we've used the Zeus cluster from the Academic Computer Centre CYFRONET AGH. The computational resources of Zeus are available via the PBS library, which provides a programming interface to the batch system and which is suitable for use in multiple programming languages. One of the benefits of using the GridSpace2 Virtual Laboratory as an environment for scripts execution is the complete automation of experiment workflows. The experiment may be composed of many parts (called *snippets*) in many programming languages (each *snippet* may be written in a different one). Subsequent parts of the experiment are performed automatically when all predecessors are finished. More detailed description of the GridSpace2 Virtual Laboratory is also available in [14,15].

3.1 Examining the FOD Model

The experiment examining the FOD model was focused on answering the question: are there many proteins representing the structure of *fuzzy oil drop* character with respect to hydrophobicity distribution? Large scale calculations were performed using the complete set of proteins present in the Protein Data Bank, aimed to identify the proteins representing the assumed structure. The expected hydrophobicity distribution was calculated according to 3D Gauss function, based only on the residues position and size of the protein molecule. The observed hydrophobicity distribution was calculated according to Levitt function [10], which takes into account empirical hydrophobic interactions of the residue with all other residues. The similarity of both distributions (expected and observed, denoted as O/T) was calculated according to Kullback-Leibler distance entropy [11]. The protein for which O/T is less than the distance between the observed distribution and a random one (O/R) was taken as the protein representing hydrophobic core of the *fuzzy oil drop* character.

The experiment was built as a set of scripts written in Ruby [21]. The following steps were identified as main parts of the workflow:

1. Create directory structure, download and extract PDB data, setup experiment parameters.
2. Decompose input data, prepare running scripts.
3. Send tasks to PBS (Torque) batch system, store received job identifiers.
4. For each job, run FOD model program for each PDB file in a sub-directory, compute the O/T and O/R values and store the results, additional information about protein (source organism, function, chain lengths) is also stored.
5. Wait for finishing all jobs.
6. Collect and summarize the results.

All structures deposited in RCSB PDB [18] are available for download, using *rsync*. Downloaded files are grouped into directories, according to the PDB identifiers (four characters code). The downloaded directory structure forms a basis for data decomposition in the experiment (every directory contains from a few to several dozen of PDB files). Each PDB file contains information about position of every atom in a molecule. The experiment requires setting up parameters, which are mainly data and result directories, FOD scripts for execution and PBS parameters. Firstly, all input data is unpacked and moved to the input directory and afterwards for each directory, the running script for FOD model is created and send to PBS (Torque) as a job. As a result, the job identifier is returned. The Torque batch system is then queried about the state of every job. When all jobs have finished, the final part of the experiment is executed. All output files are gathered and results are analysed from the point of view of accordance to the hydrophobic model. Additionally, summarized results are prepared with taking into consideration the protein source organism, chain lengths and protein function. The main script for calculating the FOD model parameters was written in Python [22] and was decomposed into the following steps:

1. For each file in input directory, read atom position for every amino acid in the molecule, files representing only nucleic acid molecules were excluded from the analysis.
2. Move the protein to the geometrical centre, simplify side chains for each residue to one point presentation called "effective atoms".
3. Rotate the molecule to put two residues with the longest distance between them along the x-axis, in the next rotation the two residues of the longest distance are put on the y-axis (only projection of the residues on the Y, Z plane are taken into account).
4. The longest distances in each direction determine the size of the Fuzzy Oil Drop, increasing the size by 9Å and dividing by 3, we are getting $\sigma_x, \sigma_y, \sigma_z$.
5. Determine the theoretical hydrophobicity as follows:

$$\widetilde{H}t_j = \frac{1}{\widetilde{H}t_{sum}} \exp\left(\frac{-(x_j)^2}{2\sigma_x^2}\right) \exp\left(\frac{-(y_j)^2}{2\sigma_y^2}\right) \exp\left(\frac{-(z_j)^2}{2\sigma_z^2}\right)$$

   Where $\widetilde{H}t_j$ expresses the hydrophobicity in the *j-th* point of space and x_j, y_j, z_j are positions of the *j-th* effective atom in 3D space.
6. Calculate the observed hydrophobicity as follows (according to Levitt function [10]):

$$\widetilde{H}o_j = \widetilde{H}_j^r + \sum_{i=1}^{N} \widetilde{H}_i^r \begin{cases} 1 - \frac{1}{2}\left(7\left(\frac{r_{ij}}{c}\right)^2 - 9\left(\frac{r_{ij}}{c}\right)^4 + 5\left(\frac{r_{ij}}{c}\right)^6 - \left(\frac{r_{ij}}{c}\right)^8\right), r_{ij} \leq c \\ 0,\ r_{ij} > c \end{cases}$$

   Where $\widetilde{H}o_j$ denotes the observed hydrophobicity in the *j-th* point in space, r_{ij} represents the distance between the *j-th* and *i-th* points ($j \neq i$), $\widetilde{H}_j^r$ is a hydrophobicity value specific to residue on *j-th* position in the protein chain, accordant to used hydrophobicity scale. The cut-off value c was set to $9\mathring{A}$. Finally, the $\widetilde{H}o_j$ value is normalized by $\widetilde{H}o_{sum}$.

7. Test the similarity of both distributions, according to Kullback-Leibler distance entropy [11]:
$$O/T = \sum_{j=1}^{N} \widetilde{H}o_j \cdot \log_2 \frac{\widetilde{H}o_j}{\widetilde{H}t_j}$$
8. Calculate the Random distribution:
$$O/R = \sum_{j=1}^{N} \widetilde{H}o_j \cdot \log_2 \frac{\widetilde{H}o_j}{R_j},\ R_j = \frac{1}{N}$$
9. Store results, proteins for which O/T is lesser than O/R are considered as accordant to the FOD model.

The structure of Fuzzy Oil Drop calculations was also presented in the *B* part of the Fig. 1. For parsing and extracting data from PDB files the Biopython package [23] was used.

3.2 Protein Sequence and Structure Comparison

The second experiment – Protein Sequence and Structure Comparison – was focused on finding a conservative area for proteins belonging to *immunoglobulins* superfamily, especially *Igg*, *Vcam* and *Icam* families. For each chain in selected proteins, the amino acid sequence, the structural code sequence and the 3D structure were examined and aligned. Sequences were aligned with *ClustalW* algorithm, while *Mammoth* was used to align the 3D structures. For each residue and every type of alignment, the W score that depicts the conservation of the area to which the residue belongs to, was calculated.

The experiment is a good example of how a user can employ external tools, available as web services, within the virtual laboratory experiments. The following steps are executed while the experiment is run:

1. Define set of PDB identifiers and selected chains for each protein to compare.
2. Download PDB files, using *DbFetch*.
3. Extract amino acid sequence from each PDB file.
4. Generate structural codes, that describe secondary level structures, with EarlyFolding – StepBack path [6].
5. Perform Multiple Sequence Alignment for Amino acid sequences (*ClustalW* [19], use "Blosum" as score matrix).
6. Perform Multiple Sequence Alignment for Structural codes (*ClustalW*, only identity matrix may be used).
7. Determine Multiple Structural Alignment (*Mammoth* [20]).
8. Calculate W score for each type of alignments, according to the following formula:
$$W = \log_{10}\left(\frac{F}{(N/M)+1}\right),$$
where F is a maximal code frequency for a particular position, N – is the total number of aligned sequences and M denotes the number of codes

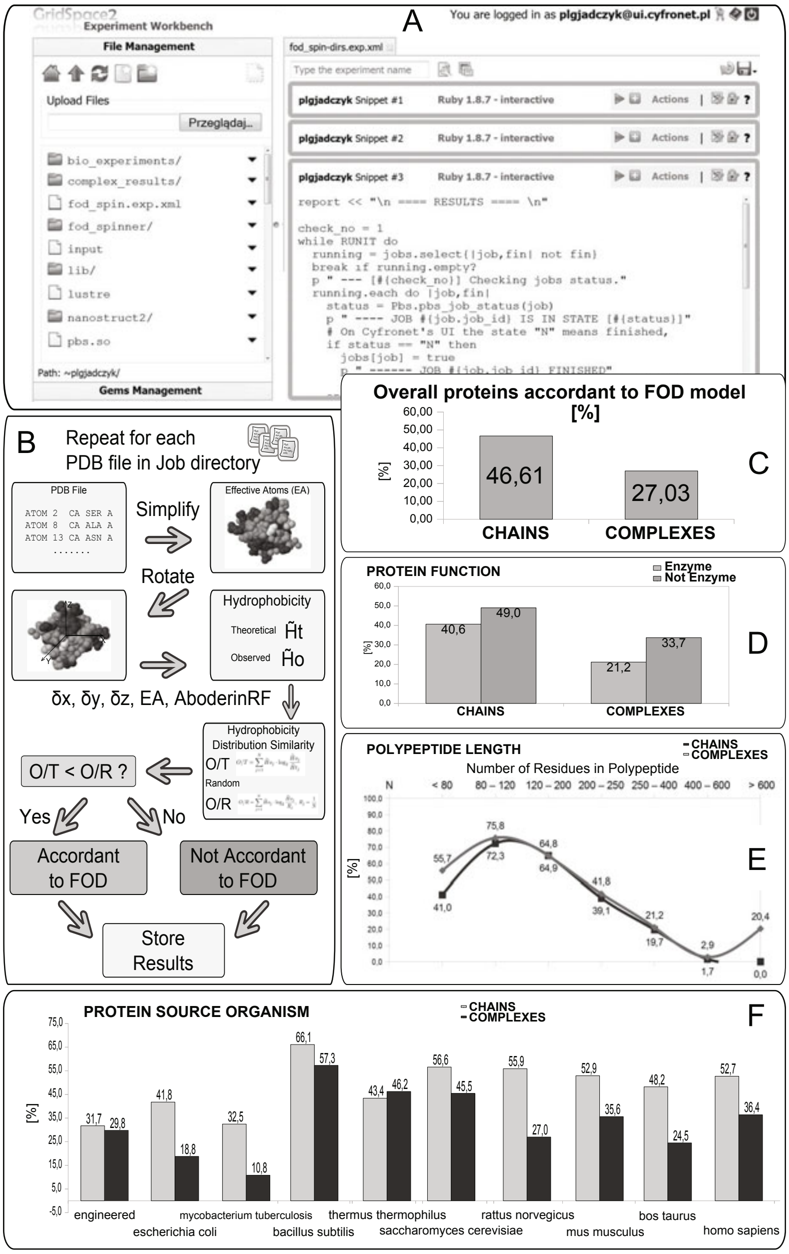
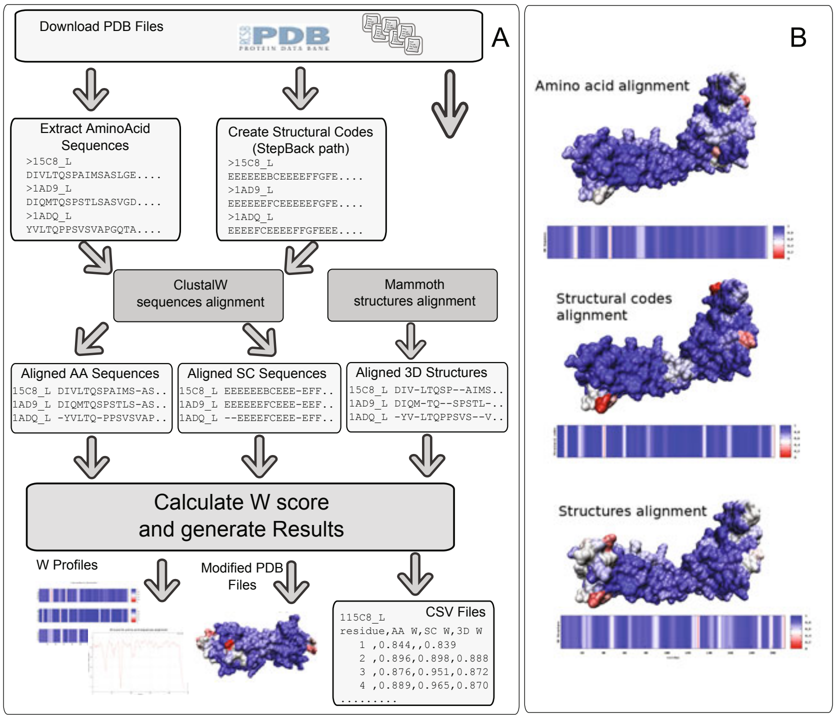

```
report << "\n ==== RESULTS ==== \n"

check_no = 1
while RUNIT do
  running = jobs.select{|job,fin| not fin}
  break if running.empty?
  p " --- [#{check_no}] Checking jobs status."
  running.each do |job,fin|
    status = Pbs.pbs_job_status(job)
    p " ---- JOB #{job.job_id} IS IN STATE [#{status}]"
    # On Cyfronet's UI the state "N" means finished,
    if status == "N" then
      jobs[job] = true
      p " ------ JOB #{job.job_id} FINISHED"
```

Fig. 1. The Fuzzy Oil Drop experiment in GridSpace2 Experiment Workbench (A). Simplified structure of the FOD algorithm (B). Set of results presenting the accordance of proteins deposited in PDB with FOD model: overall (C), with taking into account enzymatic function (D), length of polypeptide (E) and source organism (F).

($M = 8$ for structural code alignment and $M = 21$ for amino acid sequences, indels marked as "-" are also included). The resulting W value is then averaged, in this experiment the averaging window has a length of five residues, but can be easily changed by a researcher. The W value is then normalized to $[0; 1]$ range, in order to compare the results obtained in experiment runs, when protein families with different number of members are analyzed.

9. Generate charts (Gnuplot) with the W score, modified PDB files, completed with the W values and CSV files with results containing aligned sequences and the W score values for each residue in every protein.

The structure of this experiment is also depicted on the A part of the Fig. 2.


A
Download PDB Files
PDB PROTEIN DATA BANK
Extract AminoAcid Sequences
Create Structural Codes (StepBack path)
ClustalW sequences alignment
Mammoth structures alignment
Aligned AA Sequences
Aligned SC Sequences
Aligned 3D Structures
Calculate W score and generate Results
W Profiles
Modified PDB Files
CSV Files
B
Amino acid alignment
Structural codes alignment
Structures alignment

```
>15C8_L
DIVLTQSPAIMSASLGE....
>1AD9_L
DIQMTQSPSTLSASVGD....
>1ADQ_L
YVLTQPPSVSVAPGQTA....
```

```
>15C8_L
EEEEEEBCEEEEFFGFE....
>1AD9_L
EEEEEEFCEEEEEFGFE....
>1ADQ_L
EEEEFCEEEEFFGFEEE....
```

```
15C8_L DIVLTQSPAIMS-AS..
1AD9_L DIQMTQSPSTLS-AS..
1ADQ_L -YVLTQ-PPSVSVAP..
```

```
15C8_L EEEEEEBCEEE-EFF..
1AD9_L EEEEEEFCEEE-EEF..
1ADQ_L --EEEEFCEEE-EFF..
```

```
15C8_L DIV-LTQSP--AIMS..
1AD9_L DIQM-TQ--SPSTL-..
1ADQ_L -YV-LTQPPSVS--V..
```

```
115C8_L
residue,AA W,SC W,3D W
   1 ,0.844,,0.839
   2 ,0.896,0.898,0.888
   3 ,0.876,0.951,0.872
   4 ,0.889,0.965,0.870
.........
```

Fig. 2. Structure of the Protein Comparison experiment, with main steps and generated types of results (A). Sample results with marked conservation score for 1AD9, chain L, VMD was used for protein visualisation (B).

4 Results

The FOD experiment was implemented and executed in GridSpace2 Experiment Workbench. The Zeus cluster was used as an Experiment Host. A sample screen is presented on the *A* part of Figure 1. The complete set of proteins deposited in PDB (on March 2011) was used as input data. The input data set consisted of 71100 files that occupied over 11.4GB of disk space. The experiment data has been divided into 1060 parts, where each consisted of a few to several dozens PDB files to calculate the theoretical and observed hydrophobicity values, and was treated as a single job. The structural unit was defined in two ways: the protein complexes were taken as one unit, and each chain was taken separately. Percentage results of proteins accordant to the FOD model, with number of analysed structures, are presented in Table 1 (also in a chart form, on *C* part of Fig. 1).

Table 1. The percentage of proteins with the structure accordant to the assumed model

Structural unit	Examples	Accordant to model [%]
CHAINS	321829	46.61%
COMPLEXES	68207	27.03%

The following parameters were set to run the FOD model script: input data directory, selected hydrophobicity scale (AboderinRF), type of effective atom's position. The execution of complete workflow in GS2 Experiment Workbench took approximately 4.5 hours. Since the computations were totally independent, the speedup was nearly linear. The computation time depends mostly on availability of resources and the length of the job queue. Each single job was short enough to fit into the short queue (average time of the execution was 35 minutes). As a result, a file representing the database was generated. It stored the information extracted from 68100 proteins and 321800 protein chains. The information concerning: the protein enzyme characteristics (Fig. 1, *C*), the length of polypeptide (Table 2, Fig. 1, *D*) and the source organism (Table 3, Fig. 1, *E*), was also gathered and summarized.

The "Protein Sequence and Structure Comparison" experiment was executed to analyse the mechanism of signal transduction in immunoglobulins. The proteins belonging to the immunoglobulin super-family like *Icam* (1 chain selected), *Vcam* (3 chains) and *IgG* (35 chains) were analyzed. As a processor of sequences and structural codes alignment, the *ClustalW* was used. The *Mammoth* was employed for computing structures alignment. An experiment was performed for each super-family (except *Icam*) proteins, as well as for all combinations of listed super-families. The group of result files were generated for each protein: aligned sequences and structures, CSV files with *W* score for each residue in protein and modified PDB files, with *W* score values in appropriate positions, suitable to use in the visualising software, e.g. VMD and Gnuplot chart files showing the conservative score along the polypeptide length. After the complete execution of this experiment over one thousand of files were generated (including 189 pictures and CSV files and 342 PDB files). Sample results with marked conservation score for IGG-FAB fragment of engineered human monoclonal antibody (1AD9, chain L) was presented on Fig. 2, (B).

Table 2. The percentage of proteins with the structure accordant to the assumed model, classified on polypeptide length

Polypeptide length	Chains [%] (examples)	Complexes [%] (examples)
< 80	41.04% (97616)	55.72% (1452)
80 - 120	72.34% (67808)	75.78% (2337)
120 - 200	64.92% (65784)	64.77% (7846)
200 - 250	39.06% (24478)	41.76% (5019)
250 - 400	19.72% (42173)	21.16% (14727)
400 - 600	1.66% (18293)	2.88% (11098)
> 600	0.00% (5677)	20.37% (25728)

Table 3. The percentage of proteins with the structure accordant to the assumed model for the most popular source organisms

Source Organism	Chains [%] (examples)	Complexes [%] (examples)
engineered	31.73% (23023)	29.80% (1762)
escherichia coli	41.77% (20291)	18.80% (4565)
bacillus subtilis	66.07% (3914)	57.2% (648)
mycobacterium tuberculosis	32.46% (2825)	10.83% (739)
thermus thermophilus	43.42% (4169)	46.24% (1118)
saccharomyces cerevisiae	56.59% (9403)	45.48% (1515)
rattus norvegicus	55.87% (6545)	26.98% (834)
mus musculus	52.91% (17229)	35.62% (2187)
bos taurus	48.22% (6161)	24.50% (1245)
homo sapiens	52.70% (90091)	36.39% (14502)

5 Conclusions and Future Work

Two interesting problems from the field of bioinformatics were outlined in this work. Firstly, the nature of protein folding process simulation, especially the Fuzzy Oil Drop model, which takes into consideration the external force field of hydrophobic character, was presented. Methods for determining the theoretical and observed hydrophobicity along the polypeptide were proposed, and so was a method for comparing both hydrophobicity distributions that are based on the information theory. Secondly, the problem of similarity search in protein family was depicted in conjunction with the multiple sequence and structure alignment software. An algorithm for combining the alignment results for different levels of protein description and a measure of conservativeness of area in protein were presented.

Both solutions were implemented as *in-silico* experiments in GridSpace2 Virtual Laboratory, which allowed to use the PL-Grid infrastructure. Virtual experiments are available for PL-Grid users and may be executed by the researchers with different sets of input parameters, especially with another hydrophobicity scale for broadened testing of the Fuzzy Oil Drop model or with a different set of PDB identifiers to search for conservative areas in further protein families.

References

1. Baxevanis, A.D., Ouellette, F.B.F.; Bioinformatics: A Practical Guide to the Analysis of Genes and Proteins. Wiley-Interscience (2001) ISBN (0471383910)
2. Higgs, P.G., Attwood, T.: Bioinformatics and Molecular Evolution. Blackwell Publishing Limited (2005) ISBN (1405106832)
3. Konieczny, L., Brylinski, M., Roterman, I.: Gauss-function-based model of hydrophobicity density in proteins. Silico. Biol. 6, 15–22 (2006)
4. Brylinski, M., Konieczny, L., Kononowicz, A., Roterman, I.: Conservative secondary structure motifs already present in early-stage folding (in silico) as found in serpines family. Journal of Theoretical Biology 251, 275–285 (2008)
5. Brylinski, M., Konieczny, L., Roterman, I.: SPI – structure predictability index for protein sequences. Silico. Biol. 5, 22 (2004)
6. Jurkowski, W., Brylinski, M., Konieczny, L., Wisniowski, Z., Roterman, I.: Conformational subspace in simulation of early-stage protein folding. Proteins 55(91), 115–127 (2004)
7. Roterman, I.: The geometrical analysis of peptide backbone structure and its local deformations. Biochimie. 77(3), 204–216 (1995)
8. Roterman, I.: Modelling the optimal simulation path in the peptide chain folding – studies based on geometry of alanine heptapeptide. J. Theor. Biol. 177(3), 283–288 (1995)
9. Kauzmann, W.: Adv. Protein Chem. 14, 1 (1959)
10. Levitt, M.: A Simplified Representation of Protein Conformations for Rapid Simulation of Protein Folding. J. Mol. Biol. 104, 59–107 (1976)
11. Nalewajski, R.F.: Information theory of molecular systems. Elsevier (2006) ISBN 978-0-444-51966-5
12. Chou, K.C.: Review: structural bioinformatics and its impact to biomedical science. Current Medicinal Chemistry 11, 2105–2134 (2004)
13. Ciepiela, E., Harężlak, D., Kocot, J., Bartyński, T., Kasztelnik, M., Nowakowski, P., Gubała, T., Malawski, M., Bubak, M.: Exploratory Programming in the Virtual Laboratory. In: Proceedings of the International Multiconference on Computer Science and Information Technology, pp. 621–628 (2010)
14. Ciepiela, E., Nowakowski, P., Kocot, J., Harężlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing Entire Lifecycles of e-Science Applications in the GridSpace2 Virtual Laboratory – from Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)
15. Ciepiela, E., Zaraska, L., Sulka, G.D.: GridSpace2 Virtual Laboratory Case Study: Implementation of Algorithms of Quantitative Analysis of the Grain Morphology in Self-Assembled Hexagonal Lattices according to Hillebrand Method. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 240–251. Springer, Heidelberg (2012)
16. GridSpace2 Experiment Workbench – portal for PL-Grid users, https://gs2.plgrid.pl
17. Hull, D., Wolstencroft, K., Stevens, R., Goble, C., Pocock, M., Li, P., Oinn, T.: Taverna: a tool for building and running workflows of services. Nucleic Acids Research 34(Web Server issue), 729–732 (2006)
18. Research Collaboratory for Structural Bioinformatics, Protein Data Bank, http://www.rcsb.org/

19. Larkin, M.A., Blackshields, G., Brown, N.P., Chenna, R., Mcgettigan, P.A., Mcwilliam, H., Valentin, F., Wallace, I.M., Wilm, A., Lopez, R., Thompson, J.D., Gibson, T.J., Higgins, D.G.: Clustal W and Clustal X version 2.0. Bioinformatics 23(21) (2007)
20. Ortiz, A.R., Strauss, C.E., Olmea, O.: MAMMOTH (matching molecular models obtained from theory): an automated method for model comparison. Protein Science: A Publication of the Protein Society 11, 2606–2621 (2002)
21. Ruby programming language, `http://www.ruby-lang.org`
22. Python programming language, `http://www.python.org`
23. Biopython – a set of freely available tools for biological computation, `http://biopython.org/`

InSilicoLab – Managing Complexity of Chemistry Computations

Joanna Kocot, Tomasz Szepieniec, Daniel Haręźlak, Klemens Noga, and Mariusz Sterzel

AGH University of Science and Technology, ACC Cyfronet AGH, ul. Nawojki 11, 30-950 Kraków, Poland
{j.kocot,t.szepieniec,d.harezlak,k.noga,m.sterzel}@cyfronet.pl

Abstract. InSilicoLab is an application portal designed to support *in silico* experiments by easily running computational software on Grids. Unlike manual job submission or grid portals, InSilicoLab enables its users to run complex computations without technical knowledge of how to operate the grid resources. Instead, the users may focus only on the information and activities relevant to their research. This paper is a result of a feasibility study of applying the InSilicoLab concept to the domain of computational chemistry. It, therefore, includes a study of chemistry computations and their realisation in InSilicoLab, as well as a description of the generic architecture of the environment.

Keywords: in silico, experiment, computational chemistry, PSE, compputional resources, e-science.

1 Introduction

In the era of rapid technological progress, when computer centres constantly improve and extend their offer of computer resources, the barrier of a know-how necessary for using them, stays relatively high. Moreover, the development of tools aimed at facilitating the access and efficient usage of the resources still requires a lot of effort. The environment presented in this paper, called InSilicoLab [1], not only offers easy access to computational and storage resources, but it is designed especially to assist researchers in solving computational problems of their specific domain of science.

The motivation behind the development of this tool is rooted in the observation that the language used by the researchers in their daily work and the language used for operating computing infrastructures are very different. We intend to bridge this semantic gap by introducing a so-called *mediation layer*, that translates the goals expressed in the researcher's domain language into tasks understood by computational infrastructure. Defining the functionality and providing implementation of this layer is the main achievement of our work.

With the proposed solution, the researchers define the computational processes that lead them to results that are needed to solve their research problem – forming, in cooperation with the portal developers, so-called *experiments*.

An experiment is a combination of computational operations which produce valuable results based on the input data provided by the researcher. This includes running domain-specific applications on computational resources, as well as managing the input/output data and initial analysis of the results. Such experiments, being specific to a given domain of science, at the same time are generic enough to be used by various groups of researchers from that domain.

The scope of this article covers application of the InSilicoLab environment to computational chemistry – a domain in which advanced research requires intensive computations. However, the environment described hereby is rather a generic and comprehensive solution that can be applied to many other domains of e-science.

2 Computations in the Domain of Chemistry

The domain of computational chemistry is an interesting study, not only as a field of computing-intensive research, but also due to its specific nature. The computations in this domain often need several iterations using various (sets of) tools. This introduces a high level of complexity in managing the computation process and the data it consumes and produces.

In spite of this high complexity, it is possible to extract a common pattern of chemistry computations – reflecting the usage of chemistry tools, construction of the computations and analysis of their results. This pattern was a base for identifying a behavioural scenario of a chemistry experiment. The scenario includes a composition of input data for one of the popular chemistry applications, executing the application in order to obtain a number of preliminary results, picking the most promising of these results and using them as an input for detailed analysis with another application. This is described in detailed steps in Table 1.

An example implementation of this scenario performs a so-called conformation scan, which results in identifying the optimum geometry from a set of similar molecules. For this purpose, an initial molecule is transformed by rotation at a given bond, angle, or dihedral (this information is specific to the chemistry domain and defined by the user). For each transformation a new molecule is created, and for each molecule the first application is run to obtain initial results containing its computed energy value. Having these results, the scientist is able to identify an optimum conformation of the molecule (usually the one with minimum energy), and perform an in-depth analysis of this molecule with another chemistry application.

By analysing this scenario from the point of view of distributed computations we can observe the following characteristics:

– Both input and output of the computations are molecules, which are generated or/and transformed in the process.
– The number of molecules that need to be managed may be large.
– The processing of one molecule is independent from the others, therefore the computations can be performed in parallel without introducing communication overheads.

