DISTRIBUTED COMPUTING AND GRID-TECHNOLOGIES IN SCIENCE AND EDUCATION

Proceedings of the 4th International Conference

Dubna, June 28 – July 3, 2010

GRID
2010

РАСПРЕДЕЛЕННЫЕ ВЫЧИСЛЕНИЯ И ГРИД-ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ

Труды 4-й международной конференции

Дубна, 28 июня – 3 июля 2010 г.

PARALLEL.RU

РФФИ

RGP
ЗАО «Ризл Гео Проджект»
Программное обеспечение Autodesk
Разработка геоинформационных систем

NIAGARA
Distribution Company

SUPERMICRO®
SUPER MICRO COMPUTER INC.

МЕЖДУНАРОДНАЯ АКАДЕМИЯ НАУК
РУССКАЯ СЕКЦИЯ

Joint Institute for Nuclear Research
Laboratory of Information Technologies

DISTRIBUTED COMPUTING AND GRID-TECHNOLOGIES IN SCIENCE AND EDUCATION

Proceedings of the Fourth International Conference

Dubna, June 28 – July 3, 2010

РАСПРЕДЕЛЕННЫЕ ВЫЧИСЛЕНИЯ И ГРИД-ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ

Труды четвертой международной конференции

Дубна, 28 июня – 3 июля 2010 г.

Дубна • 2010

УДК [004.7+004.9](063)
ББК 32.988-5я431
32.97я431
D63

Organized by the Joint Institute for Nuclear Research,
the Laboratory of Information Technologies
under the sponsorship of the Russian Foundation for Basic Research,
Supermicro Computer,
NIAGARA,
Real Geo Project

Media partners:
PARALLEL.RU,
Russian Section of International Academy of Sciences

The contributions are reproduced directly from the originals presented by the Organizing Committee.

Distributed Computing and Grid-Technologies in Science and Education: Proceedings
D63 of the Fourth International Conference (Dubna, June 28 – July 3, 2010). — Dubna: JINR, 2010. — 452 p.

ISBN 978-5-9530-0269-1

The Proceedings of the Fourth International Conference «Distributed Computing and Grid-Technologies in Science and Education» (Grid'2010) include the reports presented at the Grid'2010, which was held in Dubna on June 28 – July 3, 2010. The Conferences are held every two years by the JINR Laboratory of Information Technologies. This Conference was devoted to the 80th anniversary of the birth of N. N. Govorun (1930–1989), an outstanding scientist, a corresponding member of the USSR Academy of Sciences, a former Deputy Director and Director of the Laboratory of Computing Techniques and Automation. The Conference is a unique one conducted in Russia on the issues relating to the use of Grid-technologies in various areas of science, education, industry and business. Presentations of the Conference are available at the Conference web-page http://grid2010.jinr.ru/program.php

Распределенные вычисления и грид-технологии в науке и образовании: Труды четвертой международной конференции (Дубна, 28 июня – 3 июля 2010 г.). — Дубна: ОИЯИ, 2010. — 452 с.

ISBN 978-5-9530-0269-1

Труды четвертой международной конференции «Распределенные вычисления и грид-технологии в науке и образовании» (Grid'2010) содержат доклады, представленные на «Grid'2010», которая проходила с 28 июня по 3 июля 2010 г. в Дубне. Конференция проводится раз в два года Лабораторией информационных технологий ОИЯИ. «Grid'2010» была посвящена 80-летию со дня рождения выдающегося ученого, первого заместителя директора и директора Лаборатории вычислительной техники и автоматизации, члена-корреспондента АН СССР Николая Николаевича Говоруна (1930–1989). Это единственная в России конференция, посвященная проблемам, связанным с использованием грид-технологий в различных областях науки, образования, промышленности и бизнеса. Презентации докладов, представленные на конференции, размещены на веб-странице конференции http://grid2010.jinr.ru/program.php

УДК [004.7+004.9](063)
ББК 32.988-5я431
32.97я431

ISBN 978-5-9530-0269-1

General Information

The 4th International Conference "Distributed Computing and Grid-technologies in Science and Education" was held from June 28 to July 3, 2010 in the Laboratory of Information Technologies of the Joint Institute for Nuclear Research (Dubna, Russia). The Conference was attended by 252 participants from 21 countries: Armenia, Belarus, Bulgaria, Hungary, Germany, Greece, Georgia, Iceland, Kazakhstan, Moldova, Myanmar, Poland, Russia, Romania, USA, Uzbekistan, Ukraine, France, Czechia, Switzerland, Sweden as well as from CERN and JINR. Russia was presented by participants from 56 universities and research centers. The Conference included 8 sections: WLCG - Worldwide LHC Computing Grid, grid-applications, grid in business, distributed computing and grid-technologies in education, GridNNN – Grid of the National Nanotechnology Network, methods and algorithms for distributed computing, grid-infrastructure and "cloud" computing. In frames of the conference round tables were organized on using grid-technologies in business and on training in grid-technologies and their application in education. A training course was held on the integrated infrastructure, tools and methods for support of the scientific applications development in Grid and the systems of voluntary distributed computing.

Advisory Committee

Abdinov O. (IoP, Baku, Republic of Azerbaijan), Abramov S.M. (PSI RAS, Pereslavl-Zalesskii, Russia), Afanasiev A.P. (ISA RAS, Moscow, Russia), Antoniou I. (Aristotle University of Thessaloniki, Greece), Bird I. (CERN), Bogdanov A.V. (IHPC&DB, St.Petersburg, Russia), Brun R. (CERN), Buzatu F. (Institute for Atomic Physics Magurele, Romania), Chetverushkin B.N. (Keldysh Institute of Applied Mathematics, Moscow), Cleymans J. (Cape Town University, SA), Dimitrov V. (Sofia University, Republic of Bulgaria), Dulea M. (IFIN-HH, Romania), Golutvin I.A. (JINR), Gusev V.V. (IHEP, Protvino, Russia), Ilyin V.A. (SINP Moscow State University, Russia), Ivannikov V.P. (ISP RAS, Moscow, Russia), Jones B. (CERN), Kadyshevsky V.G. (JINR), Kitowski J. (CYFRONET, Republic of Poland), Klementov A. (BNL, USA), Kostomarov D.P. (Moscow State University, Russia), Korolev L.N. (Moscow State University, Russia), Kopcansky P. (IEP SAS, Kosice, Slovak Republic), Kryuchkyan G.Yu. (Yerevan State University, Armenia), Lakhno V.D. (IMPB Russian Academy of Sciences, Russia), Metakides G. (University of Patras, Greece), Manh Shat Nguyen (JINR), Musial G. (Institute of Physics, AMU, Poznan, Poland), Nergui B. (Institute of Informatics MAS, Mongolia), Platonov A.P. (RIPN, Moscow, Russia), Ryabov Yu.F. (PNPI, Gatchina, Russia), Sahakyan V.G. (IIAP NAS Armenia), Shumeiko N.M. (NC PHEP, Minsk, Republic of Belarus), Shirkov D.V. (JINR), Sissakian A.N. (JINR), Smirnova O.G. (NDGF/University of Lund, Sweden), Solomonides T. (UWE, Bristol, UK), Shoukourian Yu. (IIAP NAS Armenia), Vaniachine A.V. (Argonne National Laboratory, USA), Voevodin V.V. (SRCC Moscow State University, Russia), Zhizhin M.N. (CGDS RAS, Russia), Zhuchkov A.V. (IChPh RAS, Russia), Zinovjev G. (ITP, Kiev, Ukraine).

Organizing Committee (JINR)

Ivanov V.V. – Chairman, Korenkov V.V. – Vice Chairman, Strizh T.A. – Scientific Secretary, Adam S., Aristarkhova M.V., Bulyga N.I., Fedorova E.A., Grafov A.N., Katraseva T.I., Lukyanov S.O., Novikova V.K., Podgainy D.V., Prikhodko A.V., Rudneva E.M., Rumyantseva O.Yu., Streltsova O.I., Tikhonenko E.A., Zrelov P.V.

MAIN TOPICS

- questions of creation and experience of exploitation of grid-infrastructures;
- methods and technologies of distributed computations; architecture and algorithms;
- network infrastructure for distributed data processing and storing;
- algorithms and methods of solving applied problems in distributed computing media;
- theory, models and methods of distributed data processing;
- distributed information systems: construction technologies and usage experience;
- Grid applications in science and education: physics, chemistry, biology, biomedicine, Earth sciences, etc.);
- Grid applications in business,
- cloud computing and consolidation of distributed resources.

Общая информация

Четвертая Международная конференция "Распределенные вычисления и Грид-технологии в науке и образовании", проводимая раз в два года Лабораторией информационных технологий, проходила в Объединенном институте ядерных исследований с 28 июня по 3 июля 2010 г. Конференция собрала 252 участника из 21 страны: Армении, Белоруссии, Болгарии, Венгрии, Германии, Греции, Грузии, Исландии, Казахстана, Молдавии, Мянмы, Польши, России, Румынии, США, Узбекистана, Украины, Франции, Чехии, Швейцарии, Швеции, а также ЦЕРНа и ОИЯИ. Россия была представлена участниками из 56-ти университетов и исследовательских центров. На конференции была организована работа 8 секций (WLCG – Всемирный грид для обработки данных с Большого адронного коллайдера в ЦЕРН, грид-приложения, грид в бизнесе, распределенные вычисления и грид-технологии в образовании, ГридННС – грид национальной нанотехнологической сети, распределенные вычисления: методы и алгоритмы, грид-инфраструктура и «облачные» вычисления). На нынешней конференции были проведены круглый стол по использованию грид-технологий в бизнесе и круглый стол по вопросам применения грид-технологий в образовании и обучения грид-технологиям. Во время конференции был проведен тренинг на тему «Интегрированная инфраструктура, инструменты и методы для поддержки разработки научных приложений в грид и системах добровольных распределенных вычислений».

Программный комитет

Абдинов О. (Институт физики, Баку, Азербайджанская Республика), Абрамов С.М. (ИЦМС ИПС РАН, Россия), Афанасьев А.П. (ИСА РАН, Москва, Россия), Антониу И. (Университет имени Аристотеля, Салоники, Греция), Берд Я. (ЦЕРН), Богданов А.В. (ИВВИС, Санкт-Петербург, Россия), Бран Р. (ЦЕРН), Бузату Ф. (IFIN-HH, Румыния), Ваняшин А.В. (Аргоннская национальная лаборатория, США), Воеводин В.В. (НИВЦ МГУ, Россия), Голутвин И.А. (ОИЯИ), Гусев В.В. (ИФВЭ, Протвино, Россия), Димитров В. (Софийский университет, Республика Болгария), Джонс Б. (ЦЕРН), Дулеа М. (IFIN-HH, Румыния), Жижин М.Н. (Центр геофизических данных РАН, Россия), Жучков А.В. (ИХФ РАН, Россия), Зиновьев Г. (ИТФ, Киев, Украина), Ильин В.А. (НИИЯФ МГУ, Россия), Кадышевский В.Г. (ОИЯИ), Китовски Я. (CIFRONET, Республика Польша), Клейманс Д. (Университет, Кейптаун, ЮАР), Клементов А. (БНЛ, США), Костомаров Д.П. (МГУ, Россия), Копчанский П. (ИЭФ САН, Кошице, Словакия), Королев Л.Н. (МГУ, Россия), Крючкян Г.Ю. (Ереванский университет, Армения), Лахно В.Д. (ИМПБ РАН, Россия), Метакидес Г. (Университет Патрас, Греция), Мусял Г. (ИФ Унив. Им. А.Мицкевича, Познань, Польша), Нгуен Мань Шат (ОИЯИ), Нэргуй Б. (Институт информатики МАН, Монголия), Платонов А.П. (РосНИИРОС, Москва, Россия), Рябов Ю.Ф. (ПИЯФ РАН, Россия), Саакян В.Г. (ИПИА, Армения), Сисакян А.Н. (ОИЯИ), Смирнова О.Г. (NDGF/Университет Лунда, Швеция), Солдатов А.А. (РНЦ КИ, Россия), Соломонидес Т. (УЗА, Бристоль, Великобритания), Четверушкин Б.Н. (ИПМ им. М.В.Келдыша, Москва, Россия), Ширков Д.В. (ОИЯИ), Шукурян Ю. (ИПИА, Армения), Шумейко Н.М. (НЦ ФЧВЭ БГУ, Минск, Белоруссия).

Организационный комитет (ОИЯИ)

Иванов В.В. - председатель, Кореньков В.В. - зам. председателя, Стриж Т.А. - ученый секретарь, Адам Г., Аристархова М.В., Булыга Н.И., Графов А.Н., Зрелов П.В., Катрасева Т.И., Лукьянов С.О., Новикова В.К., Подгайный Д.В., Приходько А.В., Руднева Е.М., Румянцева О.Ю., Стрельцова О.И., Тихоненко Е.А., Федорова Е.А.

ТЕМАТИКА

- вопросы создания и опыт эксплуатации грид-инфраструктур;
- методы и технологии распределенных вычислений, вопросы архитектуры;
- сетевая инфраструктура для распределенной обработки и хранения данных;
- алгоритмы и методы решения прикладных задач в распределенных вычислительных средах;
- теория, модели и методы распределенной обработки данных;
- распределенные информационные системы: технологии построения и опыт использования;
- грид-приложения в науке (физика высоких энергий, вычислительная химия, биология и биомедицина, науки о Земле, и т.д.);
- грид-приложения в образовании;
- грид-приложения в бизнесе;
- облачные вычисления и консолидация распределенных ресурсов.

CONTENTS / СОДЕРЖАНИЕ

NETWORKING, COMPUTING AND COMPUTATIONAL PHYSICS IN JINR ACTIVITIES

Gh. Adam, S. Adam, V.V. Ivanov, V.V. Korenkov, T.A. Strizh, P.V. Zrelov

Laboratory of Information Technologies
Joint Institute for Nuclear Research, Dubna
ivanov@jinr.ru

The main tasks of the Laboratory of Information Technologies (LIT) assume the provision of the JINR and its partner institutions in the JINR Member States and in other countries with top telecommunication, network and information resources, as well as top research in computational mathematics and computational physics aimed at solving specific problems arising in experimental and theoretical studies conducted at JINR or with its participation. The strength and importance of LIT results in these fields are presented, along with the basic principles and general perspectives.

INTRODUCTION

The activity of the Laboratory of Information Technologies (LIT) in the Joint Institute for Nuclear Research (JINR) is devoted to the provision of the basic research in frontier particle, nuclear, and condensed matter physics conducted with direct participation of JINR with the most advanced working tools related to the Networking, Computing, Computational Physics.

The great diversity of the research problems in the JINR Laboratories and Institutes in JINR member states asking for LIT support entails the interdisciplinary character of the LIT activity.

Experiments at the JINR basic facilities, JINR participation in the LHC experiments and in other large-scale projects asked for a substantial increase of the networking and information resources together with the deployment of a large volume of work toward the development of the JINR Grid-segment and its integration in the Russian grid-infrastructure RDIG (Russian Data Intensive Grid) and in the world-wide grid-infrastructure.

In order to fulfill these tasks, it is necessary to provide:

- development of telecommunication channels of JINR with the JINR Member States on the basis of national and regional telecommunication networks;
- fault-tolerant operation and further development of the high-speed and protected local area network of JINR;
- development and maintenance of the distributed high-performance computing infrastructure and mass storage resources;
- information, algorithmic and software support of the research-and-production activity of JINR;
- development and reliable operation of the JINR grid-segment as a component of the global grid-infrastructure;
- development of new mathematical methods and tools for modeling physical processes and experimental data analysis;
- creation of methods and numerical algorithms for modeling magnetic systems;
- elaboration of software and computer complexes for experimental data processing;
- elaboration of numerical algorithms and software for the simulation of complex physical systems;
- development of methods, algorithms and software of computer algebra;
- contribution to the development of the new generation computing tools;
- application of the developed methods and algorithms to topics in other science and technology branches (nanotechnology, biology, medicine, economy, industry, etc.).

The mathematical support involves the performance of top research in computational mathematics and computational physics, aimed at solving specific problems which arise in experimental and theoretical research carried out with the direct participation of JINR. The main part of these activities is related to the development of the mathematical description and algorithmic reformulation of the physical models such as to get significant numerical solutions; development of methods and algorithms able to extract physically insightful information from experimental data; simulation of physical processes within experimental installations; algorithm implementations into effective and reliable hardware adapted program environment.

This subject area covers a wide spectrum of studies in the high energy physics, nuclear physics, solids physics and condensed matter physics, biophysics, information technologies, conducted in close cooperation with all JINR Laboratories and research centers from the JINR Member States.

Following the decision concerning the radical improvement of the computer telecommunication links with major partner organizations in the JINR Member States during the years 2010-2015, adopted at the March 14-15, 2008 Session of the JINR Committee of the Plenipotentiary Representatives, the first steps have been undertaken towards the development of a unified grid-environment of the JINR Member States. Such an infrastructure will allow all the participating sides to effectively join their forces for solving the foreseen fundamental and applied projects in elementary particle physics, nuclear physics, condensed matter physics, computational biophysics, nanotechnologies, etc., the successful realization of which would be impossible without using highly efficient computations, new approaches to distributed and parallel computing, and large amounts of data storage. The formation of a unified grid-environment of the JINR Member States is a basis of the seven-year plan within the direction "Networks. Computing. Computational Physics".

In 2009, in frames of this work, a new telecommunication link between Dubna and Moscow, on the basis of the state-of-the-art technologies DWDM and 10Gb Ethernet, was put in operation with a capacity of 20 Gbps (two channels of 10 Gbps). In perspective, the mentioned technologies allow the creation of up to 80 channels of 10 Gbps each, resulting in a total capacity of up to 800 Gbps.

The total performance and mass storage resources of the Central Information and Computing Complex (CICC) were substantially increased. The CPU time monitoring in the year 2010 places the JINR grid-site on the 10-th place among the 163 Tier2 sites worldwide. Within the Russian Data Intensive Grid (RDIG) consortium, which comprises, besides the CICC JINR, fourteen Russian and two Ukrainian computing centres, our cluster has covered more than 43% of the RDIG share.

There is a high degree of interest to the LIT activities in the JINR Member States. We can mention protocols of cooperation with INRNE (Bulgaria), ArmeSFo (Armenia), FZK Karlsruhe GmbH (Germany), IHEPI TSU (Georgia), NC PHEP BSU (Belarus), KFTI NASU (Ukraine), IMIT UAZ (Uzbekistan), WUT (Wroclav, Poland), etc., the Hulubei-Meshcheryakov programme with Romania, the BMBF grant "Development of the grid-infrastructure and tools to provide joint investigations performed with participation of JINR and German research centers", the CERN-JINR Cooperation Agreement on several topics and the project "Development of grid segment for the LHC experiments", supported in frames of the JINR-South Africa cooperation agreement in 2006-2008.

Work was done within the participation in common projects: CERN-INTAS projects, Worldwide LHC Computing Grid (WLCG), and Enabling Grids for E-sciencE (EGEEIII) project co-funded by the European Commission (under contract number INFSO-RI-222667) through the Seventh Framework Programme. Grants were afforded by the Russian Foundation for Basic Research and five contracts were concluded with the Russian Federal Agency of Science and Innovations (FASI). Work under the SKIF-GRID project – a programme of the Belarusian-Russian Union State was performed. LIT participates, in cooperation with SINP MSU, RSC "Kurchatov Institute" and PNPI, in the Grid National Nanotechnology Network (GridNNN) project performed under the federal target programme of development of the infrastructure of the nanoindustry in the Russian Federation in 2008-2010.

NETWORKING, COMPUTING, INFORMATION SUPPORT

The key components of the infrastructure implemented by LIT comprise the JINR telecommunication data links, local area network, central information and computing complex and basic software for integration of the Institute's information and computing resources in a unified information environment accessible to all users and with heavy use of grid-technologies.

JINR telecommunication links

The available throughput of the telecommunication channel between JINR and Moscow was raised from 45 Mbps in 2003 to 20 Gbps since June 2009. The project, done in cooperation with the Russian Satellite Communications Company (RSCC), owner of the optical fiber, was realized with participation of NORTEL, JET Infosystems, Russian Institute for Public Networks (RIPN), the Computer Networks Interaction Center "MSK-IX". Figure 1 provides the channel scheme, with the three places where devices of the photonic data communication equipment were installed: the JINR Net central telecommunication node, the settlement Radishchevo, and Moscow Internet Exchange (MSK-IX).

The main JINR service-provider for access to the Internet is RBNet (Russian Backbone Network). The parent organization on the support of the RBNet network, the RIPN, is operating the international channel for science and education and is the trustworthy organization for the Internet Exchange (IX) functioning in Moscow.

The development of the segment of the international channels for science and education joining Russia with the Europe, with a throughput target of 10 Gbps in 2009, and subsequent growth in 2010-2016 is based on the connectivity with GÉANT (pan-European Communications infrastructure serving Europe's research and education community). The JINR-participating countries develop regional and national research and educational networks, many of which are being connected to the GÉANT. As a result of this joint activity, the integration of the grid-infrastructures of JINR and its Member States will be realized through the high-speed European network GÉANT. This is the overall adopted approach to the integration of the regional networks for science and education in Europe.

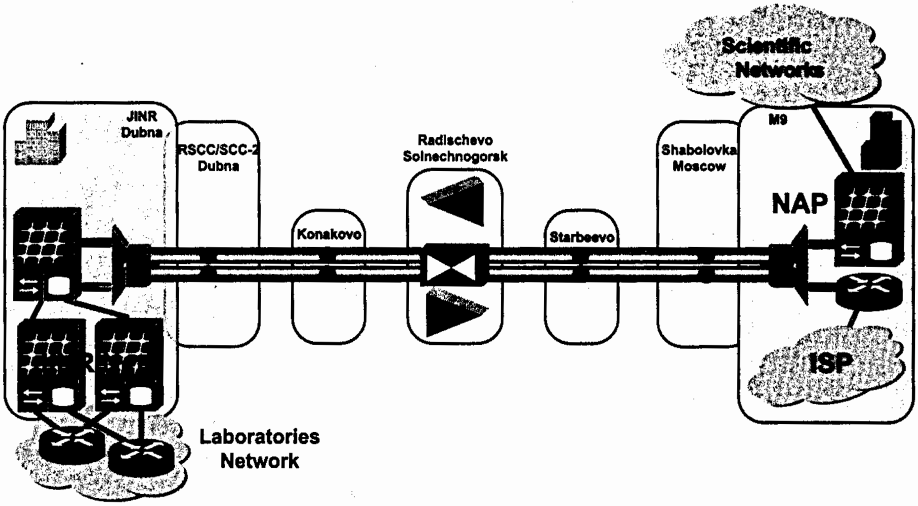


Fig. 1: Scheme of the JINR-Moscow telecommunication channel

JINR local area network

The JINR Local Area Network (LAN) currently comprises about 7000 network elements. The continuous growth of the network and computing capacities leads to certain difficulties both in the management and in providing the reliable LAN operation. To overcome them, work on creating a reliable and protected high-speed JINR LAN is done. The provision of the fail-safe work of the JINR LAN is the primary goal of the network service at LIT. The gigabit networking structure of JINR integrates the hardware and software facilities providing the basis of the JINR network and information structure, upon which the mentioned infrastructure is built up and developed (figure 2).

The gigabit networking structure solves the following tasks:

- integration of all JINR computer resources into a unified information environment;

- organization and provision of remote network access to informational – computing resources for JINR user groups, to informational resources of Russian and foreign scientific centres;
- creation of a unified information space of the JINR staff for data exchange among the Institute subdivisions and between subdivisions and JINR Directorate;
- provision of services of remote access of JINR staff to JINR resources from home PCs.

In 2010, the JINR LAN included 3696 users, more than 1500 users of mail.jinr.ru service and about 1300 users of remote access VPN. Over 120 network nodes are in round-the-clock monitoring (gateways, servers, basic switchboards, etc.). 15 servers are supported and over 40 user inquiries are served per shift. Implementation of new spam-protection systems allowed fixing about one million spam-messages a day at the central mail-servers. The present LAN characteristics are: high-speed transport (10 Gbps - 1Gbps) (Min. 100 Mbps to each PC);·controlled-access (Cisco FWM firewall module) at network entrance; general network authorization system involving many services (AFS, batch systems, Grid, JINR LAN remote access, etc.); wireless LAN access within the JINR territory.

Currently the JINR network infrastructure has direct communication lines with CERN – 10 Gbps; RBnet - 10 Gbps; RASnet - 10 Gbps; RadioMSU - 10 Gbps; GEANT - 10 Gbps; GLORIAD - 1 Gbps; Moscow - 20 Gbps.

JINR network security is achieved by the implementation of hard- and software products in the network infrastructure. To protect the computing and informational servers, user workstations and active routing and switching network equipment at JINR, the industry-approved AAA (Authentication, Authorization, and Accounting) approach is used. During the last two years, the AAA system has been successfully gradually integrated into the LIT-developed product IPDB – a network data base with multiple features of monitoring and control based on IP-addresses. The IPDB became the main tool for the network and system administrators to maintain their current administrative tasks.

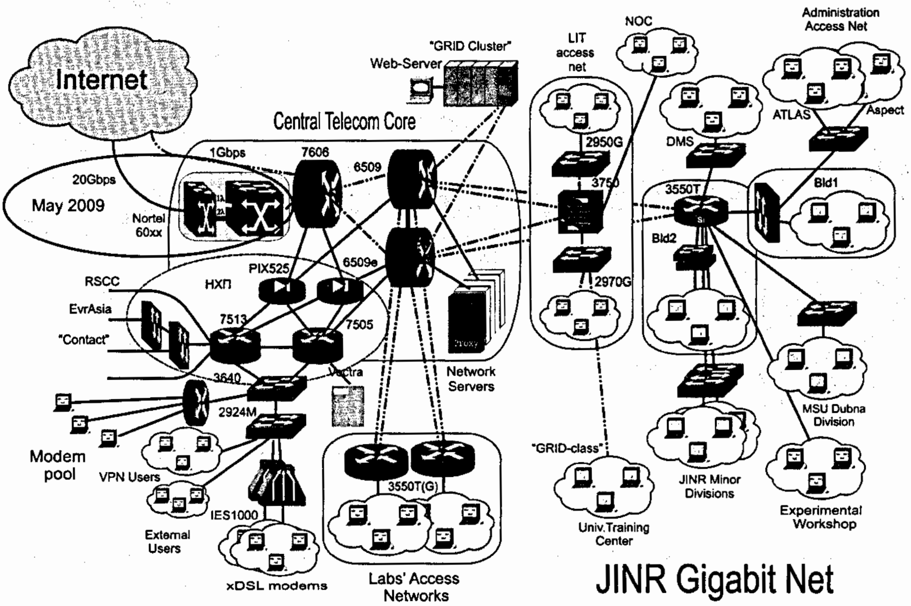


Fig. 2: Current scheme of the JINR Gigabit Network

JINR information and computing complex

The development of the JINR Central Information and Computing Complex (CICC) is based on a distributed model of data processing and data storage. Such a model is in agreement with the modern concept of establishing information processing centres for scientific research based on grid-technologies. The requirements of the LHC experiments stimulate the development of a global grid-infrastructure, together with the resource centers of all the cooperating organizations. To reach the objectives in the effective processing and analysis of the experimental data, steep increase in the performance of the CICC cluster and disk space is needed.

Starting from a CICC total performance and mass storage resources of 4.3 kSPI95 and disk space 7.7 TB in 2003, the present CICC installed computing power is 2800 kSI2K and the total disk storage capacity is ~1000 TB. Figure 3 shows the current structure of the CICC resources, access and support.

All the CICC computing and data storage resources can be used both locally (to solve sequential or parallel computing applications) and globally (for distributed computations in the WLCG/EGEE grid infrastructure and in the Russian Data Intensive Grid consortium) for all the projects the JINR physicists participate in. The first CICC performance assessments using parallel application benchmarks [1] have been followed by practical home-made optimization of the exchange of information in-between the modules of the extended configuration at the end of 2008 [2]. The results pointed to relative performance levels at those of the TOP500 computers in the world, with GigaBit Ethernet and InfiniBand interconnects respectively. The learned lessons have been used during the further JINR CICC development. This contributed to the prominent position of our system within the LHC virtual organizations.

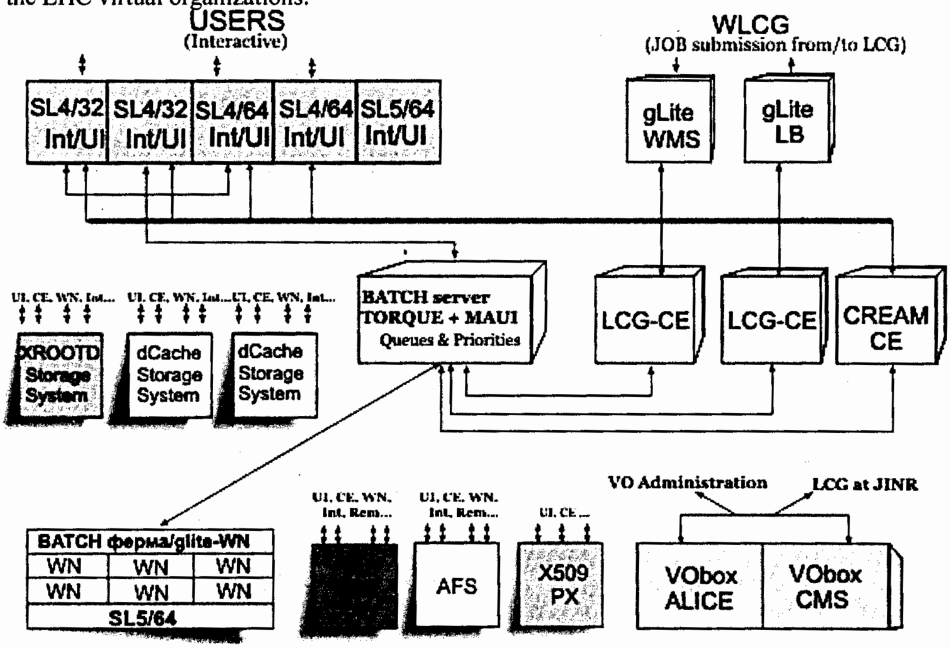


Fig. 3: Current structure of the CICC (resources, access and support)

The system software has been tuned in an optimal way, providing maximal use of computing resources and the most universal and secure access to the data storage. The *Torque* batch system and the *Maui* scheduler are used for computing resources allocation and accounting.

Basically, access to data is provided by the *dCache* system and partially via NFS. The access to the general-purpose software and user home catalogs is provided by the *Andrew File System* (AFS). The *Kerberos5* system is used for registration and authentication of local users.

JINR Grid-segment

The development of the JINR grid-segment and increase of the productivity of the JINR CICC have been started in 2003 as a means to fulfill the requirements of the large scale LHC experiments in high energy physics and relativistic nuclear physics. The JINR is an active member of the Russian consortium RDIG (Russian Data Intensive Grid) which was set up in September 2003 as a national federation in the EGEE project (http://www.eu-egee.org/). Within the RDIG consortium, which comprises, besides the CICC JINR, fourteen Russian and two Ukrainian computing centres, our cluster has covered more than 43% of the RDIG share (fig.4).

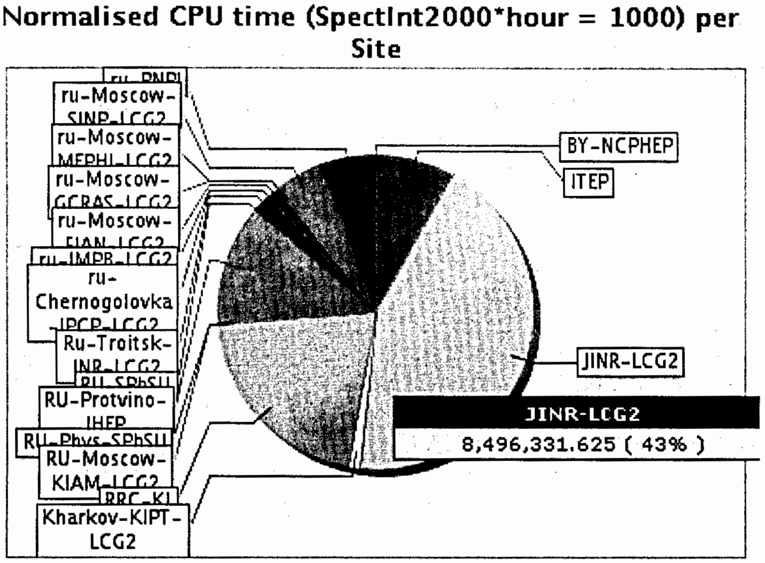


Fig.4: Normalised CPU time per RDIG sites for January–September 2010

The LHC Computing Grid (LCG) has foreseen the design and creation of distributed information and computing systems. The LCG project entered a new phase in 2006, asking for the construction of a global infrastructure of the regional centers intended for processing, storage and analysis of data at the moment of the accelerator start-up. The project is referred to as WLCG. A three-party MoU, signed by CERN, Russia and JINR in September 2007, defined the performance targets requested by the participation in this project and financial obligations assumed by Russia and JINR.

The CICC based JINR grid-segment provides to the WLCG environment basic and special services, PS and testing infrastructure, as well as software for VOs.

The JINR participation in the WLCG solved all the Pre-Challenge production tasks for the ALICE, ATLAS, and CMS experiments. For the running phase of these experiments, LIT JINR provides: computing and data storage resources; data replication to the JINR data storage system and participation in the Monte-Carlo physical events mass production in accordance with the JINR physicists' scientific program requests.

All the necessary conditions for distributed data analysis have been met at the JINR computing center (grid-site JINR-LCG2) to make possible full-fledged participation of the JINR physicists in the experiments at the LHC running phase. The contributions, during 2010, of the JINR-LCG2 site within RDIG to ALICE - 37%, ATLAS - 46%, and CMS - 37%.

The LIT JINR team has gained a valuable experience in the development and design of the grid monitoring and accounting systems [3] for RDIG (within the EGEE project) and other projects. The main point is to find scalable, reliable and interoperable solutions for the grid monitoring and accounting in real grid projects. The major part of the results obtained within this work is available as ready-to-use software packages. Tracking the current services' state as well as the history of state changes allows rapid error fixing, planning future massive productions, revealing regularities of grid operation and many other things. Alongside with monitoring, the accounting is an area which shows how the grid is utilized by virtual organizations and individual users.

A group of LIT specialists take active part in the LHC Dashboard development (grid monitoring system for the LHC experiments) [4]. To get the visualization of the monitoring, this system maps the grid infrastructure objects, processes and events on a geographic map. By using geographic information system (GIS) applications like Google Earth, quite an informative and visually attractive representation is achieved. It shows graphically real time animated information covering data flows for both Monte-Carlo Production and Tier0 export, and additional information about running jobs on the ATLAS, CMS, LHCb and Alice (jobs only) grids [5].

The LIT participates in the LCG Monte-Carlo Events Data Base (MCDB) creation. MCDB is a special knowledge base designed to keep event samples for the LHC community. The possibility to make an automated Monte-Carlo simulation chain, partially based on usage of HepML [6] and LCG MCDB [7], was already validated for use in the CMS experiment [8].

The LIT home-made events database and repository of generators were also created. Dynamical home-page [9] has been created for testing Monte Carlo Generators of physical processes. The page also allows one to estimate the main properties of hadron-nucleus and nucleus-nucleus interactions (includes FRITIOF model, HIJING model, and tools for Glauber and Reggeon theories calculation). The server HEPWEB was integrated into the Dubna-grid environment.

The TDAQ ATLAS team at LIT has brought a significant contribution to the development of the project TDAQ ATLAS at CERN [10]. The system of remote access in real time (SRART) for monitoring and quality assessment of the ATLAS data was put in operation at JINR. The methods and approaches developed for the remote access to the LHC experiments and the choice on their basis of a technique of integration of the JINR-developed SRART prototype into the experiment infrastructure are at the state-of-the-art level in the development of large-scale information projects on the creation of geographically distributed grid-system data processing. They secure conditions for participation of the scientists from JINR and other centres in the present-day studies on nuclear physics and particle physics. The work was supported by the Federal Agency on Science and Innovations of Russia, state contract No. 02.514.11.4083 and reported to the resulting conferences in 2010 [11].

The development of an educational program on grid technologies for scientists from JINR and the Member States, students, PhD-students and the teaching staff of Dubna High schools is a strategic task. To stimulate the active usage of the WLCG resources, user support consisting in special courses, lectures, and trainings was organized [12]. The educational, training and testing grid infrastructure was built-up and is intensively used for a wide range of tasks related to the training of different groups in grid-technologies as well as research and development activities in this field [13]. Grid-infrastructure for training and education is a first step towards the creation of the JINR Member-States grid-infrastructure. At the moment it involves three grid sites located at JINR and one site in each of the following organizations: the Institute of High-Energy Physics - IHEP (Protvino), the Institute of Mathematics and Information Technologies AS of Republic of Uzbekistan – IMIT (Tashkhent, Uzbekistan), Sofia University "St. Kliment Ohridski" - SU (Sofia, Bulgaria), the Bogolyubov Institute for Theoretical Physics - BITP (Kiev, Ukraine), the National Technical University of Ukraine "Kyiv Polytechnic Institute" - KPI (Kiev, Ukraine). A Wiki Web-portal of this infrastructure has been installed (https://gridedu.jinr.ru).

Information and software support of the research underway at JINR

The traditional provision of information, algorithmic and software support of the JINR research-and-production activity involves the development and support of the informational servers

WWW/FTP/DBMS of JINR and LIT, creation and storage of electronic documents related to the JINR scientific and administrative activity, development, creation and support of information web-sites of workshops and conferences, administration and support of web-sites of JINR subdivisions and various conferences in a hosting mode as well as support, modernization and maintenance of computer systems of administrative databases.

The portal technology is a key performance technology for the modern projects due to the large-scale and world-wide nature of a large part of the scientific projects, especially in experimental nuclear and particle physics. This technology is actively used at LIT in the process of development and creation of various information systems with web-interfaces. For instance, within the portals of the journals PEPAN and PEPAN Letters *http://pepan.jinr.ru* , a specialized interface has been designed for the authors, editors, referees and administrators providing interconnections with the databases of the journals. Work on the maintenance and modernization of the portal is in progress.

Another long term direction of LIT activity is the development and support of the library JINRLIB, support of program libraries developed at other scientific centres and organizations, together with information and technical help to users. The modernization of CICC software and its installation in a 64-bit variant required a full recompilation of programs from JINRLIB library. The full information on the JINR program libraries is available at the specialized WWW-server *http://www.jinr.ru/programs/* and in the LIT Information Bulletins.

MATHEMATICAL SUPPORT OF JINR STUDIES

The development of adequate mathematical models for process simulations and methods for data analysis is an integral part of the research conducted both in experimental and theoretical physics and in other fields of science and technology. This activity covers a wide spectrum of investigations defined in the JINR Topical Plans for Research and International Cooperation, in high-energy physics, nuclear physics, solid state physics, condensed matter physics, biophysics, and information technologies. LIT carries out the specific tasks in close cooperation with all JINR Laboratories along the following main directions:

- development of new approaches and methods for simulation of physics processes and for experimental data analysis;
- creation of methods and numerical algorithms for simulation of magnetic systems and charged particle beam transport;
- creation of software and computer complexes for experimental data processing and their application in JINR experiments;
- development of numerical schemes and software for simulation of complex physical systems;
- development of methods, algorithms and programs of computer algebra.

Very few results have been excerpted for inclusion in the present report. Comprehensive overviews can be found in the LIT Scientific Reports [14].

A data plotting package Gluplot, included in the JINR Program Library, can be used both as a graphical library and a standalone program which would allow scientists and students to visualize data. Gluplot handles both curves (2D) and surfaces (3D) [15]. It served, for instance, to the visualization of the simulations of freeze out surfaces and of the particle distributions inside the NICA/MPD detector.

A new tool for the approximation of functions by piecewise continuous polynomials and the analysis of the experimental data by smoothing, the basic element method (BEM), has been developed [16]. The proposed approach results in the reduction of the computational complexity of the algorithms and the increase of their stability to errors by appropriate tuning of the internal relationship structure between variables and control parameters, via an adaptive algorithm for knot detection. On the basis of the algorithm, MS Visual C# components and Windows application APCA (Autotracking Piecewise Cubic Approximation) were implemented.

The software toolkit Geant4 for simulation of particle propagation through matter is used by a large number of experiments and projects in a variety of application domains, including high energy physics, astrophysics and space science, medical physics and radiation protection. Two string models for the simulation of high energy final states were implemented into Geant4: the Quark Gluon String (QGS) and substantially improved the Fritiof (FTF) model. The implementation is accessible in the last version 4.9.3 of the Geant4 package [17].

A LIT team actively participates in the elaboration of the CBM set-up in GSI (Darmstadt). Efficient methods of event reconstruction have been proposed which accommodate the considerable problem complexity coming from the expected enormous multiplicity of the generated particles and the heterogeneous magnetic fields. For instance, approaches to track reconstruction in the STS-detector and algorithms for track recognition in the TRD-detector and for finding Cherenkov rings in the RICH-detector have been developed. The particle momentum determination methods have been elaborated; various methods for particle identification using the TRD-detector have been conceived and implemented. Work is in progress on the optimization of the geometry of the installation and development of methods of extracting "useful" events. In order to perform an efficient analysis, fast tracking algorithms are essential. A parallel tracking algorithm for the muon detector was developed. Modern technologies and optimized for parallelization event reconstruction algorithms were used. As a result, a high efficiency of track and ring reconstruction was achieved. The algorithms developed and implemented are 487 times faster for tracking and 143 times faster for ring reconstruction. A number of very efficient software modules for event reconstruction have been proposed by LIT specialists and included in the CBM framework [18]. As part of the work on designing a magnetic system for the CBM experiment, calculations have been performed for various versions of the superconducting dipole magnet [19].

The increasing interest of the experimental researches on the impact of the heavy charged particles on materials has been followed by extensive numerical investigations done in LIT on the clarification of the features of the radiative sputtering, the formation of tracks and the change of the mechanical properties of the materials under high energy heavy ion irradiation. The "track" occurrence in dielectrics was undoubtedly found to originate in mechanisms predicted within a generalized thermal spike model, which takes into account both the electron and ion lattice subsystem motions inside the material [20].

Investigations on nanostructure simulation by methods of discrete dynamics are motivated by the fact that many nanostructures, in particular certain carbon and hydrocarbon molecules, are highly symmetric discrete formations. The discrete dynamical systems of different types, deterministic and non-deterministic ones, defined on such structures have been investigated and it was found that important properties and peculiarities of the behavior of these systems are direct results of their symmetries.

The main practical tool for numerical analysis of the physical features of these compounds, a C program based on the computer algebra and computational group theory methods [21] unveiled the role of the prehistory in the present system state and allowed adequate mathematical characterization of the phase transitions in nanostructural systems.

The program packages KANTBP, POTHMF, ODPEVP [22] for computing energy values, reaction matrix and corresponding wave functions have been developed based on the Kantorovich method. They allowed the numerical analysis of various physical processes: the photo-ionization and recombination of a hydrogen atom in a magnetic field, the channeling problem for charged particles produced in a confining environment [23]; the adiabatic approach to the problem of a quantum well with a hydrogen-like impurity [24].

During the last years, LIT researchers implemented, in cooperation with colleagues from the Institute of Protein RAS, the Institute of Biophysics of Cell RAS and the Institute of Theoretical and Experimental Biophysics RAS, the project «Molecular cartography of DNA, RNA and proteins in the distributed computing environment». New algorithms have been developed and a software complex has been designed to construct molecular surfaces of spiral molecules of double-chained DNA (B-form), RNA (A-form) as well as spiral molecules of extended proteins or their fragments (for example, the widespread alpha-spiral form). Such a unified method of mapping the extended structures of nucleic acids and proteins, which supplements the known methods of 3D computer simulation of molecular structures, was realized for the first time. The designed software complex allows to perform computations, construction and analysis of the maps of relief and functional color of the molecular surfaces on the basis of corresponding spatial structures of high atomic resolution. Besides, it provides a way for massive computations of the maps of molecular surfaces within the high-performance distributed computing environment of CICC JINR. This approach reduces more than 10-20 times the time required for research on the structures of proteins and nucleic acids as compared to the manual routine work with the use of ordinary graphic interfaces. The derived method was applied to the study of the recognition, analysis and classification of binding regions of functionally important protein-DNA complexes. It allows introducing the novel data bank of the molecular surfaces and electrostatic potentials for all transcription factors of homeodomains for wide classes of species. In this way, very fast analysis of recognition areas and types of their classification became possible for the large set of protein-DNA complexes present in the international protein data bank PDB (Protein Data Bank) and the nucleic acid data bank NDB (Nucleic Acid Database) [25].

PROSPECT OF THE FUTURE

The formation of a unified grid-environment of the JINR Member States is a basic task for the Laboratory of Information Technologies within the JINR seven-year plan. This concerns all the three main levels within the grid-environment: network (high-speed backbones and telecommunication links), resource (high-performance computing clusters, and data storage systems joined in a unified grid-environment with the help of basic software and middleware), and applied (research topics solved within the grid-environment in frames of corresponding virtual organizations).

The ongoing and upcoming experiments at the JINR basic facilities (Nuclotron, NICA/MPD, etc.), the experiments at LHC, FAIR, the theoretical research need appropriate computing infrastructure. To fulfill these needs there are necessary: a substantial increase of CICC grid-infrastructure at JINR; participation in building up clusters as home workgroup computing facilities, based on the PROOF - Parallel Root Facility for parallel distributed analysis; creation of a specialized MPI cluster for parallel jobs.

A necessary condition towards the creation of a unified information environment of JINR and its Member States is the provision of information and primary software support of the research-and-production activity of the Institute. The specific activities planned on the mentioned problem include the following directions:

- centralized support of the operating systems and compilers used at JINR;
- elaboration of a unified technical policy in the field of licensing software products, the support and update of the bank of licensed and freely distributed products;
- development and maintenance of information WWW/FTP-servers;
- development, creation and support of information systems with web-interface: thematic portals, paperless document circulation, workshop and conference sites, etc.;
- creation, development and maintenance of general-purpose and specialized program libraries, first of all for topics in nuclear physics, particle physics, solid state physics, condensed matter physics, as well as computer algebra and graphic packages;
- support and development of object-oriented specialized software for modeling experimental installations and processes, as well as for experimental data processing of problems arising in nuclear physics, particle physics, solid state physics and condensed matter physics;
- support of administrative databases, creation of a unified information space for the research-and-production activities of the Institute.

In the existing grid-systems, a virtual organization (VO) defines a collaboration of specialists in some area, who combine their efforts to achieve a common aim. The virtual organization is a flexible structure that can be formed dynamically and may have a limited life-time. Instances of VOs working within the WLCG project are the VOs in LHC experiments - ATLAS, CMS, Alice, LHCb, the first three being carried out with the noticeable and direct participation of the JINR. Nowadays, as a grid-segment of the EGEE/RDIG, the JINR CICC supports computations of the VOs registered in RDIG (BioMed, PHOTON, eEarth, Fusion). Other supported VOs are HONE and Panda.

The LIT support activity of the existing VOs includes:

- cooperation with German scientific centers in grid infrastructure support and development and computing support for CBM, PANDA, and other experiments;
- support and development of the JINR WLCG-segment in frames of the global WLCG infrastructure in context of the computing requirements for running phase of the LHC experiments; participation in the relevant computing activities for ALICE, ATLAS and CMS experiments at the running phase of LHC and continuation of the software supporting for LHC (ATLAS, ALICE and CMS) and non-LHC CERN experiments;
- grid-monitoring the WLCG-infrastructure at JINR and other sites of the Russian grid-clusters;
- user support to stimulate their active usage of WLCG resources and further development of the GridLab and the corresponding educational programs (GridEdu).

The creation of new VOs gets possible and necessary under maturation of the algorithmic approaches to the problem solution, the development of corresponding mathematical methods and tools. We may assume that future VOs will be created in JINR to support the experiments within the NICA/MPD project, the nuclear physics and condensed matter physics research, the study of the nanostructure properties. The Laboratory of Information Technologies will provide qualify support to the emerging new VOs within the future activity frames defined by the interested user communities.

REFERENCES

[1] Adam Gh., Adam S., Ayriyan A., Dushanov E., Hayryan E., Korenkov V., Lutsenko A., Mitsyn V., Sapozhnikova T. , Sapozhnikov A., Streltsova O., Buzatu F., Dulea M., Vasile I., Sima A., Visan C., Busa J., Pokorny I.//Rom. Journ. Phys., Vol.53, No.5-6, (2008) pp.665-677; Adam Gh., Adam S., Ayriyan A., Korenkov V., Mitsyn V., Dulea M., Vasile I.// Rom. Journ. Phys., Vol. 53, No. 9-10, (2008) pp. 985-991.

[2] Ayriyan A., Adam Gh., Adam S., Korenkov V., Lutsenko A., Mitsyn V.// Proceedings of the XII Advanced Computing and Analysis Techniques in Physics Research, PoS(ACAT08)054, 5.

[3] http://rocmon.jinr.ru:8080/

[4] Andreeva J., Belov S., Sidorova I., Tikhonenko E. et al.// J.Phys.Conf.Ser.119:062008, 2008.

[5] Gaidioz B., Rocha R., Mitsyn S., Devesas M. // Campos: http://dashb-cms-job-devel.cern.ch/dashboard/doc/guides/service-monitor-gearth/html/user/index.html

[6] https://twiki.cern.ch/twiki/bin/view/Main/HepML

[7] http://mcdb.cern.ch

[8] Belov S. et al.// Comput. Phys. Commun., Vol. 178, No. 3, 2008, p. 222.

[9] http://hepweb.jinr.ru

[10] Abolins M. et al.// Computing in High Energy and Nuclear Physics 2007, 13:978-0230-63017-8, TRIUMF, Victoria, Canada; Corso-Radu A. et al.// CHEP 2009, 17th International Conference on Computing in High Energy and Nuclear Physics, Prague, Czech Republic; Ciovanna Lehman Miotto et al.// TIPP09 Tsukuba, Japan, KEK, Tsukuba, Japan, 2009.

[11] http://www.sci-innov.ru/icatalog_new/entry_68452.htm;
http://www.sci-innov.ru/icatalog_new/entry_79275.htm

[12] http://www.egee-rdig.ru/rdig /user.php

[13] Korenkov V.V., Kutovskiy N.A.// Open Systems. 2009. No.10. P.48-51.

[14] LIT Scientific Report 2008-2009. JINR, Dubna, 2009-196, ISBN 978-5-9530-0237-0;
http://lit.jinr.ru/Reports/SC_report_06-07/LITSR2008-2009.htm

[15] Soloviev A.G. http://www.jinr.ru/programs/jinrlib/gluplot/indexe.html

[16] Dikoussar N.D., Török Cs.// Mathematical Modelling. 2006. V. 18, No. 3. P. 23-40; Dikoussar N., Török Cs.// Kybernetika, V. 43, No. 4, pp. 533 - 546, 2007; Dikusar N.// JINR Preprint, P11-2009-123, Dubna, 2009. Submitted to the "Mathematical modeling".

[17] Geant4 release 9.3 (2008), http://geant4.cern.ch/support/ReleaseNotes4.9.3.html ; Uzhinsky V., Apostolakis J., Folger G., Ivanchenko V.N., Kossov M.V., Wright D.H.// Eur. Phys. J., C61 (2009) 237.

[18] Airiyan A., Baginyan S., Ososkov G., Hoehne C. // Tver Univ. Herald, No. 17(45), 2007, pp. 15-26; Akishina E.P., Akishina T.P., Ivanov V.V., Maevskaya A.I., Denisova O.Yu.// PEPAN Letters, V.5, No.2(144), 2008, pp. 202-218; Akishina T.P., Denisova O.Yu., Ivanov V.V., Lebedev S.A.// PEPAN Letters, V.6, No.2(151), 2009, pp. 245-259; Lebedev S.A., Ososkov G.A.// PEPAN Letters, V.6, No.2(151), 2009, pp. 260-284.

[19] Matyushevsky E.A., Akishin P.G., Alfeev V.S., Alfeev A.V., Ivanov V.V., Litvinenko E.I., Malakhov A.I.// CBM-note-2008-00, 2008.

[20] Amirkhanov I.V., Didyk A.Yu., Zemlyanaya E.V., Puzynin I.V., Puzynina T.P., Sarker N.R., Sarhadov I., Semina V.K., Sharipov Z.A., Hofman A.// PEPAN Letters, 2006, V.3, No.1(130), pp.63-75; Amirkhanov I.V., Didyk A.Yu., Sarker N.R., Sarhadov I., Semina V.K., Sharipov Z.A., Hofman A.// PEPAN Letters, 2006, V.3, No.5(134), pp.80-91; Amirkhanov I.V., Didyk A.Yu., Puzynin I.V., Semina V.K., Sharipov Z.A., Hofman A., Cheblukov Yu.N.// PEPAN, 2006, V.37, No.6, pp.1592-1644; Амирханов И.В., Дидык А.Ю., Музафаров Д.З., Пузынин И.В., Пузынина Т.П., Саркар Н.Р., Сархадов И., Шарипов З.А. // Поверхность, 2008, №5, с.1-10; Амирханов И.В., Пузынин И.В., Пузынина Т.П., Шарипов З.А.// Вестник ТвГУ. Серия: Прикладная математика, 2009, (12), с. 17-27.

[21] Kornyak V.V.// Lect. Notes Comp. Sci., 4194, Springer 2006, pp. 240-250; Kornyak V.V.// Programming and Computer Software, 33, No. 2, 2007, pp. 87–93; Kornyak V.V.// Lect. Notes Comp. Sci., 4770, Springer 2007, pp. 236-251; Kornyak V.V.// Programming and Computer Software, 34, No. 2, 2008, pp. 84–94; Kornyak V.V.: Lect. Notes Comp. Sci., 5743, Springer 2009, pp. 180-194.

[22] Chuluunbaatar O. et al// Comput.Phys.Commun., 177 (2007) 649-675; Chuluunbaatar O. et al.// Comput.Phys.Commun., 179 (2008) 685-693; Chuluunbaatar O. et al.// Comput. Phys.Commun., 178 (2008) 301-330; O. Chuluunbaatar, et al.: Comput.Phys. Commun., 180 (2009) 1358-1375.

[23] Chuluunbaatar O. et al.// Physics of Atomic Nuclei 72, (2009) 768-778.

[24] Gusev A.A. et al// Physics of Atomic Nuclei 73, (2010) 331-338.

[25] Chirgadze Yu.N., Ivanov V.V., Polozov R.V., Sivozhelezov V.S., Stepanenko V.A., Zrelov P.V.// Proceedings of the 3rd Intern. Conf. Distributed Computing and Grid-Technologies in Science and Education, JINR, Dubna, D11—2008-176, 2008, pp. 233-237; Akishina T.P., Zrelov P.V., Ivanov V.V., Polozov R.V., Sivozhelezov V.S.// Proceedings of the 3rd Intern. Conf. Distributed Computing and Grid-Technologies in Science and Education, JINR, Dubna, D11—2008-176, 2008, pp. 221-224; Иванов В.В., Зрелов П.В., Полозов Р.В., Катаев А.А., Сивожелезов В.С.// В сб. Ядерная физика и нанотехнологии. Дубна: ОИЯИ, 2008, стр. 293-311; Chirgadze Y.N., Zheltukhin E.I., Polozov R.V., Sivozhelezov V.S., Ivanov V.V.// J. Biomol. Struct. Dyn. 2009 26(6) pp.687-700; Chirgadze Y.N., Larionova E.A., Ivanov V.V.// J. Biomol. Struct. Dyn. 2009 27(1), pp.83-96.

INFRASTRUCTURE OPTIMIZATION FOR DEPLOYING A MPI CLUSTER DEDICATED TO NANO- AND BIOMOLECULAR PROCESSING[1]

Gh. Adam[1,3], S. D. Belov[1], F. Farcas[2], C. Floare[2], V. V. Korenkov[1],
V. V. Mitsyn[1], G. Popeneciu[2], R. Trusca[2]

[1] *Laboratory of Information Technologies,*
Joint Institute for Nuclear Research, 141980, Dubna, Russia
[2] *National Institute for Research and Development of Isotopic and Molecular Technology,*
400239, Cluj-Napoca, Romania
[3] *Horia Hulubei National Institute for Physics and Nuclear Engineering (IFIN-HH)*
407 Atomistilor, Magurele - Bucharest, 077125, Romania

The paper describes the progress got at INCDTIM Cluj-Napoca, in collaboration with LIT-JINR, in the implementation of a Grid infrastructure center, as well as for finding solutions to nano- and biomolecular problems using the MPI technology.

I. INCDTIM DATA CENTER

The Grid activities at the National Institute for Research and Development of Isotopic and Molecular Technologies (INCDTIM) at Cluj-Napoca started in 1996, when the collaboration with the ATLAS project in CERN was agreed. Specific responsibilities to provide computing power and storage capacity to WLCG have been assumed through the “Memorandum of Understanding” [1], which defined performance targets for the local Grid site and the network connections. The capacity of the data center is ten racks, it has raised floor and lowered roof, a generator as backup system, 20 kVA cooling system and 180 kVA power capacity for network and grid activity. The present physical hardware consists of three Blade systems and thirty 1U servers. The storage capacity is around 80 TB. The network connections are secured by a Cisco 6509E Router. The web hosting, e-mail and the database capacity of the Institute are processed through other five dedicated servers.

The networking backs all Institute activities. The site RO-14-ITIM secures Grid Computing resources, with about 80% of time devoted to the processing of jobs from the ATLAS experiment at CERN. Another 20% share secures coverage of computational physics numerical simulations with direct application to biophysics and nanostructure research in which the Institute is involved.

The computing power is backed up by a monitoring system, responsible for the good functionality of the system and datacenter. The choice of the monitoring solution proved to be an important point in the development of the datacenter.

II. COMPUTING INTENSIVE RESEARCH AT INCDTIM

After the inception of the datacenter in 2007, and the start of the Grid site in 2009, new tasks arose putting serious pressure on these units of INCDTIM and asking for fast and efficient solutions.

[1] Partial support by the Romanian Authority for Scientific Research (contracts POS-CEE Ingrid/2008 and 05-6-1048-2003/2010) and the Hulubei-Meshcheryakov Programme (JINR Dubna Orders 57/08.02.2010, p. 35 and 58/08.02.2010 p. 35) are acknowledged.

Particularly stringent requests coming from the research teams concerned the numerical modeling and parallel processing of problems in medical engineering, molecular and biomolecular modeling and structural analysis of solids.

The implementation of Quantum ESPRESSO (QE) for calculating NMR Parameters was a first priority task. QE is an integrated suite of computer codes for electronic-structure calculations and material modeling at the nanoscale [2]. It is based on the density-functional theory, plane waves, and pseudo potentials. It is used to calculate chemical shielding effects in organic crystal structures based on GIPAW (Gauge Including Projector Augmented Wave) scheme implemented in these codes. A typical result of this type of numerical processing was the derivation of the solid state NMR chemical shifts of Lisinopril (N^2-[(1*S*)-1-carboxy-3-phenylpropyl]-L-lysyl-L-proline – $C_{21}H_{31}N_3O_5$, with two molecules, i.e. 120 atoms, per unit cell) using the experimental data obtained by X-ray powder diffraction [3]. A computer power involving eight core processors, 16 GB RAM and 16 GB Swap has got the solution in a four-day run. Mention is to be made that the parameter fit securing the energy minimum asked for 99% of the available 32 GB memory.

Molecular modeling [4] computational simulations, based on the atomic description of biological molecules, get understanding of biological processes within runs which need by far longer times than in the previous example and use a lot of storage, up to 700 GB, which prevents finding numerical solutions on desktop computers.

The third research topic asks for numerical simulation of specific properties (electronic energy levels and structure, vibrational modes) of nanoscopic scale systems of interest in nano- and biotechnology. Of special interest in INCDTIM is the monitoring of the presence of dynamic molecular structures on metallic surfaces, due to the intermolecular forces between the isolated molecules or in molecular crystals.

The fourth research area is “molecular electronics” (ME) consisting of molecule-based and molecule-controlled [5] electronic device research. A simple molecular electronic device consists of a molecule connected with two metallic nano-electrodes. Computation of the current-voltage characteristic of a realistic nano-device roughly needs from 6 to 10 GB RAM and two weeks CPU time. We are using the SIESTA code for processing this kind of jobs [6]. A few days are not a very long time, but this happens only provided the number of parameters is restricted to a minimum. If we would run simulations at maximum parameter numbers, the CPU time would considerably increase.

As a rule, the available codes used by numerical simulations of computational physics are run under Unix-like operating systems. The codes used are mostly developed in FORTRAN. A great many number of such codes are developed under GNU license (such as ABINIT) or academic license (SIESTA, GAMESS). Commercial ones of interest are GAUSSIAN or MOLPRO.

The MPI (Message Passing Interface) has proved to be a very good candidate for performance enhancement using parallel programming. The modern developments of the above mentioned codes are using the MPICH2 implementation of MPI in order to reach a good scaling of the computing time with the number of processors [7].

III. INCDTIM MPI CLUSTER

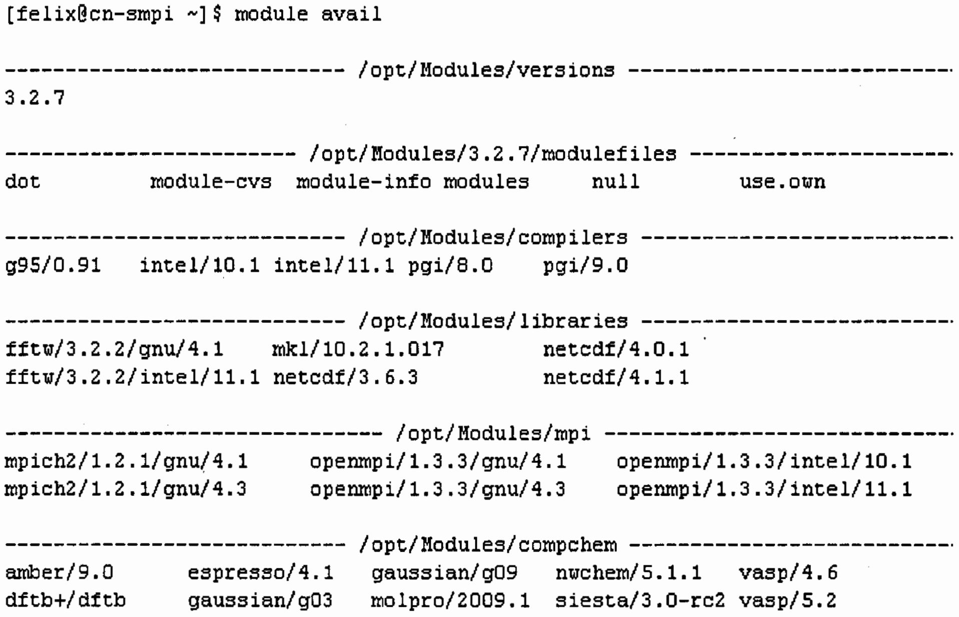
The testbed consists of one HP Blade-system. The Blade system was acquired through the European Project POS-CCE 192 nr 42/11.05.2009 [8]. The Blade system is a HP c7000 enclosure and has 16 Proliant BL280c G6 servers. The MPI system has a main server, 14 worker nodes and one configured backup server.

The operating system is Scientific Linux 5.3, chosen, among others, due to the existence of a special package dedicated to installing scientific packages afterwards.

The hardware configuration of the testbed is simple: It comprises two hard disks of 250 GB each, 16 GB RAM, and two Quad-Core Intel Xeon 2933 MHz processors. The system has an environmental certificate since its power consumption is 30 percent lower as compared to the 16 1U technology servers. The software configuration is not a Raid one. We are using both hard disks for processing jobs. There are a root partition, a location for installing the program and the home directory. Approximately 420 GB space serves for the scratch file system which is dedicated to job processing and temporary files. The overall scratch directory from all 15 servers is around 4 TB storage capacity.

The users were configured on the main server, named euler.itim-cj.ro. NFS use for the home directory enables a given user to log into each station, saving thus more space for programs. The password authentication is made via NIS so that the password and shadow files are copied on each station after the configuration is finished. The secure protocol "ssh" performs the connections among the stations.

The communication between the servers for launching a processing job is made by the TORQUE-maui combination which is installed on the system. The scientific programs as: amber, gaussian03, molpro, vasp are installed on the server in the /opt directory. We are in train to acquire Vasp 5.2 and Gaussian 0.9, modules of which are installed without access to software. For assisting each user that has access to the MPI cluster we made a local wikipedia site with information regarding the use of the cluster, which for the time being is accessible inside the INCDTIM Intranet.



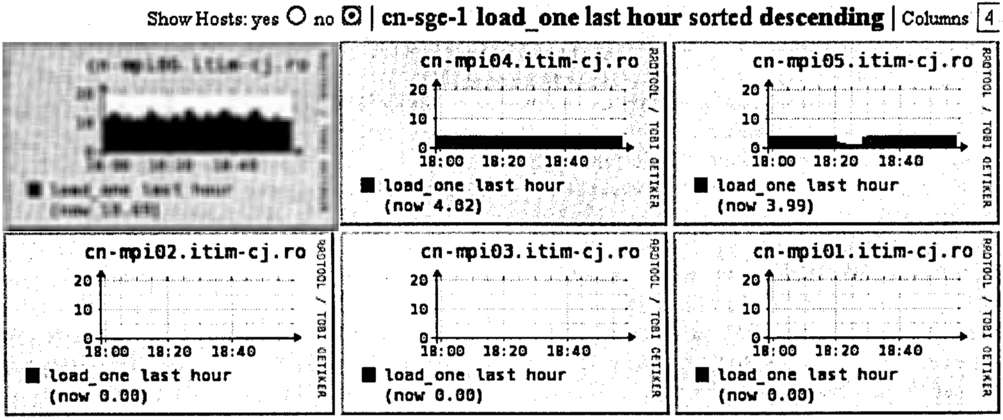
```
[felix@cn-smpi ~]$ module avail

---------------------------- /opt/Modules/versions --------------------------
3.2.7

------------------------ /opt/Modules/3.2.7/modulefiles ---------------------
dot         module-cvs  module-info modules     null        use.own

---------------------------- /opt/Modules/compilers -------------------------
g95/0.91   intel/10.1 intel/11.1 pgi/8.0    pgi/9.0

---------------------------- /opt/Modules/libraries -------------------------
fftw/3.2.2/gnu/4.1     mkl/10.2.1.017         netcdf/4.0.1
fftw/3.2.2/intel/11.1 netcdf/3.6.3           netcdf/4.1.1

------------------------------- /opt/Modules/mpi ----------------------------
mpich2/1.2.1/gnu/4.1     openmpi/1.3.3/gnu/4.1     openmpi/1.3.3/intel/10.1
mpich2/1.2.1/gnu/4.3     openmpi/1.3.3/gnu/4.3     openmpi/1.3.3/intel/11.1

---------------------------- /opt/Modules/compchem --------------------------
amber/9.0      espresso/4.1   gaussian/g09   nwchem/5.1.1   vasp/4.6
dftb+/dftb     gaussian/g03   molpro/2009.1  siesta/3.0-rc2 vasp/5.2
```

Fig. 1: Module Environment Package

For navigating through the system and choosing the packages for each research processing we installed the Module Environment Package. The program consists of a part that is able to combine the necessary modules for processing a job. The available modules are seen with the "module avail" command, figure 1.