Table 1. A generic scenario of experiment in computational chemistry domain

No.	Description of the step
1.	Specify the program parameters for the first application – these are instructions specific to particular chemistry software, e.g. Gaussian [2].
2.	Specify the input data – for chemistry computations these are molecule files. **Note:** At this stage, all the information pertaining to the chemistry domain is gathered – the next four steps are concerned only with program execution.
3.	Create job description files, which describe a computational task for the computational resources. They are entirely infrastructure-specific, and involve specification of how input/output data will be managed and how the chemistry software will be executed. To execute many parallel jobs, a job description file has to be specified separately for each job.
4.	Submit each job (with its description file).
5.	Monitor job states – after submission, the job state changes while it waits in batch queues, and is submitted to the resources themselves. Finally, the job is executed on the resources and may be running for many hours, days or even longer.
6.	Download the results of each job – the results of each computational task are stored on storage resources specified in the job description file. To view the result files, the scientist usually has to download them from the storage.
7.	Analyse the result files. The results produced by most of the chemistry applications are large text files, containing the whole log of the computations. To analyse the results, the scientist has to view the files produced by all the jobs and find useful information within them – this could be, e.g. a molecule structure calculated as optimum by the application.
8.	Extract the relevant data (optimum molecule) from one of the result files. This, in practice, means: copy a part of the result file and save it in a format that could be used for further work. **Note:** This corresponds with step 2 – the result of this step is an input molecule.
9.	Define program parameters for another application (e.g. GAMESS [3]) which will perform a more detailed optimisation of the molecule extracted in step 8. **Note:** Steps 7, 8 and 9 again involve the researcher's domain knowledge. Similarly to steps 1 and 2, they lead to defining domain-specific parameters for a chemistry application to be run.
10.	Repeat steps 3-6 (prepare and run the computations on the resources) to start other computations and download the results.
11.	Download and view the final results obtained with the second application.

– Molecules can serve as an input for various applications, which might require a different format of molecule representation. Therefore, conversion between formats is an important feature enabling integration between different applications.

For computations like these described by the above scenario, the chemistry packages, often with restricted licences, have to be installed on computational nodes. These are usually accessed in batch mode, which implies using specialised tools for interaction with the infrastructure (e.g. for submitting jobs to the queues), taking care of proper file staging and download. In case of large result files (which is usually the case in chemistry computations), their analysis is additionally hampered by a need of accessing dedicated storage.

The researcher's skills in the field of their computations (characteristic to their domain of science) are often not enough to overcome the technological barrier of the resource access, where technical knowledge of the resource management process is needed. Moreover, managing the data and tasks sent to computational infrastructure might be a very time-consuming operation, but still, it is necessary to obtain the scientific results. Therefore, the goal of creating a modern platform for computational research on resource e-Infrastructures should be to rid the researcher of all the actions that are not directly related to the essence of their research and do not belong to their research domain.

3 The Layers of Computational Experimentation

The analysis of computations in domains of science such as chemistry leads us to introducing a distinction between different layers that may be identified in such computations.

Understanding a research scenario gives us an overview of the domain, which is the subject of the researcher's work. Such scenario should be formulated with the use of objects and processes specific to the scientist's research domain. Fulfilling this condition guarantees that most of the user's work will be focused on the area where their skills are highest, which makes the work most efficient. These objects and processes expressed in terms specific to a science domain constitute a layer of the computation process designed for interaction with the researcher – a *domain layer*.

On the opposite side of the computation process lies the *resource access layer* where the operations defined by the researcher are actually performed. These may include as well operations of submitting and monitoring computational jobs, as storage read or write requests. All these operations are realised using resource-specific commands and processes.

To allow the user to manage the computations from the perspective of the domain layer, and, at the same time, realise it with resource-specific processes, an additional layer is introduced – a *mediation layer*. It is responsible for translating a domain-specific scenario to an execution on the computational infrastructure (see Fig. 1).

The translation (mediation) between the sphere of the researcher's domain and the layer of resources requires performing all the actions the user would need to run their computations on the underlying resources. The functionality that should be addressed by this process can be described either as related to computational or to storage capabilities.

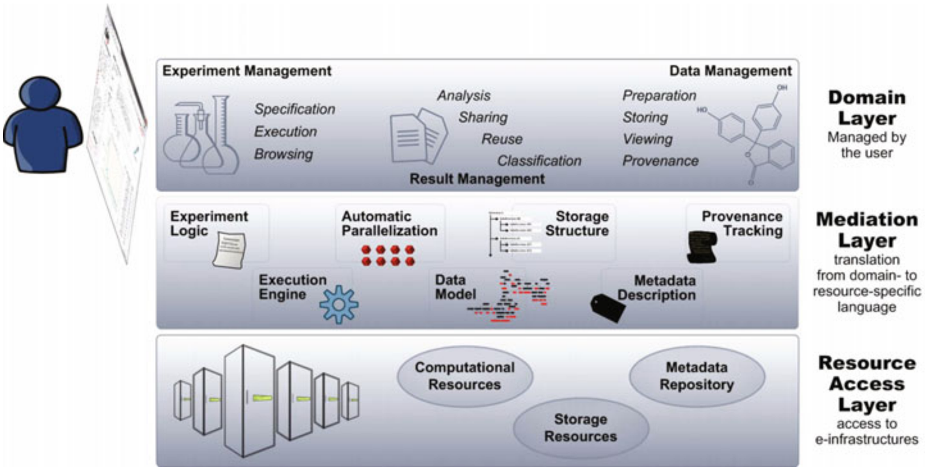


Fig. 1. The InSilicoLab architecture with the mediation layer, joining the researcher's domain sphere with computing resources

The computation-related functionality includes:

– *Experiment logic* – an entity organising the user's computations into a workflow that constitutes the *experiment* (understood as described in Section 1). It provides a description of the operations that have to be performed to create computational tasks from the user's input data, to monitor these tasks and to analyse their results.
– *Automatic parallelisation* – a service capable of automatically dividing the computations into independent computational jobs running in parallel. It defines the jobs, distributing the parts of computations so that the optimum number of computational tasks is grouped in a job.
– *Execution engine* – a service responsible for conducting and monitoring of all the operations defined by the experiment logic. The engine is an executor of the – to this moment abstract – workflow, on the computational resources.

The subset of functionality related to the infrastructure's storage capabilities is the following:

– *Storage structure* – a specific storage space organisation model, used for managing raw files created by the experiment and each of its jobs. It is applied to the storage resources to organise the data files, allowing for easier access to the files produced by the research experiment.
– *Data model* – a model of the data storage – applied to all data (other than raw files) that is used for the computations or obtained in their course.
– *Metadata description* – a model of the metadata attached to the data and files produced or used by an experiment. Such metadata may be used to organise and allow searching within data or raw files and to create relations between the data and processes.

These two groups are functionally joined by *provenance tracking* – a service responsible for recording of all the transformations and usage of data relevant to the user and their experimentation process. In this way, every experiment may be retraced and the origin of every data entity stored in the system may be identified.

The interactions between the services realising different functionalities of the mediation layer are depicted in Fig. 2.

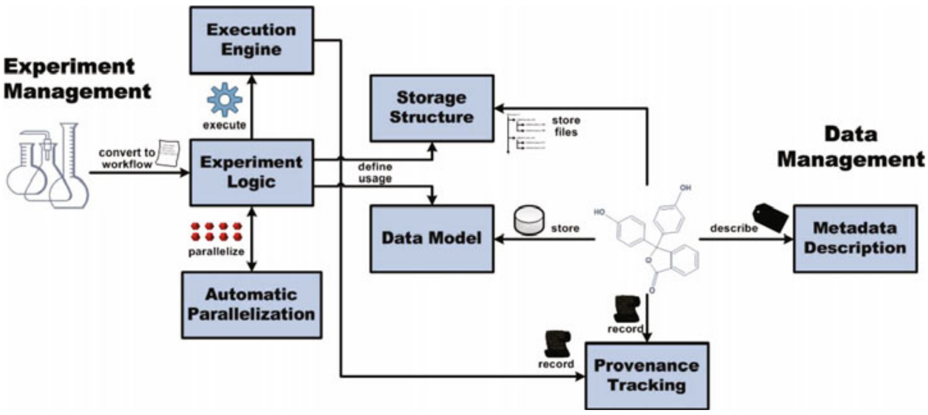


Fig. 2. The components of the InSilicoLab mediation layer, and interactions between them

4 Implementation of Chemistry Computations in InSilicoLab

The first implementation of the InSilicoLab environment provides all the tools required to perform the scenario of chemistry computations described in Section 2 with use of three chemistry packages: Gaussian, GAMESS and TURBOMOLE [4]. To access the environment, a Web portal was implemented and published – constituting an integrated presentation layer.

4.1 Realisation of the Scenario

The input files to the computations (experiment) are constructed from parameters specific to the aforementioned applications (or raw program input files) and from a molecule specification. The molecule can be provided in several formats, and translated to the one required by a particular application. The translation is based on the OpenBabel chemistry toolbox [5]. The molecules may be parameterised to enable performing conformation scans. These operations are enclosed within the domain layer, which means that they are well understood by a chemistry researcher who is willing to perform computations.

Underneath – in the mediation layer – on the base of the program input and the molecule parameterisation, conformers are automatically generated and divided into computational jobs. The jobs are encapsulated into job description files, suitable for a concrete infrastructure. All the input files are transferred to the experiment directory created on the storage system by the storage structure service. Each job is then submitted to the computational resources and monitored by the execution engine. The status of each job is continuously checked by the engine and displayed to the user. After each job finishes, its results are stored as raw files in the experiment directory on the storage system and filtered by a dedicated parser. The parser extracts the information that can be relevant to the scientist for initial analysis of the results. This information is stored by the data model service (based on SInt [6] models) in separate data store to allow easier access to it for the purpose of additional analysis or annotation.

All these operations are orchestrated by the experiment logic (an implementation of the abstract chemistry scenario in the environment). They usually require cooperation with the layer closest to the infrastructure – the resource access layer.

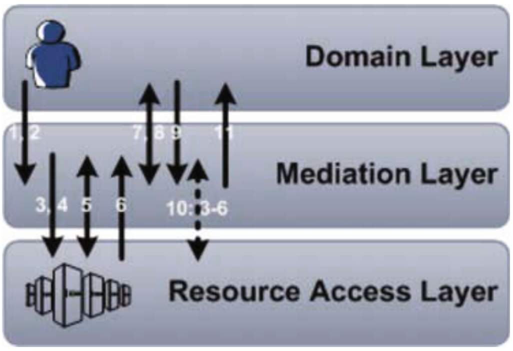


Fig. 3. The chemistry computations scenario realised in the InSilicoLab environment. The user is able to delegate all the steps that are not directly related to their domain of science to the mediation layer of the environment.

While the experiment is running, the results (stored already in the data store) are gathered and displayed jointly to the user (in form of a list as well as in a plot). Such presentation enables the scientist to identify the most promising results that should be used further to perform a detailed analysis. An alternative solution – without using InSilicoLab – would be to download all the result files and analyse them one by one. With InSilicoLab, the files and molecules identified as promising on the plot or in the result table can be viewed in place (in a Jmol [7] applet), downloaded, or reused in other computations (e.g. with another chemistry program) without additional effort. All the inputs and results are recorded in the provenance tracking service to enable repeating the experiments or tracing the data origin.

The implementation of the chemistry computations scenario within the InSilicoLab environment is illustrated by Fig. 3. The picture shows how the respective user operations (the numbers correspond with those from Table 1) are delegated to appropriate layers.

4.2 Integrated Presentation Layer

The InSilicoLab system is accessible through a Web interface, which was designed with special emphasis on intuitiveness. To achieve this goal, we implemented it, so that the users manage all the processes and objects concerning the experiment within the interface, using terms specific to their domain of science. In other words, this interface is an entrance to the domain layer of the system.

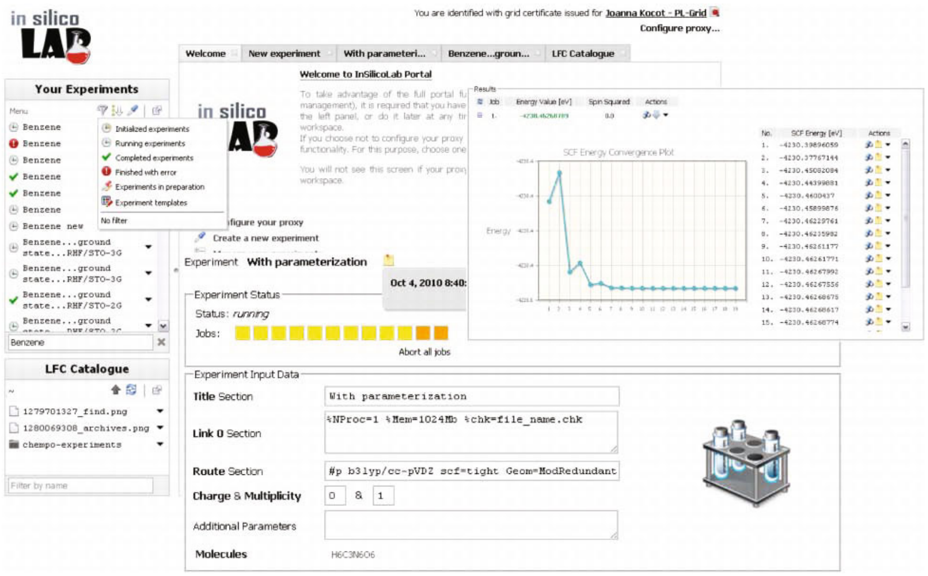
The construction of the InSilicoLab interface is meant to constitute a workspace – a powerful environment, where the researchers can find all the data used and produced by their experiments, along with the record of the experiment processes themselves. This requirement is fulfilled by integration of the following components (see also Fig. 4):

– Experiment browser – to enable the user to view and manage all the running and finished experiments.
– Input specification forms – forms specific to applications from the user's domain (e.g. Gaussian), to specify the program parameters.
– Monitoring screen – to view the experiment progress.
– Result browser – to view, compare and fetch relevant results of an experiment.
– Storage browser – to view the physical structure of the files used and created by the experiment or stored previously by the users and their research groups from outside of the portal. The browser also allows to upload new files to any accessible location on the storage.
– Data management tools – tools used to create, modify, describe and classify the data (e.g. with annotations or tags).
– Visualisation tools – tools used to visualise and analyse the data.
– Provenance record – a view of the history of all processes invoked on the user's data.

The aforementioned components, combined together, ensure the integration of the data and processes important for the user, thus allowing to significantly reduce the complexity of performing and managing their experiments.

5 Related Work

Knowing that the research in the computational chemistry domain is one of the most demanding in terms of computing resources, many supercomputing centres offer a variety of scientific packages for chemists. Unfortunately, the use of this software is hindered by the lack of intuitive interfaces.



You are identified with grid certificate issued for Joanna Kocot - PL-Grid

Configure proxy...

Welcome to InSilicoLab Portal

To take advantage of the full portal fu management), it is required that you have the left panel, or do it later at any tir workspace.
If you choose not to configure your proxy functionality. For this purpose, choose one

You will not see this screen if your prox workspace.

Experiment With parameterization

Experiment Status

Status: *running*

Jobs: 🟨🟨🟨🟨🟨🟨🟨🟨🟨🟨🟧🟧

Results

Job	Energy Value [eV]	Spin Squared	Actions
1.	[illegible]	0.0	

No.	SCF Energy [eV]	Actions
1.	-4230.39896059	
2.	-4230.37767144	
3.	-4230.45082084	
4.	-4230.44399881	
5.	-4230.4600437	
6.	-4230.45899876	
7.	-4230.46229761	
8.	-4230.46235982	
9.	-4230.46261177	
10.	-4230.46261771	
11.	-4230.46267992	
12.	-4230.46267556	
13.	-4230.46268675	
14.	-4230.46268617	
15.	-4230.46268774	

Experiment Input Data

Title Section	With parameterization
Link 0 Section	%NProc=1 %Mem=1024Mb %chk=file_name.chk
Route Section	#p b3lyp/cc-pVDZ scf=tight Geom=ModRedundant
Charge & Multiplicity	0 & 1
Additional Parameters	
Molecules	H6C3N6O6

Fig. 4. A view on the graphical tools of the InSilicoLab presentation layer

Some facilitation in the job submission process was already introduced by grid portals [8]. However, despite their usefulness in job management, these portals lack support for scientific experiment planning and evolution. There are also generic environments supporting scientific experimentation on various infrastructures, such as GridSpace2 [9], but being generic, they lack support for specific research domains and do not offer domain-specific interfaces that would be intuitive for researchers.

On the other hand, there are specialised tools invented for computational chemistry, such as WebMO [10] or ECCE [11], which do not provide access to larger e-Infrastructures. Commercial User Interfaces, such as Accelrys Material Studio [12], join both of the mentioned functionalities; however, they come with proprietary licences and are usually desktop applications what limits their usage to local computers.

6 Summary

The InSilicoLab portal is an integrated environment that, on one hand, offers access to e-Infrastructure resources, and, on the other hand, enables researchers to create and manage their scientific experiments and data using concepts specific to their field of science.

Such combination is possible thanks to the multi-layer architecture of InSilicoLab, which enables the scientists to interact with the system, operating on objects specific to their domain (such as experiments, results, analysis, etc.). The communication with computational and storage resources is shifted to a separate layer – the resource access layer – operated, in turn, by the mediation layer, which translates all the user actions and objects to ones understood by the resources of the e-Infrastructure.

The InSilicoLab portal is now available to the Gaussian VO [13] and `vo.plgrid.pl` [14], and has already been used by several users. Currently supported chemistry applications include Gaussian [2], GAMESS [3] and TURBOMOLE [4]. The applications that are planned to be integrated in the future will support the biochemistry [15] domain as well as astrophysics [16].

References

1. InSilicoLab – an application portal supporting in silico experiments on e-Infrastructures, `http://insilicolab.grid.cyfronet.pl`
2. Frisch, M.J., et al.: Gaussian 09. Gaussian, Inc., Wallingford CT (2009)
3. Schmidt, M.W., Baldridge, K.K., Boatz, J.A., Elbert, S.T., Gordon, M.S., Jensen, J.H., Koseki, S., Matsunaga, N., Nguyen, K.A., Su, S., Windus, T.L., Dupuis, M., Montgomery, J.A.: General Atomic and Molecular Electronic Structure System. J. Comput. Chem. 14, 1347–1363 (1993)
4. TURBOMOLE – Program Package for ab initio Electronic Structure Calculations, `http://www.turbomole.com/`
5. Open Babel: The Open Source Chemistry Toolbox, `http://openbabel.org`
6. Gubala, T., Bubak, M., Sloot, P.M.A.: A Semantic Integration of Collaborative Research Environments. In: Cannataro, M. (ed.) A Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare, ch. 26, pp. 514–530. Information Science Reference, IGI Global (2009)
7. Jmol: an open-source Java viewer for chemical structures in 3D, `http://www.jmol.org/`
8. Zhou, Z., Wang, F., Todd, B.D.: Development of Chemistry Portal for Grid-enabled Molecular Science. In: Proceedings of the First International Conference on e-Science and Grid Computing, pp. 48–55 (2005)
9. Ciepiela, E., Nowakowski, P., Kocot, J., Haręźlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing Entire Lifecycles of e-Science Applications in the GridSpace2 Virtual Laboratory – from Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)
10. WebMO – a free World Wide Web-based interface to computational chemistry packages, `http://www.webmo.net`
11. ECCE – Extensible Computational Chemistry Environment, `http://ecce.pnl.gov`
12. Materials Studio – a comprehensive materials modeling and simulation application, `http://accelrys.com/products/materials-studio/index.html`

13. Gaussian VO, http://egee.grid.cyfronet.pl/Applications/gaussian-vo/
14. PL-Grid – Polish Infrastructure for Supporting Computational Science in the European Research Space, http://www.plgrid.pl/
15. The Gromacs application, http://www.gromacs.org/
16. Barnacka, A., Bogacz, L., Gochna, M., Janiak, M., Komin, N., Lamanna, G., Moderski, R., Siudek, M.: PL-Grid e-Infrastructure for the Cherenkov Telescope Array Observatory. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 301–313. Springer, Heidelberg (2012)

Ab Initio Molecular Dynamics Simulations of Ketocyanine Dyes in Organic Solvents

Andrzej Eilmes

Jagiellonian University, Faculty of Chemistry,
Ingardena 3, 30-060 Kraków, Poland
eilmes@chemia.uj.edu.pl

Abstract. Ab initio Molecular Dynamics simulations of a ketocyanine dye in explicit solvents have been performed on a GP GPU Nvidia accelerators. The effects of single, double or dynamic precision used in calculations has been discussed. Accumulated Molecular Dynamics trajectories have been analyzed with the focus on orbital energies relevant for absorption spectra.

Keywords: Ab initio Molecular Dynamics, GP GPU computations, ketocyanine dyes, explicit solvent modeling.

1 Introduction

Spectral properties of organic dyes may be significantly affected by dye-solvent interactions. An example of dyes with solvent-dependent absorption and emission spectra are ketocyanine dyes [1]. Even more pronounced are the spectral effects of dye complexation by inorganic cations [2,3] leading to substantial red-shifts of the maxima in absorption or emission spectra. The effects of dye protonation [4] are similar. Ketocyanine dyes are therefore good probes to study microenvironmental effects in solution. Studies of solvation and protonation effects for electronic spectra not only attract attention of experimentalists, but are also interesting theoretically. Quantum-chemical investigations of such systems allow one to gain better insight into the nature of underlying processes and open the possibility to validate the methodology and to assess its performance.

Two extreme approaches in quantum-chemistry commonly applied to study solvent effects are the effective solvent models (continuum solvation models, e.g. Conductor-Like Screening Model [5] or Polarizable Continuum Model [6]) and the explicit solvent models (microsolvation). The first approach is computationally efficient, but requires proper parameterization of the solvent. Moreover, continuous solvent models usually fail for systems with specific solvent-solute interactions (e.g. hydrogen bonds) which can not be satisfactorily described by an effective parameterization. Explicit models, in which solute is embedded in a cluster of solvent molecules, can treat both parts of the system on equal footing and therefore do not suffer from the problem of effective parameters. The price for more realistic modeling is the computational time increasing rapidly with the number of explicit solvent molecules.

2 Related Works

Theoretical studies of microsolvation in explicit solvent are faced with a problem of appropriate generation of solute-solvent molecular structures. Classical Molecular Dynamics (MD) simulations are a fast and efficient method of modeling molecular structures of solutions for relatively long times. For ketocyanine dyes such approach was used recently in a model study of Li^+ and Mg^{2+} complexation in acetonitrile [7] following the work employing continuous solvation model for the same system [8]. However, classical MD requires proper force-field parameterization to provide physically realistic structures. Such parameterization may be derived from experimental data (by fitting parameters in order to reproduce properties of the system such as vibrational spectra, density, heat of vaporization, etc.) or/and from quantum-chemical calculations. It is usually unavailable for new systems, especially in the case where there are specific solute-solvent interactions (i.e. for the systems where also continuous solvent models are prone to fail). This is a serious disadvantage of classical MD.

On the other hand, ab initio Molecular Dynamics, e.g. Born-Oppenheimer Molecular Dynamics (BOMD), applying quantum-chemical methods to calculate energies and forces acting on atoms, which are then used to solve classical Newton's equations of motion, does not suffer from the parameterization problems. Basically, any standard quantum-chemical method, for which energy gradients are available, may be used to perform ab initio MD simulations, and quality of the results depends on the level of theory and the basis set used in quantum-chemical calculations. Therefore with no necessity to develop force-field parameterization for each type of solvent and solute, BOMD is easily applicable to a wide range of systems. It is, however, computationally demanding because it involves quantum chemical calculations of energies and gradients and for this reason its use is often limited to small systems and to short timescales.

3 Description of the Solution

Constant progress in computer technology and software development helps quantum chemists to break the barrier of computational cost for even larger systems. An obvious approach to large quantum-chemical tasks is to use cluster or grid computing [9,10]. However, in recent years new strategies became available in this field. Indeed, calculations on General-Purpose Graphics Processing Units (GP GPU) appear as a promising alternative to CPU computations in quantum chemistry, increasing the speed of calculations by an order of magnitude, thus making larger systems tractable. Whether these new possibilities can be fully exploited depends on the availability of the software capable of performing routine tasks on GP GPU.

In this work an attempt to use GP GPU computations for BOMD simulations of ketocyanine dye is reported.

TeraChem v. 1.45 software [11] was used on a machine equipped with two Nvidia Tesla M2050 GPUs. BOMD simulations were performed for a ketocyanine dye or its protonated form (dye-H) in acetonitrile (ACN) or ethanol (EtOH) solution (Fig. 1). In all quantum-chemical calculations Density Functional Theory (DFT) employing common B3LYP density functional with default TeraChem grid and the 6-31G basis set were used. Although a larger basis set with polarization functions would be desirable, current version of TeraChem does not support *d* functions in gradient calculations, which limits the choice of basis sets for BOMD simulations. This restriction is supposed to be lifted in the next release of the software [11].

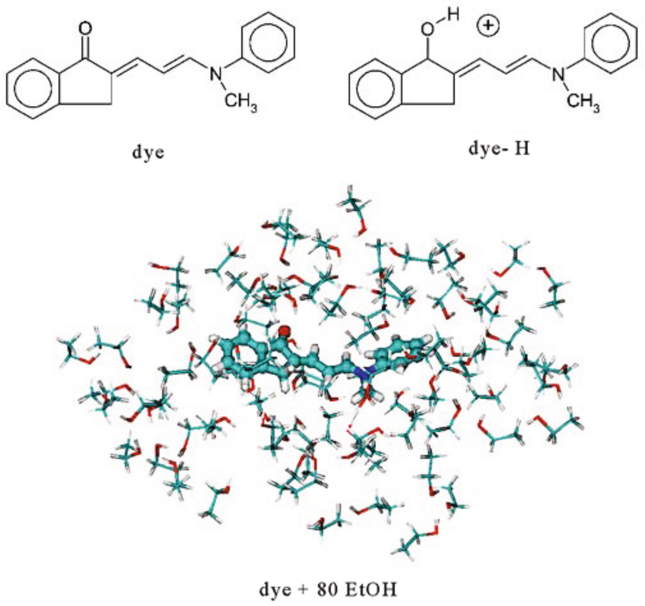


Fig. 1. Chemical formulas of the ketocyanine dye and its protonated form and a snapshot of the BOMD trajectory obtained for a single dye molecule solvated in a cluster of 80 ethanol molecules

50 acetonitrile or 40 ethanol molecules were typically used as an explicit solvent to solvate the dye. Two different initial structures were used for each system. To test the dependence of the results on the number of explicit solvent molecules, additional runs without solvent (for isolated dye molecule) and with 100 ACN or 80 EtOH molecules were performed.

In most MD runs a default TeraChem scheme of calculations with dynamic precision was used; single and double precision was tested in short additional simulations. All simulations were performed in the NVT ensemble (constant volume, temperature and the number of particles) using Langevin dynamics at T=300 K. Timestep of simulations was 1 fs as typically used in ab initio MD. After 0.5 ps of equilibration, trajectories of the systems were recorded for about 2.0 ps.

4 Results

While modern Nvidia GPUs allow to use double precision in calculations, they provide more hardware units for single precision. On the other hand, numerical tests showed that for some parts of quantum-chemical calculations single precision may be sufficient, provided that the accuracy is controlled. Therefore to optimize accuracy and computational time, TeraChem uses a dynamic precision scheme to decide (based on an estimate of upper bound of the integral) which integrals may be calculated using single precision and which require double precision [12].

Table 1. Average real time (in seconds) per 1 MD step in BOMD simulations employing different precision of GP GPU

	precision		
system	single	dynamic	double
dye + 50 ACN	129	146	231
dye-H + 40 EtOH	182	207	321

Table 1 summarizes average real (wall-clock) time per 1 MD step during calculations on two GPUs for dye in a cluster of 50 ACN molecules or protonated dye in 40 EtOH. As readily seen, double precision calculations are significantly slower and take about 70-80 % more time than single-precision computations. On the other hand, use of the dynamic precision scheme increases the average time of calculations by less than 15 %. To investigate the reliability of the results obtained using lower precision we checked how the precision affects the calculated potential energy of the system, i.e. quantum-chemical Self Consistent Field (SCF) energy, and the energy gap between the Highest Occupied Molecular Orbital (HOMO) and the Lowest Unoccupied Molecular Orbital (LUMO) (Fig. 2).