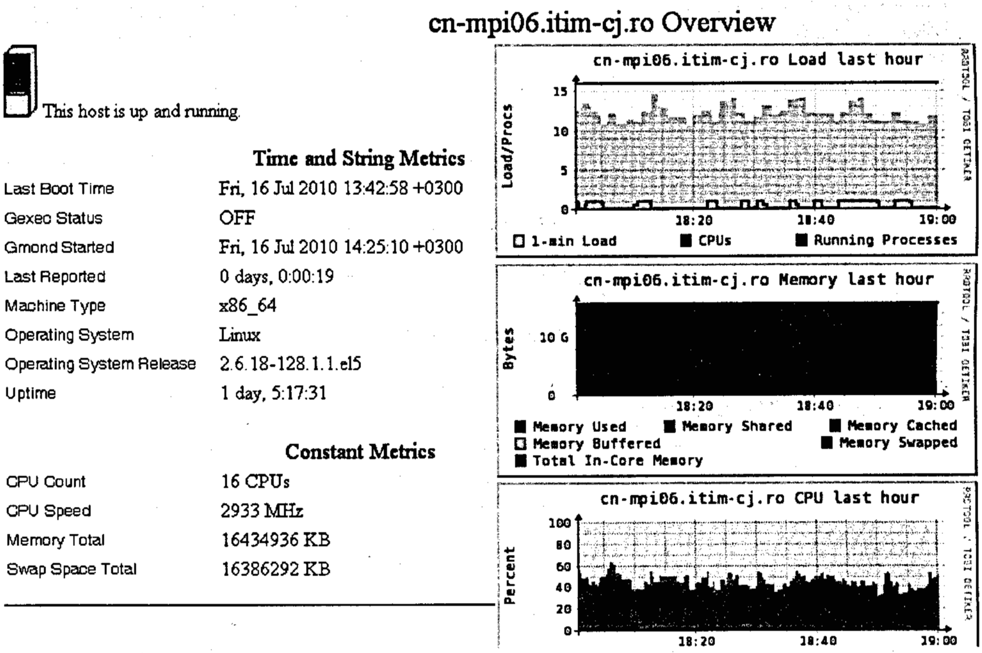
This kind of program is helpful for module loading and application usage because it is easy to install and support a large variety of shells such as: bash, ksh, zsh, sh, csh, tcsh.

The cluster monitoring is done via the Ganglia program. The cluster is functional, and four different jobs are currently running on the cluster. As shown in figure 1, some of the installed modules are backed up by programs too. In figure 2, the load of the whole system is illustrated, while in figure 3, the load on one work node is shown.



Show Hosts: yes ○ no ⦿ | cn-sge-1 load_one last hour sorted descending | Columns 4

Fig. 2: Load on the test system



cn-mpi06.itim-cj.ro Overview

This host is up and running.

Time and String Metrics

Last Boot Time	Fri, 16 Jul 2010 13:42:58 +0300
Gexec Status	OFF
Gmond Started	Fri, 16 Jul 2010 14:25:10 +0300
Last Reported	0 days, 0:00:19
Machine Type	x86_64
Operating System	Linux
Operating System Release	2.6.18-128.1.1.el5
Uptime	1 day, 5:17:31

Constant Metrics

CPU Count	16 CPUs
CPU Speed	2933 MHz
Memory Total	16434936 KB
Swap Space Total	16386292 KB

cn-mpi06.itim-cj.ro Load last hour
Load/Procs
15
10
5
0
18:20 18:40 19:00
□ 1-min Load ■ CPUs ■ Running Processes
RRDTOOL / TOBI OETIKER
cn-mpi06.itim-cj.ro Memory last hour
Bytes
10 G
0
18:20 18:40 19:00
■ Memory Used ■ Memory Shared ■ Memory Cached
□ Memory Buffered ■ Memory Swapped
■ Total In-Core Memory
RRDTOOL / TOBI OETIKER
cn-mpi06.itim-cj.ro CPU last hour
Percent
100
80
60
40
20
0
18:20 18:40 19:00
RRDTOOL / TOBI OETIKER

Fig. 3: System and Load details

IV. CONCLUSIONS AND FUTURE WORK

An operational MPI cluster, which is currently under use for solving parallel computing tasks inside the INCDTIM, was implemented.

For the time being, the implementation of the IPv6 addressing schema is still under study. This kind of addressing will be used for an extended High Performance Computing system in Romania.

The interconnect with the RO-14-ITIM Grid site is under consideration, several problems are defined and they are under active scrutiny.

REFERENCES

[1] Memorandum of Understanding for Collaboration in the Deployment and Exploitation of the Worldwide LHC Computing Grid, CERN-C-RRB-2005-01, November 2005.

[2] http://www.quantum-espresso.org

[3] Filip X., Tripon C., Borodi Gh., Oprean L., Filip C. Structural investigation of Lisinopril by Powder X Ray Diffraction and solid-state NMR, Journal of Physics: Conference Series 182 012007 (2009) [Procs. of the International Conference "Processes in Isotopes and Molecules", Cluj-Napoca, Romania, 24-26 septembrie 2009].

[4] Brown S. P., Muchmore S. W., Chem J. Inf. Model., 46, 999 (2006).

[5] Salomon A., Cahen D., Lindsay S., Tomfohr J., Engelkes V.B., Frisbie C.D. Comparison of Electronic Transport Measurements on Organic Molecules, Adv. Mater. 15 (22), 1881-1890 (2003).

[6] Farcas F., Popeneciu G., Bende A., Morari C., Belov S., Miclea L. ITIM Distributed Grid System applied in high energy, biomolecular and nanotehnology physics, In; Procs. of IEEE International Conference on Automation, quality and testing, robotics (AQTR) May 22-25 2008, theta 16th edition, vol III, pp 343-346, 2008.

[7] Gropp W., Lusk E., Skjellum A. Using MPI: portable parallel programming with the message-passing interface. Cambridge, MA, USA: MIT Press Scientific And Engineering Computation Series. ISBN 0-262-57104-8, 1994.

[8] http://www.itim-cj.ro/PNCDI/ingrid/index.html

APPLICATION OF WEAK COUPLED DISTRIBUTED COMPUTING SYSTEMS IN GEOPHYSICS[1]

I. M. Aleshin[1], V. N. Koryagin[1], K. I. Kholodkov[1], A. N. Shogin[2]

[1]Institute of physics of the Earth RAS, 123995, Moscow, Russia
[2]All-Russian Institute for Scientific and Technical Information RAS, 125190, Moscow, Russia

The use of distributed computing systems most effective in tasks that do not require strong interaction between computational units. As examples we can point out a number of geophysical applications implied the solution of inverse problems. In geophysics besides purely mathematical problems (nonlinearity of the inverse operator, the lack of stability and uniqueness of solutions), there are additional difficulties associated with lack of observational data and did not even complete the consistency of the model. The latter may be due to both insufficiency of necessary information about physical conditions in the Earth interior, and the computational complexity of calculations the direct problem. In addition, the general and fundamental difficulty that arises in solving inverse problems is the "curse of dimensionality" (the amount of computation increases very rapidly with increasing dimension of the space models), which significantly limits the range of problems that admit precise solution. Using of distributed systems dramatically improves performance of calculations that allows to increase the number of varied parameters and expand the number of problems which can have exact solution.

A possible way to organize grid-application that is used to solve the inverse problem is described in this work. As an example, we examined the problem of determination of the mantle seismic anisotropy parameters using seismic wave forms inversion. As it will be shown below, such class of problems leads to tabulation of complicated multidimensional function. Thus, a calculation within each point of defined range of parameter space performs independently that is completely suitable for calculating with weak coupled distributed computing infrastructure.

Seismic anisotropy of mantle structures can be used for research of geodynamic processes [1]. The mostly used method of it's measuring is based on analysis wave forms, SKS waves and it's related phases [2] (for review see [3]). It's based on conversion SV into SH on the boundary which separates isotropic and anisotropic environments (or two anisotropic environments).

Method of using P → S conversion on media inhomogeneity is also used to study the mantle anisotropy. The method based on converted waves application is called receiver function. After leaving the layer, straight and exchange waves propagate with different speed that allows us to estimate the depth of anisotropic layer [4, 5].

Using both wave forms SKS and receiver functions of converted waves to study anisotropic mantle properties was introduced by Vinnik et al [6]. It's main idea is based on the assumption that anisotropic effects observed in both types of data have one source therefore is can be explained by one anisotropic model.

The inverse problem is the procedure for measuring the parameters of the model which describes the system according to the observed data. In a simple, but practically interesting case, the model, located under seismic station can be introduced as set of regular layers on regular isotropic semispace. Let's define criterion function as a distance between observed data and the result of calculations within the model we have chosen. The goal of inversion is measurement of anisotropic parameters of the each layer.

As it was noted above, the precise measuring of the model parameters that based on observed data is not always possible. Moreover, often it does not even make sense. Therefore modeling can be

[1] This work was supported by the RFBR grant 10-07-00491-a

considered as realization of random process and probability for the model parameters, which adequately describe the observed system, to have some fixed values is defined by density of it's distribution, which is called a posteriori distribution function (APDF). With such approach, cost function is considered as proportional to logarithm value of APDF. It is obviously that in this case phase space of APDF coincides with space of model parameters. At the same time, one of the particular set of the values of the model parameters defines a point within phase space where "probability" of realization of given parameter set increases as soon as the difference between calculated and observed data decreases.

Therefore, the main point of the problem is calculation of a posteriori distribution function, that gives us practically complete information about interpretation of our data within the model we have chosen. In particular, we can not only define the model which minimizes the cost function, we can also investigate shape of the cost function near extremum and the value of each parameters' distribution. Moreover, we can calculate mean value of values of interest, also the other marginal estimations can be performed as well.

The execution of introduced task virtually requires tabulation of cost functions for physically interesting models. Because the calculation procedure as well as process of running the task on grid-note has some technical complexity, it is more convenient to do it through Web-based interface. It has to allow:

1. User authorization as a member of virtual organization,
2. Uploading data,
3. Parameterization setting - number of parameters, discretization degree and etc.,
4. Evaluation of calculation time for the selected number of parameters (minimum, maximum, considering load of grid-nodes),
5. Running the task on one or a few grid-nodes,
6. Downloading the task's output for further analysis.

Such configuration needs compiling of the task code and creating the number of service programs and (or) scripts in order to perform these functions.

In order to execute calculations on distributed computing system the task needs to be separated on blocks which can be processed with any sequence without any interaction between each other. In our case each such block includes calculation of criterion function values for defined number of points from space of the models. The number of points in each block is mostly defined to minimize number of resources needed to run one block.

For the problem of measuring seismic anisotropy parameters within parameterization we have selected, in practically interesting cases the model has to contain at least two anisotropic layers. It means that the function we have tabulated includes at least six variable parameters. The provided below estimations are oriented on this kind of calculations. Depending on the cluster load it takes about from 1 to 10 minutes to run one block. The period of time which takes to calculate one block of entered parameters (one point in space of models) is less than one second, the blocks contain 100000 instructions. Therefore, in this case, calculations on each model were separated on 67 blocks with 100000 instructions on each block. Such separation allowed to provide equal load for each unit of used cluster; estimated time needed to run each block were significantly less than calculation time of the task and number of blocks allowed to use all nodes of calculation cluster independently on load level and number of available nodes.

In described earlier calculations both synthetic examples real data has been used [7]. In particular it is shown that performing of joint inversion, from one hand, significantly improves the precision of measuring of the main anisotropic parameters, but from the other hand, it needs careful analysis of consistency of different types of data.

References

[1] Nicolas A., Christensen N.I. Formation of anisotropy in upper mantle peridotites a - review. In: Fuchs K., Froideveaux C. (Eds.). Composition, Structure and Dynamics of the

Lithosphere Asthenosphere System // Amer. Geophys. Union, Geodyn., 1987. Ser.16. P.111–123.

[2] Vinnik L.P., Kosarev G.L., Makeyeva L.I. Anizotropia v litosfere po nablyudeniam SKS и SKKS (in Russian)// Doklady AN SSSR, 1984. V.278. N.6. C.1335–1339.

[3] Savage M.K. Seismic anisotropy and mantle deformation: what have we learned from shear wave splitting? // Rew. Geophys., 1999. V.37. N.1. P.65–106.

[4] Kosarev G.L., Makeyeva L.I., Vinnik L.P. Anisotropy of the mantle inferred from observations of P to S converted waves // Geophys. J. R. Astr. Soc., 1984. V.76. P.209–220.

[5] Girardin N., Farra V. Azimuthal anisotropy in the upper mantle from observations of P-to-S converted phases: application to the southern Australia // Geophys. J. Int., 1998. V.133. P.615–629.

[6] Vinnik L., Peregoudov D., Makeyeva L., Oreshin S. Towards 3D fabric in the continental lithosphere and asthenosphere: the Tien Shan // Geophys. Res. Lett., 2002. V.29. P.1795. doi: 10.1029/ 2001GL014588.

[7] Aleshin I.M., Mishin D.Yu, Zhizhin M.N., Koryagin V.N., Medvedev D.P., Novikov A.P., Peregoudov D.V. Primenenie raspredelennyh vychislitelnyh system pri opredelenii parametrov seismicheskoi anizotropii kory i verhney mantii (in Russian)// Geofizicheskie issedovaniya, 2009. V.10. N.4. C.36-49.

DATA COLLECTION FOR ESTIMATION OF APPLICATIONS PERFORMANCE IN COMPUTING SYSTEMS WITH HETEROGENEOUS MEMORY

A. V. Basov[1], Joo Young Hwang[2], M. P. Levin[1]

[1] Advanced Software Group, Samsung Research Center, Moscow, Russia
[2] Advanced Software Research Team, Memory Division, Samsung Electronics,
Suwon, South Korea

1. Introduction

Nowadays, the development of the perspective embedded systems with the heterogeneous memory is in the focus of interest of various companies. A computing speed in such systems can be faster than those in traditional systems with non-volatile memory. In systems with heterogeneous memory, the memory of different type has different characteristics those include a different speed of data read/write operations. In this paper the method of data collection required for performance estimation of applications running in the systems with heterogeneous memory is described. Basing on this method the problem of application execution time measurement on the emulator of future processors is considered and analyzed.

The application operation speed can be increased significantly in the computing systems with heterogeneous memory of various types with different speed of read/write operations. In this paper we consider computing systems with heterogeneous memory of different types. For instance these systems may be based on the SGI NUMA [1] architecture or some others. Such computing systems can consist of a few homogeneous blocks. Each block can contain one or several processors and a local memory unit. These memory units aggregate against each other with special high-speed connection switches (NumaLink). These computing systems have a single address space type. The direct access to the remote memory in the considering systems is supported by the hardware. The access time of one block to the memory of another block is depended on the blocks connection layout. It should be noted, that heterogeneous memory systems can be implemented not only basing on the SGI NUMA architecture, but also by using some other architectures, particularly by using other processor types, for example such as ARM [2] and MIPS [3]. Also it should be noted, that systems with heterogeneous memory can also be implemented by using of flash memory of various types, for instance, modern PRAM memory together with traditional DRAM memory. The main difference between PRAM memory and DRAM memory consists in a non-volatile type of PRAM memory. This means that data stored in such memory are not lost after the power-off. In development of the considering computing systems with heterogeneous memory the problem of performance estimation of applications running in such systems is arisen. The solution of this problem requires solution of data distribution problem among different parts of heterogeneous memory units during application running.

In our article we describe the method of data collection that is required to provide performance analysis of applications running in the systems with heterogeneous memory. This performance analysis is provided by using a static method. The static method implies an interpretation of the information on the memory addresses those were accessed by the considering application for reading or writing, and how many times these accesses were made during the application run. After that, by using read/write characteristics of different memory types, those are used in the considering system, and by using the information on the program objects (variables, arrays of variables, structures, etc.) stored in the memory of specialized type, we can provide the estimation of the application performance on the real system. Also we can provide the estimation of the efficiency of data storing in memory for the considering applications.

2. Review of Existing Approaches

Recently there are two different methods for collection of information on memory addresses those are accessed by the program.

The first method consists in integration of the trace tool into the page-fault handler of Linux kernel. Linux kernel should monitor all data access from the instrumented application to the memory. When the instrumented application is started, all its pages are not mapped. When the application tries to access to some data in memory, the kernel should map required page. After the page is mapped, the data should be written to, or read from mapped page. The data access should be logged in the memory access trace. After the data access is performed, the page is unmapped. All above mentioned steps are repeated for the each memory access.

The main disadvantage of this approach consists in a very slow velocity of work. Also it does not allow one to provide the tracing of data written into or read from CPU caches.

The second method realizes mechanisms, those are used in well-known debugger as gdb [4], open source performance analyzer dyninst [5] and also in application memory profiler valgrind [6]. These mechanisms work in the address space of the application. They consist in the substitution of the addresses of the memory read/write instructions by the addresses of the instructions with the special wrappers. These wrappers collect all required information and pass the control flow back to the main application running in the computing system.

Let us note that methods, defined above, have a variety of significant disadvantages. Namely they are:

- By using of these methods we cannot collect evidential information on data those have been read or have been written to the physical memory, and on data those have been read or have been written to the processor cache memory;
- The considering methods do not allow provide the information collection at the early stages of the operating system kernel initialization;
- Also using of the considering methods does not allow obtain the information on the memory type that was accessed by the investigated application.

3. Using Emulator for Target Embedded Platform

To avoid previously described disadvantages, the following approach is purposed. Let us use a high-level tool, namely a hardware platform emulator to collect memory access trace. This emulator should support the emulation of considering computing system with PRAM memory architecture support. A well-known Open Source Qemu [8] hardware emulator can be used as a basement for development of such emulator.

Recently Qemu is a fast, portable emulator, which supports wide range of emulated architectures, such as x86, PPC, SPARC, SH. Let us note that the computer, on that the emulator is executed, is called a host-machine. The hardware, which is emulated by emulator, is called a guest-machine.

For Random Access Memory (RAM) the emulation of a virtual memory area is provided on the host-machine. The host-machine uses the physical Random Access Memory in the emulator. On reading or writing RAM memory operations the emulator converts the physical memory address of the guest operating system into the offset of the beginning of the memory area, allocated on the host-machine.

For the guest-environment binary code execution, Qemu uses dynamic instruction translator. This translator translates guest code into the set of so called micro-operations those are implemented as simple functions in C programming language, and then use such micro-operation for guest code execution on the host-machine.

For example, the following ARM architecture assembler instruction

```
ldr r3, [r2]
```

is translated by the emulator into the following sequence of the micro-operations:

```
void op_mov_r2_T1;
void op_ldl () { T0 = __ldl_mmu (T1); }
void op_mov_T0_r3;
```

Here, ***T0***, ***T1*** are the temporary internal registers of the Qemu emulator, those are mapped into the host-machine registers.

To trace the access to the virtual memory address, the micro operations, those are implemented by corresponding ARM memory access instructions (op_strb, op_strub, op_strw, op_ldrb etc.), are extended with the special handlers. In this case for the each micro-operation call, a corresponding handler is called. Each handler saves information on the address that was accessed by the application, the type of access (read or write access type), and the size of the accessed data (byte, word and half-word). Let us note that this approach has some disadvantages as follows:

- The usage only of the read/write memory access handler in the emulator saves information on all memory accesses; those were made by the all applications in the guest environment. It is quite complex to separate data, which are belongs to the investigating application only;
- The memory access trace containing all instructions has a very big size. Usually it is measured in gigabytes, and its size depends on the instrumented application binary code. The analyzing and processing of these traces needs a big amount of time;
- Data collecting of each memory access instruction also adds some additional overhead. It implies the necessity of saving of huge amount of the input information.

To get over the first described above disadvantage, the following approach is proposed. Let us ignore all executed instructions and data access do not issued by instrumented application. The appropriate feature was developed and added to the emulator. The description of this procedure in details is given below. The regular expression, which is matched to the name of the investigated application, or the assumed process identification (PID) of the investigated application, must be passed to the emulator. For the name and the identifier of current executed application the special handler is used. This handler is defined after the parsing of the instruction that is pointed by the ***PC*** (program counter). register of ARM processor. The handler has unlimited access to the physical and virtual memory of the guest operating system. It reads **task_struct** structure of the current executed process from the memory. In other words it reads '**task_struct current**' structure for the current process. The address of this structure can be easy evaluated by using the current value of application stack. It is used for the description of the executed process in Linux, and contains all required information about it. The **pid** field of the **task_struct** structure contains the value of the process identifier of the currently executed application and **comm** field containing the name of the executed application.

The kernel of Linux OS is implemented with **task_struct** structure always locating at the fixed address for each executed process. Its address can be computed by using the value of the stack register for the kernel-space of the current application. For instance, the stack pointer for ARM processors is a banked register. It has different values for the various processor operation modes. Further, by using offset of the **pid** field and **comm** field of the **task_struct** structure, the name and process identifier of the executing application can be read from the guest operating system memory.

To speed-up the memory access trace collector, it is not needed to produce the above mentioned activity for reading of name and process identifier of the executed application each time on the instruction decoding action. It is enough to catch the moment when the guest operating system switches from one process to another. As we using Linux as the guest operating system the process switches can be only occurred in the kernel-space. Due to this we should not read the name and PID of the executed application on the decoding of instructions of user-space application, because Linux can not switch from one process to another in the user-space.

Linux kernel is always loaded into well-known virtual memory area (for the most of platforms, this area starts from 0xC0000000 address), and therefore we can easily monitor the Linux kernel instructions, until the name or the PID will not be changed. If the corresponding fields of the tast_struct have been changed, then the process switch is occurred, and we can switch off the collection of the trace, until operating system will not switches back to the instrumented application.

The illustration of the above mentioned algorithm is given in Fig. 1.

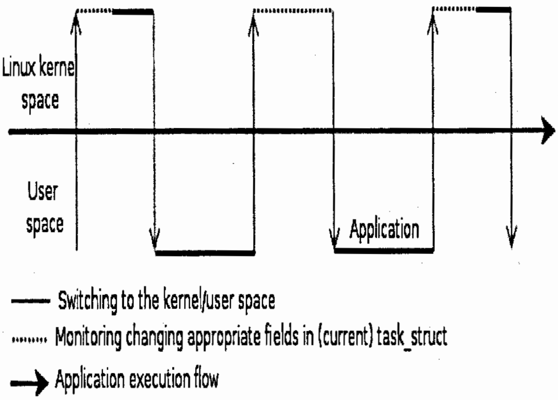

Linux kerne
space
User
space
Application
Switching to the kernel/user space
Monitoring changing appropriate fields in (current) task_struct
Application execution flow

Fig. 1: Enabling and disabling memory trace collecting, during process switches in Linux kernel

In addition, during the user-space application tracing, there is no need to save information about memory access to the virtual addresses belonging to the Linux kernel virtual memory, because according to the principals of work with virtual memory in Linux kernel, the user-space application can not get access to the kernels memory neither reading nor writing.

The described above method is enough for set up the full correspondence between the binary code of the executed application and the memory access addresses in the trace. Using such system we can trace absolutely all attempts of the virtual memory access of the executed application.

To reduce the size of the memory access trace, the resulting trace can be compressed by various ways. For example, it can be compressed on the fly, using well-known compress libraries and algorithms. To minimize the influence of the compression overhead in such way, compression can be done in separate threads on the host-machine.

On the real hardware the processor's cache memory affects on the interaction of the program with random access memory. Also it is known, that for emulation speed-up, the emulators do not have real emulation of the processor's cache memory. That is why, to have a full view, of how the application interacts with the system on the real hardware, the cache hits and cache misses should be taken into account. Therefore the cache memory must be implemented in the emulator. The cache memory has a von-Neumann architecture type. This means that it has separate instructions and data caches, which are of 16 KB size, and which support for write back mechanism.

4. Measurement of Application Execution Time

The approximately measurement of application execution time can be done by using the collected memory access trace. This trace contains not only memory addresses, which were accessed by the application, but also all executed instructions of the investigated application. Application execution time can be calculated by using this information in the following manner.

For each instruction containing in the memory access trace a number of cycles, those are needed for its execution, can be calculated. At the next step, memory access instructions (str, ldr,) are separated from all other instructions. The memory access instructions are executed by the different number of CPU cycles. The number of CPU cycles is defined by different memory access time for the considering instructions. Also, all instructions obtaining the data not from main memory, but from the CPU cache, are separated. These instructions do not wait data from main memory and get its data immediately from CPU cache.

Using all these parameters we can calculate approximately execution time of the instrumented application.

For the evaluation of the application execution time, the following formula can be used

$$T_{total} = \frac{C_{no\_mem} + C_{dram} + C_{pram}}{C_{LK}}, \qquad (1)$$

where T_{total} is a total execution time of the application, C_{no_mem} is a total number of CPU cycles, spent on instructions execution with no memory access, C_{dram} is a total number of CPU cycles, spent on instructions execution with dram memory access, C_{pram} is a total number of CPU cycles, spent on instructions execution with PRAM memory access, C_{LK} is a clock rate of used CPU.

More over, we can easily separate time which application spent in the kernel space, in the libraries and in its own code.

As mentioned above all executed instructions are stored in memory trace. Each instruction has its own address. It can be easily evaluated, those instructions are corresponded to the application code instructions, those are corresponded to kernel code instructions, and those are corresponded to library code instructions.

Addresses of application code instructions can be evaluated from the application executable file. It can be done by analyzing the application binary execution file.

The same technique can be used for kernel code analyzing. The addresses of the kernel instructions can be received by disassembling non-compressed Linux kernel image (vmlinux file).

Measuring time, that application spent in dynamic libraries is a more complicated task, because the dynamic libraries can be dynamically loaded and unloaded into any addresses. To solve this issue, libc was modified in the following manner. When dynamic library is loaded, then libc sends information about dynamic library (it name, load address and PID of application) to the memory trace collector. Also libc sends the same information when dynamic library is unloaded. The memory trace collector adds special mark to the trace, with specified information. The memory trace collector adds addresses to the memory trace consequently. The addresses of the dynamic libraries, those are in the range, specified by added marks can be easily identified.

The formula (1) can be used for time measurement for the mentioned below cases, namely, for measurement how much time application spent in its own code, in libraries code, and in Linux kernel code.

5. Conclusion

If the data accessed by the processor is in cache-memory, then a special mark is added to the memory access trace. This mark can be further used by memory access trace analyzer for more precise estimation of application performance.

Collected information can also be used for debugging such programs as the kernel of operating system, because the memory trace can be extended with data, those are read or written to the appropriate addresses. This allows one to provide debugging on the earliest stages of the operating system loading, when there is no ability to use traditional debugging tools.

References

[1] Nikolopoulos D.S., Papatheodorou T.S. The Architectural and Operating System Implications on the Performance of Synchronization on ccNUMA Multiprocessors. In: International Journal of Parallel Programming, 29, (2001), pp. 282-331.

[2] Seal A D. ARM Architecture Reference Manual (2nd Edition). Addison-Wesley Professional, 2001.

[3] Kane G., Heinrich J. MIPS RISC Architecture (2nd Edition). Prentice Hall PTR, 1991.

[4] Stallman R. M., Pesch R., Shebs S. Debugging with GDB: The GNU Source-Level Debugger. Free Software Foundation, Boston.

[5] Hollingsworth J. K., Snavely A., Ekanadham K., Sbaraglia S. EMPS: An Environment for Memory Performance Studies. In: Proceedings of the 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS'05) , 11, (2005), pp. 223 – 223.

[6] Nethercote N., Seward J. Valgrind: A Framework for Heavyweight Dynamic Binary Instrumentation. In: Proceedings of ACM SIGPLAN 2007 Conference on Programming Language Design and Implementation (PLDI 2007), 42, (2007), pp. 89 – 100.

[7] Levin M. P. Parallel programming with OpenMP. Binom, Moscow, 2008. (in Russian)

[8] Bellard F. QEMU, a fast and portable dynamic translator. In: Proceedings of the annual conference on USENIX Annual Technical Conference, (2005), pp. 41 – 41.

GRID-TECHNOLOGIES FOR BUSINESS TASKS

M. Belev[3], S. Belov[1], V. Korenkov[1,3], N. Kutovskiy[1,2], A. Nechaevskiy[1], A. Uzhinskiy[1]

[1] Laboratory of Information Technologies,
Joint Institute for Nuclear Research, 141980, Dubna, Russia
[2] National Scientific and Educational Centre of Particle and High Energy Physics
of the Belarusian State University, 220040, Minsk, Belarus
[3] Dubna International University for Nature, Society, and Man, 141980, Dubna, Russia

Basic concepts of grid technologies are described. A general approaches to building grid infrastructure and work on it are shown. Our experience in use of grid infrastructure for solving tasks in different areas, and example of the solution for CAE-jobs is presented.

During last 5-7 years researches in the grid area was rather popular. Many scientific projects have reached success using grid technologies. The question is if these technologies can be used to solve business tasks?

Grid computing combines different resources and allows users to get as many computation resources as they need. It means a great increase in calculation speed for users. Users shouldn't run their jobs consistently on their local PC but they can run as many jobs with different parameters on grid-infrastructure as they want (see Fig. 1). There is a possibility to run even parallel jobs what will also speed up their calculations.

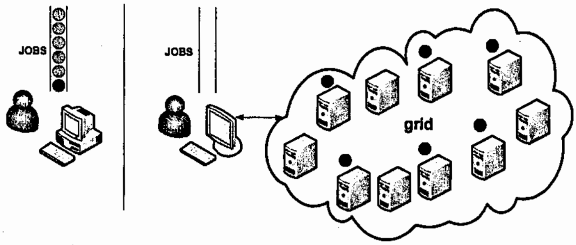


Fig. 1: Job processing schema (casual at the left and grid-oriented at the right)

Another benefit of the grid technologies is more effective utilization of the software and hardware resources.

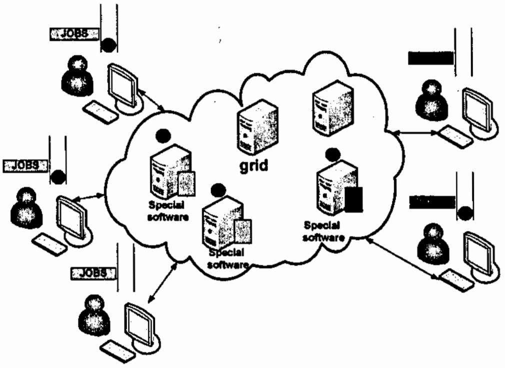


Fig. 2: Using grid for load balancing and licensing tasks

[1] Contacts: zalexandr@list.ru (Alexander Uzhinskiy)

Load balancing allows one to use standing idle resources of different department to solve current tasks. By organizing *remote access to specialized software* one can avoid installing the software on each and every PC (see Fig. 2). This schema could also help in licensing problem.

Two more features provided by grid what should be mentioned also are the wide internal access and security polices as well as secure and reliable data storage. Most of the middleware already have different security levels (for example organizations->groups->roles etc.), so one shouldn't worry about security issues. By using distributed storages and replication mechanism grid infrastructures can provide more reliable and effective data storage mechanism.

It is clear that grid has something to offer for business, but how does it work? How one can create its own grid infrastructure or use resources of the existing one?

There are some infrastructures which provide a possibility to use the computational resources of many scientific centers. EGEE [1] is one of the most popular. To use their resources the user needs to belong to the one of so called virtual organizations (VOs), should walk through some administrative procedure, get user certificate, sign it in certification authority etc. Apart from that the user's tasks must match the VO's one. There is also a chance that needed software isn't installed. So users will have to initialize procedure to install the software or even modify it to be able to use that application in grid. Probably for business it will be more interesting to create grid segments on the base of its local resources. Generally all you have to do to create grid infrastructure based on your corporative one is to add in each department server machines on which grid services will be installed and set up on each workstation special software which allows them to work as a grid working nodes (see Fig. 3).

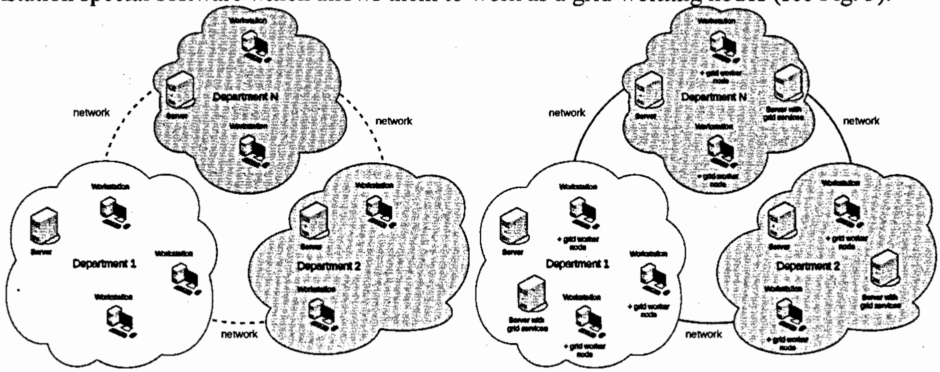


Fig. 3: Transformation of the corporative computer infrastructure (left) in to the corporative grid infrastructure (right)

But what is the best way to organize interaction with grid infrastructure? We believe that one of the most convenient ways to communicate with grid is to use web portals.

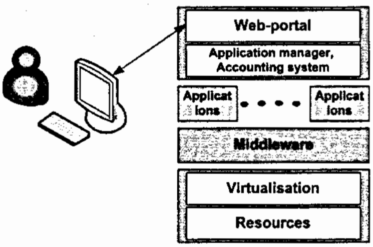


Fig. 4: User and grid-infrastructure interaction scheme

At the lowest level as it shown at Fig. 4 there are different hardware resources. Virtualization can be used to simplify interaction with them. The next level is the middleware that aggregates all the physical or virtual resources in to the one grid infrastructure. Different applications can be installed on working nodes or servers so users can get possibility to run special jobs on it. Application manager is kind of catalog where all information about installed software in grid infrastructure is stored and it is having some mechanisms to interact with them. Accounting system should keep accounting or billing information about consumed grid resources by users. Web portal is the easiest way for user to submit his job to the grid infrastructure. Users shouldn't know anything about grid, all they should know is how to work in the web using their browsers.

According to the BEinGrid [2] project some experiments have been already carried out and can be considered as successful in using grid infrastructure for solving tasks in areas like Advanced Manufacturing, Media, Finance, Retail & Logistics, Environment & eScience, Telecom, Tourism, Agriculture, Medicine, etc.

Our own experience covers areas like Computer Aided Engineering (CAE), Molecular dynamics simulations, Quantum chemistry, processing of 3D models and video data, Storing and post processing of the video-supervision systems. We have used methods of interacting with grid infrastructure like portal technologies described above. We also tried to develop some scripts and modules which work directly in Unix/shell environment but that approach is rather for more advanced users.

As an example of our work we would like to describe our experience in CAD/CAE area. Lots of companies have to calculate heat equation, geometry and meshing, electrostatic, fluid dynamics and other such kind of tasks in their work. There are huge program complexes like AutoCAD [3], Unigraphics NX [4], CATIA [5], ANSYS [6], Nastran [7], etc which allows to process mentioned tasks but there are some problems. For example the model processing is very time consuming operation, usually only single workstation is used for calculations, the failures during model processing can ruin the whole work, production distributed systems (clusters) cost too much (the price for license is proportional to number of cluster nodes), etc.

Generally the user has to pass to following stages to solve CAE job. He needs to create model geometry using «modeler». After that all parameters of the model, forces which will influenced on it have to be set as well as simulation type needs to be specified. To do so user uses «preprocessor». The next step is to perform the processing of the mathematical model what is done by «solver». At the end the results of the calculation made by «solver» can be represented using «postprocessor» (see Fig. 5).

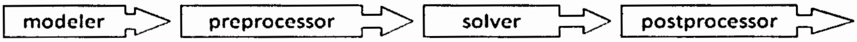


Fig. 5: CAE-job schema

The most time consuming operation is the processing of the model in «solver». The calculations can take hours, days, weeks or even longer. Our main idea was to spread solvers over working node of the grid infrastructure to reduce the calculation time (see Fig. 6).

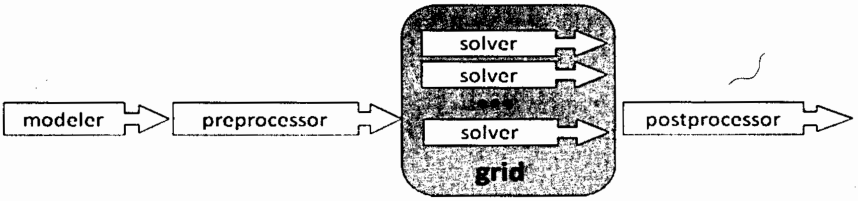


Fig. 6: Modified CAE-job schema

In our work we use freeware solvers like Elmer that allows us not to worry about licensing.

After modification the user still needs to perform all steps mentioned above in «modeler», «preprocessor» and «postprocessor». The main difference is that user shouldn't run his jobs consistently, one can change the parameters and run as many jobs as it is necessary. CAE jobs could be parallelized, therefore the use of several processors for solving job could be prominently decrease calculation time. The processing results are saved in grid what provides a reliable and secure way to store them (see Fig. 7).

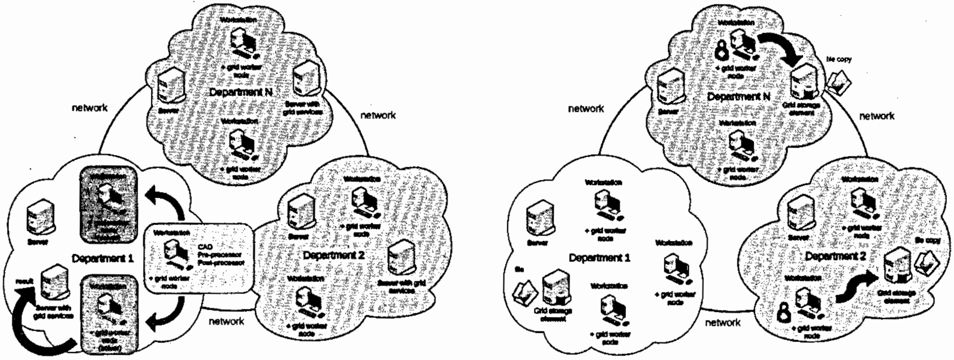


Fig. 7: Distributed «solvers» and data storage in grid-infrastructure

As a conclusion by adopting this model one can reach the following benefits:

- increase effectiveness of resource usage by using standing idle resources of departments for calculation purpose;
- minimize calculation time and increase number of the CAE jobs;
- solve licensing problem;
- increase reliability and effectiveness of data storage course of data replication mechanism;
- organize secure and effective collective operations with data.

There are a lot of interesting and promising experiments in grid2business area but it seems that none of them achieved production quality yet. There is still a lot of work to do including user-friendly interfaces (portals) to software and hardware resources. To speed up the development of the production-quality solutions business should invest in R&D projects.

We believe that those pioneers in business who is first starting to use new technologies can benefit from them and get a competitive advantage!

References

[1] EGEE official web portal, http://eu-egee.org/
[2] BEinGRID - Business Experiments in Grid, http://www.beingrid.eu/
[3] AutoCAD, http://usa.autodesk.com/
[4] Unigraphics NX, http://www.plm.automation.siemens.com/en_us/products/nx/
[5] CATIA, http://www.3ds.com/products/catia/
[6] ANSYS, http://ansys.com/
[7] Nastran, http://www.plm.automation.siemens.com/en_us/products/nx/simulation/nastran

MONITORING AND ACCOUNTING FOR GRIDNNN PROJECT[1]

S. Belov, V. Korenkov, I. Lensky, M. Matveev, E. Matveeva, S. Mitsyn, D. Oleynik, A. Petrosyan, R. Semenov

Laboratory of Information Technologies,
Joint Institute for Nuclear Research, 141980, Dubna, Russia
belov@jinr.ru

The contemporary distributed systems, such as computing grids, are complex technical systems. That is why, in order to keep an eye on their state and to count the consumption of computational resources, special automated tools should be used. In this paper we discuss the experience in the monitoring and accounting system development for the project GridNNN [1,2], aimed to provide a grid infrastructure for the National Nanotechnology Network in Russia. The grid middleware used in the GridNNN project is partially based on well known packages like Globus Toolkit 4 [3] and VOMS [4], and to fit the needs of the specific application area, several grid services were developed from scratch. In such conditions, special monitoring and accounting system was created within the project.

The monitoring is rather a general concept. Most common tasks we deal with are:

- Continuous watching for the state of grid services both common for all infrastructure and in a particular Resource Center;
- Obtaining information on resources (slots number, operation system, hardware architecture, special software packages) and their utilization;
- Access control rules by Virtual Organizations and groups inside them;
- Execution monitoring, tasks and jobs submission, state changes and return codes;
- ...

For effective control, planning and faults detection it is important to know not only the current state of the grid infrastructure but also to keep track of the state history.

Introduction

The GridNNN project aim is to create and support the national nanotechnology network of Russia. The main goal of the project is to provide an effective access to the distributed computational, informational and networking facilities for nanotechnology science and industry. The base middleware of particular GridNNN services (like MDS, GRAM) is Globus Toolkit 4, some services are developed by the project team (e.g. job handling tool Pilot [5], Information Index [6] based on Globus MDS, GRAM connection with "non-standard" Local Resource Management System, Web User Interface, etc.).

The operation of GridNNN, unlike many huge grid projects, is more centralized: there are about 15-30 resource centers (supercomputers) controlled from two operation centers, the main and the backup one having the same set of central services. The infrastructure has one central information index where all information providers have to publish their data. Another difference from most grid project is a variety of different Virtual Organizations (VOs).

Monitoring subsystem overview

Computational jobs specific for GridNNN are parallel and use the MPI technology. They demand a huge volume of computation, but do require to store or transfer a considerable amount of data. Therefore, the monitoring activities in this area are primarily concentrated on tasks and jobs

[1]This work was partially supported by the Russian Ministry of Education and Science, Contract No.01.647.11.2004

tracking (computational jobs are parts of a task and could be interconnected with each other). Jobs monitoring is naturally associated with billing (or accounting) features: it is important to know who, when and where is using the project resources.

End users should have the possibility to observe the main parameters of resource centers' environment to choose the proper center to submit jobs. Most important of them are supported VOs, hardware architecture, number of total available and free slots for running jobs, operation system version, special software packages and so on.

Then, the next main point is overall information on the state of the infrastructure. Now within monitoring subsystem simple test to check whether infrastructure services are alive were prepared. Along with it was created geoinformation real-time visualization of system's operation based on Google Earth, which allows seeing all jobs and tasks events on 3D globe in real time [8]. This feature is of impress and often uses to make graphic presentations.

Also there are several significant tacks related to monitoring but not included to the monitoring subsystem and supported by other teams in the project. Firstly, it is RAT-tests (resource availability tests), sample tasks periodically submitted to each computational resource in the system. The second type of regular checks is examination of the information published by sites to their local information indexes (and then to the central information index).

Information gathering for the monitoring and accounting

There is a special information provider, Service for Registration of Resources and Grid Services (SRRGS), to publish detailed static information on grid sites. The content of the SRRGS is managed by resource centers administrators and project coordinators. The service contains general data on resource centers (name, geographical coordinates, administrators' contacts, etc.) along with the main information on grid services provided (like entry points, type and current state of the service). The SRRGS is the main place in GridNNN to keep registration information on sites and services, and is the origin of such information for the other services in the system (they could get it from SRRGS using simple HTTPS queries).

Information System (including central and local information indexes) contains both slow changing and dynamic information on the resources. Site publishes many clue parameters like job available job slots, system architecture type, OS version, list of special software packages, VO access information – these changes are not so frequent. But there is a small piece of information that contains changing real-time information on state of job queues and available job slots on the site. Information System is based on Globus MDS 4, so to perform a request WSRF queries are used.

For each type of service in the project there is a small simple test just to check if the service is available and is responding to queries specific to if (e.g. Information Index should return non-empty response to a wsrf-queries).

All collected information coming from Information Index, SRGGS and simple service tests is then handled and stored to the monitoring database. The monitoring web interface is used to present both real-time and historical information. Data flows in monitoring are shown on Fig. 1.

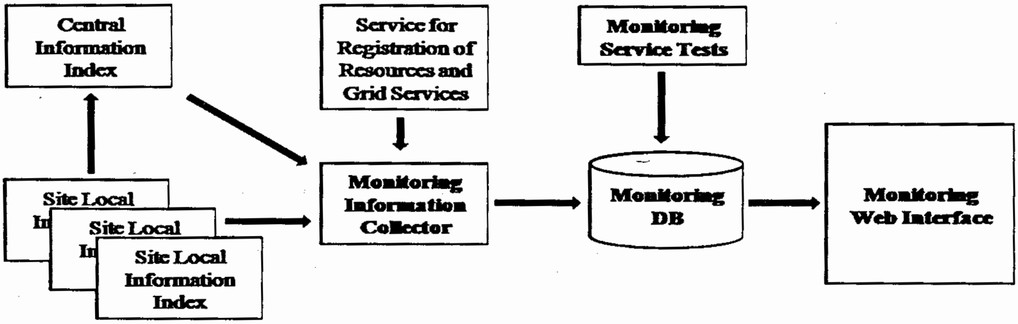

Central Information Index
Service for Registration of Resources and Grid Services
Monitoring Service Tests
Site Local Information Index
Monitoring Information Collector
Monitoring DB
Monitoring Web Interface

Fig.1: Monitoring data processing

The jobs monitoring information and the accounting data are taken from Pilot (job execution service). Pilot servers publish special accounting log containing all the events occur with tasks and their jobs (starting from task submission, sending jobs to the particular resources and to the task finishing or termination). Monitoring service is querying for new events every minute, and then parses result came in JSON format [7]. Obtained events information (task, job, user, VO, start and finish time) is linked with the same events which is already in the database, forming the states of tasks and jobs in question. Accounting data (mainly consumed CPU time) is to be taken from local Grid Resource Allocation Managers (GRAMs) in resource centers. Then it is linking with job information in database. At the end, full aggregated accounting information is in the database and is available via the web-interface (see Fig. 2).

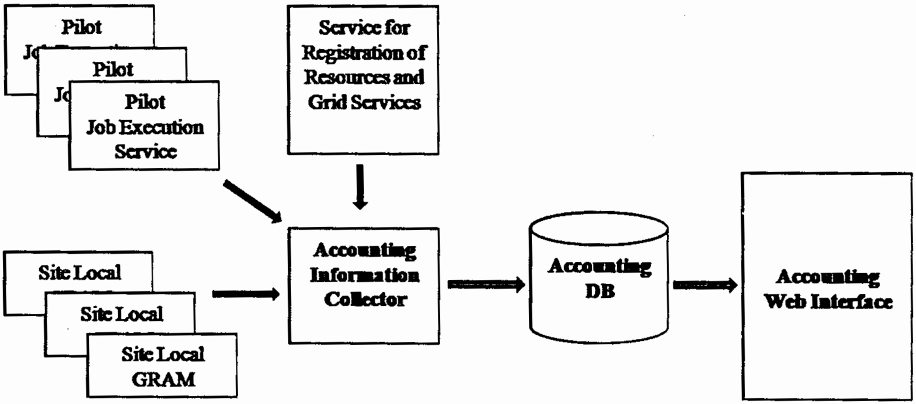

Pilot Job Execution Service
Service for Registration of Resources and Grid Services
Site Local GRAM
Accounting Information Collector
Accounting DB
Accounting Web Interface

Fig. 2: Accounting schema

For the representation of the collected information, the monitoring service has a web-interface [9]. The main parameters to be displayed on the site are states of computational jobs queues, resource characteristics, operation system version and so on (see previous paragraph). Accounting information is available on the site as several report views with tables and diagrams by resources and users.

Real time jobs monitoring allows displaying on 3D globe how and where jobs are started and finished [8]. Special script periodically (each 10 minutes) prepares information on job events based on Accounting DB and makes KML file to use it in visualization in Google Earth. It is probably the best and the most spectacular way of demonstration of the project's operation to the wide community.

Summary

The GridNNN project has its own features and peculiarities, and it differs from other grid projects. Therefore, for making monitoring and accounting for it, along with common means and approaches, some special developments are required. Similar to the whole project, the monitoring subsystem is still developing and evolving. Some practical results are already achieved, and work is still in progress.

References

[1] GridNNN project site (in Russian): http://ngrid.ru

[2] Kryukov A. et al., Architecture of Grid for National Nanotechnology Network (GridNNN), The 4th International Conference "Distributed Computing and Grid-technologies in Science and Education", June 28 - July 3, 2010, Dubna, Russia, http://grid2010.jinr.ru/files/pdf/grid2010-kryukov2.pdf

[3] Foster I. IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2005

[4] VOMS project home page, http://voms.forge.cnaf.infn.it/

[5] Shamardin L. et al, GridNNN Job Execution Service: a RESTful Grid Service, The 4th International Conference "Distributed Computing and Grid-technologies in Science and Education", June 28 - July 3, 2010, Dubna, Russia, http://grid2010.jinr.ru/files/pdf/shamardin-gridnnn_job_execution.pdf

[6] Stepanova M.M. et al, Information System of GridNNN, The 4th International Conference "Distributed Computing and Grid-technologies in Science and Education", June 28 - July 3, 2010, Dubna, Russia

[7] JavaScript Object Notation (JSON), http://json.org

[8] Mitsyn S., Belov S., Development of Real-time Visualization Service Based on Google Earth for GridNNN Project, http://grid2010.jinr.ru/files/pdf/mitsyn_prezent_ver005.pdf

[9] GridNNN monitoring site, http://mon.ngrid.ru

DEVELOPMENT OF REAL-TIME VISUALIZATION SERVICE BASED ON GOOGLE EARTH FOR GRIDNNN PROJECT

S. D. Belov, S. V. Mitsyn

Laboratory of Information Technologies,
Joint Institute for Nuclear Research, 141980, Dubna, Russia
svm@jinr.ru

In this article the problems of geoinformation visualization of grid infrastructure and monitoring are discussed.

The problems of grid infrastructure monitoring as a geographically distributed system, including usage of geographic maps and 3D model of Earth, are reviewed, as well as existing solutions of different GRID projects.

The most attention is paid to the visualization of the grid infrastructure of GridNNN project and to the specially designed service that uses new generation geographic system – Google Earth.

1. Introduction

Traditionally the problem of representation and visualization in attractive and appealing way in grid projects gets low attention, through it's a very important task in different ways. For example, such visualization service would allow presenting grid infrastructure and one of its most important properties – globality and distribution – to end users in easy-to-understand way. It enables them to understand the basic ideas of the grid systems and decide whether it is useful for their business in the beginning. Also it would enable easy orientation in monitoring data which is required to get the information about status of system and its components.

One of the ways to get such service is to involve the functionality of geoinformation systems to existing grid infrastructure monitoring solutions. While grid monitoring is responsible for data acquisition and storage, geoinformation system may be used to represent it. The GridNNN project, along with other grid projects, includes development of national-scale grid infrastructure, which is territorially distributed keeping in mind large territories of Russia.

Currently, the more and more attention is paid to new generation GIS systems – neogeographics 1], and the most popular one is Google Earth. Beneath other differences to traditional geoinformation systems, in these new systems the most attention is paid to usage of raster images. The expressiveness of such systems makes them a good candidate for user interface implementations, including grid visualization services.

The main goal of this work is the creation of visualization service of grid infrastructure as a geographically distributed system along with its monitoring, thus creating geoinformation system. The main requirements include the appealing and expressive view of distributed system and its components and visualization of their interaction and users' activity.

2. Existing solutions

To begin with, existing solutions in the area of grid monitoring which includes geographic visualization are reviewed.

One of the most widely known systems which includes visualization on the map, is MonALISA, the framework for distributed systems monitoring. The visualization is implemented in a Web Repository, which is a entry point for end users and includes MonALISA Repository, which collects and stored data, and Web service to access that data. In last few years the is a move to 3-rd party geoinformation services. For example, ALICE repository uses Google Maps for geographic visualization [3].

Other visualization system – Real Time Monitor – is created especially for geographic visualization of grid infrastructure [4]. It uses NASA free maps. Unfortunately, it does not support plugging in external data or alternatives sources, so it's not suitable for our task.

In Dashboard project a service for geoinformation monitoring data visualization has been created, which uses Google Earth as its user interface [5,9]. This application satisfies a mandatory requirement – an ability to visualize external user-supplied data. Also an important reason to use Google Earth application in GridNNN visualization service was an appealing and attractive user interface and intuitive representation.

3. Design

The designed system is a classical 3-tier application which consists of: back-end is a data source for representation (usually some database), KML file server which is an adaptor that transforms data to a format that Google Earth is able to visualize, and the Google Earth itself that periodically downloads the data packed in KML file from predefined web-address and displays it on the screen.

In principle, the back-and can be any data source. The main reason for factoring out data access functionality was the ability to replace data sources in case of changes in the back-end of monitoring subsystem.

KML file server consists of KML file generator and web-server, which provides access for clients to the KML file. KML (Keyhole Markup Language) is a file format for modeling and storage of geographic objects, which we use to add external visualization data to Google Earth [6].

As user interface Google Earth is used. The advantages of this decision include attractive view, easy installation and configuration (the application is designated for wide range of users) and ease of external data plugging.

4. Implementation

Data sources and user interface are implemented as external to KML file server components: user interface is made entirely of Google Earth application, configured for downloading KML files from specified web address, and data for visualization are provided by external against the service system which replaceability is implemented inside KML file server. Thus, the most work is done on the server part.

KML file server is implemented as a set of Python and shell scripts which periodically run by cron. The server itself consists of four parts: data access layer, KML file generator, manager and apache web-server. The KML file generator is taken from Dashboard project and with some changes integrated into the server.

The data access layer is extracted for the purpose of inclusion different data sources. At the current moment there is only one supported data source – monitoring subsystem database (based on PostgreSQL), through it is relatively easy to add new data sources.

4.1. Google Earth visualization

The most experience of Google Earth visualization described here is collected while working on the Dashboard project on the closely related work.

Visualization in Google Earth is implemented through adding KML files to it. The KML file must be generated according to KML specifications. It contains a set of geographic elements for visualization. Currently these elements are used:

- Dots (icons) – visualize sites and Pilot job management services;

- Lines – job submission and sandbox receipt.

The animation, the most attractive part of visualization, is implemented as limitation of visualization of different elements with a time interval [7]. For example, job submission, which is drawn as a line that flies from pilot to the designated site, implemented as a series of short lines that appear and disappear sequentially.

This way of animation with large amount of visualization events causes explosive growth of KML file size. In the Dashboard project, which has very large amount of jobs to track that cause many events, uses logarithmation scheme: for every unique set of geographic entities that define event localization (for example, job submission localization is defined by pilot and destination site) the logarithm of the number of events with the same type are taken, and the final number of events are randomly spread across 15-minutes time interval. The same scheme is used in the GridNNN visualization, through while the number of events is relatively low the events can be visualized 1 to 1.

On the other side, it can be clearly seen that large amount of data contained in KML files conforms to relatively simple patterns. For example, for a full set of lines related to one job submission event only coordinates and time interval are changed, and the exists functional dependency than can be expressed explicitly. If KML format would support some sort of generation of such elements on the client side, the size of KML files could be decreased by orders of magnitude. It should be noted that KMZ file format, which is a zip compressed KML and is supported by Google Earth, is several times smaller than KML.

4.2. Alternatives

As alternatives, there were considered other user interface applications as well as different schemes.

For example, instead of Google Earth web-applications like Google Maps of Yandex.Maps (Yandex.Карты) can be used. One can import its visualization data into Google Maps with a link to KML file, Yandex.Maps also supports this format together with YMapsML geographical markup language [8]. Unfortunately, during the investigation whether these applications can be used for such purpose it was discovered problems with animation, but they have an advantage over Google Earth in the way that they does not require installation or configuration. All that is needed is to have modern web browser and follow the specified link.

On the other side, a scheme with deployment of KML file generator on the user interface side was considered. Such option could rapidly decrease the internet traffic, but it also complicates installation and configuration of the user interface, thus degrade one of the most important advantages – ease of deployment on the client side.

5. Summary

In summary, the service of grid infrastructure visualization is developed that uses grid monitoring data which has appealing and nice-looking user interface; the ways for further development are specified.

Certainly grid infrastructure visualization is an important part of the whole monitoring system. It enables its users to have fuller information about the infrastructure and better understand it and evaluate its quality of operations. The appealing look helps to understand importance and usefulness of grid computing for different purposes, including business.

References

[1] GridNNN project web page (in Russian), http://ngrid.ru/ngrid/gridnnn/main

[2] www.neogeography.ru/ru/2010-05-03-14-39-41

[3] Grid sites monitoring map - MonALISA repository for CERN ALICE Grid, http://alimonitor.cern.ch/map.jsp

[4] The Real Time Monitor, http://rtm.hep.ph.ic.ac.uk/

[5] Gaidioz B., Rocha R., Mitsyn S., Devesas Campos M. Dashboard Earth User's Guide, http://dashb-cms-job-devel.cern.ch/dashboard/doc/guides/service-monitor-gearth/html/user/index.html

[6] KML Documentation Introduction - KML - Google Code, http://code.google.com/intl/en/apis/kml/documentation/

[7] Time and Animation - KML - Google Code, http://code.google.com/intl/en/apis/kml/documentation/time.html

[8] http://api.yandex.ru/maps/ymapsml/

[9] Mitsyn S.V. Visualization for monitoring of WLCG/LCG grid infrastructure as a geographically distributed system, LIT JINR Scientific report 2009 (in Russian), http://lit.jinr.ru/Reports/SC_report_06-07/p37.pdf

PROBLEMS OF DEVELOPMENT OF COMPLEX MULTI-LAYERED APPLICATIONS IN DISTRIBUTED ENVIRONMENT

A. V. Bogdanov[1], A. B. Degtyarev[2], Soe Moe Lwin[3], Thurein Kyaw Lwin[4]

[1] High-performance computing Institute and the integrated systems,
e-mail: bogdanov@csa.ru, Saint-Petersburg, Russia
[2] St.Petersburg State University, e-mail: deg@csa.ru, Saint-Petersburg, Russia
[3] St.Petersburg State Marine Technical University, e-mail: mogokthar@gmail.com,
Saint-Petersburg, Russia
[4] St.Petersburg State Marine Technical University, e-mail: phothar83@mail.ru,
Saint-Petersburg, Russia

The paper is focused on the description of the development of element of virtual testbed for marine technology virtual examination. Unit for sea environment (wind, waves, other factors) modelling is considered. Multi scale wave climate models are based upon development of distributed hardware – software complexes. Such complexes include a set of physical-mathematical models describing wave climate, including input meteorological data pre-processing and output data post-processing. The technology of distributed calculations and Grid technology provide simulation of the complex problems using remote heterogeneous computational resources, simultaneous visualization of the large amount of the scientific data. Distributed data processing and analysis provides interconnection of the scientific tools with remote computers and data bases.

1. Introduction

Development of virtual testbed for study of complex technical objects behavior requires application of many various models, describing different phenomena. We have such state because in reality there are many processes which are taking place simultaneously. Some of them are independent but some depends on each other. So in general we have joined different kinds of some mathematical models. Real time simulation of all processes which is influencing on final result of complex objects behavior (we can define virtual testbed by such a way too) could not be organized in one computer because adequate modeling requires different kinds of computer resources (for instance: high performance calculation, data processing, visualization, etc.) . At the same time different algorithms in frameworks of virtual testbed require different ways of mapping on multiprocessor architecture [1]. If we consider one complete unit of virtual testbed for marine operations – environment simulation – we obtain complex multi layer application that requires mapping on distributed architecture.

Realization of such complexes requires tremendous computational resources, powerful visualization tools, and elaborated high-performance numerical algorithms. The best way to meet all the requirements is to create a distributed computational environment – Grid, capable to offer computational resources and adequate to problem being solved. In essence, the Grid allows to combine heterogeneous information and computing resources to solve complex problems [2]. Our task is integration.

2. The Concept of Virtual Testbed

The proposed concept of virtual testbed defines process organization as a complicated multilevel system consisting of the following core components:

- Hierarchies of imitating models specifying considered problem areas
- Hierarchies of analytical models giving simplified description of various parties of modelled phenomena
- Information system including database and knowledge base based on methods and models of artificial intelligence
- Control systems and interfaces providing interaction of all system component and interactive work with operator

Virtual testbed development represents complex multistage iterative process. The basic feature of this process consists of necessity to carry out coordination (at conceptual, algorithmic, information and program levels) of heterogeneous models describing various parties of functioning of investigated objects. Choice of admissible alternatives is based on compression of assumed variants that are set by alternatives analysis in complex, especially in non-standard (supernumerary and extreme) situations. Concept of such analysis assumes that estimations of expenses for realization of achieving decisions (expenses for the charge of used resource) do not decrease virtual testbed and it become more and more exact in process of admissible alternatives set narrowing. Thus it is considered as control processes by structural dynamics of system have multilevel, multistage and multifunctional characters.

We can consider virtual testbed as a new generation of computer environment – problem solving environment (PSE) [4],[8]. Main character of such PSE is complication of information processing algorithms, results in necessity of high performance application methods searching for new effective computer procedures and there parallel implementation [5].

Virtual testbed concept is formulated as generalization and development of information processing common methods utilizing high performance computer tools. Such models foresee complex virtual testbed using both complex ship dynamics modeling and software-hardware development. We can note that the following concepts and principles are adaptability, distribution, service orientation, virtualization and fault resistance.

3. Complex Applications in Distributed Environment

The issue of running parallel applications in heterogeneous environment became more obvious after Grid technology was introduced. Observed is a problem-oriented environment for simulation of wave in virtual polygon. Large number of computational applications and problem solving environments (PSE) developed for traditional parallel systems which required modifications in order to enable efficient execution on distributed and heterogeneous environment such as Grid [3].

For example, In this paper, different models of the natural environment of WRF (Weather Research Forecasting – (a regional model of atmospheric circulation), and Wave Watch (wind-wave model) examines the problem of constructing a virtual testbed in the form of PSE. These two models were chosen to illustrate the basic features of the implementation of virtual testbed, since each of them requires working with numerous input data, significant computing resources, processing the output data (data assimilation, visualization and animation), and the output data of one of the models and after the treatment are baseline data for the other. This makes it to verify the information and computing solutions which are proposed for multi-applications in a distributed environment.The key point is to study the problem of run parallel applications in dynamic heterogeneous resources; it is because the requirement to meet the needs of the resource and the resource itself may change at running time.

The considered models (WRF, WW) were implemented, including in the form of parallel applications, traditional (i.e., static homogeneous) parallel systems. The real problem of such applications in a Grid environment is to maintain a high level of parallel efficiency. To ensure efficient use of network resources used special methods to distribute the workload.

Appropriate methods of optimizing the workload must be taken into account two aspects:

- Characteristically applications (for example, the volume of data transferred between the processes, the number of floating point and memory)

- Characteristics of resources (e.g., potential processors, network, memory, and the level of heterogeneity of dynamically allocated resources)

4. Problems Solving Environments

A Problem Solving Environment (PSE) is a specialized software system that provides all the computational facilities needed to solve a target class of problems. These features include advanced solution methods, automatic and semiautomatic selection of solution methods, and ways to easily incorporate novel solution methods. Moreover, PSEs use the language of the target class of problems, so users can run them without specialized knowledge of the underlying computer hardware and software technology [7]. The PSE provides a user-friendly environment, allowing more rapid prototyping of ideas and higher research productivity. There are two main parts in the PSE, Visual Programming Composition Environment (VPCE) and Intelligent Resource Broker (IRB). In the VPCE, a user can visually construct an application from software components. After the user constructs an application, the VPCE will generate a task graph and then submit the task graph to the IRB. Based on computational resources available in a distributed computing environment, the IRB will schedule the tasks to different resources to achieve high performance.

A PSE composed of high performance numerical methods, tools and grid-enabled middleware system for scalable and data-driven computations for multiphysics simulation and decision-making processes in integrated multiphase flow applications.

5. Problem to Solve

Collaborative work within the virtual testbed PSE is provided in two ways. Firstly, different users can start several instances of the PSE; PSE create different numerical experiments and run them independently on the testbed computational resources, replicating the simulation components and sharing access to databases, archives and other resources. Second, different users can connect to one common virtual display, thus having the same graphical output and interactive steering capabilities.

The purpose of the given paper was testing of Grid products as operative environment for the big computer centre. For that propose the experimental test-bed was created with a help of middleware and other applications based of operating system LINUX. As a result of the analysis of variety of possible combinations of computing platforms and the middleware the following choice has been made:

- The intermediate software as the manager - SGE (Sun Grid Engine)
- For Storage and a data control - application IBM DB 2 (Universal Database)
- Creation of a portal and gateway – UNICORE
- Testing of applications - WRF (Weather Research and Forecast System)

We can consider this solution as a PSE for simulation of wave in virtual polygon [6]. The scheme of such complex is shown in a Fig. 1.

Architecture of DB2 UDB is completely parallel and supports parallelized execution of the majority of operations, including inquiries, inserting, updating and removal of data, creation of indexes, loading and export of data. And functionality of DB2, the transition from the standard, not parallel environment of execution ôn parallel, does not meet any limitations at the increase of efficiency. DB2 UDB has been specially developed for successful work in a number of parallel environments including systems MPP, SMP and MPP clusters from SMP nodes.