For both systems dynamic precision gives practically the same values of potential energy as double precision calculations; the difference increases to about 0.3 - 0.4 kcal/mol in single precision which, however, still is below chemical accuracy. The picture becomes different for the HOMO-LUMO energy gap. For dye/ACN system, the dynamic and double precision methods yield almost the same values while single precision overestimates the gap by about 30 cm^{-1}. In the case of dye-H/EtOH both dynamic and single precision results behave similarly and they differ from the double precision values by -10 to 10 cm^{-1}. Nevertheless, maximum errors are small and the average deviation is almost zero, therefore again dynamic precision appears as a reasonable compromise between accuracy and speed. Different behavior of potential energy and HOMO-LUMO gap may be rationalized by taking into account that the wavefunction (thus orbital energies, especially LUMO energy) is more sensitive to perturbations than the total SCF energy. The difference between the two systems is likely to be related to hydrogen bonds present in the dye-H/EtOH and their dynamic pattern.

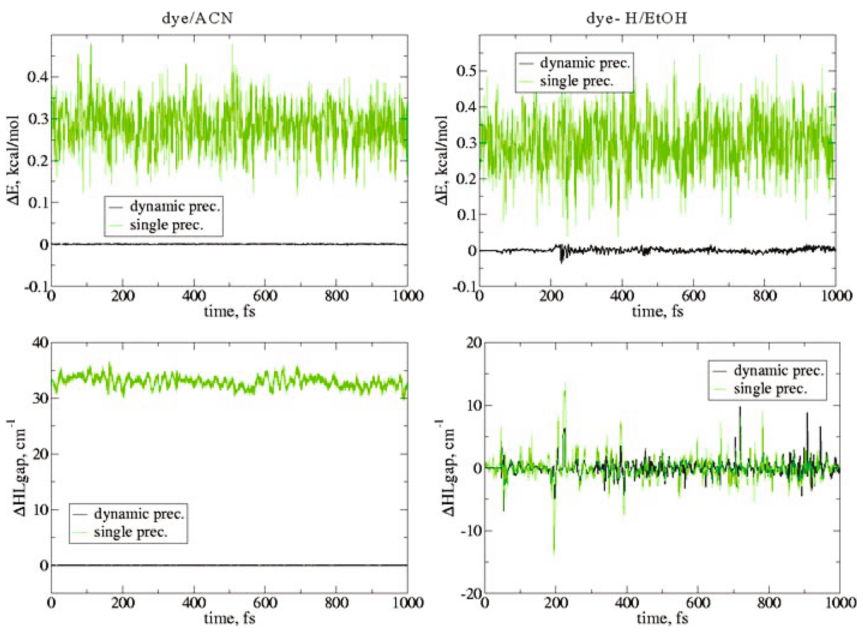


Fig. 2. Differences between potential energies (upper row) or HOMO-LUMO gap values (bottom row) calculated using dynamic and single precision and the corresponding values obtained using double precision calulations. Left column: dye in 50 acetonitrile molecules, right column: protonated dye in 40 ethanol molecules.

Test results suggest therefore that the dynamic precision scheme provides satisfactory accuracy, and even single precision calculations may be acceptable for some properties (however probably with the need of preliminary checks). All other calculations reported here were performed using the dynamic precision scheme.

Table 2 presents the values of average real time per 1 MD step. For smaller systems (40 or 50 solvent molecules) simulation of 1 fs (on both GPUs working in parallel) takes about 3-4 min of real time. With such speed, it is possible to calculate 2.5 - 3.5 ps of MD trajectory per week, which is a very satisfactory result for this system size and basis set. Likewise, simulations for larger systems are also time-efficient and could yield a reasonable length of MD trajectory in acceptable time.

Accumulated trajectories may be further analyzed to investigate the changes of potential energy and to observe reorganization of solvent molecules in the dye-solvent aggregate. Quantum-chemical nature of BOMD calculations enables one to gain more insight also into some properties related to the wavefunction of the system. We will focus here on spectroscopic properties based on HOMO and LUMO energies recorded during simulations. HOMO-LUMO separation is related to excitation energies and may serve as a first approximation for the energy of the transition observable in the absorption spectrum of the dye in solution.

Table 2. Average real time (in seconds) per 1 MD step for systems of different size in BOMD GP GPU calculations using dynamic precision

system	no. of atoms	no. of basis functions	time per MD step
dye + 50 ACN	338	1873	165
dye + 100 ACN	638	3523	531
dye + 40 EtOH	398	1783	221
dye + 80 EtOH	758	3343	695
dye-H + 50 ACN	339	1875	162
dye-H + 40 EtOH	399	1785	221

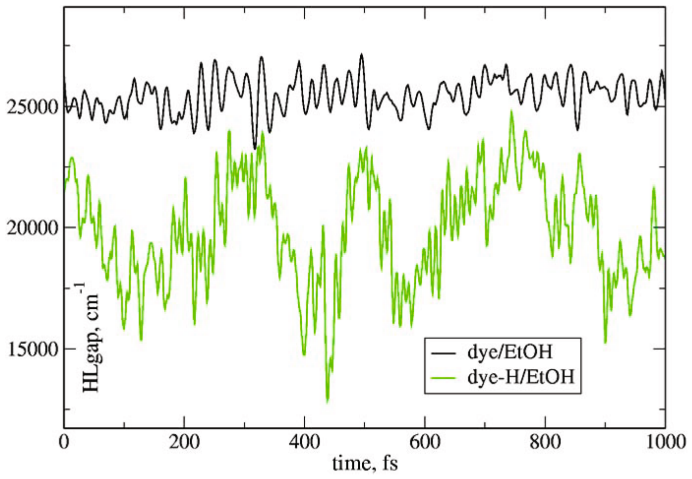


Fig. 3. Fluctuations of the HOMO-LUMO separation during 1 ps BOMD simulations for dye or protonated dye molecule solvated in 40 ethanol molecules

An example of changes of the HOMO-LUMO separation is shown in Fig. 3 for dye and its protonated form in ethanol. Oscillations with the period of about 20 fs are related to vibrations of the dye molecule. Changes on longer timescale result from the evolution of the aggregate structure (changes of the geometry resulting from the movements and reorientation of solvent molecules) and these fluctuations are more prominent for protonated dye.

Evolution of the system may vary depending on its initial structure, therefore it is always desirable to average data over several independent MD runs. In Fig. 4 we display distributions of the HOMO-LUMO gap averaged over two trajectories for each type of the system. As readily seen, positions of the maxima of distributions are similar in both solvents. Protonation of the dye significantly decreases the HOMO-LUMO separation (regardless of the solvent) in qualitative agreement with experimental observations that absorption spectrum of protonated dye is shifted to lower energies.

It is also visible that the HOMO-LUMO gap for dye-H/EtOH system varies in a much broader range than for dye/ACN, and the long tail of the distribu-

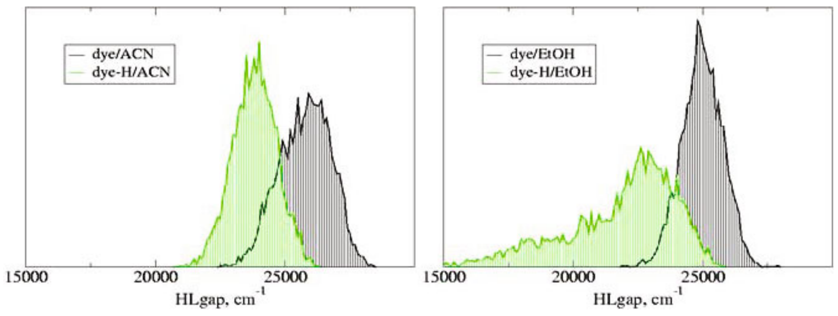


Fig. 4. Distributions of the HOMO-LUMO gap values for dye and protonated dye molecules solvated in 50 acetonitrile or 40 ethanol molecules. Results averaged over two different MD trajectories.

tion spans almost 10000 wavenumbers. This feature is likely to be caused by coupling of the hydrogen atom from the OH group of protonated dye to dynamically changing system of hydrogen bonds between solvent molecules. The major difference between solvents seems therefore to be the ability of EtOH to form hydrogen bonds which particularly affects the properties of protonated dye in solution. Such feature can not be properly described in continuous model of solvation, but may be captured by quantum-chemical calculations with explicit solvent molecules.

To check how fast the solvent effect saturates with increasing number of explicit solvent molecules one may examine plots like Fig. 5 showing the HOMO-LUMO gap distributions for isolated dye molecule and for dye embedded in 50 or 100 acetonitrile molecules. HOMO-LUMO separation decreases in the

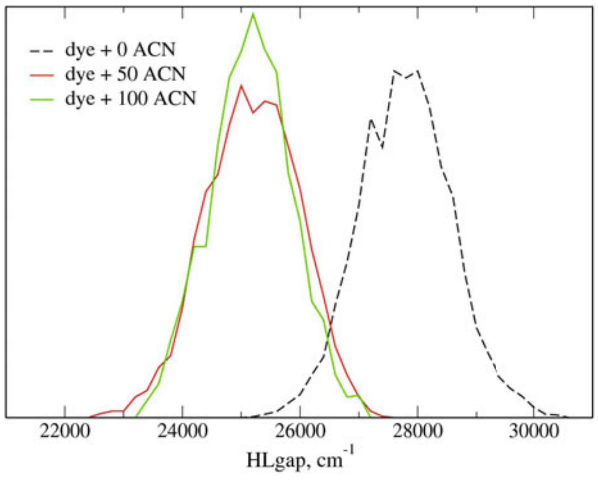


Fig. 5. Distributions of HOMO-LUMO gap values obtained from BOMD trajectories for isolated dye molecule and the dye solvated in 50 or 100 acetonitrile molecules

solvent, which is a typical solvent effect resulting in absorption red-shift. There are no major differences between systems with 50 or 100 ACN molecules, suggesting that the effect of the solvent has already saturated for about 50 solvent molecules (although averaging over more different trajectories would be desirable to support such conclusion).

5 Conclusions and Future Work

To conclude, it has been shown that the GP GPU BOMD simulations are a valuable tool to study the evolution of structure and properties of small molecular systems in quantum-chemical fashion. Performance of GP GPU accelerators significantly increases capabilities of such approach, allowing one to use larger systems, larger basis sets or more time-demanding quantum-chemical methodology, to perform simulations for longer timeframes or to collect more individual trajectories to improve the statistical analysis. In addition to methods used in classical MD to analyze geometries and structural changes, in BOMD it is also possible to examine properties closely related to the wavefunction of the system. As an example we presented such analysis of the HOMO-LUMO separation.

If necessary, individual frames from recorded trajectories may be used in subsequent quantum-chemical computations applying more advanced methodology to increase accuracy or to calculate more properties for selected dye-solvent structures. For example, in the case of dye-solvent systems studied here it is possible to use Time Dependent Density Functional Theory (TDDFT) in order to calculate transition energies and to obtain simulated absorption spectra. As the TDDFT methodology is scheduled for implementation in TeraChem software, such post-processing of BOMD data will also greatly benefit from the speed of GP GPU accelerators.

References

1. Reichardt, C.: Solvatochromic Dyes as Solvent Polarity Indicators. Chem. Rev. 94, 2319–2358 (1994)
2. Basu, J.K., Shannigrahi, M., Bagchi, S.: Lithium-Ion Ketocyanine Dye Interactions in the Ground and Excited States. J. Phys. Chem. A 110, 9051–9056 (2006)
3. Basu, J.K., Shannigrahi, M., Bagchi, S.: Ground and Excited State Complexation of Ketocyanine Dyes with Alkaline Earth Metal Ions. J. Phys. Chem. A 111, 7066–7072 (2007)
4. Sarkar, A., Kedia, N., Purkayastha, P., Bagchi, S.: Synthesis and spectroscopic investigation of a novel solvatochromic dye. J. Lumin. 131, 1731–1738 (2011)
5. Klamt, A., Schüürmann, G.: COSMO – a new approach to dielectric screening in solvents with explicit expressions for the screening energy and its gradient. J. Chem. Soc. Perkin. Trans. 2, 799–805 (1993)
6. Miertuš, S., Scrocco, E., Tomasi, J.: Electrostatic interaction of a solute with a continuum. A direct utilization of ab initio molecular potentials for the prevision of solvent effects. Chem. Phys. 55, 117–129 (1981)

7. Eilmes, A.: TDDFT study of absorption spectrum of ketocyanine dye complexes with inorganic ions: explicit solvent model. Theor. Chem. Acc. 127, 743–750 (2010)
8. Eilmes, A.: A DFT/TDDFT study of Li^+ and Mg^{2+} interactions with ketocyanine dye. J. Mol. Struct. THEOCHEM 915, 141–148 (2009)
9. Bosak, B., Komasa, J., Kopta, P., Kurowski, K., Mamoński, M., Piontek, T.: New Capabilities in QosCosGrid Middleware for Advanced Job Management, Advance Reservation and Co-allocation of Computing Resources – Quantum Chemistry Application Use Case. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 40–55. Springer, Heidelberg (2012)
10. Jadczyk, T., Malawski, M., Bubak, M., Roterman, I.: Examining Protein Folding Process Simulation and Searching for Common Structure Motifs in a Protein Family as Experiments in the GridSpace2 Virtual Laboratory. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 252–264. Springer, Heidelberg (2012)
11. TeraChem v. 1.45, PetaChem, LLC, http://www.petachem.com
12. Luehr, N., Ufimtsev, I.S., Martinez, T.J.: Dynamic Precision for Electron Repulsion Integral Evaluation on Graphical Processing Units (GPUs). J. Chem. Theor. Comp. 7, 949–954 (2011)

Polish Contribution to the Worldwide LHC Computing

Artur Binczewski[3], Michał Bluj[5], Antoni Cyz[2], Michał Dwużnik[1], Maciej Filocha[4], Łukasz Flis[1], Ryszard Gokieli[4,5,⋆], Jarosław Iwaszkiewicz[4], Marek Kowalski[2], Patryk Lasoń[1], Rafał Lichwała[3], Michał Łopuszyński[4], Marek Magryś[1], Piotr Malecki[2], Norbert Meyer[3], Krzysztof Nawrocki[4,5], Andrzej Olszewski[2], Andrzej Oziębło[1], Adam Padée[4,5], Henryk Pałka[2,∗], Marcin Pospieszny[3], Marcin Radecki[1], Radosław Rowicki[4], Dorota Stojda[6,4], Marcin Stolarek[4], Tomasz Szepieniec[1], Tadeusz Szymocha[1,2], Michał Turała[1,2], Karol Wawrzyniak[4,5], Wojciech Wiślicki[4,5], Mariusz Witek[2], and Paweł Wolniewicz[3]

[1] AGH University of Science and Technology, ACC Cyfronet AGH, Kraków, Poland
[2] Institute of Nuclear Physics PAN (IFJ PAN), Kraków, Poland
[3] Poznań Supercomputing and Networking Center (PSNC), Poznań, Poland
[4] Interdisciplinary Centre for Mathematical and Computational Modelling (ICM), Warsaw, Poland
[5] Soltan Institute for Nuclear Studies (IPJ), Warsaw, Poland
[6] Copernicus Science Centre (CSC), Warsaw, Poland

Abstract. The computing requirements of LHC experiments, as well as their computing models, are briefly presented. The origin of grid technology and its development in high energy community is outlined, including the Polish participation. The LHC Computing Grid project and its successor, the Worldwide LHC Computing Grid, are presented, including the summary of its successful operations in the first years of LHC data gathering. Against such a background, the creation and operation of the Polish Tier-2 is described, including examples of its use by the LHC experiments.

Keywords: Distributed computing, Grid, Middleware, gLite, WLCG, Tier-2, LHC, High Energy Physics.

1 Introduction

1.1 Requirements for LHC Computing

Searches for new rare phenomena in particle physics, and/or precise measurements of specific physics processes, require experiments at very high energies and very high rates. The resulting high bandwidth of data – approximately 300 MB/s – leads to petabytes per year per experiment, which have to be rapidly stored, processed and analysed. Additional computing resources, processors and

⋆ Deceased.

storage, are needed for Monte Carlo studies. The computing needs of LHC experiments (Monte Carlo simulations, large data volumes, high processing power, access to data for all collaborating parties on all continents – altogether a few thousand physicists from a few hundred institutions, etc.) have been recognized already at the time of writing proposals [1]; however, it was not clear how to solve the issue. Towards the end of the '90s a project called MONARC (Models of Networked Analysis at Regional Centres for LHC Experiments) was established [2], with the goal to find the solution. The MONARC came out with a hierarchical model of LHC computing, with several layers of "tiers", reflecting their role and size of their infrastructures: Tier-0 at CERN, a dozen of Tier-1s in large computing centres in Europe, Asia and America, and a large number of Tier-2s organized around smaller centres in the countries involved in the LHC program. However, MONARC discovered a problem: data sharing would be difficult due to the prohibitively high cost of networking; and it was even considered to transport the data to regional centres via... planes...

The needs of LHC experiments as well as the LHC computing model were summarized in the CERN LHC Computing Review [3]. Primary event processing occurs at CERN in the Tier-0 Facility. The RAW data is archived at CERN and copied (along with the primarily processed data) to the Tier-1 facilities around the world. These facilities archive the RAW data, provide the reprocessing capacity, provide access to various processed versions and allow for scheduled analyses of the processed data by physics analysis groups. Derived datasets produced by the physics groups are copied to the Tier-1 or Tier-2 facilities for further analysis. The Tier-2 facilities also provide the simulation capacity for the experiment, with the simulated data housed at Tier-1s. In addition, the Tier-2 centres provide analysis facilities and some of them offer the capacity to produce calibrations based on processing some RAW data. The CERN Analysis Facility supplies an additional analysis capacity, with an important role in the data-intensive calibration and algorithmic development work.

1.2 Worldwide Computing Grid

Towards the end of the 20th century, the cost of networking went down significantly due to the liberalisation of the networking market; in other words, for the same money one could buy much more bandwidth each year (the performance of the network started to double approximately every 8 months while before it took 8 years!...) [4]. Such changes allowed for integration of processors, storage and networking into one computing infrastructure, governed by a dedicated middleware, called the Grid [5].

This new paradigm was quickly spotted and adopted by the physics community. In the year 2000 the DataGrid project, led by CERN, was launched, being soon (2002) followed by the CrossGrid project, led by the Academic Computer Centre Cyfronet AGH [6]. The results of those projects were encouraging, and based on their initial experience a pilot project, the LHC Computing Grid (LCG), was launched at CERN. Its goal was to demonstrate at a large scale (about 20-30% of the final size) the operation and usefulness of the computing grid infrastructure for the storage and analysis of the physics data [7]. The project was supported by a series of three EU Grid projects, under the name of Enabling Grid for E-SciencE (EGEE) [8], aiming to support research communities of different fields. Both projects, LCG and EGEE, worked closely together and have developed grid middleware, called gLite, which is now in use [9].

The LCG project was successful and soon it was joined by other grid initiatives, Open Science Grid and Nordic Data Grid Facilities [10]. They use a different brand of middleware; however, they could be interfaced together. Today, they form one Worldwide LHC Computing Grid (WLCG), based on global collaboration of more than 140 computing centres in 35 countries, 4 LHC experiments, and several national and international grid projects. In 2005 a corresponding Memorandum of Understanding (MoU) was prepared for Collaboration in the Deployment and Exploitation of the Worldwide LHC Computing Grid between CERN, the provider of the Tier-0 centre and the CERN Analysis Facility as well as the coordinator of the LCG project, and all the Institutions participating in the provision of the Worldwide LHC Computing Grid with the Tier-1 and/or Tier-2 Computing Centre (including federations of such Institutions with computer centres that together form the Tier-1 or Tier-2 Center) [11]. The document describes in detail the needs of WLCG and the responsibilities of each participating party. The summary of LHC experiments requirements and pledges of computing centres of WLCG for 2009-2012 is given in Table 1 [12]; the necessary bandwidth of networking for data transfer from CERN to Tier-1 centres was estimated to be greater than 1.6 GB/s.

In preparation for LHC data gathering, in 2009 WLCG ran an intensive test of the whole grid system, called STEP09 (Scaling Testing for the Experimental Program 2009), loading the storage, processors and the networks with Monte Carlo data of the rates and volumes expected at LHC. The results were very good [14]; still, the seminar on the first results from LHC, taking place at CERN on 18 December 2009, surpassed the expectations [15]. The excellent performance continues and the LHC experiments are delivering a lot of interesting results.

2 Polish Distributed Tier-2

2.1 General Schema

Five Polish institutions were involved in the CrossGrid project: ACC Cyfronet AGH (the coordinator) and IFJ PAN from Kraków, ICM and IPJ from Warsaw, and PSNC from Poznań. In the years 2004-2010 three computing centres participated in three generations of the EGEE projects – this way Polish computer experts and physicists familiarized themselves with the grid technology. At the time of the CrossGrid project the three Polish computer centres were supported financially by the Polish State Committee for Scientific Research, which allowed for purchasing new processors and disks that became a part of the project grid testbed, combining computing resources of 16 institutions.

Polish physicists also participated in the work of the Grid Deployment Board (GDB), right from the first meeting in 2002, and later in the Collaboration

Table 1. The summary of LHC experiments, requirements and pledges of computing centres of WLCG for 2009-2012. HEP-SPEC06 used in the table is the new HEP-wide benchmark for measuring CPU performance. For example a typical unit based on two Intel Xeon L5420 (8 cores) at 2.5 GHz, with 16 GB memory and running Scientific Linux 5 x86_64 shows performance of about 70 HEP-SPEC06 [13].

Summary Ext. Tier1s		SUM 2009	SUM 2010	Sum 2011	Sum 2012
CPU (HEP-SPEC06)	Offered	245800	406164	518668	613048
	Required	217300	4974000	608920	767240
	Balance	13%	-18%	-15%	-20%
Disk (Tbytes)	Offered	34890	60336	79682	94377
	Required	37400	65100	114610	152396
	Balance	-7%	-7%	-18%	-24%
Tape (Tbytes)	Offered	40189	65938	89805	115423
	Required	29000	50900	114610	152396
	Balance	39%	30%	-22%	-24%
Summary Tier2s		SUM 2009	SUM 2010	Sum 2011	Sum 2012
CPU (HEP-SPEC06)	Offered	305324	475752	607096	746016
	Required	228100	570400	874880	1100040
	Balance	34%	-17%	-31%	-32%
Disk (Tbytes)	Offered	22847	35221	45196	58571
	Required	22720	48520	47903	61963
	Balance	1%	-27%	-6%	-5%

Board of WLCG, which provided direct information about the formation and operation of the LHC Computing Grid. The ACC Cyfronet AGH was among 14 computing centres on three continents which in 2003 created the first worldwide LCG testbed for physics [16]. This testbed demonstrated the viability of the grid concept for physics use (running "around the world, around the clock").

In 2005, following a request of GDB, the three Polish computing centres formed "distributed Tier-2" – this meant connecting to the Tier-1 at FZK Karlsruhe via dedicated links (provided by PIONIER and DFN) and delivering an agreed amount of resources (processors and disk storage) for the needs of LHC experiments (see Fig. 1). The original middleware was that of Data Grid (EDG), which was later upgraded to gLite [9]. The sites had to adopt LCG standards of high availability and reliability. The final qualification ("validation") of their performance was done by running test programs of LHC experiments and comparing their outputs with the reference results.

In 2007 Poland signed the Memorandum of Understanding for Collaboration in the Deployment and Exploitation of the Worldwide LHC Computing Grid [11]. The current pledges of Polish Tier-2 are presented in Table 2 [18].

2.2 Operations

The WLCG is continuously monitoring the performance of data transfer of all LHC experiments to remote sites, including live visualisation of active sites using

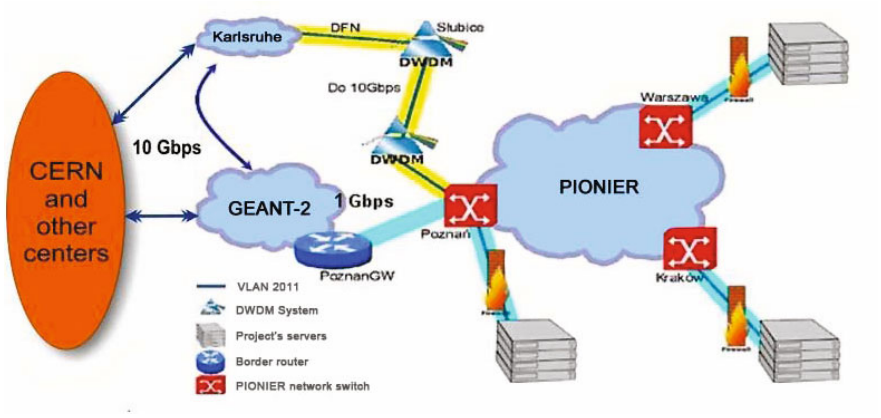


Fig. 1. Connections between the three computing centres, ACC Cyfronet AGH Kraków, ICM Warsaw and PSNC Poznań – constituting the Polish Tier-2, the FZK Karlsruhe Tier-1 and CERN [17]

Table 2. The current pledges of Polish distributed Tier-2

Poland, Polish Tier-2 Federation	2010	2011	2012
CPU (HEP-SPEC06)	10540	13050	15800
Disk (Tbytes)	599	810	1020

Real Time Monitor [7]. In addition, regular reports on the availability/reliability of all their resources as well as accounting statistics are available and distributed monthly [7]. The MoU requires the reliability and availability of Tier-2s to be above 95%, which has been now fulfilled by the majority of sites, including the Polish ones (some drops below 95% are due to new installations and/or refurbishing of the computing centres). In the future implementing service level management in PL-Grid structure [32] will help in maintaining high level of reliability of Tier-2 services provided in Poland.

3 Computing Models of LHC Experiments and the Use of Polish Tier-2

A distinctive feature of particle physics experiments is data organized in the so-called events containing information read out from the detector for a single beam-beam collision trigger. Due to high frequency of beam crossings, LHC provides enormous rates of events, nominally 40 million per second. Since the interesting events occur with very low probability, an online filtering system is used to select them to be written on permanent storage. Even then the amount of the collected data is of the order of many petabytes per year per experiment. These huge samples in initial RAW data format have to be processed and further reduced in many subsequent steps before any physics result can be derived.

Fortunately, each event can be analysed separately and this allows for parallellizing the process. In WLCG users submit separate jobs to machines in different clusters using grid tools. Due to the scale of operations, all LHC experiments developed their own tools and models of data processing on top of WLCG middleware. All four LHC experiments adopted the hierarchical MONARC model with some experiment specific customizations; they will be later described in more detail. The common feature is the use of pilot jobs to increase the efficiency of job execution by running pre-checks of the local environment before starting real production jobs. This procedure worked well in 2010, with the experiments being able to fill available resources. The largest differences between the experiments could be originally found in the area of data access and distribution. The experience gained in the past year leads experiments to plan a very similar way of dynamic data access making the maximum use of the network connectivity.

3.1 The ALICE Experiment

The ALICE computing model is described in detail in the ALICE Technical Design Report of the Computing [19]. It assumes the existence of functional Grid middleware allowing for efficient, seamless and democratic access to worldwide distributed heterogeneous computing and storage resources. Since 2001 ALICE has developed a set of middleware services called AliEn [20], which implements the above model. In the resulting architecture, the user interacts with the Grid via the AliEn User Interface and the services offered by a combination of AliEn middleware, providing a high-level of ALICE-specific services. In addition, the middleware installed at a computing centre provides basic services. The system has been in continuous operation since 2001 with periodic large scale exercises called "data challenges" performed to test its evolution and scalability. The AliEn system is built around Open Source components. It uses Web Services model and standard network protocols. The ALICE computing model should be considered in a twofold way. As for proton-proton interaction, it is similar to other experiments: i) quasi-online data distribution and first reconstruction at Tier-0, ii) further reconstruction at Tier-1s. As for AA collisions, producing much more data, the model is different: i) calibration, alignment, pilot reconstruction and partial data export during data gathering, ii) data distribution and the first reconstruction at Tier-0, iii) further reconstructions at Tier-1s.
Summarizing:

Tier-0: First pass reconstruction, storage of RAW, calibration and first pass of ESD (Electronic Summary Data),

Tier-1: Subsequent reconstruction and scheduled analysis, storage of a collective copy of RAW data,

Tier-2: Simulation and end-user analysis, disk replicas of ESD and AOD (Analysis Object Data).

Polish Tier-2 Sites in ALICE. Two Polish sites participate in the ALICE Grid – ACC Cyfronet AGH in Kraków and PSNC in Poznań. They provide 2370 and 1050 cores respectively, which corresponds to 4.1% of total resources. According to the ALICE computing model, their main role is the Monte Carlo data production. The RAW format is exactly the same for the simulated data and the data coming from the experiment. Thus, the same analysis approach is adopted in both cases. In Fig. 2. the number of ALICE jobs executed in both centres in the period of Jun 2010 – July 2011 are presented; Polish sites provided about 4.1% of total resources.