Often enough it happens that the necessary information is stored in several absolutely various databases or in the information part of file system. Such essential characteristics of DB2 UDB as objective - relational structure and corresponding expansions of DB2, integration with external sources of data by means of technology Data-Links, and also, in particular, algorithms of complex search in a structural types of data makes DB2 an ideal server for realization of the concept of the federal database including a variety of distributed sources and variety of types of data.

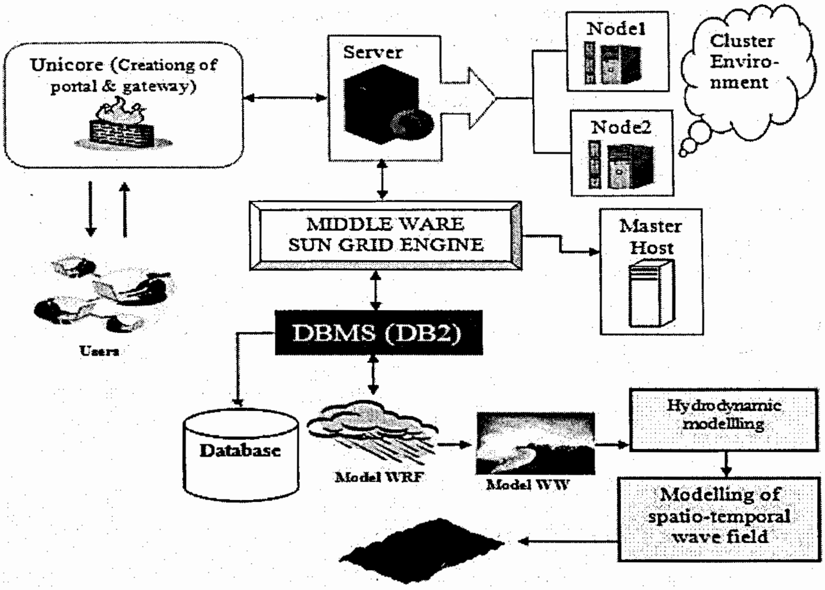


Fig. 1: Data Control and Applications on the Sun Grid Engine Platform

The demands for intellectual information systems always were high, but only recently relational DBMS have found possibilities in a sufficient measure to support such systems. The analysis of data demands the use of the consolidated data sets from numerous operative sources of the data united in a data storage. The storage of data becomes a platform to support the diverse analytical applications. As the storage of data urged to store higher and higher volumes of data with high efficiency and scaling requirements, they become the cores for DBMS.

The main tool of administration is DB2 Control Center which helps to create, delete, and modify databases, tabular spaces, tables, indexes and triggers and to receive the information on their condition and parameters. A number of components (Performance Monitor, Event Monitor and Event Analyzer) allow to carry out within adjustment and monitoring of work of DB2, are used for monitoring of productivity and the analysis of activity of various objects of a database (the table, tabular space, etc.). Visual tools allow investigating the plan of execution of inquiry to analyze what DB2 UDB addresses to data and processes them.

In this paper we also use UNICORE with others systems of grid technology as an effective way to build the secure, simple and seamless access to high performance computing resources for users in grid environment. Then we use the standard DRMAA for the integration of UNICORE with SGE to gain improvements to its current architecture in terms of portability, flexibility and compliance of standards.

The UNICORE system was developed in Germany in order to simplify the access to supercomputing resources. UNICORE is short for "UNIform access to COmputing REsources": The users can use a simple client to access computing resources, instead of obtaining a shell login, transferring the files to the target machine manually, and starting the job using the site-specific commands.

For the integration of UNICORE with the DRMS (SGE) we used the standard DRMAA (Distributed Resource Management Application API). DRMAA is a high-level Open Grid Forum API specification for the submission and control of jobs to one or more Distributed Resource Management Systems (DRMS) within a Grid architecture, giving the functionality required for Grid applications to submit, control, and monitor jobs to local Grid DRM systems. The given integration is used in

UNICORE to change the connection from NJS-TSI to NJS-TSI-DRMAA for the purpose of forwarding the jobs submitted from UNICORE to SGE. Implemented standard DRMAA for UNICORE allow to use of any DRMS, and customers UNICORE allow to create the PSE [9], [12].

These considerations define a set of problems which can be solved on these systems: large databases, complex analytics, and computation problems demanding coordinated operations over large volumes of data which can be divided into relatively independent stages of computing. The control system of such computer network is handled via scheduling of separate tasks, instead of interrelation between separate blocks of one problem.

To solve the problem, we used Sun Grid Engine (SGE) platform, based on the software developed by Genias company and known as Codine/GRM. In SGE, tasks are in a waiting zone and queue for servers to provide services for tasks. This platform allows to:

- Unite some servers or workstations in a single computing resource which can be used both for package problems, and high-performance package computing.
- Load the task in SGE and declare a profile of necessary requirements for its performance.
- Define the task queue parameters and launch it either with the higher priority, or with the longest waiting time, trying to run new tasks for the most fitting or least loaded queue.

The master host is central for cluster activity. The master host runs the master daemon and usually also runs the scheduler. The master host requires no further configuration other than that performed by the installation procedure. By default, the master host is also an administration host and a submit host.

Network administrator can receive monitoring and statistics data to optimize resources usage based on them. Administrative interface allows to set various parameters of problems launching, such as priorities, required resource hardware environment, licenses for specific software, soft performance slots, users access rights for resources etc.

The flexible structure of SGE-based network makes it possible to re-structure the parallel codes to be handled in scheduling of separate tasks, instead of interrelation between separate blocks of one problem [10], [11].

6. General Infrastructure of PSE

Generally the infrastructure of a site within a Grid testbed can be of one of the following types depending on the underlying resources:

- Traditional homogeneous computer cluster architecture: homogeneous production nodes and uniform interconnection links.
- Homogeneous production nodes with heterogeneous interconnections.
- Heterogeneous production nodes with uniform interconnections.
- Heterogeneous nodes with heterogeneous interconnections.

Conclusion

It becomes clear that the concept of problem-oriented environment is not so simple as convenient computing environment with the necessary user interface; although, the only possible tool for solving complex tasks, consists of a large set of different models and at the moment is the most reasonable solution. A complete Grid infrastructure is always the heterogeneous production nodes with uniform interconnections, characterized by heterogeneity with a wide range of processors and network communication parameters. Before working on the deployment of the full PSE, it must create a test site (Testbed), which will be tested by the proposed solutions. To organize such test site, we analyzed several Grid products. By combination of different parameters we propose Sun Grid Engine, the software that best corresponds from the point of view of our objectives. At the moment, this testbed is being created at the Chair of Computer modelling and multiprocessor systems of St. Petersburg State University.

References

[1] Alexander V. Bogdanov., Alexander B. Degtyarev , Yu. Nechaev Parallel algorithms for virtual testbed // Proceedings of International Conference «Computer Science & Information Technologies», September 2005, Yerevan, Armenia, pp.393-398.

[2] Alexander V.Bogdanov, Alexander B. Degtyarev, Elena N. Stankova, Irina V. Shoshmina, Wave weather scenarios modelling using Grid technology. //Proceedings of the 9th International Conference Stability of Ship and Ocean Vehicles STAB'2006, Rio de Janeiro, 2006, pp. .279-286.

[3] Korkhov V.V.,2009 *Hierarchical Resource Management in Grid Computing*. PhD thesis, University of Amsterdam.

[4] Houstis E.N., Rice J.R., 2000 Future problem solving environments for computational science. *Mathematics and Computers in Simulation*, Vol.54, pp.243-257.

[5] Degtyarev A.B, Nechaev Yu.I., 2007 Virtual testbed for marine objects behavior investigation // Proceedings of the 9th International ship stability workshop. Hamburg, Germany, 8p.

[6] Alexander Bogdanov, Lu Moe Khaing, Soe Moe Lwin Deployment the testbed for Grid products testing on the base of Sun Grid Engine// Proceedings of International Conference «Computer Science & Information Technologies», 28 September - 2 October, 2009, Yerevan, Armenia, pp.394-396.

[7] Houstis E.N., Gallopoulos E., Bramley R. and Rice J.R. Problem-Solving Environments for Computational Science. IEEE Computational Science and Engineering, 4(3): 18-21, 1997.

[8] Walker D.W., Li M., Rana O., Shields M.S., Huang Y., 2000 The software architecture of a distributed problem-solving environment.

[9] Riedel M., Menday R., Streit A. "A DRMAA-based Target System Interface Framework for UNICORE".

[10] Alexander V. Bogdanov, Myo Tun Tun. Development of the distributed computing systems and running applications in the heterogeneous computing environment, Saint-Peterburg, 22-25 June 2009 .Works of XVI All-Russia scientifically-methodical conference. " Telematics'2009".pp.425-427.

[11] Sun Microsystems, BEGINNER'S GUIDE TO SUN™ GRID ENGINE 6.2 InstallationandConfiguration,2008,c.2-7. https://www.sun.com/offers/details/Sun_Grid_Engine_62_install_and_config.xml?intcmp=092208-10054308-CAR

[12] La Min Htut. The integration of UNICORE and SGE to create single sign-on for users in the distributed computing environment // Proceedings of 5-th All-Russia conference of young specialists in marine intelligent technologies «Morinteh-Junior 2009». St. Peterburg.10-12, November 2009.pp 85-87.

EUCALYPTUS OPEN-SOURCE PRIVATE CLOUD INFRASTRUCTURE

A. V. Bogdanov[1], M. Dmitriev[2], Ye Myint Naing[3]

[1] High-performance computing Institute and the integrated systems, e-mail: bogdanov@csa.ru, Saint-Petersburg, Russia
[2]Saint-Petersburg State University, e-mail: dmitriev@csa.ru, Saint-Petersburg, Russia
[3]Saint-Petersburg State Marine Technical University, e-mail: yemyintnaing@gmail.com, Saint-Petersburg, Russia

This paper discusses how Eucalyptus can be used as an infrastructure to create private clouds. And also explains how an open source software infrastructure can be used for implementing cloud computing with clusters or workstation farms.

1. Introduction

Cloud computing is the mainstream of information technologies today. This is a natural move to a new perception of information infrastructure, which appeared as a result of the evolution of the technological base of hardware and software of today's computer complexes.

Modern technologies allow virtualizing the whole infrastructure as well as its certain parts. This gives an opportunity to flexibly adjust CPU clock properties, RAM and HDD capacities and bandwidth — all these seemed to be unchangeable in the past days. Virtualized infrastructure allows creating exactly the computing powers that are needed, and adjust them in real time. Thus, the dependence on real computational complexes, their power and location, loses its significance. Moreover, the dependence upon heavily occupied system administrators disappears, for the cloud structure can be changed by users themselves with the help of GUI. This is an actual service, offering the users resources and ways to control them.

The first successful solution and a trendsetter in this field is a commercial service Amazon WS. But the computer science is progressing and many other alternative cloud computing organizations appear. Most interesting among them are open projects, free to use and modify, but at the same time supporting up-to-date standarts. We consider Eucalyptus to be the most promising solution [1-2, 9].

2. Eucalyptus cloud Architecture

Eucalyptus was designed from the ground up to be easy to install and as non-intrusive as possible. The software framework is a highly modular cooperative set of web services that interoperate using standard communication protocols. Through this framework it implements virtualized machine and storage resources that are interconnected by an isolated layer-2 network. From a client application and/or user perspective, the cloud API is compatible with Amazon's AWS (both SOAP and REST interfaces are supported) although other interfaces are available as customizations.

Eucalyptus consists of five main components that work together to provide the requisite cloud services. The components communicate with each other securely using SOAP messaging with WS-Security:

- Cloud Controller (CLC)
- Walrus Storage Controller (WS3)
- Elastic Block Storage Controller (EBS)
- Cluster Controller (CC)

- Node Controller (NC)

In figure (1) CLC is the Cloud Controller which virtualizes the underlying resources (servers, storage, and network). Cloud Controller (CLC) is providing the interface to the management platform with which users of the cloud interact. This interface is comprised of a standard SOAP based API matching the Amazon EC2 API (see Amazon EC2 API below). The Cluster Controllers (CCs) form the front-end for each cluster defined in the cloud. Cluster Controller (CC) generally executes on a cluster front-end machine, or any machine that has network connectivity to both the nodes running NCs and to the machine running the CLC. CCs gather information about a set of VMs and schedules VM execution on specific NCs. The CC also manages the virtual instance network and participates in the enforcement of SLAs as directed by the CLC. All nodes served by a single CC must be in the same broadcast domain (Ethernet). NCs are the machines on which virtual machine instances run. The Storage Controller (SC) provides block storage service (similar to Amazon EBS) while the Walrus storage system spans the entire cloud and is similar to the Amazon S3 in functionality. A Management Platform provides a one-stop console for the cloud administrator to configure and manage the cloud. The Management Platform also exports various interfaces for the administrator, project manager, developer, and other users, with customizable levels of access and privileges. These features can include VM management, storage management, user/group management, accounting, monitoring, SLA definition and enforcement, cloud-bursting, provisioning, etc. [3, 4, 8].

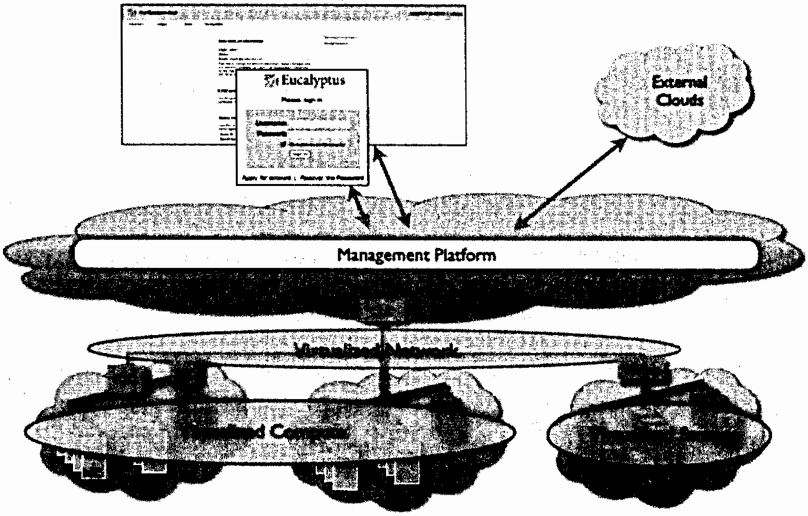


Fig.1: Conceptual Representation of the Eucalyptus Cloud

A Eucalyptus cloud installation can aggregate and manage resources from a single cluster or multiple clusters. A cluster is a group of machines connected to the same LAN. In a cluster may be single or multiple instances of an NC, each of which manages the instantiation and termination of virtual instances.

A single-cluster installation, as shown in Figure 3, will consist of at least two machines: one running the CC, SC, and CLC, and the other one running the NC. This configuration is suitable mainly for experimentation and speedy configuration [4].

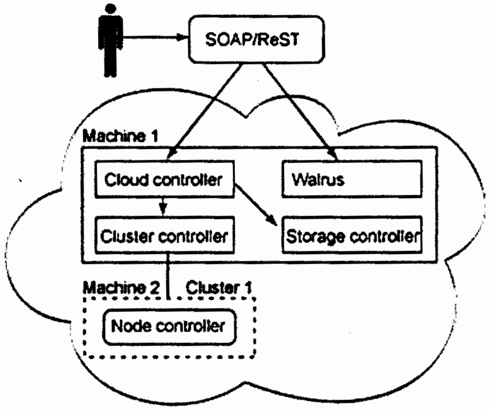


Fig. 2: Topology of a single-cluster Eucalyptus installation

A multi-cluster installation can situate each of the components (CC, SC, NC, and CLC) on separate machines. This is the preferred way to configure Eucalyptus cloud. The multi-cluster installation also gives the opportunity to significantly enhance performance by selecting machines that complement the type of controller running on it. For instance may be select a machine with a super-fast CPU for running the CLC. The choice of multiple clusters will result in higher availability, and in distribution of load and resources across the clusters. The concept of a cluster is similar to the concept of an availability zone in Amazon EC2. Figure 4 shows an example [4].

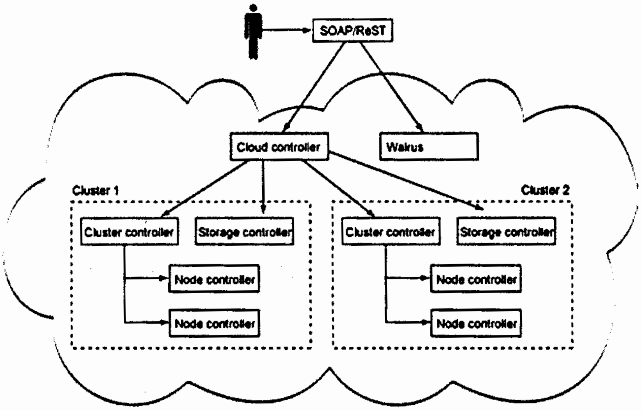


Fig. 3: Topology of a multi-cluster Eucalyptus installation

3. The Eucalyptus Open Source Private Cloud

Eucalyptus is a Linux-based open source software architecture that implements private and hybrid clouds within an enterprise's existing IT infrastructure. A Eucalyptus private cloud is deployed across an enterprise's "on-premise" data center infrastructure and is accessed by user over enterprise intranet. A private cloud is a software infrastructure that enables end-users to acquire, configure, and ultimately release data center resources on-demand, using automated self-service tools and software services within an enterprise's data center. Eucalyptus Systems delivers private cloud software. This is infrastructure software that enables enterprises and government agencies to establish their own cloud computing environments. With Eucalyptus, customers make more efficient use of their computing capacity, thus increasing productivity and innovation, deploying new applications faster, and protecting sensitive data while making savings in capital expenditure [5-6].

4. Eucalyptus Systems

Eucalyptus Systems develops technology solutions built on Open Source Eucalyptus software for private and hybrid cloud computing. Eucalyptus is quickly becoming the standard for on-site cloud computing, delivering the cost efficiencies and scalability of cloud architecture with the security and control of deploying on an organisation's own IT infrastructure. Eucalyptus adds capabilities such as end-user customisation, self-service provisioning and legacy application support to data-centre virtualization features. It makes IT customer service easier, more full-featured and less expensive. Eucalyptus Systems develops and supports Eucalyptus technology. In addition, the company delivers commercial products built on the Eucalyptus platform and provides consulting and support services to customers [10].

5. Nimbus

Nimbus, previously called Virtual Workspace Service, is an open source Globus product for creation of virtual clusters. It is similar in many aspects to Open Nebula for creation and management of Xen VM instances. Current ongoing efforts are to include support for both KVM and VMW are within Nimbus. The main difference comparing Nimbus to Open Nebula is that Nimbus uses grid credentials to authenticate user requests. A user can also start EC2 instances via the same set of credentials making it easier than Open Nebula to interact with EC2. Image management is similar to Open Nebula and Eucalyptus, raw file copies are a common methodology among these platforms. The state of a VM can be "saved" by making a copy of the current image. There is also the notion of an image repository where each user can upload VM images to his or her individual repository.

We have investigated the Eucalyptus cloud computing platform as a general solution to deploying cloud infrastructures. We document the design, installation and usability of the version1.5 platform, including Ubuntu Enterprise Cloud. Ubuntu Enterprise Cloud is an open-source software stack that complements Eucalyptus, allowing for the implementation of a private cloud infrastructure using a combination of Canonical and Eucalyptus Systems technologies. We have also developed a demonstrator that illustrates a real-world use case of the Eucalyptus cloud. The general impression of Eucalyptus is that it is easy to install and configure. It has a well-defined interface that is borrowed from AmazonEC2 [12].

6. Benefits of Eucalyptus

Eucalyptus is the only cloud architecture to support the same application programming interfaces (APIs) as public clouds, and today Eucalyptus is fully compatible with the Amazon AWS public cloud infrastructure. The Eucalyptus design gives users the flexibility to seamlessly move applications from on-premise Eucalyptus clouds to public clouds, and vice versa. Eucalyptus also makes it easy to deploy "hybrid" clouds, which use public and private cloud resources together to get the unique benefits of each.

Eucalyptus enables virtualization of servers, network, and storage in a secure manner, thereby reducing the cost, increasing the ease of maintenance, and providing user self-service. The modular design of Eucalyptus enables a variety of user interfaces, bringing the benefits of virtualization technology to a broad range of users (admins, developers, managers, hosting customers) and provides a platform for service providers to devise profitable consumption-based pricing models. VM and Cloud snapshot features provide an exhaustive set of opportunities to improve cluster reliability, template manipulation, and automation. This makes the cloud easy to use, reduces learning time for the average user, and reduces the turn around time for projects. Leverages existing virtualization technology, supports Linux-based operating systems, and supports multiple hypervisors [3, 11].

7. User Management

The Eucalyptus installation configures a secure HTTP portal on the CLC on port 8443. This web interface is used mainly to manage users and configure the cluster. A part from the web portal, users interact with Eucalyptus services to start/stop VM instances or create EMIs through third party software like the Amazon EC2 API and Amazon EC2 AMI tools. These java command-line tools can be downloaded from the Amazon EC2 site and installed on any machine that a user will be accessing the Eucalyptus services from. To setup the tools, administrators or regular users need to download their authentication credentials from the Eucalyptus web portal and setup appropriate paths and variables. The .eucarc file defines these variables. Each user and administrator of the Eucalyptus cluster will obtain a unique EC2 CERT, EC2 PRIVATE KEY, EC2 ACCESS KEY and EC2 SECRET KEY. This file should be sourced before using Eucalyptus.

Eucalyptus has been designed for its user management system to be fully web-based. A user first requests access to the Eucalyptus installation through its secure web portal by filling out a web form describing his or her profile. This request will generate an email to the administrator who will in turn login to the web portal to grant the user access. Along with the creation of a new user, X.509 credentials and secret keys will be generated for the user to authenticate with the Eucalyptus services. A user will download these keys through the web portal and setup their local .eucarc file with the appropriate paths to the local installation of the Amazon API and AMI tools. Through the web portal, an administrator can also update user profiles as well as delete or disable user accounts. Users can request for a password reset by confirming their profile email address. This system has a potential security flaw as email high jacking can easily compromise the entire user authentication scheme. A high jacker can reset the password through email and gain access to the user's account to download all credential information. Alas, this type of password reset scheme is common among most web portals, including Amazon EC2 [12].

Conclusion

Eucalyptus is a powerful free framework, which allows creating and controlling your own clouds. Its main features are: the support of API, WDSL and REST access methods, it is free of charge and it can be modified easily — having been adjusted to one's preferences, it still integrates with existing solutions in the field of cloud computing.

Using Eucalyptus gives an organization all the power and benefits of creating a cloud infrastructure: it allows virtualizing the existing computational complexes and making a flexible real-time relocation of them, both automatically and by delegating certain authorities to users of computational resources, thus cutting down expenses, raising users' convenience and the efficiency of solving their tasks.

In order to keep up with today's level of information technologies, the organizations have to turn their computing powers into clouds, and make them available for their users. We consider this way of evolution to be extremely efficient and having long-term prospects. Open frameworks like Eucalyptus possess a strong community of users and developers, have the examples of implementation and seem to be very impressive. We are planning a virtualization of all computational powers of the Faculty of Applied Mathematics and Control Processes of Saint-Petersburg State University, so these powers can be used effectively by both teachers and students, also it will help to save on support expenses and electricity supply, and will engage more students in educating process and management of the Faculty's infrastructure.

References

[1] OPEN SOURCE & CLOUD COMPUTING:ON-DEMAND, INNOVATIVE IT ON A MASSIVE SCALE : White Paper June 2009.

[2] EucalyptusSystems,Inc. FiveStepstoEnterpriseCloudComputing http://www.eucalyptus.com/whitepapers

[3] EucalyptusSystems,Inc. Eucalyptus Open-Source Cloud Computing Infrastructure - An Overview.

[4] http://www.ibm.com/developerworks/opensource/library/os-cloud-virtual1/#8.Eucalyptus%20and%20Ubuntu%20Enterprise%20Cloud|outline Cloud services for your virtual infrastructure, Part 1: Infrastructure-as-a-Service (IaaS) and Eucalyptus.

[5] http://www.eucalyptus.com/

[6] http://open.eucalyptus.com

[7] EucalyptusSystems,Inc. Cloud Computing and Open Source: IT Climatology is Born: http://www.eucalyptus.com/whitepapers

[8] Ubuntu Enterprise Cloud Architecture: By Simon Wardley, Etienne Goyer & Nick Barcet – August 2009.

[9] http://www.eucalyptus.com/news/03-19-2010

[10] http://www.ubuntu.com/partners/Eucalyptus

[11] The Eucalyptus Open-source Cloud-computing System: Daniel Nurmi, Rich Wolski, Chris Grzegorczyk Graziano Obertelli, Sunil Soman, Lamia Youseff, Dmitrii Zagorodnov.

[12] An Assessment of Eucalyptus Version1.4: Grid Research Centre, University of Calgary, Canada.

BUILDING USER ACCESS SYSTEM IN GRID ENVIRONMENT

A. V. Bogdanov[1], A. A.Lazarev[1], La Min Htut[2], Myo Tun Tun[2]

[1] *Institute for High-performance Computing and Information Systems (IHPCIS), e-mail: bogdanov@csa.ru, Saint-Petersburg, Russia*
[2]*Saint-Petersburg state university, e-mail: laminhtut@mail.ru, bonge21@mail.ru, Saint-Petersburg, Russia*

In this paper we discuss GSI which define basic characteristics of security in grid system. We use UNICORE with other grid systems as an effective way to build the secure, simple and seamless access to high performance computing resources for users in grid environment. Then we use the standard DRMAA for the integration of UNICORE with SGE to gain improvements to its current architecture in terms of portability, flexibility and compliance to standards.

1. Introduction

The emergence of High Performance Computing (HPC) such as Grand Challenges [1] problems lead to build a system, which provide users secure and seamless access to geographically distributed supercomputers, clusters, storage systems, large-scale visualization systems, advanced devices such as telescope, sensors, etc.., performing like a single virtual supercomputer for users to achieve the need of HPC jobs. This system is named a grid system, formerly metacomputing system [2]. Then grid developed with virtual organizations (VO) [3], giving flexible, secure, coordinated resources sharing facilities for users among collections of individuals, institutions, and resources. Nowadays every modern grid is basing on the Open Grid Services Architecture (OGSA) [4], which enables the integration of services and resources across distributed, heterogeneous, dynamic virtual organizations. A grid is a heterogeneous environment and may contain different resources, many administrative domains with different security policies, finally – different OS. As the heterogeneity nature, the need of security and ease of use for users, organization of user access to resources in grid environment is the real problem for an organization to face.

2. Security issues

When building any kind of distributed computing environment, we must keep in mind few things:

- Protect applications and data from system where computation is going on. In traditional system you must only protect data of local user from being compromised, but Grid-enabled systems compute a lot of data from external uses.
- Stronger authentication needed. Grid requires more responsibility from its users, they credentials should not be compromised.
- Protect local system from remote executive content. If someone has access to our system, no matter what he does, Grid components must work properly.

3. Basic Characteristics Of Secure Access In Grid

Every day millions of people use WWW in various ways, including e-mail, reading news, downloading of music and films, purchases in online shops or simply finding any information. Using a standard Web-browser, the user can get access to information and data, stored on the Web-servers located somewhere in various points of the world. Unlike the Internet, the Grid gives external users and organizations full access to resources, increasing risk of infringements of protection.

To prevent access to resources and the information of users without respective privileges is, in a broad sense, the goal of security of any system. In Globus system, the de facto standard of grid, the Grid Security Infrastructure - GSI defined the requirements for control of illegitimate access or MITM attacks.

Fig 1. represents a simple Grid, consisting of several sites which constitute VO. Let's review the basic characteristics which provide secure access to resources of Grid. We will consider the following example: the user wishes to submit a job to high performance computing resources in Grid system.

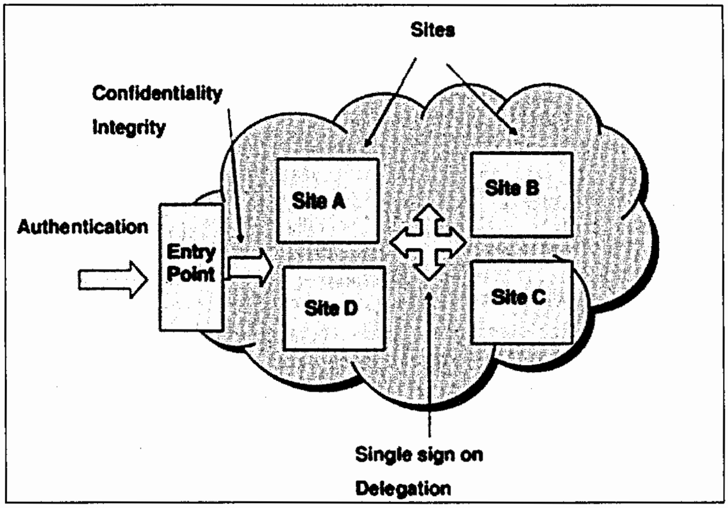


Fig.1: A Simple Grid from the point of view of security

The user starts a job in Grid which comes to an entry or gatekeeper of Grid system. There should be mechanisms that authenticate the user at this point. When the job is started in Grid, it is required to provide confidentiality and integrity so that nobody could see the content of the information or change it. At the very least, mechanisms are necessary for a single entry or login (Single Sign-on) and delegations [5].

4. Single Sign-on (SSO)

Here we will try to explain why single sign-on is the important thing when we make access to resources while solving the complex scientific or engineering jobs in grid environment. Every complex job solved in grid environment can be divided into 3 steps: preprocessing, processing, and post processing. At the stage of preprocessing we will often need the space for intensive volume of data in storage tier and high-throughput network for transferring these massive data. For processing or computing stage, very high computing power is necessary. At the post processing stage we will need powerful graphic server for visualization. And all these stages cannot be lunched on any single computer or cluster; so it is necessary for user to start each stage manually, to take data and to forward to target system. Thus, one of scientific problems is the need of the program environment that gives access for users to work with distributed and heterogeneous resources by a principle of “a single entry” supported by user-friendly interface.

Single Sign-On is the access strategy that allows an end user to log on through a single round of entering credentials and have access to a range of different systems on the network without the need to enter additional credentials.

Compared with legacy approach to user sign-on to multiple systems, using SSO the following advantages can be achieved [6]:

- reduction in the time taken by users in sign-on operations to individual domains, including reducing the possibility of such sign-on operations failing;
- improved security through the reduced need for a user to handle and remember multiple sets of authentication information;
- reduction in the time taken, and improved response, by system administrators in adding and removing users to the system or modifying their access rights;
- improved security through the enhanced ability of system administrators to maintain the integrity of user account configuration including the ability to inhibit or remove an individual user's access to all system resources in a coordinated and consistent manner.

So Single Sign-On in grid means that the user should be registered and authenticated only once at the beginning of a session, getting access to all authorized resources of base level of architecture of Grid.

5. Public Key Infrastructure (PKI)

To provide the SSO a grid site must trust other grid sites. Public Key Infrastructure (PKI) provides users a way to do secure communication in insecure public network using public/private key pair. PKI involves a trusted third party, which is called a certifying authority (CA). In the grid the implementation of a PKI is intended to provide mechanisms to ensure trusted relationships are established and maintained. The specific security functions in which a PKI can provide foundation are confidentiality, integrity, non-repudiation, and authentication.

The framework of a PKI consists of security and operational policies, security services, and interoperability protocols supporting the use of public-key cryptography for the management of keys and certificates. The generation, distribution, and management of public keys and associated certificates normally occur through the use of Certification Authorities (CAs), Registration Authorities (RAs), and directory services, which can be used to establish a hierarchy or chain of trust. CA, RA, and directory services allow for the implementation of digital certificates that can be used to identify different entities. The purpose of a PKI framework is to enable and support the secured exchange of data, credentials, and value (such as monetary instruments) in various environments that are typically insecure, such as the Internet.

The CA issues a digital certificate (end user certificate) to users who may be an individual or an organization. The digital certificate uniquely identifies a user. The digital certificate follows the structure specified by the X.509 system. In an X.509 system a distinct name for the user of the certificate is bound with its public key by a CA. The private key of the certificate is securely kept with the owner of the certificate while the digital certificate containing the public key is available for public use. A piece of data signed by the private key can be decrypted only using the public key and vice-versa. These forms are the basis of the PKI [8].

6. Delegation

Delegation is a common requirement for a wide range access of Grid applications. There may be a need for services to perform actions on the user's behalf. A computational job may require accessing database many times. In that case there is a need to delegate the authority to some service which will perform the action on the user's behalf. When dealing with delegation of authority from an entity to another, care should be taken so that the authority transferred through delegation is scoped only to the task(s) intended to be performed and within a limited lifetime to minimize the misuse of delegated authority. In the grid X.509 proxy certificates are used for delegation.

7. Test-bed Organization

As a possible solution to these requirements for the purpose of testing grid applications at the department of Applied Mathematics and Control Process of Saint Petersburg State University we designed a test-bed including 1) a system for user access, 2) a Distributed Resources Management

System (DRMS), 3) computing clusters, and 4) storage and data management system. Today we have a possibility to choose between many grid products. After the analysis we have chosen the following products:

- System for user access – UNICORE,
- Distributed Resources Management System -- Sun Grid Engine from Sun Microsystems,
- Storage and data management system -- DB2,
- Linux cluster.

8. Why UNICORE (UNIform access to COmputing REsources)?

We choose UNICORE for user access system because firstly UNICORE supports security and access requirements of modern grid system. Secondly it is open and extensible. UNICORE can integrate with other gird middleware such as Globus, with others grid technology such as grid portals which is used to deploy as a user gateway to grid resources. And we intend to use UNICORE and grid portals integration in future for the improvement of client side. Thirdly, UNICORE is implemented in Java and platform independent. At last UNICORE is free software under BSD license.

The UNICORE system was developed in Germany to simplify access to supercomputing resources. UNICORE is short for "UNIform access to COmputing REsources": The users can use a simple client to access computing resources, instead of obtaining a shell login, transferring files to the target machine manually, and starting the job using the site-specific commands.

Detailed configuration of UNICORE components in our test-bed is shown in Fig 2.

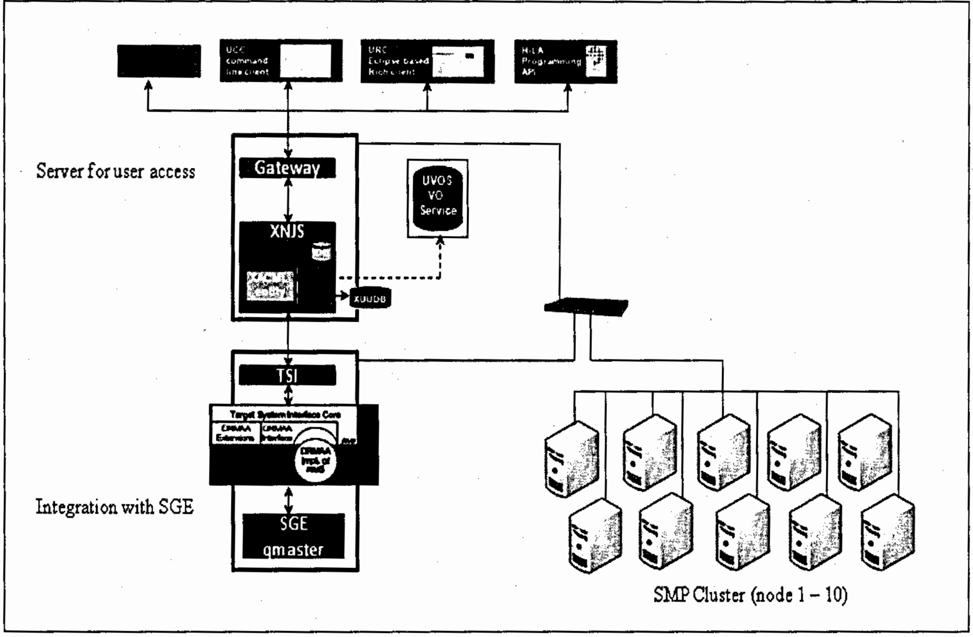


Fig.2: Detailed configuration of UNICORE components in test-bed

UNICORE client: Access to resources and services for the end user is one of the most important components in Grid system. This component is known as User Interface (UI).

In this scheme unicore client is a graphical user interface (GUI) that provides seamless access for users. For preprocessing the user can choose the resources which are managed by SGE to run the jobs: execution node, memory, processing time, application, etc. All inquiries from UNICORE will be forwarded to SGE. At the processing time users can carry out monitoring and management of the submitted jobs, using the part of the interface named job monitoring. Finally, for post processing stage user can get the results of successful jobs. So using UNICORE the users can accomplish all 3 stages discussed in above from one place. Now UNICORE allows to use various clients on demand: Eclipse Based Client, Application Client, Command Line Client, Grid APIs (for example, HiLA), and users can use Portal Clients (for example, GridSphere) as an extension. The Portal is used as an entry point on a site through which it will be possible to get into Grid. The Portal is accessible via usual browser (Firefox, Internet Explorer, Opera etc.) so it is not necessary to install additional client programs which can work only with UNICORE.

UNICORE security layer or service layer: From the security aspect, UNICORE takes advantages from using mature security mechanisms X.509 certificates. The UNICORE gateway receives incoming client connections and authenticates them. After the client has been authenticated, the gateway provides more information about the available systems to the client, i.e. the XNJS that have registered with the gateway. All the connections within the UNICORE are based on more secure TLS connections. For authorization of users the XNJS uses the XUUDB user database to perform the mapping from X.509 certificates to the actual users' logins and roles. Full X.509 certificates are used as base line, while the access control is based on XACML policies. Besides, Virtual Organization (VO) service can be used for user authorization, using the SAML standard. Deployment of SAML for SSO assumes that the user can use standard Web browser (HTTP or HTTPS protocols) and passe authentication on source site.

UNICORE target system layer: The Target System Interface (TSI) takes specific job requests and executes them on the target system, using the local user determined by the XNJS. The TSI connects to the XNJS using a plain-text TCP connection.

For the integration of UNICORE with the DRMS (SGE) we used the standard DRMAA (Distributed Resource Management Application API). DRMAA is a high-level Open Grid Forum API specification for the submission and control of jobs to one or more Distributed Resource Management Systems (DRMS) within Grid architecture, giving the functionality required for Grid applications to submit, control, and monitor jobs to local Grid DRM systems. The given integration is used in UNICORE to change the connection from NJS-TSI to NJS-TSI-DRMAA for the purpose of forwarding the jobs submitted from UNICORE to SGE [7, 9].

9. CA

Own Grid CA can be created by a simple tool such as openssl. The given package supports making RSA keys, DSA, DH and X.509 certificate (which is need for UNICORE grid), signing them and forming Certificate Signing Request. With the connection to UNICORE every components of UNICORE must have X.509 certificate signed by CA, which is trusted by all UNICORE components and CA certificate. Then user certificate signed by CA is delivered to end users.

10. UVOS (UNICORE Virtual Organization System)

VO is the fundamental key concept in building of user access to gird resources. VOMS (Virtual Organization Management Service) is a system for managing authorization data within multi-institutional collaborations to generate Grid credentials for users. Such functionalities are supported in UNICORE with UVOS.

But there are some considerations of user access system in grid relating with VO. Compared with the Cloud Computing, grid has bottleneck with VO. Generally VO is created in gird environment for a certain task or application (e.g. one VO for financial modelling or VO for weather forecasting). So for every application we want to run in grid environment we have to create VO with appropriate policies for users to run the application.

Fundamental aspect to large grids such as EGEE, if there are 200 applications to run, administrators must create 200 VOs and have to manage over 10000 users. This is still too heavy work for user access system in grid environment. Cloud has advantage in this with “pay per use” policy giving required resources for users to run the application on demand.

But on the other hand Cloud has disadvantage in security. At present days security model in cloud computing seems simpler than security model in grid. In user access system of cloud computing infrastructure of registration is typically dependent on web form (through SSL) to create and manage account information for end users, enabling for users to change own password and receive new password via email unsecured and unencrypted connection over the internet. New users can use cloud easily with a credit card and/or email. But grid is stricter with its security infrastructure. For example, as cloud, although grid also uses web form to manage the user accounts, still needs personal conversations, to check the personality. Security approach to grid may need more time but it gives additional security level to prevent unauthorized access.

Conclusion

As a result, basing on UNICORE, the system which provides secure, seamless access for users to all Grid resources managed by local DRMS is offered, providing a single entry for users for authentication, authorization, submitting jobs, launching applications and receiving of results. The integration with SGE is more effective for collecting resources and extension of XNJS in UNICORE to submit the jobs to the target system. At this moment such ideology is used for organization of access for users to high-performance computing resources at the department of Applied Mathematics and Control Process of Saint Petersburg State University and their integration with others systems based on Grid technology.

References

[1] Grand Challenges; http://parallel.ksu.ru/docs/Parallel/faq/22.txt

[2] Ian Foster, Carl Kesselman; Globus: A Metacomputing Infrastructure Toolkit.

[3] Ian Foster, Carl Kesselman, Steven Tuecke; The Anatomy of the Grid: Enabling Scalable Virtual Organizations.

[4] Ian Foster, Carl Kesselman, Jeffrey M. Nick, Steven Tuecke; The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration.

[5] The Open Group; Introduction to Single Sign-On. http://www.opengroup.org/security/sso/sso_intro.htm

[6] Anirban Chakrabarti; Grid Computing Security, p. 49-65, 2007.

[7] Riedel M., Menday R., Streit A. A DRMAA-based Target System Interface Framework for UNICORE.

[8] Joel Weise, Sun BluePrints: Public Key Infrastructure Overview, August 2001.

[9] UNICORE Site; http://www.unicore.eu/

DEVELOPMENT OF THE DISTRIBUTED COMPUTING SYSTEMS AND RUNNING APPLICATIONS IN THE HETEROGENEOUS COMPUTING ENVIRONMENT

A. V. Bogdanov[1], A. Lazarev[1], Myo Tun Tun[2], La Min Htut[2]

[1] *Institute for High-performance computing and integrated systems, e-mail: bogdanov@csa.ru, Saint-Petersburg, Russia*
[2]*Saint-Petersburg state university, e-mail:bongelay@gmail.com, laminhtut@mail.ru, Saint-Petersburg, Russia*

A grid can abstractly be viewed simply as a collection of computational resources, able to compute tasks. Ones such set of resources is available, the challenge is to manage them as a single huge machine. Once it's done, the end users can view the grid as an opaque resource to work with, without being concerned about where the processes run.

The Sun Grid Engine provides a management solution for a particular type of grid that consists of many heterogeneous computational resources working together for users of a single project: the Cluster grids. For this type of grids the ability to control where and how the processes run becomes essential.

In this presentation we discuss different problems of Sun Grid Engine tuning to put heterogeneous resources together into a single opaque resource.

1. Introduction

Controllability of distributed resources is reached through virtualization, by means of transition to higher level - from boxes, ports and disks to virtual volumes of a data storage, virtual computing resources and virtual sub network. Today considerable amount of dynamic management data is gathered by computing resources inside the actual end user computer. So, for example, products by Sun Microsystems are based on multithreading supporting dynamic domains and Solaris Resource Manager, supporting both local and distributed computing resources. In such situation, it`s not so important for the user, which specific node of a network executed his task; it simply consumes defined amount of the virtual processor power available in a network.

2. Sun Grid Engine Component Architecture

These considerations define a set of problems which can be solved on these systems: large databases, complex analytics, computing problems demanding coordinated operations over large volumes of data which can be divided into relatively independent stages of computing. The control system of such computer network is handled in scheduling of separate tasks, instead of interrelation between separate blocks of one problem.

To solve the problem, we used Sun Grid Engine (SGE) platform, based on the software developed by Genias company and known as Codine/GRM. In SGE, tasks are in a waiting zone and queue for servers to provide services for tasks. This platform allows to:

- unite some servers or workstations in a single computing resource which can be used both for package problem, and high-performance package computing.
- load the task in SGE and declare a profile of necessary requirements for its performance.
- define the task queue parameters and to launch it either with the higher priority, or with the longest waiting time, trying to run new tasks for the most fitting or least loaded queue.

Network administrator can receive monitoring and statistics data to optimize resources sage based on them. Administrative interface allows to set various parameters of problems launching, such as priorities, required resource hardware environment, licenses for specific software, soft performance slots, users access rights for resources etc.

The diagram below shows the components of a Sun Grid Engine cluster at a high level:

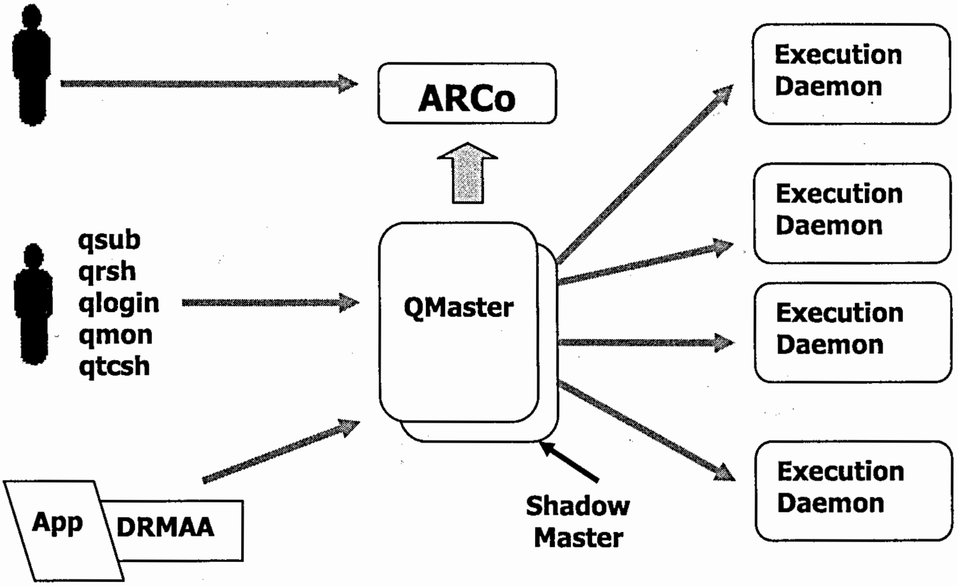


Fig.1: Sun Grid Engine Component Architecture

At the center of the diagram is the QMaster. The QMaster is the central component of a Sun Grid Engine computational cluster, accepting incoming jobs from users, assigning jobs to resources, monitoring the overall cluster status, and processing administrative commands. The QMaster is a multi-threaded daemon that runs on a single host in the compute cluster. To reduce unplanned cluster downtime, one or more shadow masters may be running on additional nodes in the cluster. In the case that the QMaster or the host on which it is running fails, one of the shadow masters will promote the host on which it is running to the new QMaster node by starting a new QMaster daemon locally [1].

Each host in the cluster that is to execute jobs will host an execution daemon. The execution daemon receives jobs from the QMaster and executes them locally on its host. The capacity to execute a job is known as a job slot. The Sun Grid Engine software does not place a limit on the number of job slots that an execution daemon can offer, but in most cases the number of job slots is determined by the number of CPU cores available at the execution host. When a job has been completed, the execution daemon notifies the QMaster so that a new job can be scheduled to the now free job slot. At a fixed interval each execution daemon will send a report of its status to the QMaster. If the QMaster fails to receive several consecutive load reports from a particular execution daemon, the QMaster will mark that execution host and all its resources as no longer available and will remove it from the list of available job scheduling targets.

Jobs are sent to the QMaster in a variety of ways. DRMAA provides a programmatic interface for applications to submit, monitor, and control jobs. The Sun Grid Engine software comes with C and Java™ language DRMAA bindings, making it possible to use DRMAA from a wide range of applications. qmon is the Sun Grid Engine graphical user interface. From qmon users and administrators can submit, monitor, and control jobs as well as manage all aspects of the cluster. qsub is a command-line utility for submitting batch, array, and parallel jobs. qsh, qrsh, qlogin, and qtcsh are all command-line utilities for submitting various kinds of interactive jobs.

The last component shown in the diagram is ARCo, the Accounting and Reporting Console. ARCo is a web-based tool for accessing Sun Grid Engine accounting information automatically stored in an SQL database. Using ARCo, end users and administrators can create and run queries against the cluster's accounting data. Administrators can also pre-configure ARCo with common queries for their end users to run.

3. Sun Grid Engine hosts

The first step to design a SGE cluster is to define the topology of the resource collection by selecting the right machines for the right tasks. SGE uses basically 4 hosts types: Master host, Execution host, Administration host and Submit host.

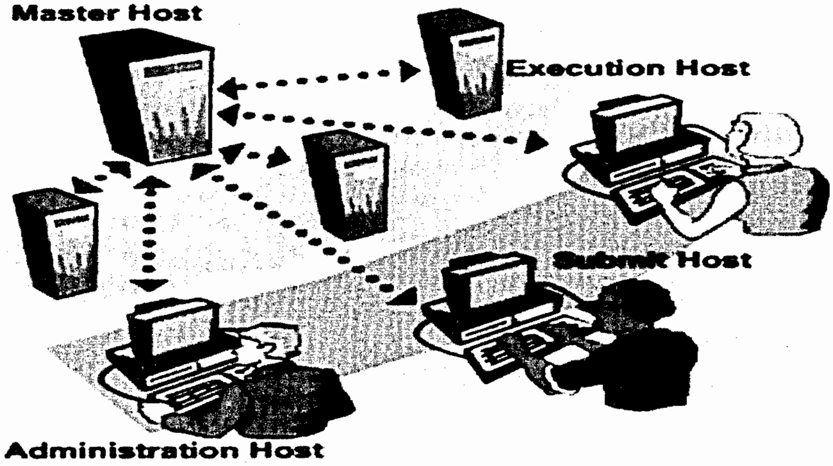


Fig.2: Sun Grid Engine hosts

Each host can be at the same time a member of more than one category and according to that, it will run the appropriate SGE daemons. The only restriction to this subdivision of hosts is that it can exist only one Master host inside a single high-coupled SGE grid (called a Cell). The master host is basically responsible to maintain all the information about the cluster and take decisions on jobs scheduling, acting as the director of the grid. The execution hosts are nodes that have the permissions to run SGE jobs using a set of SGE queues, they represent the computational core of the cluster, sharing the concrete resource and providing CPU power, storage and memory capabilities. The administration host perform any kind of administration functionality like for example monitoring, usually this tasks are performed from the same machine that is the Master host. The submit hosts are those nodes allow to schedule batch jobs to the cluster, these kind of jobs are all the tasks that don't need interaction from the user, therefore the scheduling consists only on submit the job and save the output to a file rather than manage an interaction between the user and the job in the case of for example programs with a graphical user interface.

4. Configuring Parallel Environments

A parallel environment (PE) is a software package that enables concurrent computing on parallel platforms in networked environments. A variety of systems have evolved over the past years into viable technology for distributed and parallel processing on various hardware platforms. The following are two examples of the most common message-passing environments:

- PVM – Parallel Virtual Machine, Oak Ridge National Laboratories
- MPI – Message Passing Interface, the Message Passing Interface Forum

Public domain as well as hardware vendor-provided implementations exist for both tools. All these systems show different characteristics and have segregative requirements. In order to handle parallel jobs running on top of such systems, the grid engine system provides a flexible, powerful interface that satisfies various needs.

The grid engine system provides means to run parallel jobs by means of the following programs:

- Arbitrary message-passing environments such as PVM or MPI.
- Shared memory parallel programs on multiple slots, either in single queues or distributed across multiple queues and across machines for distributed memory parallel jobs.

Any number of different parallel environment interfaces can be configured concurrently.

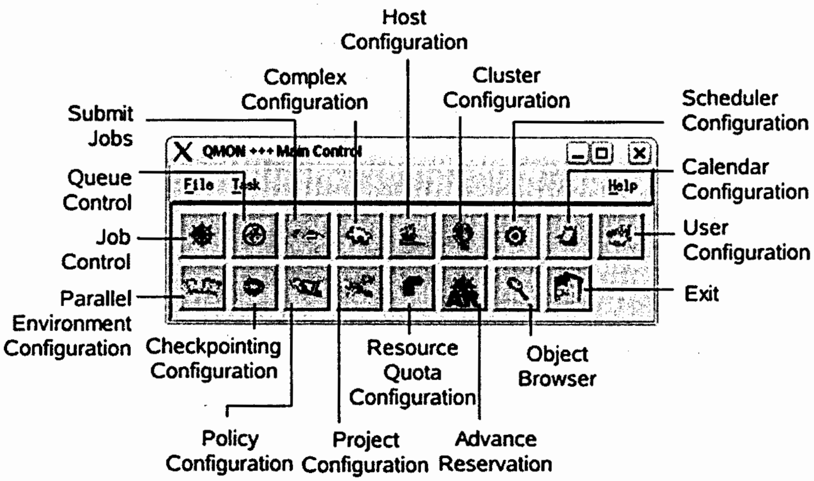


Fig.3: Configuring Parallel Environments

5. Three methods of mpich2

This new MPICH2 implementation of the MPI-2 standard was created to supersede the widely used MPICH(1) implementation. Besides implementing the MPI-2 standard, another goal was a faster startup. To give the user a greater flexibility, there are 3 startup methods:

mpd: As the primary startup method. It's based on the script language Python to startup a so called ring of machines. Giving mpdboot a list of nodes it will startup daemons on the requested machines, which can be used immediately for the execution of parallel programs inside this ring. This is convenient for the interactive use of a parallel program, as the only thing which must be prepared is a list of to be used nodes.

Due to limitations in mpdboot, it must have a connection to the daemons on the nodes via stdin/stdout, until the mpd fork into daemonland.

smpd: This startup method can be used in a daemon based or daemonless mode. The daemon based startup is not creating all the daemons on the nodes according to a nodelist on its own (like it is done by the mpdboot command in the mpd startup method), but the daemons have to be started before the execution of the main program, e.g. by a script.

A daemonless startup is very similar to the startup of the tasks in the former MPICH(1). Although it includes the same scripts from the original $SGE_ROOT/mpi, so that it can easily be used with a still installed $SGE_ROOT/mpi without any side effects.

gforker: Programs started under gforker are limited to one machine and supports only forks for additional processes.

6. Tight Integration of Parallel Environments and Sun Grid Engine

Configuring Parallel Environments With QMON mentions that using sge_execd and sge_shepherd to create parallel tasks offers benefits over parallel environments that create their own parallel tasks. The UNIX operating system allows reliable resource control only for the creator of a process hierarchy. Features such as correct accounting, resource limits, and process control for parallel applications, can be enforced only by the creator of all parallel tasks.

Most parallel environments do not implement these features. Therefore parallel environments do not provide a sufficient interface for the integration with a resource management system like the grid engine system. To overcome this problem, the grid engine system provides an advanced parallel environment interface for tight integration with parallel environments. This parallel environment interface transfers the responsibility for creating tasks from the parallel environment to the grid engine software.

The distribution of the grid engine system contains two examples of such a tight integration, one for the PVM public domain version, and one for the MPICH MPI implementation from Argonne National Laboratories. The examples are contained in the directories sge-root/pvm and sge-root/mpi, respectively.

7. Remaining issues and problems

There may be a problem running Sun Grid Engine on heterogeneous environment. If you have more than one cluster with Sun Grid Engine installed, you should use higher level software to organize proper job submission.

At that moment, the most popular example of that kind of software is Globus Toolkit. It is an open source project, which includes software for security, information infrastructure, resource management, data management, communication, fault detection, and portability [9].

Also, Sun Grid Engine software package does not contains any kind of Web-based frontend to submit and monitor jobs. If you need one, Grid Engine Portal should be used. It is an integrated web-enabled platform that provides virtualized, unified and secure access to Grid Services for end-users.

Grid Engine Portal enables the submission of jobs, query the status of these jobs, get notification on their completion etc.

Conclusion

Studying the Sun Grid Engine infrastructure, the first thing that is easy to recognize is the hard relationship that binds SGE to a familiar UN*X environment. The structure of the system is quite simple and it often uses already existed and trained technologies without trying to provide complex features and with the constant idea to keep the things simple.

The flexible structure of SGE-based network makes it possible to re-structure the parallel codes to be handled in scheduling of separate tasks, instead of interrelation between separate blocks of one problem.

References

[1] Sun Microsystems, BEGINNER'S GUIDE TO SUN™ GRID ENGINE 6.2 Installation and Configuration, 2008, c.2-7. https://www.sun.com/offers/details/Sun_Grid_Engine_62_install_and_config.xml?intcmp=092208-10054308-CAR

[2] Andrea Sottoriva, Julien Bonjean. Sun Grid Engine architecture overview, MSc Grid Computing Universiteit van Amsterdam, The Netherlands, 2006.

[3] Bob Porras. Sun Grid Engine 6.2 – what opensource can do for you, 2008, , c.1 – 4. http://blogs.sun.com/bobp/entry/sun_grid_engine_ru

[4] DanT. Sun Grid Engine 6.2 Information, 2008. http://blogs.sun.com/templedf/entry/sun_grid_engine_6_2

[5] Critchlow J. Installing Sun Grid Engine Software, 2008. http://wikis.sun.com/display/GridEngine/Planning+the+Installation

[6] Lubomir Petrik. Sun Grid Engine 6.2u3 Released, 2009. http://blogs.sun.com/lubos/entry/sun_grid_engine_6_2u31

[7] Sandra.Konta. Sun Grid Engine Administering Guide, 2009. http://wikis.sun.com/display/gridengine62u3/Administering+Guide+%28Printable%29

[8] N1 Grid Engine 6 Administration Guide http://docs.sun.com/app/docs/doc/817-5677/6ml49n2c0?a=view

[9] Globus Toolkit Documentation. http://www.globus.org/toolkit/

UNCONVENTIONAL USE OF DISTRIBUTED DATABASES FROM SERVER CONSOLIDATION TO CONSOLIDATION RESOURCES

A. V. Bogdanov[1], Thurein Kyaw Lwin[2], A. Shuvalov[3], Soe Moe Lwin[4]

[1] High-performance computing Institute and the integrated systems, e-mail: bogdanov@csa.ru, Saint-Petersburg, Russia
[2,4]St.Petersburg State Marine Technical University,e-mail:trkl.mm@mail.ru,
[3]Saint-Petersburg state university, anatoly.shuvalov@gmail.com, Saint-Petersburg, Russia

This paper describes the consolidation of large distributed computational complexes on the basis of database. The organization of access to the distributed computing resources, especially in the solution of the large complex problems, is a challenge for any general purpose computer centre. Consolidating data in distributed heterogeneous systems is an important and challenging task. The existing approach to solving this problem is the most suitable approach to the organization of federal databases.

1. Introduction

Data consolidation is a strategically important task for effective management of corporate and scientific information within distributed computing environment. Due to the continuous increase complexity of applications and volumes of data generated, the company spend on such tasks more money, time and effort. Analysts at Gartner estimated the market size of integration at the end of 2007 is 1.44 billion dollars, with annual growth of more than 17%.

The main reason for consolidation is the geographical distribution of data sources, as well as their syntactic, systemic, semantic and structural heterogeneity. Funds Consolidation used in corporate enterprise systems, with integration of heterogeneous information sources to support applications, business analysis, forecasting and management, as well as in distributed scientific systems for sharing access for knowledge bases and research findings. Data Consolidation covers practices and architectural approaches and software tools to ensure having consistent access and delivery of data for all range of applications and business processes. Therefore, the strategy and program tools used for consolidation fully depend on the characteristics of each particular system. The purpose of this study is to examine the software and approaches to overcome the problems associated with geographic remote data sources, as well as their syntactic and systemic heterogeneity. The problems of semantic and structural heterogeneity can also be solved through the creation of ontology and patterns of compliance using the methods of data cleaning that goes beyond the scope of this article.

The cloud computing is bringing together multiple computers and servers in a single environment designed to address certain types of tasks, such as scientific problems or complex calculations. This structure builds up a lot of data, distributed computing nodes and storage. Typically, applications executed in a distributed computing environment, apply to only one data source. However, when simultaneous access to multiple sources is required, difficulties arise because these sources may contain heterogeneous data and methods of access, and located at a distance from each other. In addition, for users engaged in an analysis of accumulated data, it is convenient to apply to a single source of information, creating queries and get results in the same format [1]. Thus, the main approaches to the problem of storage of information in distributed computing systems are heterogeneous and remote data sources. The solution is to create a centralized access point, providing a single interface to access all sources of data computing clouds in real time. You must select the most appropriate approach and a relevant platform that provide consolidation.

2. Consolidation Technology

All existing approaches to the consolidation of distributed data sources can be divided into two types of Centralized approach and Federated approach.

2.1. Architecture of centralized databases

The centralized approach to the consolidation of distributed data sources is duplication of data from all sources in the central database. Such databases are called data stores. Usually the data warehouse used by a relational database with advanced tools for integration with external sources [2]. Availability of data combined in a single source speeds user access to data and facilitates normalization and other similar processes compared to the case of data scattered in different systems. However, integration of information in a centralized source requires the data that are often in different formats, are reduced to a single format, a process that can lead to errors [2]. Also for the repository it can be difficult to work with new sources of data in unfamiliar formats. Moreover, the processing costs are often increased because of the need to duplicate data and process the two sets of data.

2.2. Architecture federated databases

Federated database - a mechanism to access and manage heterogeneous data, hiding from user a particular data source, but providing a uniform interface instead, similar to the classical relational database. The most applicable approach to creating a platform for federated database is an approach to develop the existing relational database management system to ensure its interaction with external data sources. This database is a central node of a federal database that keeps all the necessary information on the sources of data, and forwards requests to the sources of their parts [2]. System database directory of the central node should contain all necessary information about data sources in general and on each of the objects in particular [2]. Such information should be used by the optimizer of SQL-queries to build the most efficient query execution plan.

2.3. Comparison of federal and centralized approaches

Feature of federal databases is a logical integration of data when the user has a single point of access to the totality of data, but the data itself physically remain in the original source [2]. This feature is a key difference from the centralized approach that uses physical integration, where data from disparate sources are duplicated on a common node that is accessed by all users. The federated approach involves storing data in their respective sources, while the central node performs the query translation, taking into account the characteristics of the source [2].

In case of cloud computing, federated database is a more appropriate choice for the following reasons:

1. Federated technology is less prone to distortions and integrity errors, because the data remain in their original locations.

2. In federated architecture it is easier to add new sources, which is especially important in dynamic systems.

3. The federated approach, in contrast to a centralized, always guarantees the receipt of actual data from the primary source, whereas in the centralized approach, a copy of the data in the central site may become outdated.

It should be noted that in complex cases that require the intersection of large data sets from different sources, federated database must provide the ability to store the information centrally, thus ensuring a hybrid approach.

3. Forms of Consolidation

Database consolidations can take many different forms. A physical consolidation focuses on reducing the number of physical servers, disk storage, database instances and databases. Geographic consolidation involves centralizing servers in one location. Logical consolidation entails centralizing applications or data by their business functionality. And vendor consolidation trims the number of database suppliers. Database consolidation projects can be triggered by several different factors. Often, an enterprise will have acquired new applications and databases through mergers and acquisitions. Or it may have multiple versions of a database purchased over time. Finally, when a company opts to re-architect its underlying infrastructure, perhaps moving to a grid infrastructure, for example, database consolidation may be appropriate [3].

A database consolidation project is not a trivial task. Like other major IT projects, a database consolidation project has six phases--analysis, design, development, test, implement, and monitor. The goal of the evaluation is to determine the performance of the existing infrastructure, assess which parts of the infrastructure should be retained, and develop a blueprint for the new architecture. Close cooperation will insure that the consolidation project will achieve its goals. Once the blueprint is developed, the hardware infrastructure must be configured and the databases migrated to the new platform. Finally, the applications must be moved.

In a database consolidation project involving heterogeneous databases, maintaining existing applications can represent a potential hidden cost. Each major database vendor uses its own version of SQL, and reworking existing applications can represent as much as 30 percent of a database consolidation project [3]. Clearly, consolidating on a universal database platform with multiple language compatibility offers significant advantages. It can radically decrease the cost of application maintenance and cut the time needed to complete the consolidation project.

4. Consolidating Servers

Server consolidation is a big topic for data center managers. Server consolidation is an approach to the efficient use of computer server resources in order to reduce the total number of servers or server locations that an organization requires. The practice was developed in response to the problem of server sprawl, a situation in which multiple, under-utilized servers take up more space and consume more resources than can be justified by their workload [4]. Servers are still the primary focal point for consolidation because they are so obvious. Whether you have 100 servers or 5000 servers, you probably have too many to manage effectively. Today's distributed computing environment lends itself to a proliferation of servers. Reducing and controlling the number of devices to manage and simplifying ways to manage them is the goal of most IT groups [5].

4.1. Identifying Patterns in an End-to-End Architecture

The end-to-end architectures that are prevalent today, tiers of servers are specialized for particular tasks. When you look at consolidating servers, you need to look for patterns in your server population. When you identify these patterns within tiers, you can start to devise a consolidation strategy [5]. Scalability is the key, here. Because you are expected to deliver predictable service levels in response to unpredictable workloads, it is important that you use the right type of scalability for each part of a consolidated architecture. The following sections describe common patterns in an end-to-end architecture. For consolidation discussions, we generally assume that there are three server types, or tiers:

- The presentation tier is the closest tier to the end user.
- The business, or middleware, tier is where applications or middleware run in conjunction with the other tiers.
- The resource tier is where large, scalable servers run mission-critical applications and databases.

Although architectures with these characteristics have been around for a while, most corporations still have many servers running monolithic applications. In many cases, these are older servers running mature applications [5]. These servers are generally excellent candidates for server and application consolidation.