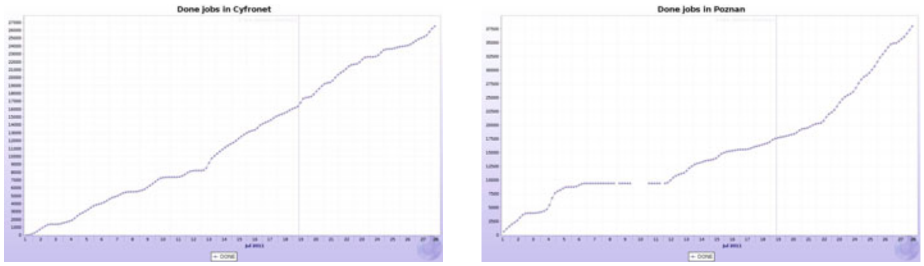


Fig. 2. The number of ALICE jobs executed in ACC Cyfronet AGH (left, maximum at 27 000) and PSNC (right, maximum at 37 500) in July 2011

3.2 The ATLAS Experiment

ATLAS is a general purpose experiment aiming at a direct discovery of new phenomena in proton-proton and heavy ion collisions at the highest energies explored so far [21]. In 2010 the LHC accelerator operated at the centre of mass energy of 7 TeV for proton-proton collisions and 2.76 TeV for Pb-Pb (lead ion) collisions. In both runs the efficiency of the accelerator exceeded expectations for this first year of operations and allowed for recording large samples of proton-proton and Pb-Pb collisions.

The collected RAW data consisting of information from the ATLAS detector have to be later processed by reconstruction software and further reduced in specific analysis steps before any physics results can be derived. The ATLAS computing model [22] relies on WLCG Grid infrastructure supplemented with dedicated tools and services to enhance and customize its functionality. In 2010 computing was still organized in the original MONARC model of the hierarchy of Tier-0,1,2,3 sites, each having a specific set of tasks. The Tier-0 at CERN was used for RAW data acquisition, first processing and data distribution to the Tier-1 sites. The Tier-1 sites cooperated each with its own cloud of the Tier-2 sites, distributing data among them for further processing and integrating output results from simulations to common datasets. Their main task, however, was to run large scale RAW data reprocessing and physics group analyses using the latest versions of experiment specific software. Monte Carlo production and analyses of individual ATLAS members were performed at the Tier-2 sites. For each type of processing the data were pre-placed by Distributed Data Management (DDM) services according to the ATLAS policy and the jobs were sent to sites where the input data were available. The DDM architecture [23] is implemented with DQ2 tools and other services which distribute files using Grid File Transfer Service (FTS), registering them in LCG File Catalog at the file level and in ATLAS specific catalogues at the level of datasets. The job distribution is done by the workload management services implemented in PanDA software [24], using pilot jobs to increase the efficiency of job execution by running pre-checks of the local environment before starting real production jobs. The pilot jobs are sent to grid clusters by the so-called Pilot Factories located at Tier-1s or selected Tier-2 sites in the cloud. An additional element needed for running the reconstruction is data containing information on the state of the ATLAS detector at the time the RAW data was recorded, the so-called conditions. This data is stored in central and distributed databases while access to it is provided via Frontier software with Squid caching in order to reduce access time. Currently, most of the critical WLCG services are located at Tier-1 sites while most of the ATLAS-specific ones are located at CERN. There is now a tendency to centralize all services critical for ATLAS at CERN and to use Tier-1 and Tier-2 sites in a more balanced way.

In Poland ATLAS uses computing resources at ACC Cyfronet AGH in Kraków and PSNC in Poznań, having access to 333 TB of disk space and 22666 HEP-06 CPU at ACC Cyfronet AGH and 14 TB disk space and 4496 HEP-06 CPU at PSNC – these resources are mainly provided by the PL-Grid project [25]. Both sites participate regularly in all ATLAS test activities and in regular production. PSNC participates in large scale Monte Carlo generations. The ACC Cyfronet AGH is the main production site for ATLAS in Poland. It runs both central production jobs as well as user analysis jobs which require access to large datasets on local storage. In addition, ACC Cyfronet AGH has been selected to provide special group disk resources for the Heavy Ion group in ATLAS, which involves staff from the IFJ PAN in Kraków. In the first year of LHC operations and after the first Heavy Ion run at the LHC, this space was heavily used storing up to 55 TB of group reduced data and allowing for fast analysis and the first publication submitted in 2010. Another special type of resources available at ACC Cyfronet AGH is 10 TB private disk space for Polish Grid users, members of the ATLAS experiment. This space is not considered as part of the pledges by the ATLAS experiment but is very important for the local members of the experiment since it has the capability to participate in automated data transfers from all other ATLAS resources and is fully available only for local users' needs.

In 2010 the ATLAS experiment made its first measurements of interactions involving jets, W and Z bozons and top-quark. This allowed to validate the theoretical predictions concerning Standard Model processes at the highest energy available. At the end of 2010 ATLAS participated in the run with Pb-Pb collisions. One of the main goals was attempting to create so-called quark-gluon plasma that filled the Universe $\approx 10\ \mu$s after the Big Bang. If the plasma state is produced, one can expect that particle jets produced from hard scattering of partons inside the plasma will be partially absorbed, leading to asymmetry

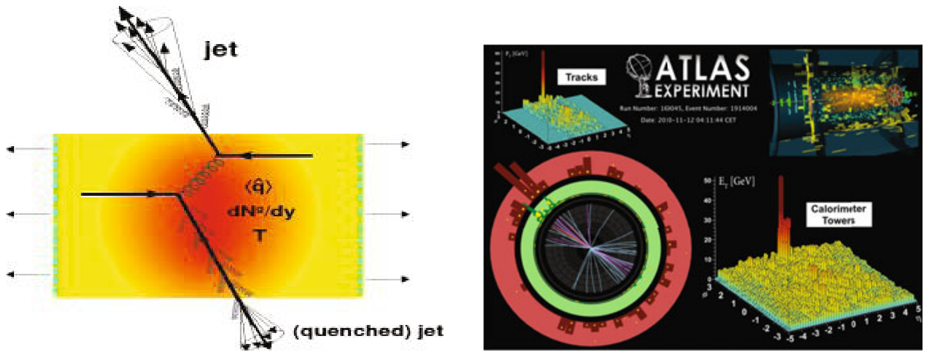


Fig. 3. Schematic view of Pb-Pb collisions – partons cross different paths in the plasma (left) [26], and strong asymmetry of their energy deposited in calorimeters (right) – due to plasma quenching [27]

in the transverse energy distribution of jets. Indeed, such asymmetry has been observed (see Fig. 3) and the results have been published [26]. Overall, in this first year ATLAS published 30 papers and over 150 conference notes.

3.3 The CMS Experiment

To analyse huge amount of data collected by the CMS experiment, a distributed computing model based on the hierarchy of computing tiers is used. In the CMS implementation only about 20% of computing resources are located at CERN and 80% outside of CERN – out of those, half at Tier-1 centres and half at Tier-2 centres. Currently, about 140000 CPUs in total are available for CMS. According to the CMS computing model, Tier-0 and Tier-1 centres serve as resources for the whole experiment and are dedicated to primary reconstruction, re-reconstruction, data and simulation archiving, data and simulation serving and data skimming (i.e. data reduction). Tier-2 centres serve as places where more end-user activities occur and are primarily dedicated to Monte Carlo production and physics analyses. According to computing centre specifications a nominal Tier2 centre should be equipped with CPUs giving 4 kHEP-SPEC06 each, 200 TB storage space and 1-10 Gb/s WAN Internet connection. The whole CMS software is similar in many aspects to that used by other LHC experiments. There are however some specific tools: CRAB (CMS remote analysis builder) [28] and PhEDEx [29] (Physics Experiment Data Export), described in more detail below.

CRAB. The CMS computing model specifies how the data is to be distributed and accessed to enable physicists to run their analyses on the data. The analysis is performed in a distributed way using the Grid infrastructure. In CMS, jobs are submitted using CRAB, which is a tool, designed and developed by the CMS collaboration, that allows the end-user to transparently access distributed data. CRAB knows how to interact with the local user environment, the CMS data management services and with the Grid middleware. It takes care of all aspects of job submission: especially the data and resource discovery. It splits the user's task into several jobs and distributes them to different Grid environments. It performs process tracking and output handling. The end-user does not need to have an in-depth knowledge of the underlying technical details to be able to perform CMS computing tasks. The CRAB tool can be used as a direct interface to the computing system or can delegate the task to a server, which takes care of the job handling. CRAB is able to perform automatic job resubmission in case of failures and notifies the user about the task status. With the same interface, it enables access to local data and batch systems such as load sharing facility (LSF). CRAB has been in production since spring 2004 and it has been extensively used in studies to prepare the Physics Technical Design Report, in the analysis of reconstructed event samples generated during the Computing and Analysis Challenges and in real data fig1..

Currently, there are more than 400 unique CMS users submitting CRAB jobs per week. The CMS Computing Technical Design Report estimated roughly 100k grid jobs per day. During the second half of 2010 the job submissions routinely exceeded this estimate by more than 40% and CRAB turned out to perform very well. The operations team supervises the interaction with users, tracks problems, submits problem tickets and operates the CRAB servers.

Data Management. Data management for the CMS experiment is performed by PhEDEx – a very sophisticated transfer management system designed to handle very large scale transfers. PhEDEx has been designed to ensure data safety, perform large-scale data replication, tape migration and staging of data. The system has reached a very stable status and helps to handle data transfer management tasks with minimum operator effort. The main strong points of PhEDEx are:

- Decoupling between the core workflow agents and the task agents interacting with the baseline transfer and storage services offered by the EGI infrastructure such as Storage Resource Management (SRM) and FTS. This way technology-specific transfer agents are strongly integrated with the baseline services, while at the same time the impact of technology evolution on the core agents is minimized.
- Easy development of a generic framework to handle local interaction with storage, with plug-ins for storage-specific implementations. Plug-ins for the most popular storage solutions used in EGI are provided (CASTOR, dCache, DPM, SRM, posix).
- Development of agents lessens the operational load needed to maintain a local PhEDEx installation, by monitoring the agent state, sending automated notifications and taking automatic actions such as agent restart.
- Deployment of a web data-service to retrieve monitoring information and place new requests into the system. This allows for the development of external tools intended to automate operations with little impact on the core components.

The agent structure of PhEDEx turned out to be extremely powerful, making it very easy to distribute agent-workload over many machines. The system has reached a very stable and mature status and has been proved to scale to data volumes beyond the level required for LHC data gathering.

CMS Computing in Poland. The CMS Computing infrastructure in Poland has been built first of all at ICM in Warsaw using resources dedicated to WLCG (Worldwide LHC Computing Grid) provided by grid projects since March 2002 (CrossGrid [6], EGEE [8], PL-Grid [26]). It serves the purpose of performing central computing tasks of the CMS experiment (e.g. Monte Carlo production) as well as providing Polish CMS community with computing resources needed to perform physics analyses. The storage for the CMS Experiment at Warsaw Tier-2 site is based on the Disk Pool Manager system that implements SRM protocol. The CMS VOBOX machine that provides services needed by the CMS Computing model (PhEDEx and FroNTier) has been set up and operates at the Warsaw Tier-2 site. Required releases of the CMSSW (CMS experiment's software) are automatically installed at the site. Last year the T2_PL_Warsaw site located at ICM contributed to about 0.6% of all events processed by all Tier-2 CMS sites (Fig. 4).

ICM also provides CMS experts with the PL-Grid ticketing system. Apart from the Tier-2 site operated at ICM, the Faculty of Physics of the University of Warsaw and IPJ operate local Tier-3 cluster consisting of about 100 CPUs and providing about 50TB of disk space. The cluster is used mainly for final interactive physics analyses.

3.4 The LHCb Experiment

The configuration of the LHCb computing centres follows the standard multi-Tier WLCG structure. The CERN site serves simultaneously as Tier-0 and Tier-1. There are six other national Tier-1 centres: CNAF (Italy), FZK (Germany), IN2P3 (France), NIKHEF (Netherlands), PIC (Spain) and RAL (United Kingdom). A number of regional Tier-2 sites are located in the countries involved in the LHCb collaboration. The details of LHCb computing model are described in [30]. The specific feature of LHCb configuration is that the majority of data processing occurs in Tier-1 centres where the data reside, while Monte Carlo generation is performed at Tier-2s (the produced Monte Carlo data is uploaded to the associated Tier-1). In this way all input data for various processing steps is concentrated in the limited number of large centres, thus allowing efficient resources usage for both centrally managed productions and random submission of user analysis jobs. One can distinguish three main activities. Two centrally managed tasks concern the off-line data processing and Monte Carlo production. The datasets produced by these central tasks are used for the third-party activity, the end-user analysis.

Off-line Data Processing. The RAW data produced by the experiment is transferred to the CERN Tier-0 for further processing and archiving. A copy of

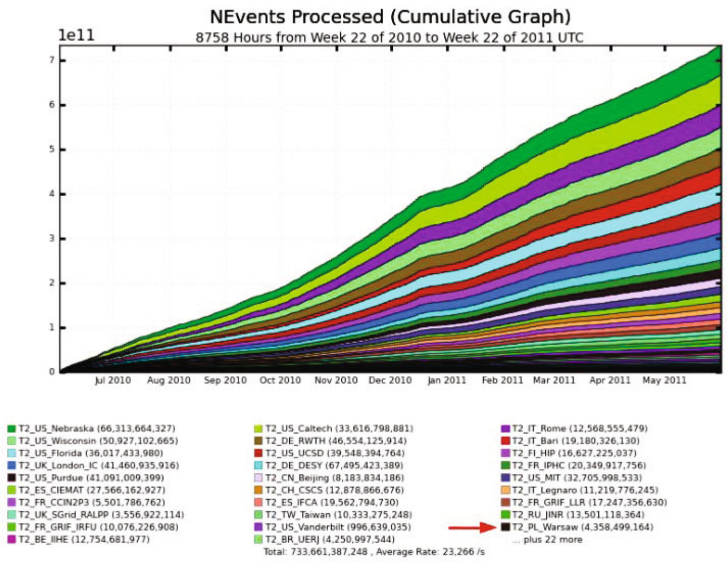


Fig. 4. The number of events processed by CMS Tier-2 sites in March 2010 – May 2011. The ICM site is represented by T2_PL_Warsaw. The plot was generated using CMS Dashboard – the CMS accounting service.

RAW data is distributed among Tier-1s. Each site receives a share of the data based on the percentage of CPU pledged. There are several phases in the processing of event data. The various stages normally follow each other in a sequential manner, but some stages may be repeated a number of times. Each phase produces data in a new format or reduces the amount of data by selecting events. The first step is the event reconstruction. It results in the generation of new data, the Data Summary Tape (DST). At this stage the amount of data is huge and has to be reduced before running the end-user analysis jobs. This reduction step is called data stripping. The minimum information is written out during reconstruction just to allow the physics pre-selection algorithms to be run at a later stage. This is known as a reduced DST (rDST). The rDST information is analysed and the events that pass the selection criteria are fully re-reconstructed, recreating the full information associated with the event. In addition, an event tag collection is created for faster reference to selected events. It contains a brief summary of each event as well as the results of the pre-selection algorithms and a reference to the actual DST record. Next this reduced dataset from stripping pass is made available for the user analysis. The event reconstruction makes use of calibration and alignment constants to correct any temporal changes in the response of the detector and its electronics and in its movement. The reconstruction step will be repeated to accommodate improvements in the algorithms and also to make use of improved determinations of the calibration and alignment of the detector in order to regenerate new improved rDST information. The improvements, when available, are applied to the new data coming from the detector. It is foreseen to repeat the full processing chain for the whole data sample collected over a year.

Monte Carlo production. The understanding of detector effects is required to derive any physics results and determine its uncertainty. It is achieved by the simulation of physics data involving a number of steps to produce simulated RAW data. The format of simulated RAW data is exactly the same as for the data coming from the real detector. Therefore simulated data can be processed in exactly the same way as described for the off-line data processing. The Monte Carlo production is expected to be an ongoing activity throughout the year.

End-user analysis. Access to the data after the stripping step is granted to all collaborating users. The data analysis is performed in a batch mode. Usually, the users are organized in groups performing similar studies on shared data. A typical analysis starts from the stripped DSTs and further reduces the sample by narrowing event selection or by scaling down the data size like in the case of Ntuple format. Then individual physicists may run its analyses programs on reduced data many times to obtain the best results for publication in a scientific journal.

The DIRAC system (Distributed Infrastructure with Remote Agents' Control). The DIRAC project (see [31] and references therein) provides a complete set of tools to support all LHCb data processing needs: off-line data processing, Monte Carlo production and real data distribution to the final end-user analysis tasks. It integrates a coherent system of resources and grid services to carry out its computing tasks in the distributed environment. The main DIRAC functionality is provided by the Workload Management System. It was based from the very beginning on the "Pull" paradigm coupled with the idea of the Pilot Jobs. The advantages of this type of job scheduling are now well demonstrated and all four LHC experiments have applied this schema with somewhat varying details. This mechanism enabled experiments to significantly decrease the rate of job failures.

The **Polish Tier-2 sites** providing resources for the LHCb are located in Kraków (ACC Cyfronet AGH) and Warsaw (ICM). According to the Tier-2 role in the LHCb computing model, the main activity of the Polish centres is the production of Monte Carlo data. The produced DSTs are transferred to Tier-1 in FZK (Germany). For many years the Polish sites participated in the computing tests like data challenges (DC06), scaling test (STEP09) and all Monte Carlo productions. The average contribution of Polish sites' resources was at the level of a few percent as expected by the load share within the collaborating group. The CPU usage broken down by the countries since the beginning of 2011 is shown in Fig. 5; the Polish contribution amounts to 3.9% of the total usage.

The final physics results are mostly extracted on the equipment below the Tier-2 level – on Tier-3 clusters or individual workstations. At IFJ PAN, a Tier-3 cluster with full grid functionality for LHCb and ATLAS virtual organizations was installed in 2006 and was operated in close cooperation with ACC Cyfronet AGH Tier-2. This enabled local users to access, in a convenient way, the resources

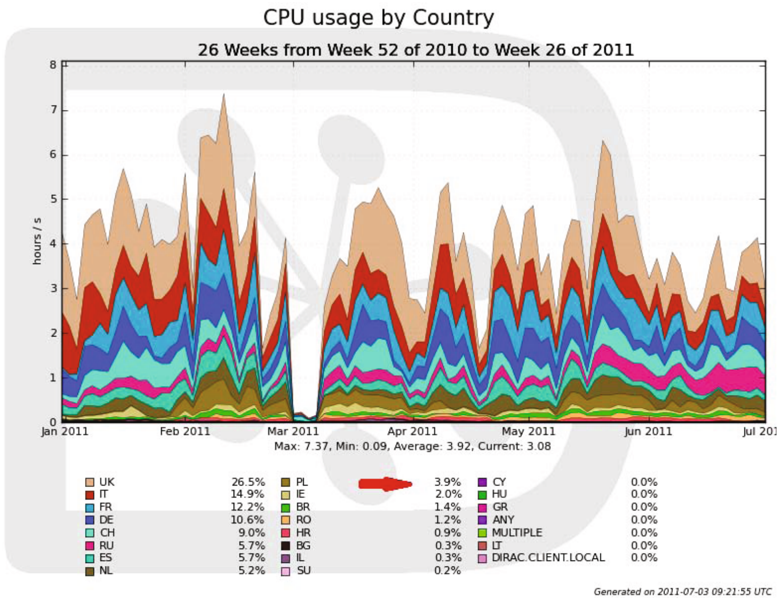


Fig. 5. The LHCb CPU usage per country for the first six months of 2011. The Polish contribution indicated by the red arrow amounts to 3.9% of the total CPU usage of the LHCb collaboration. The plot was prepared by means of the DIRAC accounting service.

of WLCG and, at the same time, also local computing resources (in a traditional way).

4 Summary and Outlook

As stated by the CERN's Director of Research and Computing, S. Bertolucci [15], WLCG shows excellent performance, which allows the LHC experiments to promptly present and publish interesting physics results. The effort has to be continued as the LHC accelerator is steadily increasing its luminosity, which means more collisions and more data to be recorded and analysed to discover New Physics...

Acknowledgments. The three computing centres, ACC Cyfronet AGH in Kraków, ICM UW in Warsaw and PSNC in Poznań, their management and staff, should be thanked for their constructive cooperation in creation and operation of the Polish LHC distributed Tier-2. Supportive approach of the directorates of IFJ PAN Kraków and IPJ Warsaw is also appreciated. The financial contributions of the EU Commission as well as the Polish State Committee for scientific Research and the Polish Ministry of Science and Higher Education (formerly the Ministry of Scientific Research and Information Technology) are acknowledged.

We are also grateful to many people from CERN, the FZK Karlsruhe and other WLCG collaborating institutions for their friendly advice and support.

References

1. ATLAS Letter of Intent CERN/LHCC/92-4; CMS Letter of Intent, CERN/LHCC/92-3; ALICE Letter of Intent CERN/LHCC/94-14; LHCb Letter of Intent CERN/LHCC/95-005
2. Aderholz, M. et al.: MONARC Phase 2 Report, CERN-LCB-2000-001 (March 2000)
3. Large Hadron Collider Committee, Review of Computing Resources for the LHC Experiments, CERN-LHCC-2005-006 (March 2005)
4. Mount, R.: Presentation at the ICFA Digital Divide Workshop, Daegu, Korea (May 2005)
5. Foster, I., Kesselman, C.: The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann, San Francisco (1999)
6. EU Data Grid project, http://eu-datagrid.web.cern.ch/eu-datagrid/; EU CrossGrid project, http://www.cyf-kr.edu.pl/crossgrid/
7. LCG project, http://lcg.web.cern.ch/LCG/
8. EU EGEE project, http://www.eu-egee.org/
9. Light Weight Middleware for Grid Computing, gLite, http://glite.cern.ch/
10. Open Science Grid, http://www.opensciencegrid.org/, Nordic Data Grid Facilities, http://www.ndgf.org/ndgfweb/home.html
11. MoU for Collaboration in the Deployment and Exploitation of the Worldwide LHC Computing Grid, CERN-C-RRB-2005-01
12. Foffano, S.: Presentation at the LHC C-RRB, CERN-RRB-2009-45 (April 2009)
13. http://w3.hepix.org/benchmarks/doku.php?id=bench:results_sl5_x86_64_gcc_432
14. Bird, I.: Presentation at the LHC C-RRB, CERN-RRB-2009-126 (October 2009)
15. Interview with Sergio Bertolucci, CERN Director for Research and Computing, CERN-MOVIE-2009-160, December 18 (2009)
16. Mickel, K.-P.: Presentation at the Cracow Grid Workshop, Kraków, Poland (October 2003)
17. Olszewski, A.: Presentation at the ICFA Digital Divide Workshop, Kraków, Poland (October 2006)
18. MoU for Collaboration in the Deployment and Exploitation of the Worldwide LHC Computing Grid, CERN-C-RRB-2005-01 /Rev.1 (March 29, 2011)
19. ALICE Collaboration: Technical Design Report of the Computing, CERN LHCC-2005-018
20. Saiz, P., et al.: AliEn – ALICE environment on the GRID, NIM A502, 437 (2003)
21. Aad, G., et al.: The ATLAS Experiment at the CERN Large Hadron Collider. JINST 3, S08003
22. ATLAS Computing Technical Design Report – TDR, CERN-LHCC-2005-022
23. Branco, M., et al.: Managing ATLAS data on a petabyte-scale with DQ2. J. Phys.: Conf. Ser. 119, 062017
24. Maeno, T., et al.: PanDA: distributed production and distributed analysis system for ATLAS. J. Phys.: Conf. Ser. 119, 062036
25. PL-Grid project, http://www.plgrid.pl/
26. d'Enterria, D.: Jet quenching, arXiv:0902.2011v2 [nucl-ex]

27. ATLAS Collaboration: Observation of a centrality-dependent dijet asymmetry in lead-lead collisions at $\sqrt{s}_{NN} = 2.76$ TeV with the ATLAS detector at the LHC, Phys. Rev. Lett. 105, 252303
28. Spiga, D., et al.: CRAB: the CMS distributed analysis tool development and design. Nuclear Physics B – Proceedings Supplements 177-178, 267–268 (2008)
29. Egeland, R.: PhEDEx data service. J. Phys. Conf. Ser. 219, 062010 (2010)
30. Computing TDR, CERN-LHCC-2005-019
31. Tsaregorodtsev, A., et al.: DIRAC3 – the new generation of the LHCb grid software. Journal of Physics: Conference Series 219, 062029 (2010)
32. Szepieniec, T., Tomanek, M., Radecki, M., Szopa, M., Bubak, M.: Implementation of Service Level Management in PL-Grid Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 171–181. Springer, Heidelberg (2012)

PL-Grid e-Infrastructure for the Cherenkov Telescope Array Observatory

Anna Barnacka[1,2], Leszek Bogacz[3], Michał Gochna[4], Mateusz Janiak[1], Nukri Komin[5], Giovanni Lamanna[5], Rafał Moderski[1], and Małgorzata Siudek[6]

[1] Nicolaus Copernicus Astronomical Center, Warsaw, Poland
[2] DSM/IRFU/SPP, CEA/Saclay, Gif-sur-Yvette, France
[3] Jagiellonian University, Faculty of Physics, Astronomy and Applied Computer Science, Kraków, Poland
[4] University of Warsaw, Faculty of Physics, Astronomical Observatory, Warsaw, Poland
[5] LAPP, Université de Savoie et IN2P3/CNRS, Annecy, France
[6] Center for Theoretical Physics, Warsaw, Poland

Abstract. We present results of Monte Carlo simulations for the Cherenkov Telescope Array – the next generation observatory of very high energy gamma rays. These calculations are intended to verify design concept for various elements of the array: single telescope design, camera electronics, array configuration, camera and array trigger, etc. The simulations consist of two parts: simulation of the distribution of Cherenkov photons from atmospheric air showers and calculation of response of the telescopes during the detection process. In design study phase of the experiment a large amount of simulations of air showers has been performed with the use of EGEE grid infrastructure, especially centers in Lyon and ACC Cyfronet AGH. Currently we are investigating the details of design of a small telescope – one of several kinds intended for the array.

Keywords: very high energy astrophysics, Monte Carlo simulations, extensive air showers, telescope design, the Cherenkov Telescope Array.

1 Introduction

Measuring cosmic radiation at very high gamma-ray energies, above 100GeV, is a way to investigate the most energetic processes in the Universe [1,2]. There is no celestial body that is hot enough to produce such radiation in a way of thermal emission. Such energies can be produced in several processes such as collision of highly relativistic particles, coming from shock waves of stellar explosions, with interstellar gas. Energy spectrum of gamma-rays would reproduce spectrum of those particles that caused emission. Such energies may also come from decays of heavy particles such as hypothetical dark matter particles, interacting with magnetic field. Therefore the very high energy (VHE) gamma-ray astronomy provides information necessary to investigate these processes. The catalogue of celestial objects detected in VHE range includes such objects as active galactic nuclei, binary systems and remnants of exploding stars. It is worth noting that the extreme conditions present in such objects cannot be created in experiments on Earth.

Observing the Universe at VHE requires much more than a simple telescope with a CCD camera. Gamma-ray instruments of the latest generation such as H.E.S.S. [3], MAGIC [4] and VERITAS [5] consist of multiple telescopes with analogue cameras based on photomultipliers. This is the only way to observe Cherenkov light from extensive air showers that are produced when a high energy particle enters Earth's atmosphere. These experiments have delivered spectacular astrophysical results. Now, based on the experience of current instruments the community of more than 100 institutes from 22 countries is going to build a new generation of ground-based gamma-ray observatories – the Cherenkov Telescope Array (CTA) [6]. The array will consist of many tens of telescopes whose order of sensitivity will be of a vastly higher level as compared to the instruments currently in use. Analyzing light from many telescopes will enable CTA to precisely determine the energy of a particle at the moment of entering the atmosphere, as well as its precise direction. It is expected to provide major impact in astrophysics, particle physics and cosmology.