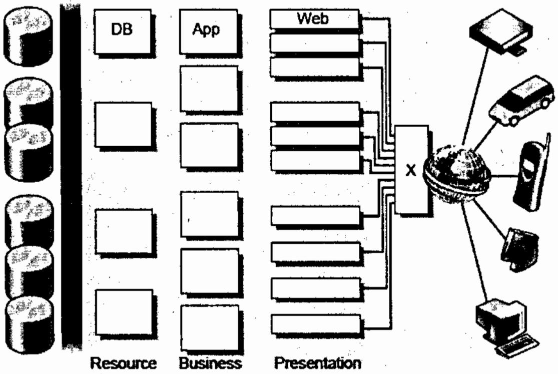


Fig. 1: End- to- End Architecture

5. Data Consolidation

Data consolidation is the main approach, which uses data warehousing applications for building and maintaining operational data warehouses and enterprise storage [5]. Consolidation of data can also be used to create the dependent data marts, but in this case, the process of consolidation is only one data source (e.g., enterprise storage). In the data warehouse environment one of the most common technology is ETL (extract, transform, and load - extract, transform, and load) [5]. Another common technique of data consolidation is content management (enterprise content management, abbr. ECM). Most ECM solutions aimed at consolidating and managing unstructured data such as documents, reports and web-page.

6. Consolidating Data Centers

Many organizations are looking to consolidate multiple data centers into one site. These consolidations range from simple city-wide consolidations to complex region wide consolidations [5]. Shutting a data center is a huge task, and before you even start down the path, it is vital that you can articulate and defend your reasons for doing it. Further, once a data center is shut down, the costs of reopening it can be enormous. From there, data center consolidations are similar to other types of consolidation, except that assessment (especially application, networking, and physical planning) and implementation become much more complex [6].

7. Optimization

As networks, applications, and services grow more complex and users expect to conduct unified communications without a compromise in functionality or performance, a company's distributed legacy infrastructure is hard-pressed to withstand the strain. Toss in the occasional corporate merger or acquisition that expands the enterprise and ratchets up network and application disparity and the situation borders on untenable. Consolidation promotes several avenues to optimization. One of them is the aforementioned transport. With a more centralized approach, there are fewer "pipes" to monitor, the architecture is more straightforward and easier to control, and traffic patterns and volumes are more visible and clearly defined. This environment offers the option to implement more advanced protocols and management strategies that maximize bandwidth utilization and performance of the overarching network and its applications. Data center consolidation also goes hand-in-hand with application virtualization. The objective of application virtualization is to segregate applications from servers. Instead of running on a physical server with which it is co-located, an application executes on a virtual server which can reside anywhere in the enterprise, such as in the consolidated data center [7]. As a result, fewer physical servers are needed, because each is multi-tasked to handle many applications, each of which performs as if the server were dedicated to it. When properly planned and maintained, the adoption of shared services is transparent to the end users of the applications, yet delivers a more manage able quality of service. Automation solutions in the datacenter, for example, can restart failed applications, dynamically allocate new servers, conduct scheduled backups, and perform configuration management of the operating environment. Automation brings a number of advantages, including process consistency and enforcement of corporate rules and regulations, accelerated process execution, and minimization of human error. It also allows for more efficient adaptation to changing conditions, and it increases the productivity of the IT and operations teams whose manual input and support for the automated processes and systems is no longer required.

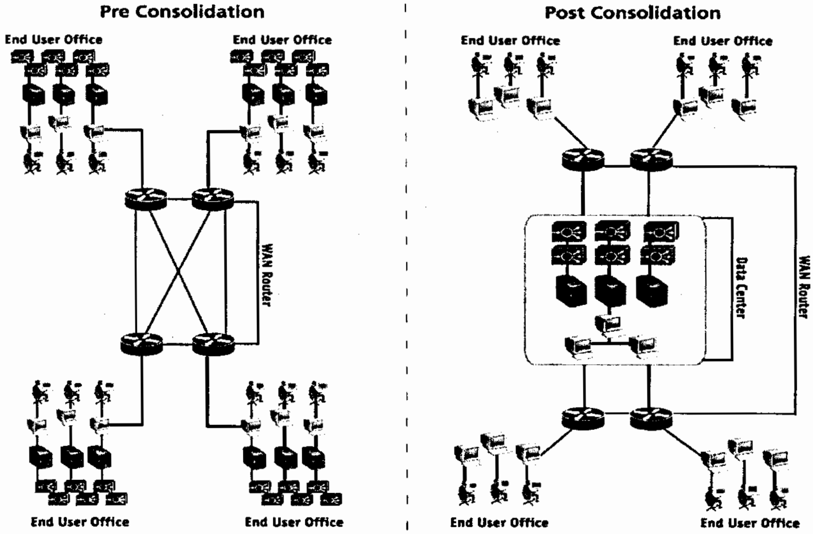


Fig. 2: Datacenter consolidation is necessary not only to simplify the infrastructure, but to optimize it so quality of service can be maintained and ultimately improved

Conclusion

Consolidating data in distributed heterogeneous systems is an important and challenging task. The existing approach to solve this problem is most suitable approach to the organization of federal databases. Creating and managing such a structure requires the use of specialized software, which in turn must meet several requirements for transparency, heterogeneity, security, performance, etc. In the market integration software there are a number of solutions from major manufacturers, based on industrial relational DBMS, you can use to organize a federal structure of data access. From a technical point of view, data integration has traditionally submitted to a centralized repository and tools Extract, Transform and Load (ETL). However, the main disadvantages this approach is the large overhead storage information and delays in receipt of information. With the increasing number consolidated sources deny overhead grows proportionally. This study shows that within distributed computing systems, huge amount of applicable federal approach does not require integration of all data in a single source. The author also explains why this approach is more flexible and provides the fastest way to connect new sources of data, which is particularly important in dynamic changing systems. Also this study describes a hybrid approach that combines benefits of the approaches to the repository and federated access to data. In this case, part of the data is replicated to a central database, and part of it is still stored in the original sources, depending on the type and strategy used. The leaders in this area are IBM and Informatics, providing comprehensive support for solving problems of data consolidation. Finally, it is worth noting that despite of rapid growth, it remains many unsolved problems that require thorough study and new solutions. The process of consolidating data in this phase of development technology requires a large amount of manual work overcome semantic structural discordant and setting performance. Therefore, in the near future efforts of companies, producing many of consolidating data will be used to increase the level of automation and self-management of their products.

References

[1] Alexander Bogdanov, Thurein Kyaw Lwin // System integration of heterogeneous complexes for scientific computing, based on the use of DB2 technology //Proceedings of International Conference «Computer Science & Information Technologies», 28 September - 2 October, 2009, Yerevan, Armenia, pp.397-399.

[2] Bogdanov A.V., Shuvalov A. // Консолидация данных в системах распределенных вычислений (бэта-версия)// Consolidating data in distributed computing systems.

[3] Database Trends and Applications, Solution for the information Project Team, Volume21, Number 5 //www.dbta.com

[4] Высокопроизводительные вычислительные алгоритмы (учебное пособие)/ А.В.Богданов, М.И.Павлова, Е.Н.Станкова, Л.С.Юденич. http://www.csa.ru/old/analitik/distant/q_start.html

[5] Consolidation in the datacenter, David Hornby, Global Sales Organization Ken Pepple, Enterprise Services Sun BluePrints™ OnLine—September.

[6] Ken Milberg Planning Your Data-Center Consolidation Strategies for a hassle-free deployment//www.ibmsystemsmag.com/aix/enewsletterexclusive/25075p1.aspx

[7] Ensuring the Success of Datacenter Consolidation over the Long Haul / Fluke white paper //www.flukenetworks.com /ensuring the success of datacenter Consolidation

ATLAS COMPUTING AT JINR

E. A. Boger, M. A. Demichev, A. G. Dolbilov, N. I. Gromova, Yu. P. Ivanov,
Yu. A. Nefedov, M. M. Shiyakova, A. S. Zhemchugov

Joint Institute for Nuclear Research, 141980, Dubna, Russia

The main goal of the ATLAS experiment[1] is to investigate various aspects of elementary particle properties in high-energy interactions of protons at the Large Hadron Collider (LHC), which was built at the European Laboratory for Particle Physics (CERN). The experiment started collision data taking in November 2009. Currently, it takes data at the record proton collision energy 7 TeV. Data flow from the detector at full accelerator luminosity is estimated to reach 3.5 Pb/year. The experiment's data processing includes event reconstruction, simulation and physics analysis.

Due to large amount of data to be processed, the ATLAS Computing Model [2] assumes that computing resources are distributed across multiple locations. The tier structure is implemented to combine distributed resources, with distinct roles of the various tiers. Joint Institute for Nuclear Research (JINR) participates Russian ATLAS Tier-2, in scope of the EGEE-RDIG consortium. According to the Tier-2 role, JINR computing facilities should provide analysis capacity for physics working groups to process 20% of the full sample of data (AOD) and computing resources for the Monte-Carlo simulation.

ATLAS computing infrastructure at JINR includes tools for central Monte-Carlo simulation data production, experimental data management and distributed data analysis. JINR site is integrated into the ATLAS Distributed Data Management System (DDM) since 2007. Recent production versions of the ATLAS software are installed centrally and available at the CE. Distributed analysis tools Ganga and Pathena are available for the ATLAS users at JINR. A dedicated queue ANALY_JINR has been created to run analysis jobs at the CE more effectively. To assure the robustness of the computing system, JINR participates in various functional tests of the data transfer system and stress-tests of the ATLAS analysis structure. Most significant tests, which gave an important hints to adjust parameters of the local computing farm were the combined computing challenge of all LHC experiments (STEP09) in June 2009, and the ATLAS user analysis test (UAT09) in October 2009.

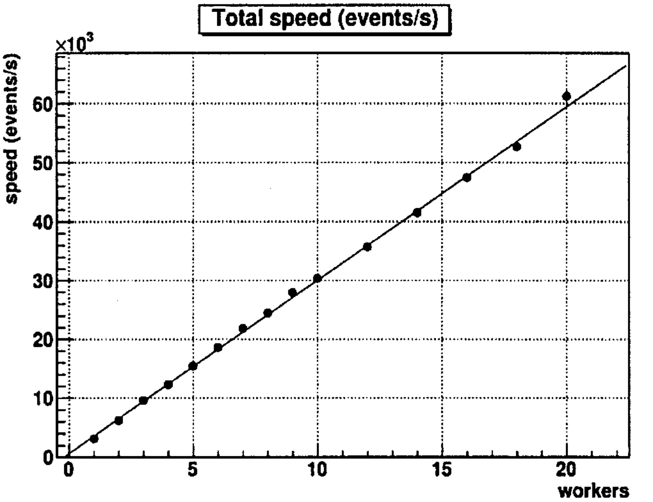


Fig. 1: Performance at the JINR LNP PROOF cluster: speed of data processing (simple analysis) versus number of workers activated

LHC startup in November 2010 verified crucially readiness of the JINR site to store and process ATLAS data. After the LHC startup till now more than 200k datasets have been transferred successfully to JINR site, with average throughput of 5 MB/s.

A new Tier-3 activity was started in 2010. The main goal is to setup a cluster optimized for the data analysis, and integrate it with the ATLAS computing system. A dedicated computing cluster at the Laboratory of Nuclear Problems of JINR is used as a Tier-3 testbed. Current implementation of the Tier-3 is based on the consistent use of parallel data storage system Xrootd[3] and parallel computing system PROOF [4]. Preliminary tests of the Tier-3 infrastructure demonstrate substantial increase of preformance during the physics data analysis (see Fig. 1).

User support is considered as an important task. Several tutorials on the use of ATLAS Grid infrastructure and data management tools have been organized in Dubna for physicists both from JINR and other Russian scientific centers.

References

[1] ATLAS Collaboration. The ATLAS experiment at the CERN Large Hadron Collider, JINST S08003, 2008.

[2] ATLAS Computing Technical Design Report, CERN-LHCC-2005-022, 2005.

[3] The Scalla/xrootd Team, The Scalla Software Suite: xrootd/cmsd, http://xrootd.slac.stanford.edu

[4] Brun R. and Rademakers F. ROOT — An object oriented data analysis framework// Proceedings of AIHENP'96 Workshop Lausanne, Sep. 1996, Nucl. Instr. and Meth. A 389 (1997) 81–86; see also http://root.cern.ch

D-GRID REFERENCE INSTALLATION

O. V. Dulov

Steinbuch Centre for Computing, Karlsruhe Institute of Technology
oleg.dulov@kit.edu

The D-Grid reference installation is a standard site for the German grid initiative. With respect to changing demands from the community, new versions of the reference installation are released every six months. The implementation of the ITIL recommendations in case of configuration, change and release management, and usage of virtualization technologies are extended topics, which are included into the project.

Introduction

The D-Grid reference installation [1] is conceptualized, designed, set up and administered at the Steinbuch Centre for Computing (SCC) Karlsruhe Institute of Technology [2] as a new service for German grid initiative [3]. D-Grid initiative is started from September 2005 is a bundle of twenty five Grid-oriented projects, funded by the German Federal Government with the purpose to build, operate and use a German-wide Grid infrastructure for Science and Economy. About 120 German institutions (Helmholtz Association, Max Planck Society, Fraunhofer Association, Universities) and some commercial partners are participating into the initiative. SCC leads the largest "D-Grid Integration Project"[4].

The reference installation is a part of the Integration Project and includes other D-Grid projects, providing an example of collaborative work inside the community. Purpose of the project is to demonstrate and document how to build the grid site by adopting the heterogeneous requirements from the community, and mapping them into a middleware stack. As the D-Grid requirements are changed, the Reference installation is implementing these changes, to serve as reference system and test platform for various grid middleware.

Generally, there is a composition of two parts, which construct the project: a *Mini-cluster* for software installation and a *technical documentation* for the system administrators and the D-Grid users. Cluster consists of hardware devices (physical and virtual) to install the grid-related software for tests. The system software maintenance sequence can be considered as follows: (1) install the software; (2) identify errors and requirements acceptance, (3) carry out changes according to tickets from helpdesk if necessary. These changes may entail new tests or even new installation.

Knowledge management and sharing – are the terms which can describe the core idea behind the reference installation. In the same time, the system administration tasks (as monitoring, or using the ITIL [5] recommendations and others) are also the focus of the project, and according to the current state, the semantic-based CMDB design and future implementation is the main focus of the interest for this project's activity. System's infrastructure was constructed by using the cfengine [6] system configuration tool and subversion [7] version control system.

Every support team or end user of the reference system can initiate changes by writing a ticket in the Grid user support portal [8]. When necessary, these will lead to changes in the system configuration, or in the technical documentation, or both. The current reference installation implementation does not include "standard" procedures or frameworks for testing. However, there is the Nagios [9] based initiative in D-Grid to work on the introduction of such a framework.

Infrastructure

In order to build the D-Grid Reference installation, the following infrastructure (see Fig. 1) to provide the control under the system was designed.

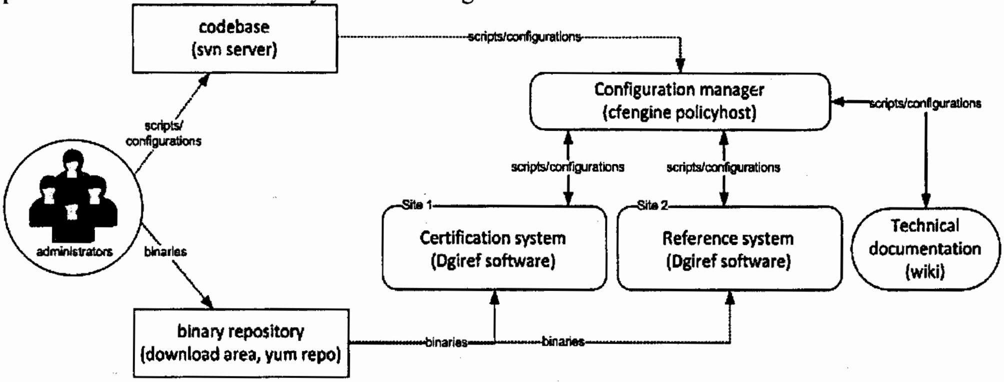


Fig. 1: D-Grid Reference installation infrastructure

There are two core ideas under this infrastructure:

(1) to represent the system administrator as the software developer, and

(2) try to control as much as possible by using only one tool.

Concerning to the system administrator's tasks execution, the following principle is used: all software binaries packages are located into one binary repository [10], and all scripts for execution or configuration are controlled by the Version control system and located into codebase repository. The binary repository has different types of software packages (rpm, tar, bin), and the simple index with search options. This is the implementation of the ITIL's Definitive Media Library (DML), which is used for the Release, Change and Configuration management.

The configuration platform is based on the cfengine configuration and change management software. The cfengine itself is a suite of programs for integrated autonomic management of either individual or networked computers and can be seen as cron-frontend, automated system administrator or even general tool for automatically making sure that promises are kept. There are some tasks which assigned to the configuration platform:

- update site configurations from codebase repository and apply them,
- manage files/directories rights, their content,
- distribute files, certificates, manage users,
- install, remove, update software, run executable scripts,
- manage services, processes, and some others.

The virtual machines management can be provided as the special task case for the configuration. Creation and configuration a new virtual machine and the virtualization platform itself can be done from the same set of configuration scripts and by using the same (central) configuration management tool. Changes themselves can be implemented automatically and is controlled under the Version control system. The D-Grid reference installation provides the prototype of these features also.

The special note is about the Documentation automation process. As will be shown later in this paper, every section for manual consists of two parts: description and script. Currently the script part – it is not exactly the executed script (it can be improved), but the commands, which the system administrator have to do (or another way around – the reference system maintainer have done). This content is coming automatically to the technical documentation website by using the MediaWiki API (Application Programming Interface), and plays a significant role into the error analysis and control.

Site architecture

The project includes two environments: *development* and *use*, which are originally not identical from the software or from the hardware perspective. Development environment is a set of virtual machines, based on the qemu emulation and Kernel Virtual Machines (KVM) technology – qemu-kvm and currently share the same hardware resources together with the use environment. The development environment consists of the servers, created on demand by the automated configuration management, based on cfengine version 3.

Thanks to the virtual environment, the Operating system change (for example from Redhat-based linux to the Debian-based linux) can be also applied, according to the supported by the virtualization technology. Machine parameters, such as 32 bits or 64 bits architecture, can be easily adapted, but it is not the case for the processor family (for example Intel Xeon, AMD Opteron, etc.), while it is predetermined by the underlying hardware.

Hardware

The D-Grid reference installation resources, provided by the community include the hardware cluster with twenty two machines in rack. All of them based on 64 bits architecture, two of them possess large hard drives and small memory, and have the disk RAID array to improve the reliability through redundancy. The rest have built in high-end memory and slender disk space.

Ten machines are used as the cluster Worker Nodes and ten for the grid middleware and local resource management and cluster configuration. Every machine use two network interfaces: one for external network configuration, another for internal.

The advantage of using the qemu-kvm virtualization technology can be implemented only if the machines provide the hardware support of it. The reference installation rack includes the AMD-V chipsets, which have support hardware-assisted virtualization.

Software

The Site architecture includes the following three blocks:

(1) Compute resource management,
(2) Storage resource management,
(3) Configuration management.

The Compute resource management block includes the grid middleware stack, resource manager system, computing cluster and the interactive node. Local Resource management is done by the Torque [11] and maui [12] advanced cluster scheduler. The Network File Server (NFS) host can extend the cluster architecture for files distribution between other components. The same host configuration as the worker node including to the interactive node.

The interactive node provides an entry point to the system for users to be able to manage their workload and is not a User Interface for Job submission. That allows the connection and access from the User Interface into the D-Grid reference installation site and to work with the grid applications and also to compile and test jobs using local libraries. This node is configured as a access point for the grid administrators and programmers.

A unique feature of D-Grid infrastructure is that all functional aspects are covered not by just one single middleware, but several middleware stacks are available. For the Compute resource management, the current stack includes: Unicore [13], gLite [14], and Globus toolkit [15] grid middleware. Job submission frontends manage the users' tasks (jobs) entering from user interface and assign them to the cluster for execution. The user interface (UI) provides the protocols for communication.

Storage resource management block consists of the Data management frontends that manage a potentially huge amount of storage space (in the magnitude of Petabytes) in grid environments. Additionally they may serve as recipient for data output from running jobs on linked clusters. In a similar way to the job submission front ends this type of middleware can also include an own user interface to access and retrieve stored data with appropriate tools for interaction. Currently, the following Data management middleware are into the reference installation architecture: dCache-SE [16] and OGSA-DAI as an independent component from the Globus toolkit.

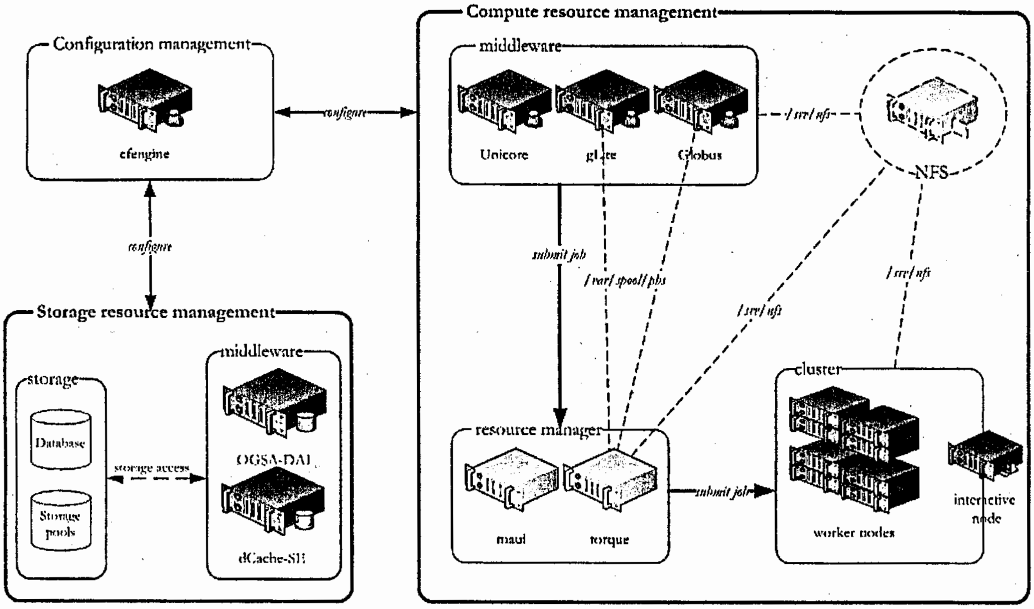


Fig. 2: D-Grid Reference installation Site Architecture

The Configuration management is provided by the cfengine version 3 and, as it was shown into the Infrastructure section, this is the special part of the site architecture. Actually, the cfengine configuration agents are located into all the hosts into the system, but one host is using so-called "master". This master host is responsible for the sharing data for non-master agents and have a connection into the codebase Version control repository.

The cfengine architecture internally supported that every agent can play a "master" role and the system can be configured for more than one "master". There are some additional benefits to use such a configuration system. One of such benefit can be an organization of the system administration work and separation tasks into two groups: system administrator tasks and system architect or service construction tasks. Both of them can be done by the same system administrator or (better) separated between two (groups). The tasks to create programming scripts can be done by the system administrators; the cfengine configuration scripts development can be done the system architect.

Another benefit is the possibility to collect the host information in form of the host reports. This feature can be done by using the commercial version of the cfengine, or can be implemented only with the open source version from cfengine community with some additional development effort. This collection of information can be published and represent the ITIL idea of the Configuration Management DataBase (CMDB), precisely – semantic-based CMDB.

Technical Documentation

The technical documentation is the end product of the reference installation. It is based on the MediaWiki [17] open source wiki engine and includes difference types of information, oriented to the community resource providers and their users. There are different types of the documents into the reference installation wiki. One type is the D-Grid user-oriented guidelines, which includes the valuable date concerning the certification, Virtual Organizations information, D-Grid access procedures, information concerning the gridmap file, and some others resources.

Another type of the documentation is the manuals and includes the system administrator-oriented documents. Every manual document has a clear defined *sectioned* structure. The content separation into the "pure technical content", like bash scripts with comments; and "common sense content", which includes configuration notes, links to other content, common words, descriptions and so on for every manual's section.

Every page in Wiki has own namespace, which helps to identify the content of this page. By using the MediaWiki Categories, the content is grouped into the Release documentation with the possibility to generate the pdf document, thus, constructing the Release manual, including all Release-related information into one file.

Release cycle

The reference system is build by using a release cycle, during six month each. New release starts with discussion, where all D-Grid resource providers are invited to contribute via e-mail. After the requirements are constrained, the beta release phase is initiated. While the required software packages are ready, they are installed during to the release phase (see Fig. 3.).

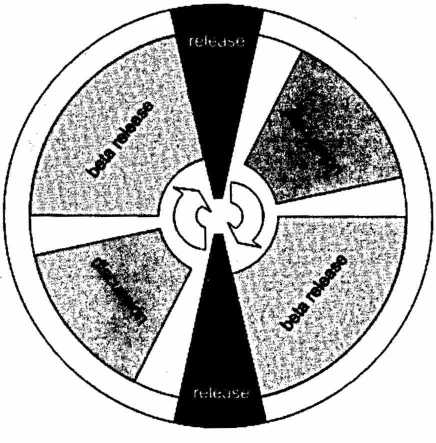


Fig. 3: D-Grid Reference installation release cycle

Throughout the whole process the system maintainer of the reference installation is moderating discussions or performing actions on hard- and software respectively. During discussion phase the D-Grid community is invited to consult the software and hardware components for the upcoming new release. The main conversation is based on exchange of e-mails via a dedicated mailing list; important statements are collected in the D-Grid reference installation's Wiki.

After the discussion is closed, all suggestions are summarized in a wish list and submitted to the D-Grid Technical Advisory Board (TAB) [18]. The TAB members represent the D-Grid resource provider centers and make a decision which part of requirements will be in- or excluded. The reference system maintainer can request the TAB about the requirements changes during the beta release.

Beta phase workflow includes the software packages request from the Reference installation support-centers with the appropriate documentation to follow. Support Centers are coordinated by the D-Grid Support Coordination workgroup (SuKo) to provide better communication quality inside the community. Current list of Support Centers includes:

- Research Center Jülich (Unicore middleware);
- Research Center München (Globus Toolkit middleware);
- Karlsruhe Institute of Technology (gLite, dCache, ogsadai middleware).

The beta-release phase includes the installation of the new version for the reference installation on a pre-production system and performing checks for conflicts between different components. This is done by the maintainer of the reference system with support by the appropriate middleware supporters. It is still possible to change the version for a software package in case of issues that turn out irresolvable otherwise.

When release phase has started the beta-release installation is repeated on the reference system. From that point onward, versions and software changes are not possible anymore (except for fixing critical bugs). After final migration tests and operability checks the technical documentation is published on the reference installation's Wiki and the D-Grid resource providers are demanded to update their systems.

Proceeding

In the scheduled time, release software packages with their technical documentation are received by the system maintainer and the installation into the development environment is started. The working procedure is the following: during the beta phase, the support team (or system maintainer) is written the descriptions for the appropriate software package. The reference installation system maintainer (or support team) create the bash scripts, based on the descriptions to automate the installation and configuration procedures.

The package can be excluded from release if there is a serious problem to maintain it into the system. The results of the beta-phase are:

a) bash scripts used to install and document the system are created;

b) reference system is installed into the development environment.

The bash scripts from beta-phase are the part of the technical documentation. There is the procedure which uses the MediaWiki API to copy this to the wiki-based technical documentation system. Such a procedure avoids the data duplication or some misunderstandings between installation procedure itself and their description.

During the release phase, the system maintainer makes a migration from certification into the reference system. This can be done manually (by using written bash scripts during beta-phase) or automatically (for example by cfengine). The technical documentation is updated with some important data and published in the end of this phase with the option to generate one pdf file with release-related information.

Summary

The D-Grid reference installation is used for different purposes. The most important is to test new grid components (or their new versions) and detect if there are compatibility issues or inconsistencies. The technical documentation and related software repositories are available online and can be used to maintain the grid-site by using provided resources. Additionally the system can be used for reconstruction of errors in a well-defined environment.

At the moment, more than 30 Virtual Organizations (VO) are supported by the reference installation. Any member of these VOs can access the system by using their user interface and check out the setup. However, the main target group for the reference system consists of the software developers of VOs and system administrators of the resource providers.

References

[1] D-Grid Reference installation: http://dgiref.d-grid.de/
[2] Official KIT (SCC) website: http://www.scc.kit.edu/
[3] Official D-Grid website: http://www.d-grid.de/
[4] D-Grid Integration Project: http://www.d-grid.de/index.php?id=539&L=1
[5] ITIL framework: http://www.itil-officialsite.com/home/home.asp
[6] Configuration management system: http://www.cfengine.org/
[7] Version control system: http://subversion.tigris.org/
[8] D-Grid User support Portal: https://helpdesk.ngi-de.eu/
[9] Open source monitoring platform: http://www.nagios.org/
[10] D-Grid Reference installation binary repository: http://dgiref.d-grid.de/AutoIndex/index.php
[11] Torque LRMS: http://www.clusterresources.com/products/torque-resource-manager.php
[12] Maui cluster scheduler: http://www.clusterresources.com/products/maui-cluster-scheduler.php
[13] Unicore grid middleware: http://www.unicore.eu/
[14] Glite grid middleware: http://glite.web.cern.ch/glite/
[15] Globus toolkit grid middleware: http://www.globus.org/
[16] Data management middleware dCache: http://www.dcache.org
[17] Open Source Wiki system: http://www.mediawiki.org
[18] D-Grid TAB: http://www.d-grid.de/index.php?id=464&L=1

SOME EXPERIENCE IN RUNNING MOLECULAR DYNAMICS SIMULATION APPLICATION IN GRID

E. B. Dushanov[1], Kh. T. Kholmurodov[2], N. Kutovskiy[1,3]

[1]*Laboratory of information technologies, Joint Institute for Nuclear Research, 141980, Dubna, Russia*
[2]*Laboratory of radiation biology, Joint Institute for Nuclear Research, 141980, Dubna, Russia*
[3]*National Scientific and Educational Centre of Particle and High Energy Physics of the Belarusian State University, 220040, Minsk, Belarus*

Introduction

The modern trends in IT are directed to the integration of different kind of resources of the organizations including high-performance clusters and storages into joint global infrastructure with help of so called grid middleware. Such approach provides the access to more computational resources for users. It increases an efficiency of its utilization because of more uniform load and allows to improve a reliability of data storage by means of making a replicas on distributed storages.

The resources of g rid infrastructure are used by so call Virtual Organizations (VOs). Each VO has its own certain aims. Users have to belong to one of VOs in order to access the grid resources. The membership in VO requires some formal procedures to be done as well as user's activities must correspond to VO aims. Often it can be inadvisable for users to perform all required steps to become VO member in order to carry out the ability to run his application in grid.

Molecular dynamics method

The molecular dynamics method (MD) was initially developed in theoretical physics and found the large extension in the material science as well as starting from 1970s years in study of biochemistry and biophysics of DNA and protein structure. The MD methods are playing the important role in the determination of the proteins structure and specification of its properties. The interactions between molecular objects can be described by a force field of classical molecular dynamics, quantum chemistry models or by a hybrid approach which include the elements of various theories (for example, quantum mechanics/molecular mechanics – QM/MM and etc.) [1-3].

The most popular packages for MD simulation are DL_POLY, AMBER, CHARMM (commercial version of CHARMm), GROMACS, GROMOS and NAMD. These packages are the programs for multipurpose use designed for wide range of problems from elementary and binary system to complex macromolecular and biological proteins molecules.

The efficient utilization of most of these packages requires a modern powerful computational resources with parallel job submission support. MD calculations with use and optimization of DL_POLY application were successfully performed on the JINR Central Information and Computer Complex – CICC (the results can be found at [4]). As a rule the examination of complex molecular systems (one of it is shown at Fig. 1) requires large computational resources what can be a problem for single organization to provide them. One of the possible solution for insufficient resources problem is to use grid resources.

Running DL_POLY in grid

The exploration of the possibility to run DL_POLY* application in grid was initially done on training, probing and testing grid infrastructure in the Laboratory of Information Technologies [5]. One of the purpose of that infrastructure is to provide for users and developers the possibility to explore and debug different applications in grid environment. All formal procedures such as the users registration on User Interface and in VO, the issue of the user certificates are reduced to minimum.

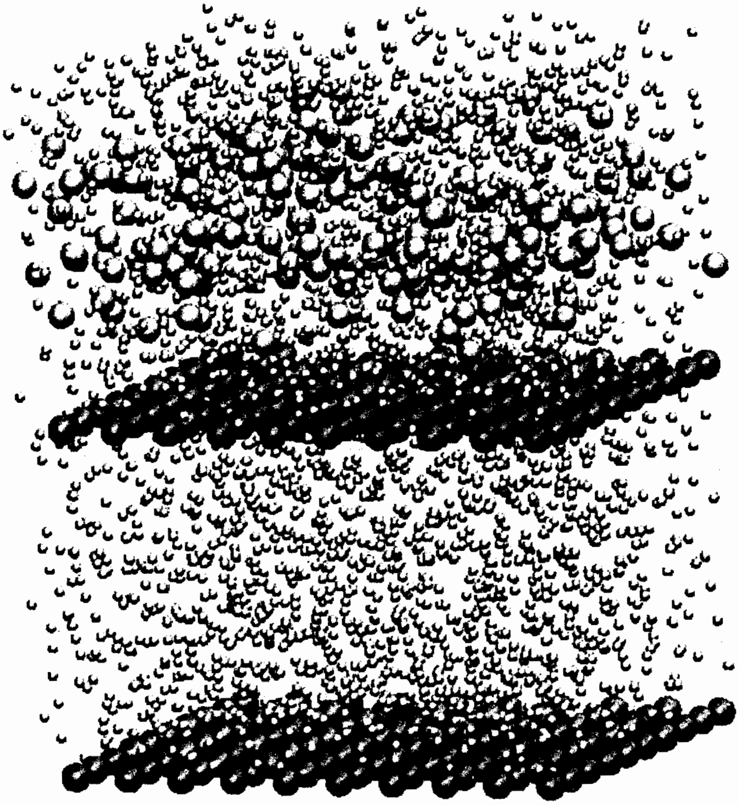


Fig. 1. Membrane with two changes, 202 and 2746 molecules of solution and solvents, respectively

During that phase the file with job description serving like a source of such parameters as the name of the executable file, the number of worker nodes, MPI type, input data location etc in order to run job on the cluster as well as intermediate scripts (they invoke the compilation of application, the extraction of input data files, setting environment variables) were written and debugged.