Because of the complexity of the instrument a proper preparatory phase is required before the actual construction of the facility. The CTA will operate at least three kinds of telescopes, each one equipped with a high sensitivity digital camera based on photomultipliers. Using many kinds of telescopes is a way to increase the energy spectrum, which will be possible to observe. Cameras will detect even single Cherenkov photons, then a digital trigger will determine if the event it observed is scientifically interesting or not. Sampling of collected data will take a few nanoseconds. Overall performance of the system depends on many parameters such as the configuration of telescopes, the size of a single telescope (aperture, focal length, field of view), the parameters of cameras (size of a single pixel, number of pixels, sampling) and many more. To determine the best configuration at optimal cost values of billions of numerical simulations are required. That means hundreds of TB of storage and years of a single CPU time. During the operational phase CTA will produce at least 3 PB of data per year. All this data will need to be analyzed, archived and shared with the community. Additional massive Monte Carlo simulations will be required to provide calibration of the scientific data. The high data rate of CTA together with the large computing power requirements for the Monte Carlo simulations demand dedicated computer resources which can be handled well using the grid approach. The EGI grid (www.egi.eu) [7] infrastructure and middleware for distributed computing, data storage and access are considered the most efficient solution for CTA e-infrastructure.

2 State-of-the-Art

At the end of XX and the beginning of XXI centuries physics, astronomy, astrophysics, computational chemistry, and earth sciences communities desperately looked for new solutions involving achievements from the field of globalizing computing. A number of Grid projects have been developed (e.g., Globus, Condor, iVDGL, DataGrid, NorduGrid, SAGrid, etc.) and derivative projects, such as Open Science Grid, Grid@Asia, etc. Among the largest projects was a three-generational EU-funded project "Enabling Grids for E-science" (EGEE, `eu-egee.org`) [8]. EGEE has developed an international production quality grid infrastructure for multiple applications in many disciplines such as geophysics, Earth observation, astrophysics and computational chemistry. The PL-Grid national infrastructure [9] is based on the experience gained during participation in various European projects, including EGEE-III.

One of the most famous experiments making use of the grid infrastructure is the Large Hadron Collider (LHC) [10] project. 4 LHC detectors are supported by the worldwide LHC Computing Grid (LCG) – a global collaboration of more than 140 computing centres in 35 countries, and several national and international grid projects [11]. The mission of the LCG project is to build and maintain data storage and analysis infrastructure for the entire high energy physics community that will use the LHC at CERN. The successful collection, distribution, and analysis of data from the four LHC experiments represent a major achievement and demonstration of the grid success.

An example of astronomy projects with high-rate data flow in real time is eVLBI project [12]. e-VLBI technique enables real-time data transfer from remote radio telescopes to the central processing facility via optical fibre cables, as opposed to the "traditional" implementations of VLBI in which the data is first recorded at the telescopes on tapes or discs and then it is physically delivered to the processor. Data processor is replaced by the distributed software correlator spread all over the grid environment. There is also a specially designed application to control and manage observations. The e-VLBI system is in the development phase. Therefore, the concept of performing VLBI experiments in the grid environment has not been verified yet.

In the field of VHE astrophysics CTA is the first project to check the possibility of using grid infrastructure in a large part of its operations. Since now, all VHE gamma-ray observatories operated as closed facilities and used their own dedicated computer resources – usually in the form of computer clusters. The amount of data and required computing power was small enough to be handled by a single institution with support for a limited number of scientists from a closed collaboration group. The traditional approach used in current VHE experiments is not applicable to CTA. The tape-based technology is aging, becoming worse supported and more expensive. On the other hand, telecommunications capacity in Europe has dramatically fallen in price and the dedicated European networks for research and education, having worked together for many years, initiated a shared infrastructure to connect them together with broadband capacities of at least 10 Gb/s. CTA will – for the first time in this field – be operated as a true observatory, open to entire astrophysics and particle physics communities, and providing support for easy access and analysis of data. Service provided to professional astronomers will be suplemented by outreach activities and layman data interfaces. Grid-based infrastructures, such as EGEE/EGI, are considered to provide the CTA observatory with the e-infrastructure which could be used for data management purposes. This is the first project attempting to apply a grid infrastructure to this kind of research. Developing a global integrated approach between optimization of grid access conditions and research network is an important topic present within the CTA design study. CTA is an innovative project which brings together the long-term presence and EGEE experience of the particle physics community with the grid solutions for public distribution of observatory data.

A feasibility study of grid solutions application in CTA is in progress within a dedicated CTA Computing Grid (CTACG) project [13]. CTACG is aimed at optimizing the application of grid technology for the CTA simulations, data processing and storage, off-line analysis and the Virtual Observatory interface through a dedicated global CTA EGI Virtual Organization. It also aims at unifying the computing capacity existing in institutes and organizations members of the CTA consortium within the EGEE/EGI framework, in order to satisfy the computing requirements for Monte Carlo simulations and data management. The main issues inherent to the observatory workflow, which could benefit of grid applications and which concern the CTACG project are: (1) Monte Carlo simulations, (2) data flow, data transfer and storage, (3) data reduction, data analysis and open access. The competitive results have shown that the grid approach is so far the best solution to fulfill all requirements. Intensive numerical computations, large storage availability, easy distributed data access and significant computing resources are provided by grid technologies. Some specific applications are developed to provide a common CTA-dashboard for Monte Carlo data production and analysis devoted to the monitoring of grid jobs as well as to support users in data access and data analysis at different levels. Applications of grid technologies in the context of services for distributed computing resources exploitation for Monte Carlo studies have already been achieved. The IN2P3 French Computing Center (CCIN2P3, `cc.in2p3.fr`) for nuclear, particle and astroparticle physics is one of the largest governmental research computing centers in the world and one of the centers of the LCG project. CCIN2P3 is also one of the six current Core Infrastructure Centers (CIC) providing the oversight for the grid operations, using a variety of monitoring tools as well as direct problem reports to find and analyze operational problems. The CTA Virtual Organization (VO) `vo.cta.in2p3.fr` was created in 2008 by the French LAPP-CTA (`www.lapp.in2p3.fr`) group in cooperation with the CCIN23P. The IN2P3-LAPP laboratory hosts a computing center supporting grid applications and has the role of CTA VO Grid Operations Center (CTA-GOC) taking over the management tasks, being responsible for the grid-operation coordination and the workload management. It has been also involved in several activities first within the European Data Grid Project, then in EGEE/EGI. At LAPP it was possible to benefit from grid middleware and software applications already deployed for LCG and now extended to CTACG. LAPP, through a constantly up-to-date version of the gLite middleware, provides and maintains central resources exploited by the CTA VO such as Workload Management Service (WMS), Logging and Bookkeeping service (LB), User Interfaces (UI), Storage Element (SE), and Computing Element (CE). CCIN2P3 also provides and supports two more critical services for the CTA VO administrations, namely the File Catalog (LFC) and the registration portal Virtual Organization Membersihp Services (VOMS).

The Polish PL-Grid site at ACC Cyfronet AGH is currently one of the largest centers supporting `vo.cta.in2p3.fr`. It provides more than 100 TB of storage space together with more than 200 CPU cores. As from 2011 ACK Cyfronet is also a formal member of the CTA Consortium and conducts research and development of the grid platform for CTA numerical computations [14].

3 Description of the Solution

The CTA project is currently in its preparatory phase. One of the main purposes of this phase is to find the best, optimal layout of the array. Many parameters need to be estimated both for a single telescope (mechanical details, such as construction geometry, mirror layout and also camera details, electronics, etc.) and for the whole array – telescopes layout and spacing. To study performance of the array one needs to perform massive Monte Carlo simulations which consist of several steps. Each of these steps is trying to provide the best possible description of reality starting from physics of high energy particle interactions and ending on analog-to-digital converter (ADC) details and trigger logics.

First and the most time and resources consuming simulations are done with CORSIKA (COsmic Ray SImulations for Kascade) [15,16] – a large numerical code which uses Monte Carlo technique to simulate extensive air showers that originate and develop in higher parts of Earth's atmosphere. Monte Carlo technique is a class of computational algorithms that rely on repeated random sampling of input parameters. It may be used whenever a solution cannot be obtained in a pure, analytic form and must be computed as an approximation converging to exact result as the number of simulations grows. Randomized inputs are being used to perform further deterministic calculations. In case of CORSIKA the parameters being randomized are not only energy and direction of an incident high energy particle, but also some details of particle interactions such as azimuthal direction of emitted Cherenkov photon, the height of the first interaction and type of the target particle. The air showers are triggered by high energy (HE) cosmic rays and gamma photons, which are of our particular interest. HE cosmic rays and photons cannot propagate all the way to the Earth's surface. Due to interactions with the atmosphere they loose their initial energy and give birth to secondary particles which become primary for another interaction. Resulting particle shower consist of a large number of secondary particles (leptons and hadrons) with particular energy and direction. Most of these charged particles travel through the atmosphere at the speed greater than the phase velocity of light in the air, and thus emit Cherenkov radiation – faint, blueish light resulting from air particles returning to the ground state after being polarized by a charged particle. The Cherenkov photons form a cone around the

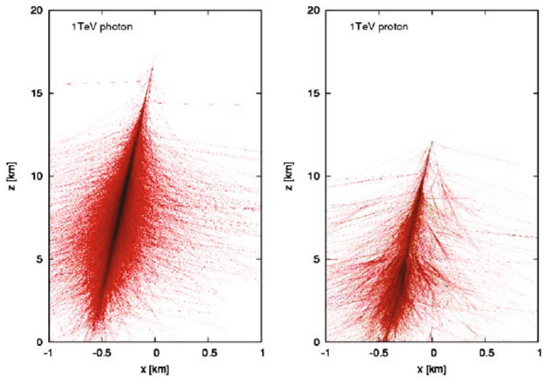


Fig. 1. Two examples of simulated extended air showers developing in the Earth's atmosphere. Each line represents a path of elementary particle created during the shower evolution. Left panel presents the shower generated by a single photon of energy 1TeV, while the right panel shows the shower originating from a single proton of the same energy.

shower axis which is then registered by the array of telescopes. CORSIKA aims to reproduce and simulate all those steps. It uses different models of high and low energy interactions (such as QGSJET and URQMD) to precisely calculate all the shower details, including tracking of position, direction and energy of each particle. Two examples of simulated showers are presented in Fig. 1. The additional package takes care of emission of the Cherenkov light. CORSIKA input is one file with primary particle energy, direction, etc., and some technical details including array layout as telescopes are simulated roughly as 3D spheres. As an output we obtain a file with all the Cherenkov photons available for the telescopes at the ground level and, as an option, precise shower details.

CORSIKA CPU and memory requirements depend, to a great extent, on the type of primary particle and, of course, on desired number of showers to be simulated. While low energy ($\sim 100\,\mathrm{GeV}$) gamma photons simulations with several hundred thousand showers can take just a few days on a single CPU, for high energy events such as 100 TeV this time can go up to a month or even more due to much bigger number of secondary particles and thus interactions to be computed. Also the output file with Cherenkov photons can have sizes about several GB for a single telescope up to several hundreds of GB or even more for an array. For a single task, simulations for several values of gamma energies and several other particle types (such as protons, electrons and muons) need to be performed. It requires large computational resources and storage space. A grid e-infrastructure is therefore ideal for this task. It provides required resources not possible to obtain with a single computer cluster since single analysis task typically requires hundreds of simultaneous CORISKA runs. CORSIKA does not provide built-in parallel computing options and it is just used many times with different sets of parameters.

While the CORSIKA runs are the most time consuming, the second performance estimation step is even more challenging. It is based on a `sim_telarray` software which simulates all the hardware response to Cherenkov photons [18]. The aim of this step is to get a final, raw output as well as the final output of a fully operational array. A graphical example of such output is shown in Fig. 2. `sim_telarray` aims to simulate every step of hardware and software behavior: telescope optics (including full ray-tracing), photomultiplier tubes, ADCs and all the electronics, either analogue or digital, different level of trigger logics and image analysis and reconstruction. The input of `sim_telarray` is CORSIKA output file and a number of various configuration files with all the hardware and software details. Usually one needs to compare the results for a number of different telescope configurations, array layouts, camera sampling, electronic details, trigger algorithms, etc., so a single CORSIKA output is used many times by `sim_telarray` whose runs are much less time and memory consuming. There are two typical ways to get CORSIKA and `sim_telarray` work together, either by using previously saved Cherenkov output as an input for `sim_telarray` or by directly piping Cherenkov photon bunches to `sim_telarray`. In the second mode, telescope response analysis is being done on-the-fly so it saves a lot of storage since CORSIKA output is not stored but used directly by many instances of `sim_telarray`. On the other hand the first solution gives the opportunity to use the same simulated Cherenkov photons many times for different analyses although it requires large storage capabilities, as full simulations for an array with several hundreds of telescopes require TBs of storage. In addition, Cherenkov photons are only stored for defined telescopes positions represented by 3D spheres, so each array layout must be simulated separately. In our approach we switch between those two methods depending on particular task: single telescopes simulations are fast so we might not bother ourselves to store Cherenkov light, while array simulations require lots of CPU time to perform Cherenkov light emission routines, so it is worth to store Cherenkov photons and use them multiple times. The second method also requires careful estimation of future needs in order to prepare the most general-purpose output.

As an output from `sim_telarray` one gets a list of events with details of reconstruction – by comparing reconstructed data with simulated data one is able to obtain parameters or prepare plots to tell the best set of construction parameters. The main performance parameters for an array of telescopes are: sensitivity curve – a minimal detectable flux from a particular region of interest, either point source or a diffuse source, angular and energy resolution – key parameters responsible for the precision of locating a source and energy determination, effective area – area from which Cherenkov photons can be collected, point spread function (PSF) – a measure of telescope optics performance and background rate – number of unwanted background amongst gamma rays. Analysis of `sim_telarray` output is done using `read_tel` software which attempts to reconstruct events seen by the telescope by applying various image analysis methods, image cleaning, etc. Analysis can also give information for different groups responsible for various parts of telescope design – for example the total

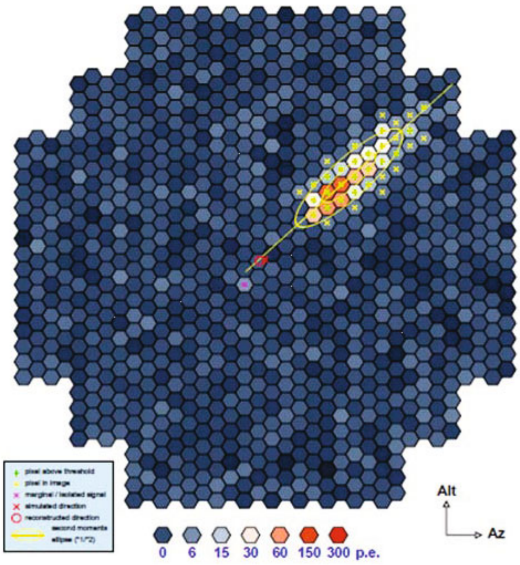


Fig. 2. Example output from `sim_telarray` showing telescope camera triggered by a single incident gamma ray. All pixels are excited due to night sky background

background rate from cosmic rays and night sky background (NSB) can provide crucial information for hardware electronics design.

CTA array will consist of three main telescope types: large, medium and small size telescopes (LST, MST and SST). We focus on a small size telescopes optimization. SSTs are designed to operate in the highest energy band, up to hundreds of TeVs, and to provide good angular resolution to study diffuse sources.

Monte Carlo simulations can not only help with the construction geometry, electronics, optics design, etc., but are also, or maybe mainly, used for verifying trigger strategies. Ground-based imaging Cherenkov telescopes are subjected to many sources of light – amongst them gamma rays are just a small fraction. Main source of collected light comes from NSB – either zodiacal light, moon, stars or light pollution from cities and cosmic rays. It is not possible to process all the data so the set of algorithms to trigger just gamma rays must be introduced. There are different levels of trigger algorithms and each of them aims at reducing background while keeping as much gamma photons as possible. Having the best telescope construction and array layout one still needs the best trigger conditions. This requires additional massive simulations with different trigger algorithms.

Another very important optimization criterion which has to be considered is the telescope cost; the most important part of it is the camera cost which is about 50% of total budget. Camera consists of a large number of photomultipliers (pixels), and thus, pixel number and its size are the most crucial parameters in telescope design. Pixel size has not only impact on cost, but also on the whole telescope performance. It defines the telescope's focal length, mirror and dish size and field-of-view.

To sum up, the Monte Carlo simulations are a very complicated task and require large resources (CPU time and storage) which can be provided by the PL-Grid e-infrastructure. Lots of parameters to be optimized require preparing hundreds of configuration scripts for CORSIKA and `sim_telarray`, many job submissions and then careful analysis.

4 Results

The first part (Prod-1 phase) of the massive Monte Carlo simulations for the CTA project started in 2008 by use of resources available for virtual organization `vo.cta.in2p3.fr`. Its main goal was to simulate showers initiated by cosmic rays. These are mainly charged particles ($\sim 99.9\%$), like protons ($\sim 89\%$), α-particles ($\sim 10\%$), ionized nuclei of heavy elements ($\sim 1\%$), electron/positrons ($\sim 1\%$) and a small fraction of γ-rays. The expected CTA ability to distinguish between gamma and hadron showers is by a significant amount better than in any current operated instrument. Therefore there is a need to simulate a huge number of showers ($\sim 10^{10}$ protons) before estimating the remaining background and drawing conclusions on the performance of a particular configuration. As it was described in the previous sections, the simulations are carried by CORSIKA software compiled with a special option to trace the emitted Cherenkov photons. As the input one chooses the primary particle type, the interaction model and the set of physical parameters of the telescope array, such as e.g. the altitude, positions and fiducial radius of individual telescopes. During the Prod-1 phase the generic configuration of 275 telescopes of 5 different types has been considered (see Fig. 3). A candidate configurations being a subset of this (typically about 80 telescopes of total estimated cost of approximately 80 MEUR), marked by letters A-K are later chosen for detailed analysis.

As the result of the CORSIKA simulation information about the showers, which theoretically could be detected by at least one telescope, is stored. Till now (June 2011) there were 7.9×10^3 runs done including 7.9×10^8 showers. From this amount 2.3×10^7 were triggered by at least one telescope. It is worth mentioning that the simulations are highly resource demanding – for a typical run one needs at least 4GB of RAM and over 5GB of storage. It turned out to be difficult to fulfill on some nodes, resulting in interruptions during high energy cascade calculations. Those affected simulations were rejected due to not introducing a systematic error. Even after that the amount of data currently available for further analysis exceeds 100 TB.

The produced data is used to evaluate the individual configurations of the telescopes with the `sim_telarray` software. The results are saved in `eventio` format [19]. Comparison between the picture seen by the telescopes with the information about the primary particle direction and energy allows to estimate important parameters of the telescope array, such as e.g. flux sensitivity or angular resolution. Presented here integral flux sensitivity is the minimum flux of γ-ray events per unit of time and unit of area, which, in a given observation time, results in a statistically significant excess above the background of cosmic-ray

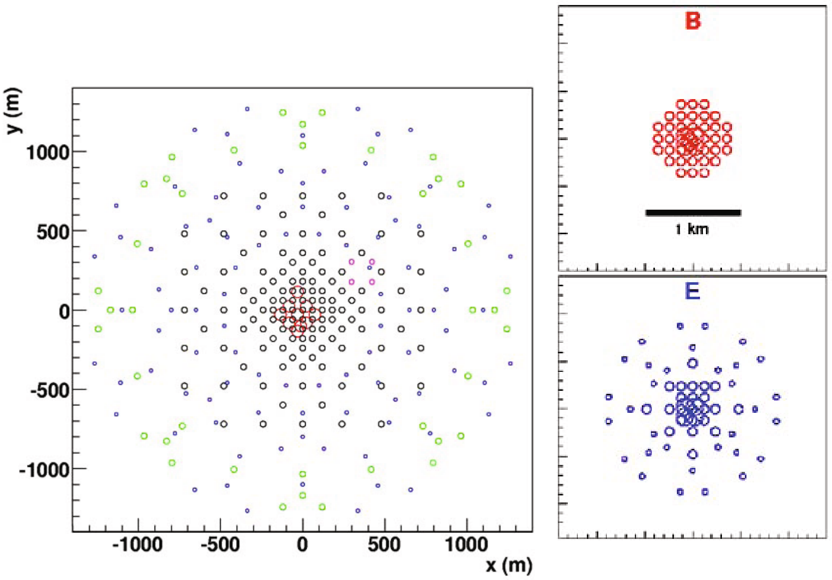


Fig. 3. The generic configuration of 275 telescopes for Prod-1 phase and two example candidate configurations B and E. Taken from [17].

initiated showers above a certain energy level. Example results obtained for tree configurations B, C, and E are shown in Figs 4 and 5.

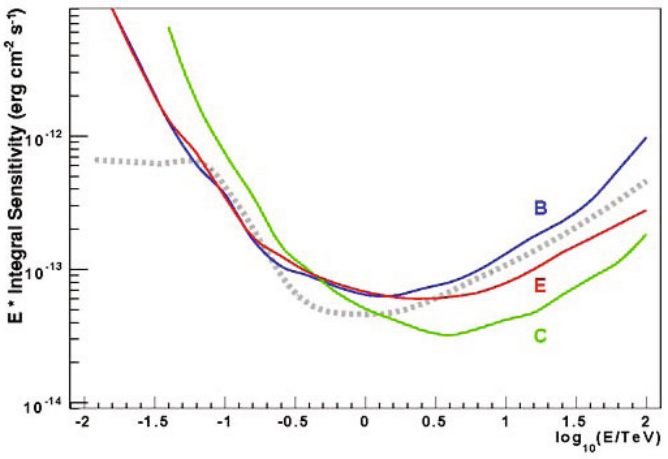


Fig. 4. Integral sensitivity (multiplied by energy) as a function of energy, E, for the candidate configurations B, C and E, for point sources observed for 50 hours at a zenith angle of 20°. The goal curve for CTA (dashed line) is shown for comparison. Taken from [17].

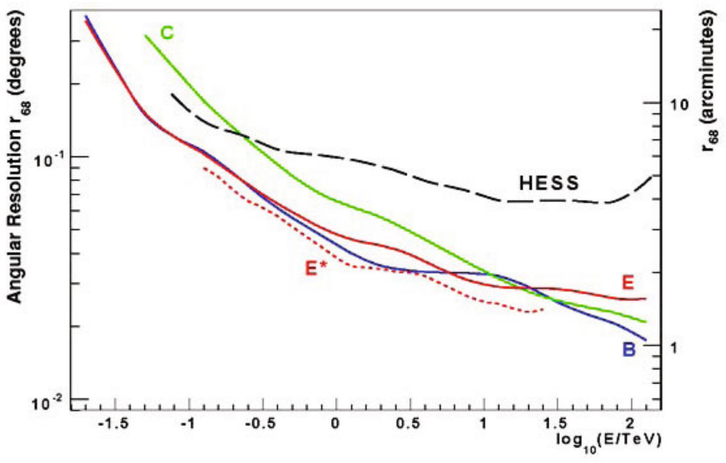


Fig. 5. Angular resolution (68% containment radius of the gamma-ray point spread function) versus energy for the candidate configurations B, C and E. The resolution for a more sophisticated shower axis reconstruction method for configuration E is shown for comparison (dashed red line – E*). The angular resolution of H.E.S.S. is shown as a reference. Taken from [17].

In the next stage of MC simulations (Prod-2 phase) which is about to start, it is planed to perform more detailed analysis of the most promising configurations from the previous run. The main effort at the moment is to put into definition the most realistic parameters of the telescopes which are going to be considered for production, and to modify the software to allow for testing of more sophisticated trigger algorithms.

5 Conclusions and Future Work

The performed simulations prove the grid approach to be very promising for the CTA operations. The amount of acquired data was not possible to obtain with more traditional approach based on single computer clusters. The study also allowed for the development of dedicated tool for job submission and storage – `EasiJob` [13]. The PL-Grid resources allowed to perform construction optimalization for the small size telescope within a reasonable time. The results have important implications for decisions to be taken currently, regarding the parameters for prototype telescopes. The next step of the study would be further investigation of the optimal performance including different hardware camera solutions and mirrors for the small size telescope, but with the greatest emphasis on developing new trigger algorithms which will allow detection of the highest gamma ray showers with maximal efficiency. Also, we will investigate the layout of the array depending on the number of telescopes, their configuration, type and overall performance. Further studies will explore the possibility to use grid infrastructures for international data transfer and storage as well as data access, processing and analysis.

References

1. Aharonian, F., Buckley, J., Kifune, T., Sinnis, G.: High energy astrophysics with ground-based gamma ray detectors. Reports on Progress in Physics 71, 096901 (2008)
2. Hinton, J.A., Hofmann, W.: Teraelectronvolt Astronomy. Annual Review of Astronomy and Astrophysics 47, 523 (2009)
3. Vasileiadis, G.: The H.E.S.S experimental project. In: Proceedings of the Fifth International Workshop on Ring Imaging Detectors. Nucl. Instr. and Methods in Phys. Research, vol. 553, p. 268 (2005)
4. Ferenc, D.: The MAGIC gamma-ray observatory. In: Proceedings of the Fifth International Workshop on Ring Imaging Detectors. Nucl. Instr. and Methods in Phys. Research, vol. 553, p. 274 (2005)
5. LeBohec, S., et al.: Deployment of the VERITAS observatory. J. Phys.: Conf. Ser. 47, 232 (2006)
6. Glicenstein, J.-F.: The Cherenkov telescope array, an advanced facility for ground based gamma-ray astronomy. In: Proceedings of the Seventh International Workshop on Ring Imaging Cherenkov Detectors. Nucl. Instr. and Methods in Phys. Research, vol. 639, p. 46 (2011)
7. Ghiselli, A., Mazzucato, M.: Grid: From EGEE to EGI and from INFN-GRID to IGI. Il Nuovo Cimento 32, 233 (2009)
8. Gagliardi, F.: The European Grid Infrastructure EGEE Project. In: Astronomical Data Analysis Software and Systems (ADASS) XIII, vol. 314, p. 357 (2004)
9. Kitowski, J., Dutka, Ł.: Status and Current Achievements of PL-Grid Project. In: CGW 2009 Proceedings, p. 295 (2009)
10. Engelen, J.: The Large Hadron Collider Project. In: Proceedings of the EPS-13 Conference "Beyond Einstein – Physics for the 21st Century", vol. 22 (2005)
11. Binczewski, A., Bluj, M., Cyz, A., Dwużnik, M., Filocha, M., Flis, Ł., Gokieli, R., Iwaszkiewicz, J., Kowalski, M., Lasoń, P., Lichwała, R., Łopuszyński, M., Magryś, M., Malecki, P., Meyer, N., Nawrocki, K., Olszewski, A., Oziębło, A., Padée, A., Pałka, H., Pospieszny, M., Radecki, M., Rowicki, R., Stojda, D., Stolarek, M., Szepieniec, T., Szymocha, T., Turała, M., Wawrzyniak, K., Wiślicki, W., Witek, M., Wolniewicz, P.: Polish Contribution to the Worldwide LHC Computing. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 285–300. Springer, Heidelberg (2012)
12. Garrington, S.: e-MERLIN and e-VLBI. In: Future Directions in High Resolution Astronomy: The 10th Anniversary of the VLBA, ASP Conference Proceeding, vol. 340, p. 595 (2005)
13. Komin, N., et al.: CTACG – The Cherenkov Telescope Array Computing Grid. In: EGI User Forum (2011), http://www.egi.eu/indico/conferenceDisplay.py?confId=207

14. Ciepiela, E., Nowakowski, P., Kocot, J., Harężlak, D., Gubała, T., Meizner, J., Kasztelnik, M., Bartyński, T., Malawski, M., Bubak, M.: Managing Entire Lifecycles of e-Science Applications in the GridSpace2 Virtual Laboratory – from Motivation through Idea to Operable Web-Accessible Environment Built on Top of PL-Grid e-Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 228–239. Springer, Heidelberg (2012)
15. Capdevielee, J.N., et al.: Extensive air shower simulations with the CORSIKA program. Very High Energy Cosmic-Ray Interactions 276, 545 (1993)
16. Heck, D.: Hadronic Interaction Models and the Air Shower Simulation Program CORSIKA. In: International Cosmic Ray Conference, vol. 1, p. 233 (2001)
17. The CTA Consortium: Design Concepts for the Cherenkov Telescope Array (2010) arXiv:1008.3703v2
18. Bernloehr, K.: CTA simulations with CORSIKA/sim_telarray. In: High Energy Gamma-Ray Astronomy: Proceedings of the 4th International Meeting on High Energy Gamma-Ray Astronomy, AIP Conference Proceedings, vol. 1085, p. 874 (2008)
19. Bernloehr, K.: Eventio – a machine-independent hierarchical data format and its programming interface. CORSIKA Documentation (2005), http://www-ik.fzk.de/corsika/

Training in the PL-Grid as Key Component to Attract Users to Grid e-Infrastructures

Marcelina Borcz[1,2], Krzysztof Benedyczak[1,2], Adam Padée[1], Rafał Kluszczyński[1], Grzegorz Marczak[1,2], Mirosław Zdybek[3], Maciej Pawlik[3], Maciej Filocha[1], and Piotr Bała[1,2]

[1] University of Warsaw,
Interdisciplinary Centre for Mathematical and Computational Modelling,
Pawińskiego 5a, 02-106 Warsaw, Poland
[2] Nicolaus Copernicus University,
Department of Mathematics and Computer Science,
Chopina 12/18, 87-100 Toruń, Poland
[3] AGH University of Science and Technology, ACC Cyfronet AGH,
Nawojki 11, 30-950 Kraków, Poland
bala@icm.edu.pl

Abstract. This chapter provides an overview of the training facilities in the PL-Grid project. It is well known that procedures necessary to obtain certificates and then install software or get access to the User Interfaces are complicated and due to security reasons cannot be simplified. For these purposes we have crated extended training facilities which allow to get easy access to the Grid. We have installed dedicated infrastructure including the computational resources, certificate authority, user portal and an on-line training system. The efficiency of the training infrastructure, materials and procedures has been proved in the numerous training sessions.

Keywords: PL-Grid, training, UNICORE, gLite.

1 Introduction

One of the most important problems in grid resources utilization is access barrier. The potential users are usually very interested in utilizing the grid resources, however they have problems to get through the access procedures. Recent developments of the grid middleware significantly lowered technological barriers but still some effort is necessary to enter and use the grid. In the PL-Grid project this problem has been addressed in a number of ways [1]. User management has been handled by the PL-Grid Portal [2] which allows for easy user registration and management. Another important activity implemented to attract users are trainings. In the PL-Grid project users can take part in traditional trainings or participate in on-line trainings. For these purposes a dedicated learning management system and a training infrastructure have been installed and used. Special procedures have been developed to simplify and speed up access to the training facilities.

The details of the grid resources dedicated for training purposes are similar to the production grid resources and are described in the separate chapter [3] therefore we will not present them here. This article focuses on the trainings organization and integration of the learning resources and training infrastructure with the PL-Grid Portal as the main entry point for actual and prospective users.

2 State of the Art

The promotion of a grid infrastructure and trainings can be performed in numerous ways. The simplest one is evangelization of actual and potential users with presentations, posters and flyers. This activity is often profiled according to the users' background and the presentations are performed at thematic conferences and events to meet potential users rather than developers. Another way is attracting users through publications in journals and newspapers including general ones. It is well known that these activities require a lot of effort, both in terms of manpower and money and final effect is rather limited. The main reason of the low impact is difficulty in utilization of new knowledge by the users and lack of organizational and commercial support of new solutions.

The main barrier in entering grid are difficulties met at the start, caused by the long procedures to obtain certificates, handle and install them. The verification procedure takes time and cannot be simplified since the user is granted access to a large and complicated infrastructure. Usually before the procedure ends potential users lose their interest and end up not obtaining access to the grid. The only solution to this problem is either simplification of the procedure or creation of a dedicated training infrastructure which can be accessed by users almost immediately. During training period the verification the user can be finished which allows for smooth transfer from testing to production environment.

Training infrastructure allows beginners to play and experience how to use the grid without being officially enrolled in a grid production environment. In case of gLite [4], the EGEE [5] and related projects deployed a dedicated training infrastructure named GILDA [6]. Beside this permanent infrastructure several efforts have been done to setup temporary grid infrastructures to be used during training and dissemination events. Right now almost all proposed solutions are based on a virtualization approach. The availability of virtualization software like Xen [7], KVM [8], VMware [9] or VirtualBox [10] has boosted the use of virtualization for temporary infrastructures dedicated to training.

A good example of recent developments is GRIDSEED [11], a tool to setup a temporary training infrastructure. The system makes it easy to setup a training testbed to start experiences with grid infrastructure based on gLite middleware. GRIDSEED users and even trainers are largely shielded from the intricate details of Grid middleware installation and configurations. Users can learn how to use the grid and not how to install the middleware. GRIDSEED provides its own Certification Authority so no administrative requests have to be issued before it is possible to start using the grid environment.

Similar approach has been taken by the UNICORE [12] community which provided testbed infrastructure run by the ICM within Chemomentum project

[13,14]. The training infrastructure has been built of Plug and Play Certification Authority and a testbed which provided users with limited resources. The users obtain temporary certificates which allow for testbed access. The application for certificates and testbed resources have been implemented on the web [14].

3 Description of the Solution

In the PL-Grid project we have put significant effort to attract users and to provide training to them. Based on the previous experience we have implemented a variety of methods including well known exploitation in the scientific community through talks, presentations and personal communication. The current approach has been accomplished with on-line trainings in form of webcast presentations. Training activities have been organized using the PL-Grid Portal which provided extended training management functionality. A prospective user can register in the portal and apply for training access.

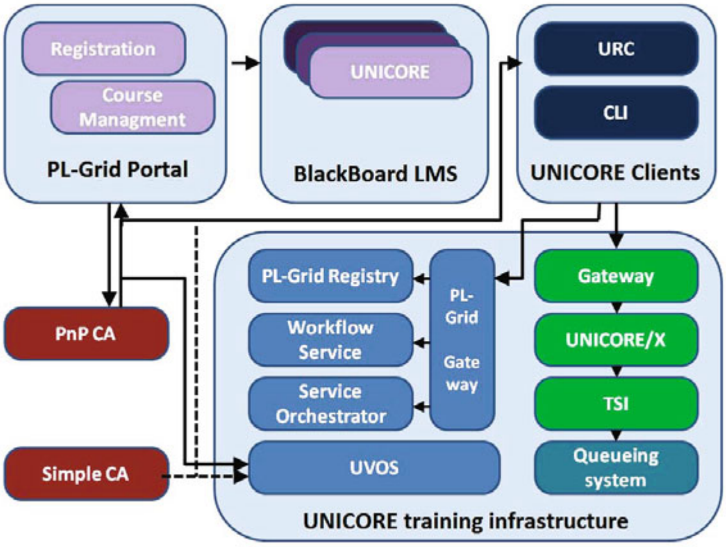


Fig. 1. Key components of the UNICORE training facilities

The PL-Grid Portal has been integrated with Simple Certification Authority (Simple CA), which provides certificates to access the training infrastructure. The user applies for a certificate through the web interface and with the same interface receives automatically generated files with the certificates in formats suitable for gLite and UNICORE. The public part of the certificates is automatically installed on the testbed which allows for almost immediate access to the training resources. This procedure has also been performed by the users during registration for the Pl-Grid hands-on classes.

Additionally, the organizers of the training sessions can use the PL-Grid Portal to generate certificates for the last minute joiners to distribute them on site.

Special effort has been made to establish asynchronous on-line training facilities in order to allow users access to the training infrastructure at any time. For these purposes the PL-Grid Portal has been integrated with the Blackboard system [15]. The registered users of the PL-Grid Portal semi-automatically obtain an account in Blackboard and can be enrolled in on-line grid courses available there. This solution allows to use top level e-learning systems and still handle the user registration process using the PL-Grid Portal.

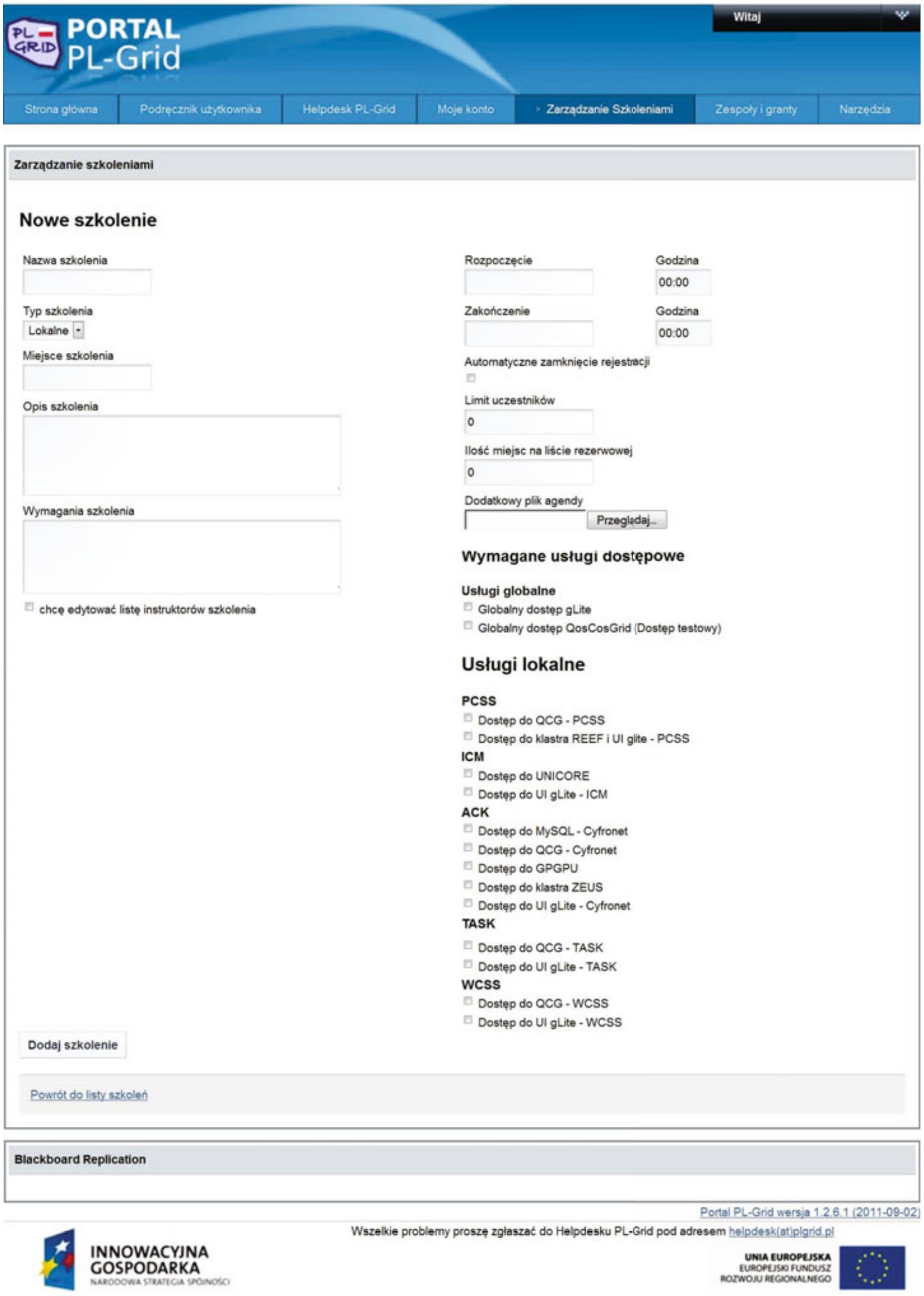
The key components of the training facilities for UNICORE are presented in Fig. 1. The PL-Grid Portal, PnP CA and Blackboard are also used to organize gLite trainings. The client and infrastructure tiers are replaced by corresponding gLite tools.

4 Results

Generally speaking in PL-Grid we can distinguish between two types of trainings: on-line and local ones. Both types of courses are created by the instructors using the PL-Grid Portal. Every authorized PL-Grid project member can easily create a training using the form available in the Training Management portlet. Using the dedicated form presented in the Fig. 2 the course instructor needs to specify: the training type (on-line or local) and its description. In case of local courses the date and time of the training should additionally be defined, as well as the place and number of participants. An agenda file can also be attached. The instructor may indicate that access to particular services is required. For example, if a UNICORE course is planned and the trainer would like to use the production resources, the UNICORE access should be marked. In this way, all the course participants will be added to appropriate databases (LDAP, UVOS) and will have access to the resources necessary for training.

A local training is advertised in advance and usually is organized as a response to special interest shown by a particular research group or organization. Usually such training is performed using computer labs provided by the computer centers or local organizers. On-line trainings are more flexible since they are available through distance learning systems and users can participate in them at any time. The disadvantage is lack of direct contact with the instructor and therefore lack of immediate help while problems occur. The specificity of on-line trainings therefore requires high quality demonstration material. All problematic procedures have to be clearly presented and explained.

Users sign up to the PL-Grid Portal which contains the list of available courses and their descriptions. To minimize the entry barrier, registration to the courses can be performed by all users registered in the PL-Grid Portal even if they have not been verified nor registered as PL-Grid infrastructure users. Access to the actual PL-Grid resources requires user verification which cannot be performed automatically and takes some time. Therefore this procedure can be performed in parallel with the training.



Zarządzanie szkoleniami

Nowe szkolenie

Nazwa szkolenia

Typ szkolenia: Lokalne

Miejsce szkolenia

Opis szkolenia

Wymagania szkolenia

☐ chcę edytować listę instruktorów szkolenia

Rozpoczęcie — Godzina: 00:00

Zakończenie — Godzina: 00:00

Automatyczne zamknięcie rejestracji ☐

Limit uczestników: 0

Ilość miejsc na liście rezerwowej: 0

Dodatkowy plik agendy [Przeglądaj...]

Wymagane usługi dostępowe

Usługi globalne
- ☐ Globalny dostęp gLite
- ☐ Globalny dostęp QosCosGrid (Dostęp testowy)

Usługi lokalne

PCSS
- ☐ Dostęp do QCG - PCSS
- ☐ Dostęp do klastra REEF i UI glite - PCSS

ICM
- ☐ Dostęp do UNICORE
- ☐ Dostęp do UI gLite - ICM

ACK
- ☐ Dostęp do MySQL - Cyfronet
- ☐ Dostęp do QCG - Cyfronet
- ☐ Dostęp do GPGPU
- ☐ Dostęp do klastra ZEUS
- ☐ Dostęp do UI gLite - Cyfronet

TASK
- ☐ Dostęp do QCG - TASK
- ☐ Dostęp do UI gLite - TASK

WCSS
- ☐ Dostęp do QCG - WCSS
- ☐ Dostęp do UI gLite - WCSS

[Dodaj szkolenie]

Powrót do listy szkoleń

Blackboard Replication

Portal PL-Grid wersja 1.2.6.1 (2011-09-02)

Wszelkie problemy proszę zgłaszać do Helpdesku PL-Grid pod adresem helpdesk(at)plgrid.pl

INNOWACYJNA GOSPODARKA
NARODOWA STRATEGIA SPÓJNOŚCI
UNIA EUROPEJSKA
EUROPEJSKI FUNDUSZ ROZWOJU REGIONALNEGO

Fig. 2. Screenshot of a training creation portlet. The instructor has to specify the course name, description, time and location as well as necessary resources.

Signing up for any kind of trainings does not force to register as a PL-Grid user. Even a not logged-in person can see all available trainings and their descriptions in the portal. There are two possibilities to enroll in the course: registration as a PL-Grid user, which requires additional verification, or registration for the tutorial only.

Course instructor after creating the training has the ability to approve the applications, write e-mails to the participants and, in the case of local courses, generate training certificates. While a new training participant registers in the portal, a notification is sent to the course instructor. They can approve or reject the application and the user is informed about the decision by e-mail.

4.1 Local Training

Local trainings are courses based on direct contact with the instructor. For this kind of trainings we use certificates valid for a short period of time, usually several days (short-term certificates). Such certificates can be generated for the participants using the PL-Grid Portal by the instructor. They are valid for gLite and UNICORE both production and training infrastructures. These certificates can be handed to the users after verification of the person's identity, which can be carried out at the beginning of the tutorial. The big advantage of the short-term training certificates is that they have the same format as the production ones. Therefore it is easy for the users to repeat the procedures after they register in PL-Grid and get SimpleCA certificates. Since the PL-Grid Portal allows for attaching training materials to the tutorial, participants can download and use them after the course in case of any problems. There is also a user handbook integrated with the portal which contains the most important information about the project and middlewares.

Local courses are time-limited, so the instructor has to focus on the parts significant to the users. This type of trainings is very often prepared at the request of some group of people interested in a certain research area. Such course is created in the portal and announced on the PL-Grid webpage. It is important to gather participants similarly oriented which allows to profile presented applications depending on participant interest. A biologist would like to know how to run bioinformatics tools such as BLAST [16] or Clustal [17] while a chemist could be interested in molecular dynamics applications such as Amber [18] or Gromacs [19]. If participants are completely new to the grid, the trainer should present how to get middleware, a certificate and how to start running jobs. On the other hand, a tutorial for a group of system administrators should focus on the system architecture and installation.

Local trainings are an important activity in the PL-Grid project. This fact is reflected by the number of users trained. During the 2 year period local trainings have been provided to 660 users. Some of the users have attended multiple training sessions therefore the unique number of trained people is less, but still large and comparable with the number of users of traditional HPC resources in Poland. The training sessions were quite uniformly distributed in time, with

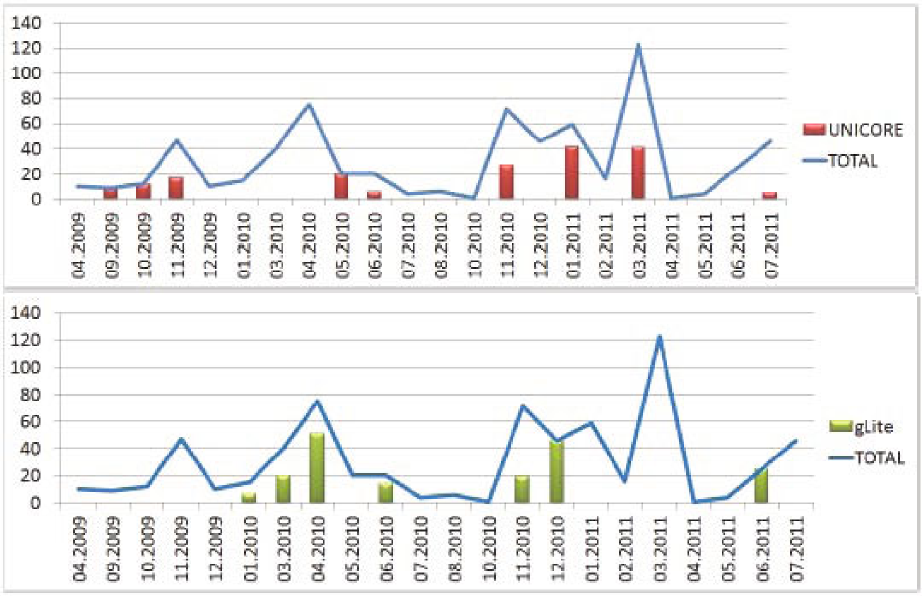


Fig. 3. Number of users trained monthly in the PL-Grid project. Data presented for the UNICORE (upper) and gLite (lower) local trainings. Line presents total number of trained users.

smaller activity during summers due to vacation season. The number of users trained monthly is presented in Fig. 3.

Each training was profiled individually depending on the participants. However we can distinguish some types of trainings: basic trainings presenting PL-Grid aims, middleware presentations (UNICORE, gLite), PL-Grid Portal usage trainings, and finally, trainings focused on running particular applications on the grid.

4.2 On-line Trainings

Another type of courses are on-line trainings. They are many advantages of this type of trainings: users can try the grid at any place and time. They can follow the whole course or just focus on the parts that are important for them. The drawback of such trainings is lack of direct contact with the instructor. However, in case of any problems, users can contact helpdesk or even the instructor personally since their e-mail address is provided in the course.

On-line trainings are available through the e-learning platform. Among several tools for on-line learning, including Olat [20] or Moodle [21], the Blackboard platform was chosen. It is maintained by the Academic Computer Centre Cyfronet AGH. The system has been integrated with the PL-Grid Portal, so every user – who signs up for a training – gets an account at Blackboard automatically. There are several courses available in the system. User can sign up for them through the Portal as it was described in the previous section.

In case of the UNICORE middleware the content of the on-line lessons can be practiced on the production and training infrastructure. The first one is available only for registered users who have a valid Simple CA or Polish Grid certificate. These certificates also allow access to the training infrastructure, so PL-Grid users may practice on it without consuming the production resources. People who would like to try the middleware but are not registered users can use short-term tutorial certificates. It is possible thanks to the installation of the Plug-and-Play Certification Authority (PnP CA), which was developed during the Chemomentum project and is available as open source software [22].

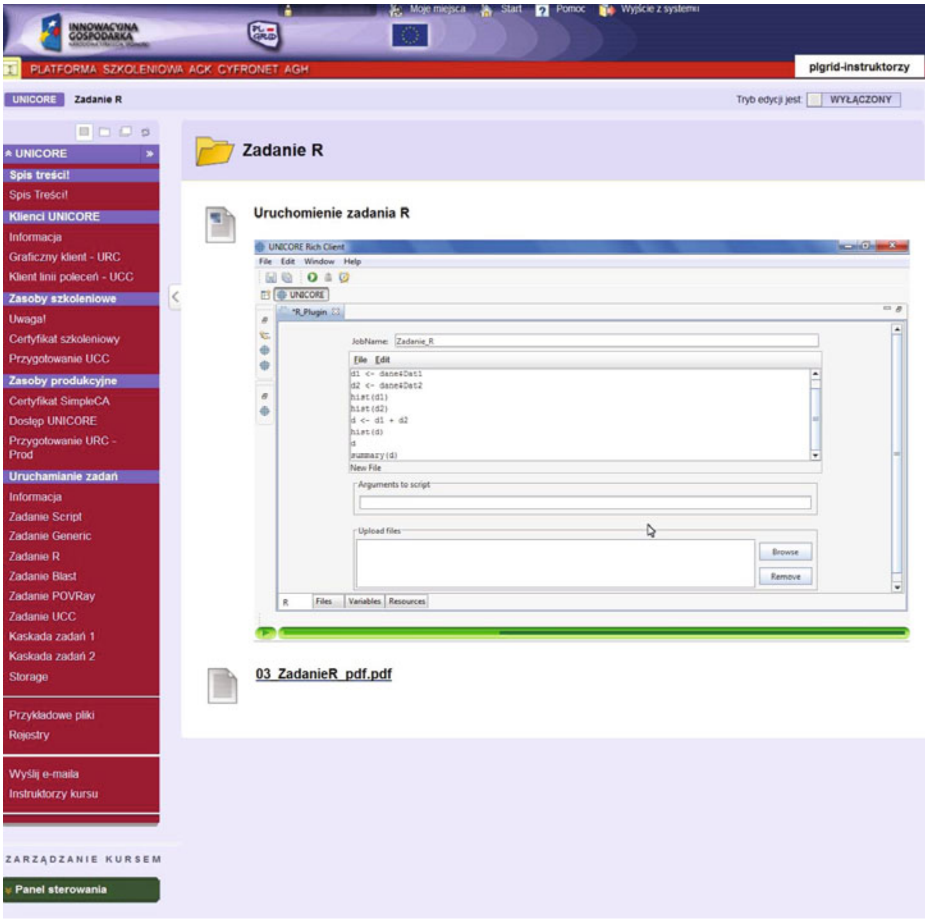
To generate a certificate user has to create a certificate signing request (CSR) file which can be done using the UNICORE Rich Client or with standard unix tools. Next, that file has to be sent to the CA using a dedicated web form. The certificate is sent automatically to the user by e-mail. The certificate can also be requested using the PL-Grid portal where the user only needs to provide necessary data in a web form. Generated certificate can be downloaded and decoded using dedicated portlets.

The whole procedure has been accepted by users as it usually takes only several minutes to obtain a valid certificate, install it in the UNICORE Client and start using the training infrastructure.

4.3 UNICORE Training Lessons

Most of the on-line course materials describing UNICORE middleware are prepared using the Wink program freely available on the web [23]. Prepared webcasts are available in the Flash format (see Fig. 4). They are created as a sequence of screenshots showing all mouse movement needed to complete the task. The comments added in textboxes help to explain some actions (see Fig. 5). To allow the users quick access to the materials, for example to print them, all training files are also provided in PDF documents.

The key element of the UNICORE training is to get and configure the UNICORE client. The download and installation the UNICORE Client is straightforward, the main problem is proper dealing with the certificates. In the on-line course there are separate files presenting how users can generate the Simple CA certificate and how non-registered users can get a certificate for the training infrastructure. This step is especially important for self training, since during traditional courses there is usually no need to use PnP CA because users register in advance and are provided with the certificates generated by the course manager. Once user obtains a certificate, he/she has to create a keystore file used by UNICORE clients. This process is also presented as webcast. The last part of the configuration process is provision of a registry address. The whole process is presented and allows the user to install and configure the UNICORE client with access to the grid infrastructure. Webcasts are prepared and verified based on traditional face-to-face trainings and seem to be straightforward even for non-experienced users.

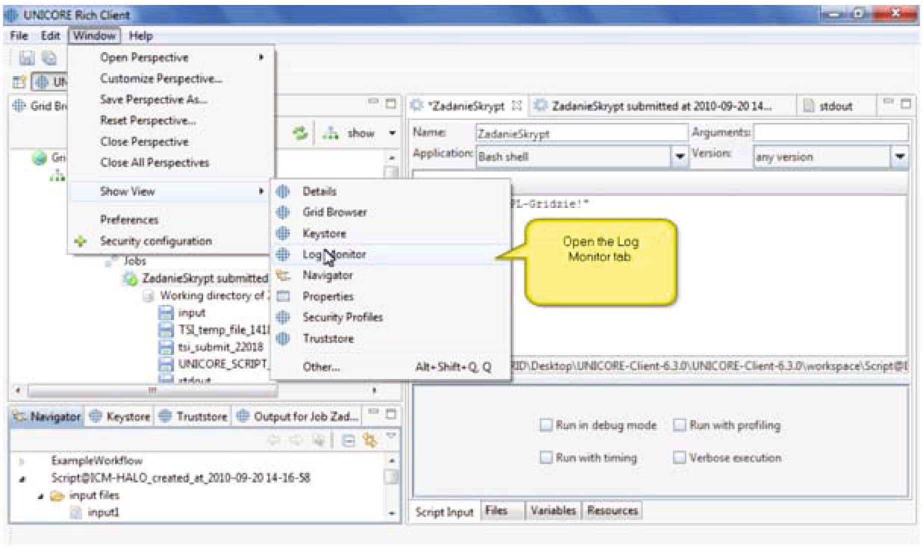


JobName: Zadanie_R

```
d1 <- dane$Dat1
d2 <- dane$Dat2
hist(d1)
hist(d2)
d <- d1 + d2
hist(d)
d
summary(d)
```

Fig. 4. Screenshot of the wink webcast in the Blackboard lecture management system

The second part of the training is usage of the basic and advanced capabilities of the UNICORE client. In the on-line training there are several presentations showing the capabilities of the UNICORE system. They are based on the example of using different applications such as BLAST, R [24], POVRay [25]. Some tutorials are dedicated to the workflows. They present how to design and run them in the grid using the UNICORE middleware. Examples of input data for used applications are provided. Most of the presentations focus on the UNICORE Rich Client, as it is chosen by the users more often than the UNICORE Commandline Client. However, there are also tutorials presenting the second type of the UNICORE client.



Name: ZadanieSkrypt
Arguments:
Application: Bash shell
Version: any version

Fig. 5. Screenshot of the wink webcast with the comments explaining some actions are presented

5 Conclusions

The PL-Grid training infrastructure has been successfully implemented and put into operation. It has been used to organize training events and to allow users to access on-line training materials. The procedures available in the portal have been created based on the numerous training events performed within the project. The feedback received from users and trainers allowed us to optimize the training infrastructure and operational procedures in order to minimize disadvantages. The on-line training materials and dedicated training infrastructure for the UNICORE and gLite grids have been created. The training infrastructure and associated procedures have been used to attract users providing fast and easy access to the training resources. Direct consequence of minimizing the access barrier was an increase of users' interest in the production grid infrastructure. A large number of trained users shows that training activities in the PL-Grid project were designed and implemented successfully.

References

1. Pająk, R., Mosurska, Z.: Dissemination Activities Conducted within the PL-Grid Project as a Successful Means of Promotion of its Offer and Achievements. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 326–338. Springer, Heidelberg (2012)
2. Kitowski, J., Turała, M., Wiatr, K., Dutka, Ł.: PL-Grid: Foundations and Perspectives of National Computing Infrastructure. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 1–14. Springer, Heidelberg (2012)

3. Benedyczak, K., Stolarek, M., Rowicki, R., Kluszczyński, R., Borcz, M., Marczak, G., Filocha, M., Bała, P.: Seamless Access to the PL-Grid e-Infrastructure Using UNICORE Middleware. In: Bubak, M., Szepieniec, T., Wiatr, K. (eds.) PL-Grid 2011. LNCS, vol. 7136, pp. 56–72. Springer, Heidelberg (2012)
4. Laure, E., Hemmer, F., Prelz, F., Beco, S., Fisher, S., Livny, M., Guy, L., Barroso, M., Buncic, P., Kunszt, P., Di Meglio, A., Aimar, A., Edlund, A., Groep, D., Pacini, F., Sgaravatto, M., Mulmo, O.: Middleware for the next generation Grid infrastructure. In: Aimar, A., Harvey, J., Knoors, N. (eds.) Computing in High Energy Physics and Nuclear Physics, Interlaken, Switzerland, p. 826 (2004)
5. Gagliardi, F., Jones, B., Grey, F., Bégin, M.-E., Heikkurinen, M.: Building an infrastructure for scientific Grid computing: status and goals of the EGEE project. Phil. Trans. R. Soc. A 363(1833), 1729–1742 (2005)
6. GILDA (Grid INFN Laboratory for Dissemination Activities), https://gilda.ct.infn.it/
7. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Wareld, A.: Xen and the Art of Virtualization. In: Proceedings of the Nineteenth ACM Symposium on Operating Systems Principles. ACM, Energy and Nuclear Physics (2003)
8. Warnke, R., Ritzau, T.: qemu-kvm & libvirt. Books on Demand GmbH, Norderstedt (2010)
9. Devine, S., Bugnion, E., Rosenblum, M.: Virtualization system including a virtual machine monitor for a comwith a segmented architecture. US Patent (1998)
10. http://www.virtualbox.org
11. Gregori, I., Patil, M., Cozzini, S.: GRIDSEED: A Virtual Training Grid Infrastructure. In: Cozzini, S., Lagana, A. (eds.) ICTP Lecture Notes Series, vol. 24 (November 2009)
12. Streit, A., Bala, P., Beck-Ratzka, A., Benedyczak, K., Bergmann, S., Breu, R., Daivandy, J.M., Demuth, B., Eifer, A., Giesler, A., Hagemeier, B., Holl, S., Huber, V., Lamla, N., Mallmann, D., Memon, A.S., Memon, M.S., Rambadt, M., Riedel, M., Romberg, M., Schuller, B., Schlauch, T., Schreiber, A., Soddemann, T., Ziegler, W.: UNICORE 6 – Recent and Future Advancements. Annales des Télécommunications 65(11-12), 757–762 (2010)
13. Bala, P., Baldridge, K., Benfenati, E., Casalegno, M., Maran, U., Rasch, K., Schuller, B.: UNICORE – a successful middleware for Life Sciences Grids. In: Cannataro, M. (ed.) Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and HealthCare IGI, pp. 615–643 (2009)
14. Chemomentum Project, http://www.chemomentum.org
15. Blackboard, http://www.blackboard.com/Platforms/Learn/Overview.aspx
16. http://www.ncbi.nih.gov/BLAST
17. http://www.clustal.org
18. Pearlman, D.A., Case, D.A., Caldwell, J.W., Ross, W.S., Cheatham, T.E., Debolt, S., Ferguson, D., Seibel, G., Kollman, P.: AMBER, a package of computer programs for applying molecular mechanics, normal mode analysis, molecular dynamics and free energy calculations to stimulate the structural and energetic properties of molecules. Computer Physics Communications 91(1-3), 1–41 (1995)
19. Berendsen, H.J.C., van der Spoel, D., van Drunen, R.: GROMACS: a message-passing parallel molecular dynamics implementation. Computer Physics Communications 91(11), 43–56 (1995)
20. http://www.olat.org

21. Dougiamas, M., Moodle, P.T.: Using Learning Communities to Create an Open Source Course Management System. In: Lassner, D., McNaught, C. (eds.) Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications, Chesapeake, VA, pp. 171–178 (2003)
22. `http://pnp-ca.sf.net`
23. `http://www.debugmode.com/wink/`
24. `http://www.r-project.org/`
25. `http://www.povray.org/`

Dissemination Activities Conducted within the PL-Grid Project as a Successful Means of Promotion of Its Offer and Achievements

Robert Pająk and Zofia Mosurska

AGH University of Science and Technology, ACC Cyfronet AGH,
ul. Nawojki 11, 30-950 Kraków, Poland
{r.pajak,z.mosurska}@cyfronet.pl
http://www.cyfronet.pl

Abstract. The dissemination of information as well as promotion of results are the key activities for a scientific project co-funded by European funds. This paper will summarize the dissemination-related activities conducted within the PL-Grid project. It will also point a number of crucial elements of this work – namely, roles and responsibilities of the Project partners, methods of communication, structure of the dissemination efforts and means and measures of their success.

Keywords: dissemination, promotion, offer, results, materials, events.

1 Introduction

The PL-Grid project was aimed at establishing a country-wide grid infrastructure for scientists in Poland, at the same time, enabling their wide international collaboration. In the framework of the Project, a set of tools has been created and deployed, which enabled designing and running scientific applications on powerful computational resources by use of distributed data sources. As one of the PL-Grid's main tasks was attracting interest of potential users from different scientific disciplines and academia, a number of diverse dissemination activities were conducted – all in order to ensure applicability of the Project results, increase usability of its resources, services and offered scientific packages, and improve the Project's recognition within the Polish scientific community and in Europe. From the very beginning of the Project, we undertook different actions and used various means to encourage the scientists and research groups representing various fields of science – potentially interested in using the PL-Grid infrastructure for computations and large scale simulations – to take advantage of the Project's offer.

2 Related Work

The members of the PL-Grid Dissemination Team (DT) were previously involved in the CrossGrid, Int.eu.grid and BalticGrid projects, where their role was to support, and, later, to coordinate the work packages dealing with education, training, dissemination and outreach activities. An international cooperation with other partners involved in all these projects helped us to better understand the need for well-organized dissemination, and its significance in terms of advertising achievements of a project among general public and potential beneficiaries of these results. A cooperation with other related projects (DataGrid, GridStart, EGEE, EGI, BELIEF) and National Grid Initiatives – NGIs (LitGrid, LatvianGrid, EstonianGrid), as well as observing various types of dissemination activities conducted within them, helped us to acquire additional knowledge and experience. This enabled promoting the PL-Grid project in a most suitable and efficient way.

3 Description of the Solution

3.1 Organization of Work and Responsibilities of the PL-Grid Dissemination Team

In order to fulfill its aims and plans, various carefully-focused groups were established by the PL-Grid Project Management through formal and informal mechanisms. One of these groups was the PL-Grid Dissemination Team – responsible for planning, coordinating and monitoring of the dissemination activities within the Project. The team has received a broad support from all the Project consortium members, following the instructions of the Project Director (PD) and general Project policy. The PL-Grid Dissemination Team was composed of:

– the Dissemination Team Leader (DTL) – from ACC Cyfronet AGH,
– the Dissemination Contact Persons (DCP) – representing all Partners.

The DTL was in charge of defining strategic options of the Project dissemination and implementing the dissemination policy in close cooperation with all the participants of the PL-Grid Consortium. The Dissemination Team Leader was also responsible for planning, organizing and coordinating of all dissemination activities performed within the Project. Other Partners, especially their DCPs, supported the DTL in fulfillment of its tasks through spreading information on PL-Grid and its achievements via lectures, seminars and tutorials during conferences and workshops, and through Partners' websites and local media. In addition, they provided the DTL with information concerning results of their promotional activities within the Project, which was later included in the PL-Grid website and used during preparation of dissemination materials.

The role of the Project Director and Task Leaders (TLs) within the cooperation with the Dissemination Team was twofold (see Fig. 1):

– approving dissemination plans, initiatives, materials and documents;
– providing information on the work progress in each Project Task (guides and instructions for users, results, achievements and news) – this information was subsequently used for the PL-Grid website and as material for the Project newsletters, brochures, posters, presentations, movies, etc.

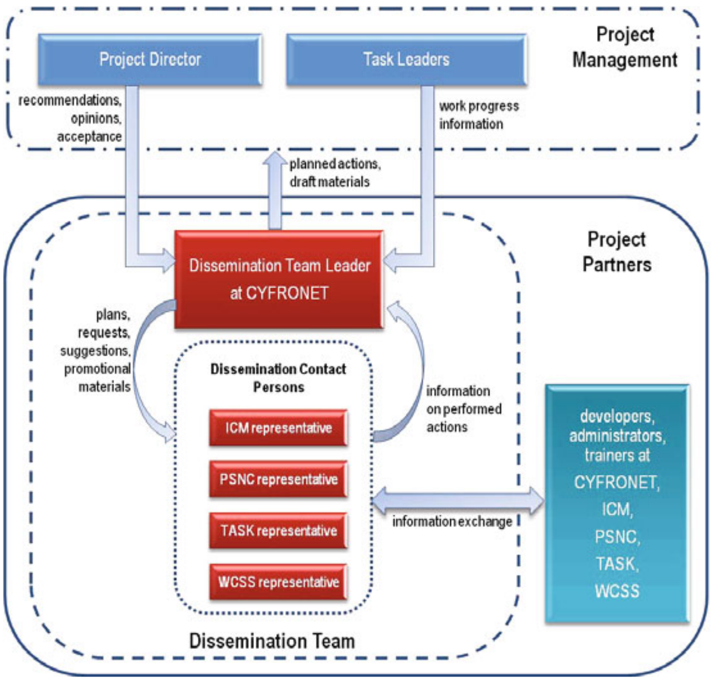

Project Director
Task Leaders
Project Management
recommendations, opinions, acceptance
work progress information
planned actions, draft materials
Project Partners
Dissemination Team Leader at CYFRONET
plans, requests, suggestions, promotional materials
Dissemination Contact Persons
ICM representative
PSNC representative
TASK representative
WCSS representative
information on performed actions
information exchange
developers, administrators, trainers at CYFRONET, ICM, PSNC, TASK, WCSS
Dissemination Team

Fig. 1. Cooperation among the Dissemination Team, Project Management and Partners

3.2 Structuring the Dissemination Efforts

To properly address different target audiences of the results achieved within PL-Grid, the Project dissemination activities have been split into two distinct levels: public (including national and international sub-levels) and internal (within the PL-Grid Consortium). The following activities were carried out in the framework of the public dissemination (they will be described in detail in Section 4):

– at national level – creation, development and maintenance of the Project website, promotion of the Project at grid- and IT-related national events; organization of seminars and meetings with potential users; creation and distribution of disseminative materials and promos; establishment of contacts with local scientific community; building the PL-Grid corporate image,
– at international level – maintenance of the English version of the Project website; spreading information about PL-Grid during international events; production and distribution of disseminative materials written in English; elaboration and publication of press articles in international media; maintenance of the Project profiles at community portals; cooperation with related projects.

The goal of internal dissemination within PL-Grid was to develop solid links and means for information exchange among the PL-Grid partners. This involved spreading, making available and promoting the Project results within the Consortium, in order to increase the PL-Grid dissemination potential outside the Consortium.

All the Project participants have been treated as promoters of PL-Grid and, therefore, had to be informed, updated and involved in the Project achievements and plans in a systematic and consistent way. This task was realized mainly during the Project technical meetings, through talks given by the DTL and through direct contacts among the Consortium members. All necessary materials for supporting the above-mentioned task (news, newsletters, leaflets, movies, etc.) were made available on the Project public website and on the internal Wiki pages (for the stage of preparation and consultation). The latter tool was oriented towards the management and handling of internal affairs of the Project and comprised all essential collaboration information, materials and documents (reports, formal documents, issues related to Project meetings, etc.).

Several electronic mailing lists were also established for the PL-Grid members, with aim to provide a mechanism for internal communication and interaction among various teams.

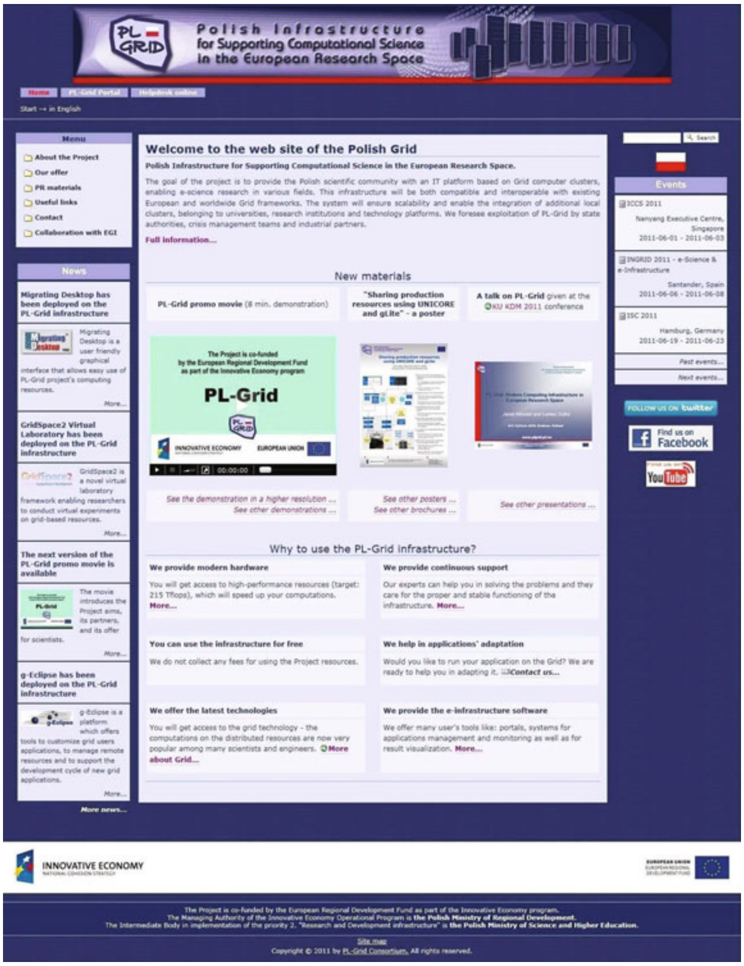
4 Results – Public Dissemination Activities Successfully Conducted within the PL-Grid Project

The main aim of the PL-Grid project was to provide grid infrastructure for scientific computing to researchers who worked with scientific calculations, simulations and modeling. In the framework of the Project, a number of activities promoting it, its aims and results were conducted. These activities aimed at wide dissemination of information about possibilities of supporting the research in Poland with high performance computing, on the infrastructure constructed by the Project consortium.

4.1 Project Website and Presence in Community Portals

The PL-Grid website [1] was – from the very beginning of the Project – one of the most important means for increasing awareness of the grid technology and the PL-Grid project itself amongst wider public. It was also used to attract people to the Polish grid infrastructure – in order to make them PL-Grid users. The main goals of the development of the Project website (see Fig. 2), maintained by ACC Cyfronet AGH – the coordinator of PL-Grid, were:

- to provide the general public with basic information such as: the Project goals and tasks, PL-Grid partners and contact information;
- to disseminate the Project achievements to potential users of the Project resources, as well as to inform them about the PL-Grid offer with various disseminative materials;
- to timely deliver information on news and related events, especially training, conducted within PL-Grid.



Welcome to the web site of the Polish Grid

Polish Infrastructure for Supporting Computational Science in the European Research Space.

The goal of the project is to provide the Polish scientific community with an IT platform based on Grid computer clusters, enabling e-science research in various fields. This infrastructure will be both compatible and interoperable with existing European and worldwide Grid frameworks. The system will ensure scalability and enable the integration of additional local clusters, belonging to universities, research institutions and technology platforms. We foresee exploitation of PL-Grid by state authorities, crisis management teams and industrial partners.

Full information...

New materials

PL-Grid promo movie (8 min. demonstration)

"Sharing production resources using UNICORE and gLite" - a poster

A talk on PL-Grid given at the KU KDM 2011 conference

The Project is co-funded by the European Regional Development Fund as part of the Innovative Economy program

PL-Grid

00:00:00

See the demonstration in a higher resolution ...
See other demonstrations ...

See other posters ...
See other brochures ...

See other presentations ...

Why to use the PL-Grid infrastructure?

We provide modern hardware
You will get access to high-performance resources (target: 215 Tflops), which will speed up your computations. **More...**

We provide continuous support
Our experts can help you in solving the problems and they care for the proper and stable functioning of the infrastructure. **More...**

You can use the infrastructure for free
We do not collect any fees for using the Project resources.

We help in applications' adaptation
Would you like to run your application on the Grid? We are ready to help you in adapting it. **Contact us...**

We offer the latest technologies
You will get access to the grid technology - the computations on the distributed resources are now very popular among many scientists and engineers. **More about Grid...**

We provide the e-infrastructure software
We offer many user's tools like: portals, systems for applications management and monitoring as well as for result visualization. **More...**

Migrating Desktop has been deployed on the PL-Grid infrastructure
Migrating Desktop is a user friendly graphical interface that allows easy use of PL-Grid project's computing resources.
More...

GridSpace2 Virtual Laboratory has been deployed on the PL-Grid infrastructure
GridSpace2 is a novel virtual laboratory framework enabling researchers to conduct virtual experiments on grid-based resources.
More...

The next version of the PL-Grid promo movie is available
The movie introduces the Project aims, its partners, and its offer for scientists.
More...

g-Eclipse has been deployed on the PL-Grid infrastructure
g-Eclipse is a platform which offers tools to customize grid users applications, to manage remote resources and to support the development cycle of new grid applications.
More...

More news...

ICCS 2011
Nanyang Executive Centre, Singapore
2011-06-01 - 2011-06-03

INGRID 2011 - e-Science & e-Infrastructure
Santander, Spain
2011-06-06 - 2011-06-08

ISC 2011
Hamburg, Germany
2011-06-19 - 2011-06-23

Past events...
Next events...

The Project is co-funded by the European Regional Development Fund as part of the Innovative Economy program.
The Managing Authority of the Innovative Economy Operational Program is **the Polish Ministry of Regional Development.**
The Intermediate Body in implementation of the priority 2. "Research and Development infrastructure" is **the Polish Ministry of Science and Higher Education.**

Site map
Copyright © 2011 by PL-Grid Consortium. All rights reserved.

Fig. 2. The PL-Grid project website

It was particularly important to deliver up-to-date information on the Project progress in making the grid infrastructure available and in developing the tools and grid services, through the website. All the information related to this topic was gathered in the "Our offer" section, which described:

- grid infrastructure – the means of access to high-performance computing clusters and large storage resources through the PL-Grid Portal;
- scientific software packages – applications from various fields of science (biology, quantum chemistry, physics, numerical computations and simulation, etc.) available in the framework of the PL-Grid infrastructure;
- tools and services – a set of advanced programs which could be used, e.g. for organization of computational experiments, visualization of application results or grid resource management;

– user support – technical support in development and running of scientific applications on the PL-Grid computational resources, and in adaptation of tools; as well as direct and on-line training – including training on request and course materials;
– dissemination activities, materials and reports.

One of the measures of the success of dissemination efforts conducted within the Project is the number of visits on the Project website. To allow monitoring of section usage patterns within the website, a web statistics tool – WebLog Expert [2] has been used (starting from May 2009). Sample data covering the period from May 2009 to November 2011 is presented in Fig. 3. Another interesting information gained from web statistics is the total number of connections to the PL-Grid website coming from unique IP addresses. More than 18,000 connections from Poland were registered until the end of November 2011, which means – assuming that each IP belongs to a different person – that several thousands of people have somehow been informed about PL-Grid and visited the Project website to learn more.

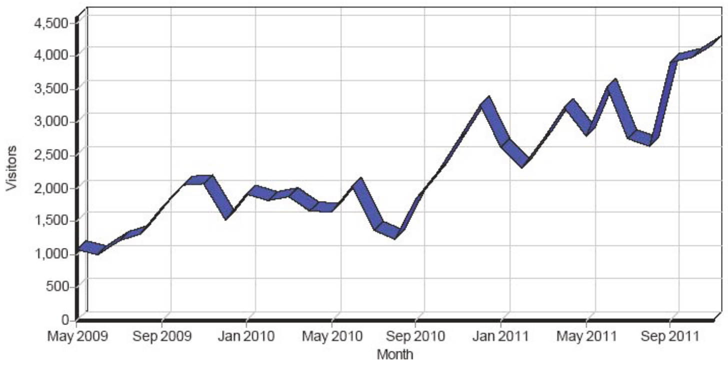


Fig. 3. The PL-Grid project website usage statistics

To broaden the group of potential users of the Project results, PL-Grid has also been advertised at webpages of the Partner institutions and has been present in various community portals. Up-to-date information on the Project has been systematically published on Twitter [3], Facebook [4] and YouTube [5].

4.2 Organization and Participation in Related Events

One of the Project's promotion tasks was organization of seminars at research institutions as well as workshops and grid conferences in Poland. Among the latter were: Cracow Grid Workshop, the conference of the High Performance Computers' Users, i3: Internet, Infrastructure, Innovations conference, Open Days at Partner institutions. During these events, the participants had an opportunity to meet the PL-Grid experts and obtain detailed information on the available infrastructure and its possible use in high performance computations.

The Polish initiative of the computational infrastructure and its offer were also advertised through active participation of the Project representatives in other, international conferences related to grid technologies and high performance computing: International Conference on Parallel Processing and Applied Mathematics (PPAM), the European Grid Initiative (EGI) Technical Forum and User Forum, eChallenges Conference, Supercomputing Conference, etc. The participants of these events were able to learn about the Project, its achievements and further perspectives of the e-Infrastructure development in Poland by listening to corresponding talks, visiting exhibition stands and examining leaflets, posters and demonstrations of the PL-Grid tools and services.

4.3 Exhibitions

PL-Grid exhibition stands were held during various relevant events: EGEE'09, 21-24.09.2009, Barcelona; i3 conference, 4-6.11.2009, Poznań; Open Day at ACC Cyfronet AGH, 16.11.2009, Kraków; 7th Conference on Computer Methods and Systems (CMS'09), 26-27.11.2009, Kraków; ICM Day at the University of Warsaw, 26.02.2010, Warsaw; KU KDM 2010, 18-19.03.2010, Zakopane; EGEE User Forum, 12-15.04.2010, Uppsala; MCSB 2010 – Cybernetic Modelling of Biological Systems, 21-22.05.2010, Kraków; EGI Technical Forum, 14-16.09.2010, Amsterdam; CGW'2010, 11-13.10.2010, Kraków; Open Day at ACC Cyfronet AGH, 25.10.2010, Kraków; eChallenges e-2010 conference, 27-19.10.2010, Warsaw; 2^{nd} i3 conference, 1-3.12.2010, Wrocław; KU KDM 2011, 9-11.03.2011, Zakopane; EGI User Forum, 11-14.04.2011, Vilnius; ISC 2011, 19-23.06.2011, Hamburg and NANO 2011, 4-7.07.2011, Gdańsk.

The participants of these events were able to visit the PL-Grid booth, where they had an opportunity to talk to the Project representatives and to obtain information on PL-Grid and its offer. They could also watch demonstrations of selected PL-Grid tools or several movies promoting the Project displayed at the booth. In addition, several types of promotional materials were distributed among the visitors.

4.4 Meetings with Potential Users

Another important dissemination activity within PL-Grid was the organization of direct meetings with people representing key scientific disciplines – like biology, quantum chemistry, physics, astrophysics and materials science. More than ten such events were organized at the Project partner institutions, being a great opportunity to increase awareness of the possibility of using PL-Grid resources to the research these people conduct, and to attract them to the Grid. The scientists had a chance to visit the grid cluster rooms as well as to learn about the work carried out within PL-Grid, including tools and services developed to simplify executing users' applications on a large scale infrastructure. On the other hand, the Project members could obtain information about the requirements and expectations concerning the PL-Grid infrastructure and scientific software they use.

4.5 PL-Grid Corporate Image and Promotional Materials

The Dissemination Team had to build a strong corporate image and style of the Project in order for PL-Grid to be easily recognizable. The following activities have been performed to realize this task:

- the Project logo was designed and used in all the dissemination materials – from websites to formal PL-Grid documents, reports, brochures, posters and other materials and promos;
- the style elements and templates of dissemination materials, presentations and documents were prepared for use by all the PL-Grid members that deal with public and internal dissemination activities – to develop eye-catching promotional materials.

Having prepared the Project logo and style elements, it was possible to realize the next important part of dissemination activities within PL-Grid – namely, elaboration and distribution of promotional materials.

A rich set of brochures, information posters (see Fig. 4) and demonstration videos has been prepared, made available at the Project website and distributed among the potential users. Some of these materials were elaborated in form of instructions, to help efficiently use the new PL-Grid facilities available to the potential users – e.g. brochures: "Project offer", "How to become a PL-Grid user", "User support in PL-Grid"; guides: "PL-Grid User Guide", "PL-Grid Portal User Guide", "PL-Grid Helpdesk User Guide", "PL-Grid Computational Grants"; posters: "PL-Grid", "PL-Grid – technical poster", "Tools and services"; and several roll-ups and promotional movies.

Polish Infrastructure
for Supporting Computational Science
in the European Research Space
www.plgrid.pl
Polish Infrastructure
for Supporting Computational Science
in the European Research Space
How we started
What we aim at
Our partners
What we offer
PL-Grid structure
www.plgrid.pl
Polish Infrastructure for Supporting Computational Science in the European Research Space
Tools and Services
www.plgrid.pl
user tools
tools for management of VOs, grid resources and security
portal applications

Fig. 4. Examples of the PL-Grid posters

More than 100 oral and poster presentations [6] were given by the Project representatives during related events in Poland and abroad.

Many press articles have been elaborated and published in:

– national media – portals: Molnet [7], Innopomorze [8] and ACC Cyfronet AGH website [9], [10], [11]; bulletins: AGH [12] and WCSS "Pryzmat" [13]; scientific journals: Forum Akademickie [14] and PAK [15]; newspapers: Dziennik Bałtycki [16];
– international media – The Parliament Magazine [17], EGI [18] and iSGTW [19] newsletters.

In addition, a number of scientific publications have been published in Computational Methods in Science and Technology 2010 [20] and in the Proceedings of: CGW'09 [21], [22], [23], [24], [25], CGW'10 [26], [27], [28], [29], [30], IMCSIT'2010 [31], PARENG'2011 [32] and ICCS'2011 [33], [34], [35].

Other activities involved production of several commercial advertisements in travel guides [36], [37] and newspapers [38], [39].

One of the best means of distributing the Project topical issues among the PL-Grid users was the PL-Grid Newsletter (see Fig. 5) – devoted to the most important innovations in hardware, software and services which were made available to the PL-Grid users.

PL-Grid Newsletter
kwiecień 2011
Infrastruktura PL-Grid – stan obecny i najbliższe prace
Portal PL-Grid – obecnie udostępnione usługi
PL-Grid Newsletter
czerwiec 2011
Infrastruktura PL-Grid – prace instalacyjne
Udostępniane oprogramowanie
PL-Grid Newsletter
sierpień 2011
Wykorzystanie zasobów PL-Grid
Nowe funkcjonalności w Portalu PL-Grid

Fig. 5. The PL-Grid Newsletter – first three editions

In the framework of the PL-Grid project's dissemination activities a variety of promotional gadgets were also created – including USB pen drives, mouse pads, pens, pencils, T-shirts, etc. Together with the dissemination materials mentioned above, they were distributed among the potential users of the PL-Grid computational infrastructure – participants of the meetings, seminars, conferences and training organized by the PL-Grid Dissemination Team and other events during which the Project was promoted.

4.6 Other Activities

Various non-standard activities were also undertaken to attract interest of new groups of scientists in Poland and to increase the general visibility of the Project.

PL-Grid members reported the Project to participate in the "Funds and Science" competition in the category "Higher education infrastructure", organized press campaigns and participated in science festivals. These initiatives resulted in many articles or references concerning PL-Grid in mass media [40].

To optimize scientific networking, R&D management, exchange of information and dissemination activities, the PL-Grid project actively cooperated with other, grid-related projects. The most intensive and diverse collaboration was established with the EGI.InSPIRE project, in the framework of the e-Infrastructure interoperability and dissemination issues. The PL-Grid members participated in the EGI events (conferences, user forums) with talks, posters, demonstrations and exhibition stands, as well as promoted PL-Grid results in relevant dissemination instruments initiated by EGI (newsletters, brochures, webpages, etc.).

5 Conclusions and Future Work

In this paper we presented the dissemination activities carried out within the PL-Grid project. All these efforts were planned and coordinated by the Dissemination Team Leader; however, suitable implementation of the plan required close cooperation of all the PL-Grid partners. This resulted in various, well-performed activities, carried out at national and international dissemination levels, which contributed significantly to increasing of awareness and knowledge about the Project and the grid technology in Poland.

During the whole Project life, the Partners put a lot of effort to contacting diverse groups of people potentially interested in grid computing, and informed them about the possibility of using the PL-Grid computational infrastructure and its services. Various methods and means were utilized to attract interest, transfer necessary knowledge and convince potential users of the benefits of using the Grid. The following results of these activities have been registered: participation of a great number of people – not involved in the Project – in conferences, seminars, meetings and discussions organized by the PL-Grid Consortium; thousands of tickets in the Helpdesk system; and – most importantly – registration of many new users within the Project infrastructure.

Substantial computational resources of PL-Grid were installed and made available to users in November 2010. Within those days – as a result of dissemination efforts – many new articles or references concerning the PL-Grid project appeared on the Internet, in press and radio stations. It all resulted in an increased interest in the PL-Grid offer, and number of visitors of the Project website raised significantly (see Fig. 3). At the same time, we observed growth of the number of users who registered into the PL-Grid infrastructure.

Other events, like: inclusion of Polish Grid into EGI, deployment of the Helpdesk system and PL-Grid Portal with a rich set of services for users, publication of new tools facilitating and supporting the infrastructure use – were also strongly highlighted at the PL-Grid website. A dynamic and up-to-date website not only was a key element in maximizing the visibility, but also provided support to users and people potentially interested in grid computing.

All these disseminative activities, together with training – which supported the PL-Grid developers and administrators in performing their tasks – caused that more than 500 researchers became registered users of PL-Grid within several months after the installation of the main resources.

Thanks to our activities at international events, contacts with related EU projects and informative website in English, PL-Grid is also known outside of Poland, especially among grid communities in Europe. This gives Polish scientists a better chance for extending international cooperation in the future.

References

1. PL-Grid, http://www.plgrid.pl
2. WebLog Expert, http://www.weblogexpert.com/
3. PL-Grid profile at Twitter, http://www.twitter.com/PLGrid
4. PL-Grid profile at Facebook, http://www.facebook.com/pages/Projekt-PL-Grid/106745852703943
5. PL-Grid channel at YouTube, http://www.youtube.com/user/PLGrid
6. PL-Grid related presentations, http://www.plgrid.pl/materialy_pr/
7. Kaczor, A.: PL-Grid. Molnet portal (2010), http://www.molnet.eu/index.php?option=com_content&view=article&id=272
8. Tylman, R., Nakonieczny, M.: Polska Infrastruktura Informatycznego Wspomagania Nauki w Europejskiej Przestrzeni Badawczej PL-Grid. Innopomorze portal (2010), http://www.innopomorze.pl/infrastruktura-gridowa.html
9. Mosurska, Z., Ozięblo, A., Lasoń, P.: Superkomputer klastrowy Zeus z AGH najmocniejszy w Polsce. ACC Cyfronet AGH website (2010), http://www.cyfronet.pl/?a=komunikat20100601
10. Mosurska, Z., Ozięblo, A., Lasoń, P.: Superkomputer Zeus z AGH w pierwszej setce najpotężniejszych komputerów świata. ACC Cyfronet AGH website (2010), http://www.cyf-kr.edu.pl/?a=komunikat20101115
11. Ozięblo, A.: Zeus po raz kolejny w pierwszej setce najpotężniejszych superkomputerów świata! ACC Cyfronet AGH website (2011), http://www.cyfronet.pl/?a=komunikat20110620
12. Kitowski, J., Mosurska, Z., Wiatr, K.: Projekt Polska Infrastruktura Informatycznego Wspomagania Nauki w Europejskiej Przestrzeni Badawczej PL-Grid dla środowisk naukowych w Polsce. AGH Bulletin 02/2010, 4–5 (2010), http://www.biuletyn.agh.edu.pl/biuletynypdf/2010_Biuletyn_PDF/026_02_2010.pdf
13. Balcerek, B., Wawrzyniak, I.: Wrocławskie Centrum Sieciowo-Superkomputerowe w PL-Grid. WCSS bulletin "Pryzmat" 05/2010, 59–61 (2010), http://pryzmat.pwr.wroc.pl/pryzmat/2009-2010/pryzmat238.pdf
14. Wiatr, K.: Superkomputery dla polskiej nauki. Forum Akademickie 09/2010 (2010), http://forumakademickie.pl/fa/2010/09/superkomputery-dla-polskiej-nauki/
15. Mosurska, Z., Kitowski, J., Pająk, R., Wiatr, K.: PL-Grid – Polska Narodowa Inicjatywa Gridowa dla naukowców w Polsce. PAK (Pomiary, Automatyka, Kontrola) 10/2010, 56 (2010), http://www.pak.info.pl/biblioteka/pobierz.php?idPlik=1140&typ=3

16. Netka, K.: Moce obliczeniowe – produkt dla naukowców. Dziennik Bałtycki 04/2010 (2010), http://www.dziennikbaltycki.pl/
17. Newhouse, S., Turała, M., Kitowski, J., Mosurska, Z., Pająk, R.: European Grid Infrastructure: powering computing for European science. The Parliament Magazine (332), 124–125 (2011), http://www.theparliament.com/digimag/issue332
18. Mosurska, Z., Kitowski, J.: News from the National Grid Initiatives – Poland. EGI eNewsletter 3/2009, 3–4 (2009), http://web.eu-egi.eu/fileadmin/user_upload/e-newsletter_3_nov09_final.pdf
19. Kitowski, J., Mosurska, Z., Pająk, R., Venton, D.: Project profile: PL-Grid. iSGTW newsletter 09/2009, (193) (2009), http://www.isgtw.org/feature/project-profile-pl-grid
20. Kurowski, K., Piontek, T., Kopta, P., Mamoński, M., Bosak, B.: Parallel Large Scale Simulations in the PL-Grid Environment. Computational Methods in Science and Technology (Special Issue 2010), 47–56 (2010)
21. Król, D., Funika, W.: Semantic-based SLA-oriented Performance Monitoring in the ProActive Environment. In: CGW 2009 Proceedings, pp. 151–157. ACC Cyfronet AGH, Kraków (2010)
22. Funika, W., Jakubowski, B., Jaroszewski, J.: Integration of the OCM-G Monitoring System into the MonALISA Infrastructure. In: CGW 2009 Proceedings, pp. 158–163. ACC Cyfronet AGH, Kraków (2010)
23. Kitowski, J., Dutka, Ł.: Status and Current Achievements of PL-Grid Project. In: CGW 2009 Proceedings, pp. 295–303. ACC Cyfronet AGH, Kraków (2010)
24. Radecki, M., Szepieniec, T., Flis, Ł., Krakowian, M., Tomanek, M., Ziajka, W.: Operational Architecture of PL-Grid Project. In: CGW 2009 Proceedings, pp. 304–309. ACC Cyfronet AGH, Kraków (2010)
25. Szepieniec, T., Tomanek, M., Twaróg, T.: Grid Resource Bazaar: Efficient SLA Management. In: CGW 2009 Proceedings, pp. 314–319. ACC Cyfronet AGH, Kraków (2010)
26. Kitowski, J., Mosurska, Z., Pająk, R., Dutka, Ł.: Recent Advancements of National Grid Initiative in Poland (PL-Grid). In: CGW 2010 Proceedings, pp. 38–49. ACC Cyfronet AGH, Kraków (2011)
27. Radecki, M., Ziajka, W., Pawlik, M., Szymocha, T., Szelc, M., Flis, Ł., Tomanek, M., Szepieniec, T.: PL-Grid Enhancement for NGI Tools. In: CGW 2010 Proceedings, pp. 136–141. ACC Cyfronet AGH, Kraków (2011)
28. Król, D., Kryza, B., Skałkowski, K., Nikołow, D., Słota, R., Kitowski, J.: QoS Provisioning for Data-Oriented Applications in PL-Grid. In: CGW 2010 Proceedings, pp. 142–150. ACC Cyfronet AGH, Kraków (2011)
29. Funika, W., Szura, F.: Automation of Decision Making for Monitoring Systems. In: CGW 2010 Proceedings, pp. 164–171. ACC Cyfronet AGH, Kraków (2011)
30. Rzepka, M.: Approach to Monitoring Grids with System of Automatic Reporting and Administration (SARA). In: CGW 2010 Proceedings, pp. 172–183. ACC Cyfronet AGH, Kraków (2011)
31. Ciepiela, E., Harężlak, D., Kocot, J., Bartyński, T., Kasztelnik, M., Nowakowski, P., Gubała, T., Malawski, M., Bubak, M.: Exploratory Programming in the Virtual Laboratory. In: Proceedings of the International Multiconference on Computer Science and Information Technology, IMCSIT 2010, pp. 621–628 (2010)

32. Dziubecki, P., Grabowski, P., Krysiński, M., Kuczyński, T., Kurowski, K., Piontek, T., Szejnfeld, D.: New Science Gateways for Advanced Computing Simulations and Visualization using the Vine Toolkit. In: Proceedings of the Second International Conference on Parallel, Distributed, Grid and Cloud Computing for Engineering, PARENG 2011, paper 80. Civil-Comp Press, Stirlingshire (2011)
33. Kopta, P., Kulczewski, M., Kurowski, K., Piontek, T., Gepner, P., Puchalski, M., Komasa, J.: Parallel application benchmarks and performance evaluation of the Intel Xeon 7500 family processors. In: Procedia Computer Science 4, 372–381 (2011); Proceedings of the International Conference on Computational Science, ICCS 2011
34. Malawski, M., Meizner, J., Bubak, M., Gepner, P.: Component Approach to Computational Applications on Clouds. Procedia Computer Science 4, 432–441 (2011); Proceedings of the International Conference on Computational Science, ICCS 2011
35. Stelmach, M., Kryza, B., Słota, R., Kitowski, J.: Distributed Contract Negotiation System for Virtual Organizations. Procedia Computer Science 4, 2206–2215 (2011); Proceedings of the International Conference on Computational Science, ICCS 2011
36. Mosurska, Z., Zając, M., Pająk, R.: PL-Grid. Travel guide The Best of Kraków 2010 (2010) ISBN 978-83-926949-3-9
37. Mosurska, Z., Zając, M., Pająk, R.: PL-Grid. Travel guide The Best of Kraków 2011 (2011) ISBN 978-83-926949-5-3
38. Pająk, R., Mosurska, Z., Pilipczuk, M.: Polska Infrastruktura Informatycznego Wspomagania Nauki w Europejskiej Przestrzeni Badawczej PL-Grid. Gazeta Wyborcza (February 16, 2011)
39. Pająk, R., Mosurska, Z., Pilipczuk, M., Kitowski J.: Polska Infrastruktura Informatycznego Wspomagania Nauki w Europejskiej Przestrzeni Badawczej PL-Grid. Newsweek Polska 29/2011 (July 24, 2011)
40. Superkomputer Zeus z AGH najszybszy w Polsce (a list of articles concerning PL-Grid in mass media). ACK Cyfronet AGH website (2010), http://www.cyf-kr.edu.pl/?a=komunikat20101115_00

Glossary of Terms

Basic Execution Service (BES). A specification which defines Web Service interfaces for creating, monitoring, and controlling computational entities such as UNIX or Windows processes, Web Services, or parallel programs within a defined environment. Clients define activities using the Job Submission Description Language (JSDL).

(source: OGSA Basic Execution Service (OGSA BES) Specification, Ver. 1.0)

Cherenkov Telescope Array (CTA). The CTA project intends to build the next generation ground-based very-high-energy gamma-ray instrument. It will serve as an open observatory for the general astrophysics community and will provide deep insight into the non-thermal high-energy universe.

Common Platform Enumeration (CPE). A specification that provides a uniform (though perhaps not intuitive), structured naming scheme for known IT systems and platforms. It is based on the general URI (Uniform Resource Identifiers) syntax. In addition to a formalized naming format it also includes a language for providing descriptions of complex platforms (CPE).

Common Vulnerabilities and Exposures (CVE). One of the most widely recognized repositories of IT security vulnerabilities, providing a broad enumeration of publicly known security weaknesses in software produced by many different vendors, expressed in an understandable, uniform manner. Currently the database contains over 40 thousand entries and this number is continuously increasing.

Common Vulnerability Scoring System (CVSS). A specification which provides a common language of severity assessment for IT security vulnerabilities that may exist in a system or infrastructure. It uses three compound metrics (Base, Temporal and Environmental) and takes into account factors such as: access complexity, potential impact, ease of exploitability, whether authentication is necessary, etc. Partial scores from individual metrics are then combined using an algorithm specified with CVSS.

Computational Grant. An allocation of computing time or storage space on computing resources that is provided to researcher(s) to support research on some declared topic. In PL-Grid a computational grant is assigned to a specific period in time and requires an accounting process.

Computer Security Incident Response Team (CSIRT). A group that handles computer security incidents.

Data-intensive application. A computer application which performs a large number of I/O operations and, as a consequence, heavily exploits storage resources. A common feature of all data-intensive applications is that their execution time is mainly dependent on the performance of storage resources rather than computing resources. Most enterprise applications (e.g. transaction systems, web applications) as well as numerous scientific applications (e.g. mesh computations, reality simulations) produce or consume vast quantities of data – thus, they are considered data-intensive applications.

Distributed Resource Management Application API (DRMAA). A generalized API for distributed resource management systems (DRMSs) used to facilitate integration of application programs. The scope of DRMAA is limited to job submission, job monitoring and control, and retrieval of the finished job status.

(source: GWD-R.133 – Distributed Resource Management Application API Specification 1.0)

e-Infrastructure. A research environment in which all researchers – whether working in the context of their home institutions or national or multinational scientific initiatives – have shared access to unique or distributed scientific facilities (including data, instruments, computing and communications), regardless of their type and geographical location.

(source: `http://e-irg.eu`)

Eclipse Parallel Tools Platform (Eclipse PTP). An open-source platform build on top of Eclipse that provides an environment specifically designed for parallel application development. Eclipse PTP offers a standard, portable parallel IDE that supports a wide range of parallel architectures and runtime systems, a parallel debugger, integration with a wide range of parallel tools as well as an UI that simplifies end-user interaction with parallel systems.

(source: `http://www.eclipse.org/ptp/`)

European Grid Infrastructure (EGI). Federation of resource providers set up to deliver sustainable, integrated and secure computing services to European researchers and their international partners.

(source: EGI Glossary of Terms)

Executable Paper. A novel concept in the field of scientific publishing where a standard (text-based) publication is extended with tools which enable readers or reviewers to reenact the computations reported upon in the publication, review their results and validate the software used to obtain such results. The development of computing environments which support executable papers, made possible by the advent of advanced Web authoring tools and distributed computing technologies, has been picking up pace in the recent years and is the subject of focus of major scientific publishing houses.

Federated e-Infrastructure. An e-Infrastructure which is composed of autonomous (i.e. maintaining their own policies) e-Infrastructure providers.
(source: EGI Glossary of Terms)

GP GPU computing. The concept of using General Purpose Graphic Processing Units to perform computations in applications traditionally handled by Central Processing Units.
(source: `http://en.wikipedia.org/wiki/GPGPU`)

Grid. An IT infrastructure concerned with the integration, virtualisation, and management of services and resources in a distributed, heterogeneous environment that supports collections of users and resources (Virtual Organisations) across traditional administrative, trust and organisational boundaries (real organisations).
(source: OGF GFD-I.181)

Gridbean (extension used in the UNICORE middleware). An object responsible for generating a job description for grid services and providing a graphical user interface for handling input and output data. Typically, a gridbean is composed of a job description generation module and a graphical user interface module. The gridbean stores parameters with corresponding values and uses them to generate the job description. The job description is requested by the client application prior to submission, and is then received from the gridbean.
(source: P. Bala, K. Baldridge, E. Benfenati, M. Casalegno, U. Maran, K. Rasch, B. Schuller: UNICORE – a successful middleware for Life Sciences Grids. In: M. Cannataro (Ed.) Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and Healthcare IGI 2009, pp. 615-643)

GridSpace2. A novel virtual laboratory framework enabling researchers to conduct virtual experiments on Grid-based resources and other HPC infrastructures. GridSpace2 facilitates exploratory development of experiments by means of scripts which can be expressed in a number of popular languages, including Ruby, Python and Perl. The framework supplies a repository of gems enabling scripts to interface low-level resources such as PBS queues, EGEE computing elements, LFC directories and other types of Grid resources. Moreover, GridSpace2 provides a Web 2.0-based Experiment Workbench supporting joint development and execution of virtual experiments by groups of collaborating scientists.

Incident (in Service Operation). An unplanned interruption to an IT service or reduction in the quality of an IT service.
(source: ITIL glossary)

Intrusion Detection System (IDS). A group of computer security tools comprising all systems that are able to detect and report malicious activities within the observed infrastructure(s).

Intrusion Prevention System (IPS). The next step in the evolution of IDS, extending its capabilities to include active counter-measures against malicious system(s) upon detection. Typical actions include blocking network traffic to/from problematic host(s) and sending abuse reports to network owners.

ITIL. A set of Best Practice guidelines for IT Service Management. ITIL is owned by OGC and consists of a series of publications providing guidance on developing Quality IT Services, and on the processes and facilities needed to support them.

Job Profile. An XML-based job definition language which allows the user to specify the requirements of large-scale parallel applications together with the complex parallel communication topologies.
(source: K. Kurowski, T. Piontek, P. Kopta, M. Mamoński, B. Bosak: Parallel Large Scale Simulations in the PL-Grid Environment. In: Computational Methods in Science and Technology, Special Issue 2010, pp. 47-56)

Job Submission Description Language (JSDL). An extensible XML language for describing the requirements of computational jobs for submission to resources, particularly in Grid environments, though not restricted to the latter. The JSDL language contains a vocabulary and a normative XML schema that facilitates the expression of those requirements as a set of XML elements.
(source: Job Submission Description Language (JSDL) Specification, Ver. 1.0)

Large Hadron Collider (LHC). The world's largest and most powerful particle accelerator, operated by CERN.

Meta-alert. A group of alerts aggregated by the correlation process. A meta-alert is a directed, acyclic, connected graph with exactly one root node. A meta-alert may consist of alerts and other meta-alerts, organized in parent-child relations.

Multi Criteria Decision Analysis (MCDA). A subdiscipline of operations research that explicitly considers multiple criteria in decisionmaking environments. Whether in our daily lives or in professional settings, there are typically multiple conflicting criteria (such as the cost and quality) that need to be evaluated when making decisions.

(source: `http://en.wikipedia.com`)

National Grid Initiative or Infrastructure (NGI). A legal organisation that (a) has a mandate to represent its national Grid community in all matters falling within the scope of EGI.eu, and (b) is the sole possessor of the mandate described in (a) for its country, thus providing a single contact point at the national level.

(source: EGI Glossary of Terms)

National Vulnerability Database (NVD). The NVD repository provides combined knowledge about publicly known IT security vulnerabilities. It relies on several other specification: a general description of the vulnerability from the CVE database, the list of affected software based on appropriate CVE entries and the severity level described with the help of the CVSS specification. The database is maintained by the NIST Computer Security Division. It is publicly available and updated approximately every two hours.

OGSA. The Open Grid Services Architecture (OGSA) is a set of standards defining the way in which information is shared among diverse components of large, heterogeneous grid systems. In this context, a grid system is a scalable wide area network (WAN) that supports resource sharing and distribution. OGSA is a trademark of the Open Grid Forum. OGSA definitions and criteria apply to hardware, platforms and software in standards-based grid computing. OGSA is, in effect, an extension and refinement of the service-oriented architecture (SOA). It addresses ongoing issues and challenges such as authentication, authorization, policy negotiation and enforcement, administration of service-level agreements, management of virtual organizations and customer data integration.

(source: `http://searchsoa.techtarget.com/definition/Open-Grid-Services-Architecture`)

Penetration test. A method of evaluating the security of a computer system or network by simulating an an attack by malicious outsiders (who do not possess authorized means of accessing the organization's systems) and malicious insiders (who have some level of authorized access).

PL-Grid Infrastructure. A federated infrastructure built and maintained by the PL-Grid consortium. Services include various grid tools and other IT services related to computing and storage resources. PL-Grid plays the role of the Polish National Grid Initiative in EGI.

Portal. A website designed to serve as a point of access for a specific group of users. It aggregates various tools and features and presents them in a uniform manner.

Public Key Infrastructure (PKI). A set consisting of hardware, software, actors, policies and procedures required to create, manage, distribute, use, store and revoke digital certificates.

Quality of Service (QoS). The collective effect of service performance that determine the degree of satisfaction of a user of the service. Note that the quality of service is characterized by the combined aspects of service support performance, service operability performance, service integrity and other factors specific to each service.

(source: TMF)

Resource Centre (RC). The smallest resource administration domain in an e-Infrastructure. It can be either localised or geographically distributed. It provides a minimum set of local or remote IT Services compliant with well-defined IT capabilities, necessary to make resources accessible to users. Access is granted by exposing common interfaces to users. Sometimes referred to as a site.

(source: EGI Glossary of Terms)

Science gateway. A set of tools designed to help solve a specific scientific problem. Components in a science gateway are configured to work together and assist the scientist. Typically, the process begins with a task definition, continues with task execution and ends with analysis of results.

Security framework. A piece of software responsible for multiple features, including: user authentication and access control (authorization) based on given policies; ensuring confidentiality, integrity and non-repudiation of data stored and processed by the system; providing some level of interoperability with other security systems (e.g. external e-Infrastructures), etc.

Security Information Management (SIM). Software that collects data from multiple agents distributed throughout an infrastructure. Upon being collected in one place, the data can be processed and correlated.

Semantic-based Autonomic Monitoring and Management (SAMM). A monitoring and management system which applies the ontology-based approach to resource and action description.

(source: W. Funika, M. Kupisz, P. Koperek: Towards autonomous semantic-based management of distributed applications. In: Computer Science Annual of AGH-UST, vol. 11, 2010, pp. 51-63, AGH Press, Kraków, 2010; ISSN 1508-2806)

Service. Means of delivering value to users by facilitating the outcomes they wish to achieve without requiring ownership of specific resources.
(source: ITIL V3, adapted)

Service Level Agreement (SLA). An agreement between an IT service provider and a customer. The SLA describes the IT service, documents service level targets and specifies the responsibilities of the IT service provider and the customer.
(source: ITIL, adapted)

Service Operation. Deployment and maintenance of IT services in an e-Infrastructure, as well as any related processes.

Site. See Resource Centre.

Software code audit. A comprehensive analysis of source code in a programming project with the intent of discovering bugs, security breaches or violations of programming conventions.

Static Security Control. An approach to control security in complex IT infrastructures, basing on analysis of configuration data obtained from each system or application. An alternative approach is dynamic security control, based on agents running on particular systems. While static control generally provides less information than dynamic control, it can combine many pieces of information that remain unchanged (or are rarely modified), permitting control of systems where no additional software (e.g. monitoring agents) may be installed and executed.

Storage resource. A system or device which stores data. Storage resources can be volatile (e.g. operating memory) or persistent (e.g. hard disk). Volatile storage resources require continuous power – otherwise they lose the stored information. They are characterized by high speed and low capacity, thus they are usually used for caching. Persistent storage devices do not lose information upon being powered down. They are characterized by slow access rates and large capacities, which means that their application area is wider than in the case of volatile storage resources. Examples of persistent storage resources include magnetic tapes, hard disks, compact disks, flash disks, etc.

UNICORE. The UNICORE software provides a general framework for running user applications. Jobs to be run on the same system can be grouped in job groups. Jobs can have complicated structures that can be run on different systems with various job subgroups and file transfers ordered with user-defined dependency. The user input is mapped to system-specific commands and options by the UNICORE infrastructure. UNICORE is compliant with the Open Grid Services Architecture and uses standards such as WSRF 1.2 and job submission definition language JSDL 1.0. UNICORE is based on a three-tier model, involving users, servers and target systems. The user tier consists of a graphical user interface written as a Java application. This interface enables jobs to be prepared and controlled. It is also used to set up and maintain the user's security environment. UNICORE security is based on the Secure Socket Layer (SSL) protocol and X.509 certificates.

(source: P. Bala, K. Baldridge, E. Benfenati, M. Casalegno, U. Maran, K. Rasch, B. Schuller: UNICORE – a successful middleware for Life Sciences Grids. In: M. Cannataro (Ed.) Handbook of Research on Computational Grid Technologies for Life Sciences, Biomedicine and HealthCare IGI, pp. 615-643 (2009))

UVOS. UNICORE VO Service (UVOS) is a client-server system, developed for use as an additional tool for large distributed systems. Grid systems, especially UNICORE grid middleware, are the mainstay of the UVOS system. Although UVOS can be used with different systems, it is designed primarily for UNICORE.

(source: K. Benedyczak, M. Lewandowski, A. Nowiński and P. Bała: UNICORE Virtual Organizations System. In: Parallel Processing and Applied Mathematics, Lecture Notes in Computer Science, 2010, Vol. 6068/2010, pp. 155-164)

Vine Toolkit. An open-source software framework that is used to create Grid-aware web applications.

Virtual Laboratory. An interactive environment for creating and conducting simulated experiments: a playground for experimentation.

(source: The Virtual Laboratory Environment @ Algorithmic Botany)

A heterogeneous distributed problem solving environment that enables a geographically distributed group of researchers to work together on a common set of projects.

(source: Lester, `http://lester.rice.edu/`)

A heterogeneous distributed environment, which allows a group of scientists from different sites to work on one project. As in all other laboratories, the equipment and techniques are specific for a given field of activity.

Virtual Machine (VM). Virtualization technology that enables running operating system instances in domains that share common system resources. Such domains are isolated environments, managed by a hypervisor (a.k.a Virtual Machine Monitor – VMM). The hypervisor performs emulation of arbitrary hardware resources, enabling a number of Virtual Machines to run concurrently. Each VM has the illusion of owning hardware and is able to support an arbitrary guest operating system. Hypervisors usually do not have fixed limits on the number of OS instances and support fine-grained resource control settings.

Virtual Machine Appliances. Pre-engineered and validated units of the IT infrastructure that conform to standards and best practices set forth by design patterns, facilitating deployment, scalability, security and ease of integration with other services. Prepared VM appliance packages can be bundled as units of assemblies with configuration templates defined by administrators, stored in repositories and provisioned on demand. A VM appliance is a basic deployment object, partially configured to expose a database, an application server or other element of functionality in the scope of various operating systems and their versions. Reference configurations are prepared and tested on development servers. Following validation, certification and approval, snapshots are created and stored in a repository with metadata describing the installed software and its VM versions.

Virtual Machine Set. A network of interconnected VM instances with preinstalled software (VM appliances). Each VM Set can be deployed on the provider's infrastructure and offered to the user for a specified period of time. The specification of a VM Set consists of five parts, covering the properties of the desired (i) VM Appliances, (ii) network interconnections, (iii) shared storage, (iv) user access policy, and (v) lease period.

Virtual Organisation (VO). A group of people (e.g. scientists, researchers) with common interests and requirements who need to work collaboratively and/or share resources (e.g. data, software, expertise, CPU, storage space) regardless of geographical location. They form a VO in order to access resources to meet these needs, agreeing to a set of common rules and policies that govern their access and security rights.

(source: EGI Glossary of Terms)

Wallet (IT security). A mechanism allowing secure storage of user credentials such as passwords or keys. Its main goal is to provide confidentiality and integrity of the stored data (e.g. by use of cryptography). It also needs to provide authentication and authorization mechanisms to ensure that the data is accessible only to legitimate persons.

Web Services Notification (WSN). A set of specifications that standardize the way Web services interact using "Notifications" or "Events". These specifications provide a common way for a Web service to disseminate information to other Web services, without requiring prior knowledge on these other Web services. They can be thought of as a "Publish/Subscribe model for Web services".

(source: `www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsn`)

Worldwide LHC Computing Grid (WLCG). The Worldwide LHC Computing Grid (WLCG) is a global collaboration which aims to build and maintain a data storage and analysis infrastructure for the high-energy physics community operating the Large Hadron Collider at CERN.

WSRF. The purpose of the Web Services Resource Framework (WSRF) is to define a generic framework for modeling and accessing persistent resources using Web services so that the definition and implementation of a service and the integration and management of multiple services is made easier. WSRF introduces the idea of an XML document description, called the Resource Properties document schema, which is referenced by the WSDL description of the service and which explicitly describes a view of the shopping cart, printer, print job, or any other entity with which the client interacts.

(source: `http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-`cd-02.pdf)

Subject Index

Author Index