The result of that stage was a successful run DL_POLY package in grid environment (two MPI implementations were tested: MPICH2 and OpenMPI.

* DL_POLY is a general purpose serial and parallel molecular dynamics simulation package developed at Daresbury Laboratory by W. Smith, T.R. Forester and I.T. Todorov. (http://www.cse.scitech.ac.uk/ccg/software/DL_POLY/)

The next step was to create dedicated VO for molecular dynamical simulation tasks and arrange some resources in one of the global production grid infrastructures. The most corresponding one was the national nanotechnology network – GridNNN*. The VO moldyn (short from ***mol**ecular* ***dyn**amics*) was created. DL_POLY was successfully run on the parallel cluster of the GridNNN resource center (RC) – Russian Research Center "Kurchatov Institute".There is an ongoing process in signing Service-level agreements (SLA) with few others RCs of GridNNN infrastructure.

Conclusion

One of the package for MD simulation – DL_POLY was successfully adopted to run in grid environment (initially it was done on the probing grid-infrastructure and after that on the production one – GridNNN). The gained experience allows to state that spent efforts for porting and running application in grid is incomparable with the time and quality results advantage what are provided by the use of such modern IT approaches as grid technologies.

References

[1] Allen M.P., Rapaport D.J.D. C. The Art of Molecular Dynamics Simulation, 1996.

[2] Todorov I.T., Smith W. and Trachenko K. DL_POLY_3: new dimensions in molecular dynamics simulations via massive parallelism// J. Mater. Chem., 2006. V.16, P.1911.

[3] Kh.T. Kholmurodov (Ed.). Molecular Simulation in Material and Biological Research, 2009. Nova Science Publishers Ltd. (N.Y.).

[4] Korenkov V.V., Mitsyn V.V., Dushanov E.B., Ayriyan A.S., Lutsenko A.I. Testing the new JINR CICC supercomputing cluster, LIT Scientific Report, 2007. P. 43-46.

[5] Korenkov V.V., Kutovskiy N.A. Educational grid-infrastructure, Open systems// Open systems, №10 (156), 2009. P. 48-51 (in Russian).

* That project aims to integrate small and medium supercomputers into a unified distributed computing environment (http://ngrid.ru).

FROM OPEN ACCESS ARCHIVES TO INTEGRATED DIGITAL LIBRARIES

I. A. Filozova, G. Musulmanbekov

Laboratory of Information Technologies,
Joint Institute for Nuclear Research, 141980, Dubna, Russia

1. Introduction

Now growing number of the academic institutes all over the world create own repositories, accumulating and arranging them in the form of Open Access (OA) for the world community. What caused the Open Access movement? There were two faced problems. The first part is the ***affordability problem***; subscription fees have by far outgrown inflation rate the previous two decades. The second part of the crisis consists of the ***accessibility problem***; subscriptions are bundled in databases controlled by large commercial agencies, like Elsevier and others. These gives very restrictive access to who can access the database and when. Transition to OA to the scientific literature is defined by Open Access Initiative (OAI) principles: digital access, free for any user, decrease in restrictions on license access and copyrights [1]. These basic principles of OA are declared in the documents of Budapest Declaration Open Access Initiative and Berlin Declaration on Open Access to Knowledge in the Sciences and Humanities.

In this paper we trace the development of OA repositories starting from OA archive 'arXiv.org' to OAI-compliant or Digital Libraries (DL) and implementation of DL into GRID infrastructure. As example of DL, the OAI-compliant repository of Joint Institute for Nuclear Research is described. Furthermore, trends of development of OA repositories into the Open Information Space and Open Science are discussed.

2. Evolution of OA archives – OAI-compliant repositories

Open Access to Scientific Research – a way to make scientific results available to all scientific community by the Internet, so that any person can get access to product from any place and at any time at an own choice. OA does not cancel the copyright and does not contradict it. The personal non-property author rights are not alienated and remain with him irrespective of a publication way. The benefits of Open Access:

Scientists and Researchers:

- expansion readership and increasing readability;
- increasing publication citation;
- scientific impact;
- grouth of the author popularity and fastening of a scientific priority.

Organization:

- management of their digital resources;
- increase of the scientific prestige.

Society:

- return on investment in research;
- removing barriers to information sharing;
- creation of additional information services for different users categories.

OA is realized by two ways: publication in open access journals; depositing documents into public scientific archives. The first OA archive (which saw daylight in 1991) is arXiv subject repository. Originally arXiv contained high energy physics (HEP) related e-prints merely, but now the repository has grown to become the reference repository for several other fields, amongst the major ones mathematics and computer science. Today the repository carries more than 530 000 e-prints. Their usage statistics shows a steady rise in monthly submissions since its birth, the number now being more than 5000 monthly submission. It is mostly a plain repository, which has got little emphasis on sophisticated search mechanisms or offerings of bibliographic information. Next step is build up of OAI-compliant repositories (institutional and subject). OAI-compliant means that the article metadata (the title, authors, keywords etc) are created in the format laid down by the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH). Search engines can then harvest the metadata from all archives making their metadata visible in this form, and present it to users in an appropriate way. In comparison with the OA archives, like arXiv.org, OA repositories hold the functionalities of Digital Libraries (DL): library management services such as a user-friendly interface, powerful search functions and collaborative features. However, many efforts in development DLs using several conception of what a Digital Library is, produced very many heterogeneous entities and systems, thus rendering the interoperability, reuse, sharing, and cooperative development of digital libraries extremely difficult. Recently authors of the paper [2] offered a new vision of this conception by presenting it as populated by three different "systems" and by carefully establishing the roles of these constituent "systems" and the relationships among them. The three systems are the Digital Library (DL), the Digital Library System (DLS), and the Digital Library Management System (DLMS), as depicted in Fig. 1. In particular, the DL is the abstract system as perceived by end-users, the DLS emerges as the software system providing all the functionality that is required by a particular DL, while the DLMS is the software system that provides the infrastructure for producing and administering a DLS and integrates additional software offering extra functionality.

In the next section, as example of OAI-compliant repository implementing functionality of DL, the repository of Joint Institute for Nuclear Research, JDS, is described.

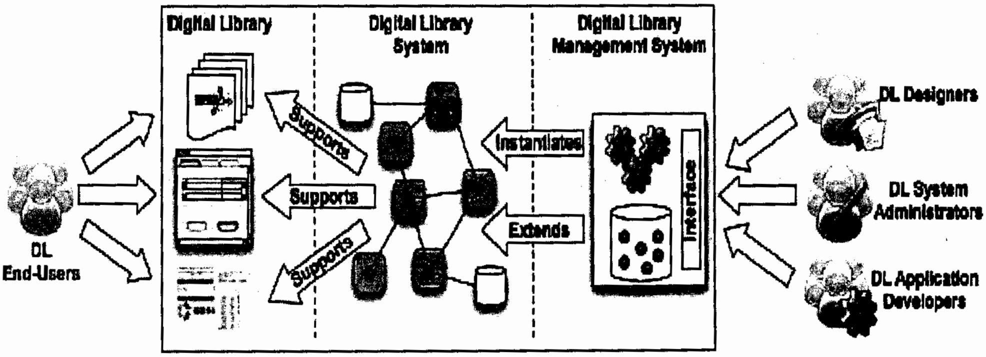


Fig. 1: Building blocks of a Digital Library

3. JINR repository

Building the institutional repository has as its objects to make accessible the scientific and technical results of JINR researchers for the international scientific community, to increase the efficiency of using information resources of JINR Publishing Department and Scientific and Technology Library, to increase the level of informational support of JINR employees by granting an access to other scientific OA archives and to estimate the efficiency of scientific activity of JINR employees. To realize these goals the JINR OAI-compliant repository, JINR Document Server (JDS) [3, 4], is being built. JDS is implemented as an institutional repository, the content of which will be composed of the following objects:

1. The technological and scientific-related documents of the following types:

- Publications issued in co-authorship with JINR researchers;
- Archive documents that describe all the essential stages of the JINR research activity;

2. Documents providing informational support for scientific and technological research performed in JINR.

The main JDS goals are:

- To make accessible the scientific and research results of JINR employees for the international scientific community;
- To increase the efficiency of the using of JINR Publishing Department and Science and Technology Library informational resources;
- To increase the informational support level of JINR employees by granting an access to other scientific OA archives;
- To estimate the efficiency of scientific activity of JINR employees.

The JDS requirements to DLS and DLSM are:

- Possibility to harvest and upload documents from other archives;
- Subject classification;
- Wide set of management services for authors and users;
- Web-interface;
- Access rights differentiation;
- Integration with the international catalogues, registers;
- Integration with JINR internal information resources;
- Multilingual interface support;
- Metadata support;
- Possibility of any format files loading;
- Possibility of the open reviewing and discussion for the all interested users even before article acceptance in reviewed journal;

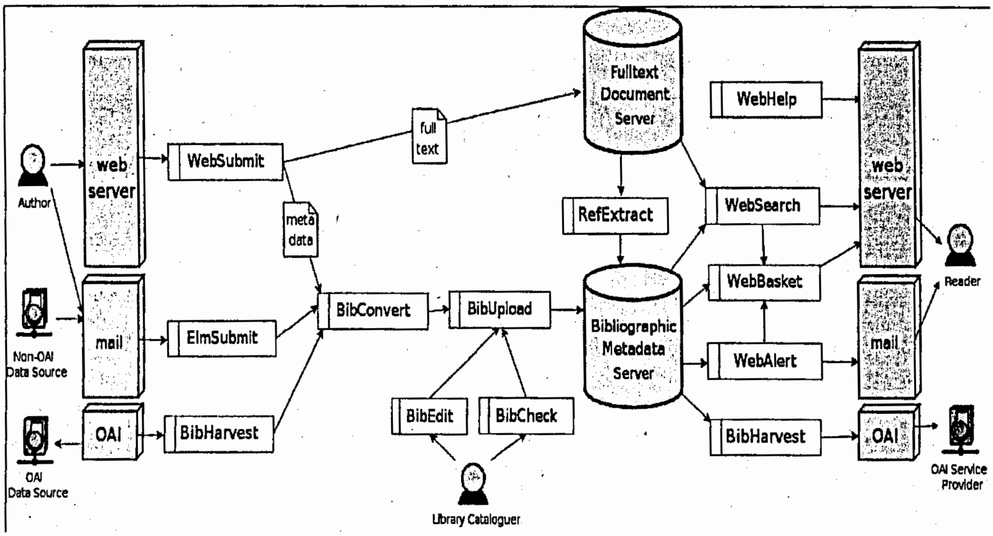


Fig. 2: CDS Invenio architecture

On the market there are many systems designed to create and manage DLs, both commercial and free: EPrints, DSpace, Bepress, OPUS, CDS Invenio, etc. According to statistics given by the Registry of Open Access Repositories (ROAR) a leadership in this area take two free packages EPrints and DSpace. The analysis, performed by authors, showed that the studied systems almost are equal to the opportunities provided by, and fully satisfy the criteria put forward above. International status of JINR, close cooperation between JINR and CERN, have defined a choice in favor of CDS Invenio. CDS Invenio, the integrated digital library system, including both DLS and DLSM, is a suite of applications which provides the framework and tools for building and managing an autonomous digital library server [5]. The software is readily available to anyone, as it is a free software, licensed under the GNU General Public License (GPL). The technology offered by the software covers all aspects of digital library management (Fig. 2). It complies with the Open Archives Initiative metadata harvesting protocol (OAI-PMH) and uses MARC21 as its underlying bibliographic standard. Its flexibility and performance make it a comprehensive solution for the management of document repositories of moderate to large size.

4. Integration of Digital Libraries with GRID technology

The way in which what is commonly named "Digital Libraries", DLs, is perceived has evolved a lot over the last 15 years. This is also true for the expectations on DLs. DLs are now moving far beyond any connotation of the term "library" and are rapidly evolving to become general systems dedicated to cover the whole spectrum of the knowledge management task. DLs are now envisioned as systems that are at the centre of any intellectual activity and have no logical, conceptual, physical, temporal, or personal barriers on information. In particular, they are shifting from *content-centric* systems in charge of simply organizing and providing access to particular collections of data to *person-centric* systems aiming at providing facilities for communication, collaboration and any kind of interaction among scientists, researchers, and the general audience interested in topics of pertinence to the knowledge the DL has been set up for [6].

This vision poses new requirements on DL systems. They must deal with many forms of data ranging from digital counterparts of traditional documents, such as PDF files, to complex and multimedia objects combining text, images, audio and/or videos, sensor data, structured and semi-structured data residing in various kinds of information sources. They must act as an integrated working environment providing a seamless and personalized access to a variety of information sources as well as to all the facilities and instruments deemed as relevant for fulfilling the user requirements. These facilities may range from "standard" Digital Library facilities, like search and browse, to co-operative working tools and community specific services that require high computing and storage capabilities. The new DL systems must also be able to easily adapt themselves to the evolving user requirements both in terms of the data the user has access to and the processes the user is entitled to perform, i.e., they must be able to appropriately combine data and services in user specific processes.

Through the handling of variety of information sources digital libraries have the potential to revolutionize the way in which joint research is conducted. In order to make this possible, digital library systems based on new technologies must be introduced. Research work today is often a collaborative effort carried out by groups of individuals belonging to different organizations remotely spread worldwide. Motivated by a common goal and funding opportunities, these groups dynamically aggregate into virtual organizations (VOs) that share their resources e.g. knowledge, experimental results, instruments, etc., for the duration of their collaboration, creating a rich and powerful research environment. Now it is necessary to create an advanced knowledge infrastructure that will serve the needs of dynamic virtual organizations. This infrastructure will allow members of VOs to access shared knowledge, services and computational resources in a secure, coordinated, and cost-effective way. It can be realized by integrating Grid and Digital Library (DL) technologies. The merging of these two different technologies will result in an innovative level of functionality providing the foundations for next generation collaboration environments able to serve many different research and industrial applications.

In particular, the Grid framework will provide the context for implementing the notion of virtual digital libraries (VDLs), i.e. transient, on-demand DLs based on shared computational, multimedia and resources, both content and applications resources. This combined technology will maintain a network of existing resources on the Grid. A virtual organization will be enabled to dynamically create and modify its own DL by specifying a number of requirements on the information space (e.g. publishing institutions, content domain, document type, level of replication) and on the services (e.g. service type, configuration, lifetime, availability, response time). A reliable and secure virtual DL that satisfies the given requirements will be transparently instantiated and made accessible to authorized users through a portal. Many virtual DLs, serving different user communities, will be active on the same shared Grid resources at the same time. The composition of a DL will be dynamic since it will depend on the currently available and registered DL resources and on many other quality parameters such as usage workload, connectivity, etc. This development model will make it possible to avoid heavy investments, long delays and radical changes in the organizations setting up these applications, thus fostering a broader use of DLs as means for communication and collaboration. Integration of DLs with GRID will exploit the results of the EGEE (Enabling Grids for E-science in Europe (http://public.eu-egee.org/) project, which will deliver a Grid production infrastructure shared by a very large number of European organizations. This is one of aims DILIGENT project (http://www.diligentproject.org). DILIGENT will enrich this infrastructure with the necessary features for creating and handling an open and 'on-the-fly' modifiable set of services as required by the DL application framework. Further evolution of Virtual Digital Libraries are *Virtual Research Environments* .These environments provide scientists with collaborative and customized environments supporting results production and exchange around the globe in cost-efficient manner. This program is continuation of DILIGENT and is realized by projects D4Science and D4ScienceII.

References

[1] Open Archives Initiative. http://www.openarchives.org.

[2] Leonardo Candela, Donatella Castelli, and Pasquale Pagano, in Digital Libraries: Research and Development, Ed. Constantino Thanos, Springer, 2007, p. 22.

[3] Borisovski V.F. et al. On Open Access Archive for publications of JINR staff members. Proc. XI National Russian Research Conference “Digital Libraries: Advanced Methods and Technologies” (RCDL’2009), Petrozavodsk, Russia, Sept.17-21, 2009. KarNC RAN, 2009. P. 451-457. (in Russian).

[4] Filozova I.A., Korenkov V.V., Musulmanbekov G. Towards open access publishing at JINR // International Symposium on Nuclear Electronics & Computing (22; 2009; Varna), XXII International Symposium on Nuclear Electronics & Computing (NEC'2009), Varna, Bulgaria, Sept. 7-14, 2009.

[5] CDSware Overview, http://cdsware.cern.ch/invenio/index.html.

[6] Candela L. et al., Int. J. Digit. Libr. 7, 2007. P. 59.

RAVEN - A RANDOM ACCESS, VISUALIZATION AND EXPLORATION NETWORK FOR THE ANALYSIS OF PETABYTE SIZED DATA SETS

N. Gagunashvili[1], H. K. Gudmundsson, N. Whitehead
University of Akureyri, Akureyri, Iceland
nikolai@unak.is, hkg@unak.is, nicolaw@unak.is

M. Britsch, M. Schmelling
Max-Planck-Institute for Nuclear Physics, Heidelberg, Germany
markward@mpi-hd.mpg.de, michael.schmelling@mpi-hd.mpg.de

H. Neukirchen
University of Iceland, Reykjavik, Iceland
helmut@hi.is

The analysis and visualization of the LHC data is a good example of human interaction with petabytes of inhomogeneous data. A proposal is presented, addressing both physics analysis and information technology, to develop a novel distributed analysis infrastructure which is scalable to allow real time random access to and interaction with peta-bytes of data. The proposed hardware basis is a network of intelligent "CSR"-units, which combine Computing, data Storage and Routing functionalities. At the software level the project would develop efficient protocols for broadcasting information, data distribution and information collection upon such a network, together with a middleware layer for data processing, client applications for data visualization and an interface for the management of the system.

1 Introduction

During the construction of the CERN Large Hadron Collider (LHC) [1] it was realized that the analysis of the data produced by the LHC experiments requires a computing infrastructure which goes far beyond the capabilities of a single computing center, and which since then has been built up in the framework of the Worldwide LHC Computing Grid [3, 4].

Despite the fact that many new concepts regarding data distribution and sharing of computing load have been implemented, the computing models for the analysis of the LHC data are still very close to the approach by earlier generation particle physics experiments. They focus on filtering the huge initial data sets to small samples which are relevant for particular physics question, which then are handled locally by the physicist doing the analysis (see e.g. [5]).

While making efficient use of limited resources, this scheme has some obvious shortcomings.

- At a given time direct access is possible to only a small fraction of the total event sample. This reduced sample also has to serve to define and check the selection criteria for the selection jobs. As a consequence the selection may be biased or inefficient.
- The time constant for full access to the data is given by the frequency of the selection runs which go through the complete data set. Programming errors or missed deadlines for code submission can easily result in months of delay for the affected analyses.
- High statistics measurements, i.e. analysis which use information from more than a small fraction of all events, are not feasible. The same holds for finding exceptional rare events which are not caught by selection criteria based on prior expectations.

What is needed is a framework which allows random access on petabyte-size datasets. It should have a scalable architecture which allows to go to real time information retrieval from the entire data set. The initial use case of this infrastructure will be faster and more efficient access to the data. Beyond that, however, also novel ways of interacting with the data and new ways of data visualization will evolve.

2 Requirements

In particle physics the basic units which make up a data set are so-called "events". At the LHC this is the information recorded from a single bunch crossing of the two proton beams. At LHCb [2], for example, with a typical size of 50 kB per event and 2×10^{10} events recorded per year, the annual data volume amounts to O(1) PB. The analysis of the data is conceptually simple in the sense that all events are equivalent, i.e. at the event-level it parallelizes trivially. Also the information content of a single event has a relatively simple structure, consisting of lists of instances of a few basic elements such as "tracks", "calorimeter clusters" or "vertices" which contain the measured information about the final state particles created in a high energy collision. Different events will differ in the number of those objects and the relations between them.

The basic mode of data analysis in particle physics is characterized by two steps. In the first step the data set is scrutinized for events containing a specific signature. Events with this signature then are analyzed in detail, either by extracting some characteristic information or by iterating the selection process with additional criteria.

It is evident that depending on the selection criteria the size of the event sample used in a specific analysis can vary by many orders of magnitude. On the other hand, the maximum communication bandwidth available to return information back to the user will essentially by fixed, i.e. the interaction between user and the full data set must be such that the network traffic stays below a certain limit.

The quantities of interest in a typical particle physics analysis are probabilities or probability density functions for a certain process or configuration to occur. Numerical estimates are obtained by means of histograms, i.e. simple counters for how often a certain condition is observed. The analysis framework thus must be able to handle this kind of cumulative information, which even for very large event samples reduces to a limited set of numbers.

In addition to cumulative information from many or even all events, the system must be able to transmit some or all information from a few selected events. This is of particular relevance for very rare types for final states, such as for example events with a candidate Higgs decay or other exotic processes and which require an in depth analysis of single events.

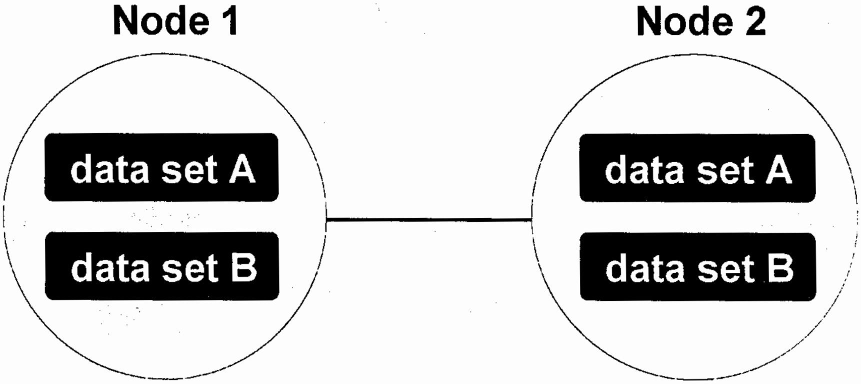


Fig. 1: Simple of example of redundant data storage on two nodes. If both nodes are present, the analysis starts in parallel on different subsets. Node 1 will start on data set A, node 2 will start with B. If one node is unavailable, either because it's down or busy with another task, then the other node will process the entire data set

The combination of the two access modes becomes particularly relevant in the context of interactive searches for special event types starting from the full data set. Here powerful visualization tools and user interfaces are required, which provide an intuitive representation of the properties of the event set, together with the possibility of interactive select-and-zoom schemes to focus on certain candidates.

3 Design Aspects

The requirements outlined above suggest a design similar to that of a biological brain: a dense network of many "simple" nodes combining data storage, processing and the routing of information flow. For the use case of particle physics, each node would store a small fraction of the total event sample, have the possibility to run an analysis task on those events and route information back to the user having submitted the analysis query. In the following these nodes will be referred to as Computing-Storage-Routing (CSR) units, which at the hardware level are standard commodity CPUs. With an appropriate middleware-layer a network of such CSR-units will then constitute a RAVEN system.

One important aspect of RAVEN is redundant and encrypted data storage. While encryption should simply ensure the confidentiality of the data also in case that public computing resources are used, redundant storage assures that the entire data set can still be processed even if some nodes becomes unavailable. A simple sketch how duplication of data between two nodes can serve these purposes is shown in Fig. 1.

For a particular analysis or visualization task, instructions would be broadcast to all CPUs. These instructions will then be executed on the local event samples, and the information retrieved from those events routed back to the user.

As discussed before, with respect to the information that is returned one has to distinguish between cumulative data, and per-event data. Since all data have to go back to a single node, per-event data should either be of only limited volume per event or should be transmitted for only a subset of all events. Cumulative data on the other hand, such as histograms, flowing back through the network can be accumulated on-the-fly such that the total amount of information transmitted over the network stays comparatively small, even for very large event samples. Figure 2 illustrates the case.

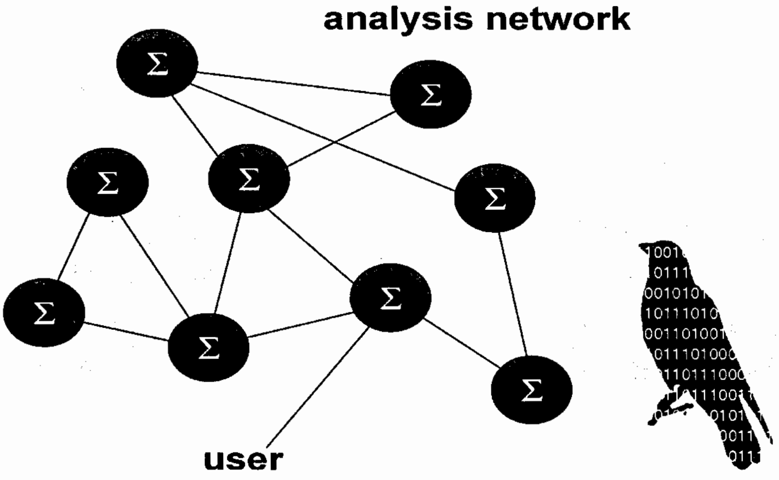


Fig. 2: Sketch of a RAVEN network. Histogram data, for example, produced by the analysis jobs on the different CSR-units are routed back to the node connected to the user, and updated on-the-fly on the way back. The links show which nodes are aware of their neighbors, i.e. the network topology routing and data distribution have to deal with

4 Implementation Aspects

A central feature of the design of a RAVEN system is its scalability, which almost automatically comes from the fact the different events are independent and thus can be spread over as many CSR-units as are available. Scalability allows to develop RAVEN on a small test system and later expand the working system to the size required for a particular application, possibly also taking advantage of cloud-computing infrastructures.

A particular implementation dealing with 1 PB of data spread over 104 CSR-units would correspond to 100 GB per node. Assuming a processing speed of 100 MB/s, which seems possible today, the data set could be processed within a quarter of an hour. A test system should typically have one percent of the capacity of the 1 PB system.

One problem that has to be addressed for RAVEN is the creation of ad-hoc routing and communication topologies for a given analysis query, which is used both to distribute the query to all nodes and to collect the results of the analysis. Here one challenge also arises from the fact that arbitrary next-neighbor topologies must be allowed. Furthermore, since many analyses will only access subsets of the full data set, the system should be able to process multiple queries simultaneously.

Another issue is the distribution of data, analysis code and actual query of a specific analysis. The biggest challenge is the distribution of the full data set. Here different data items have to go to different nodes, which in view of the total data volume that has to be distributed is a highly non-trivial task. The data distribution scheme also should take care of the redundant storage scheme. Finally it would be desirable that a RAVEN system automatically detects new CSR-units joining the system and migrates part of the data to the new resources.

While the distribution of the full data set will happen only rarely, updates of the analysis code will be more frequent, though still rare compared to analysis queries. The latter two can be distributed via a broadcast mechanism. The splitting into analysis code and query is motivated by the goal to minimize the network traffic. Instead of distributing the full analysis code, which for a typical LHC experiments amounts to O(1) GB, with each query, a middleware approach is foreseen. Here the (in general machine dependent) analysis code forms a software layer on top of the operating system, which provides a machine independent high level language to perform the actual physics analysis.

While the mapping of the classical analysis models based on histograms or n-tuples on a RAVEN infrastructure is relatively straightforward, the system calls for novel approaches to exploit its real-time capabilities in new visualization tools for the interaction of a human being with petabytes of data. A simple example would be an interactive interface on a parallel-coordinates [6] presentation of multivariate event data, which allows to select outlier events.

The performance of the system can be optimized by making sure that events falling into the same class with respect to a specific selection are distributed as evenly as possible. An analysis query addressing only that subset then will harness a large number of CPU simultaneously and finish with minimal time. Providing analysis jobs with the possibility to tag events as belonging to a certain class, one could also envisage a system which is able to automatically migrate data between nodes in order to minimize access times.

Another level of optimization would be to store event related information which is created by a specific analysis for further use. Information that should be kept in a persistent store can either be specified by the user, or selected automatically, e.g. storing by default all information that is determined with computational cost above a certain threshold.

In addition to the actual functionality of a RAVEN-system a management interface is required to monitor the performance of the system like load balancing, resource usage and network traffic, and which allows to add or remove nodes from the system.

5 Prior Work

Realization of the RAVEN project will benefit greatly from already existing knowledge in networking, middleware design, distributed data storage and computing. Projects which are in principle interesting from the point of view of RAVEN are for example BitTorrent [7, 7] for broadcasting information over a network, the Apache Hadoop [9] project addressing scalable, distributed computing, the BOINC [10, 11] framework for volunteer computing and grid computing, or the xrootd [12, 13] server for low latency high bandwidth data access in the root [14, 15] framework, which defines the de-facto standard for data analysis in particle physics. Additional input could come from the Grid-middleware developers e.g. gLite [16, 17] or the Linux community [18].

6 Summary

A proposal has been presented which targets the problem of high performance parallel analysis of the LHC data. The architecture of such a RAVEN system ensures full scalability with random access to the entire data set. Further key features of the system are redundant storage, on-the-fly accumulation of results and a rigorous middleware-approach to the data analysis plus the development of new tools for visualization and interaction with massive data set.

Acknowledgments

The authors would like to thank Torsten Antoni (GridKa), Stefan Heinzel (RZG), Andreas Heiss (GridKa), John Kennedy (RZG), Stefan Kluth (MPP), Wolfgang Müller (HITS) and Andreas Reuter (HITS) for useful discussions and constructive criticism concerning the RAVEN proposal.

References

[1] Evans L. and Bryant P. (editors): LHC Machine. JINST 3 (2008) S08001.
[2] The LHCb Collaboration, A. Augusto Alves Jt. et al.: The LHCb Detector at the LHC. JINST 3 (2008) S08005.
[3] The LHC Computing Grid: LCG Website, http://lcg.web.cern.ch/lcg/
[4] Eck, C., et al.: LHC computing Grid: Technical Design Report. Version 1.06. LCG-TDR-001 CERN-LHCC-2005-024.
[5] The LHCb Collaboration, Antunes Nobrega R. et al.: LHCb Computing Technical Design Report. CERN/LHCC 2005-019.
[6] Inselberg, A.: The Plane with Parallel Coordinates. The Visual Computer 1 (1985) 69-91.
[7] BitTorrent, Inc.: BitTorrent Website, http://www.bittorrent.com
[8] Cohen, B.: Incentives Build Robustness in BitTorrent. 1st Workshop on Economics of Peer-to-Peer Systems, University of California, Berkeley, C A, USA (2003).
[9] The Apache Software Foundation: Apache Hadoop Website http: //hadoop.apache.org/
[10] BOINC project: BOINC website, http://boinc.berkeley.edu/
[11] Anderson, D. P.: BOINC: a system for public-resource computing and storage. Fifth IEEE/ACM International Workshop on Grid Computing 2004.
[12] XRootD project: Scalla/XRootD Website, http://project-arda-dev.web.cern.ch/project-arda-dev/xrootd/site
[13] Hanushevsky A., Dorigo A. and Furano, F.: The Next Generation Root File Server. Proceedings of Computing in High Energy Physics (CHEP) 2004, Interlaken, Switzerland, 2004.
[14] The ROOT team: ROOT Website, http://root.cern.ch/
[15] Antcheva I., et al.: ROOT - A C++ framework for petabyte data storage, statistical analysis and visualization. Computer Physics Communications 180 (2009) 2499-2512.
[16] gLite Open Collaboration: gLite Website, http://glite.web.cern.ch
[17] Laure E., et al.: Programming the Grid with gLite. Computational Methods in Science and Technology 12 (2006) 33-45.
[18] The Linux Foundation Website, http: //www. linuxfoundation.org

RDMS CMS TIER 2 CENTERS AT THE RUNNING PHASE OF LHC[1]

V. Gavrilov
Institute of Theoretical and Experimental Physics, Moscow, Russia

I. Golutvin, V. Korenkov, E. Tikhonenko, S. Shmatov
Joint Institute for Nuclear Research, Dubna, Russia

V. Ilyin, O. Kodolova
Skobeltsyn Institute of Nuclear Physics, Moscow State University, Moscow, Russia

1. Introduction

Russia and Dubna Member States (RDMS) CMS collaboration [1], founded in the year 1994 and now involving more than twenty institutes from Russia and the Joint Institute for Nuclear Research (JINR) member states, take an active part in the CMS (Compact Muon Solenoid) collaboration [2] at the Large Hadron Collider (LHC) [3] at CERN [4]. RDMS CMS scientists, engineers and technicians were actively participating in design, construction and commissioning of all CMS sub-detectors in forward regions. The RDMS CMS physics program has been adopted taking into account an essential role of these sub-detectors for the corresponding physical channels. RDMS scientists made a large contribution for preparation of study QCD, Electroweak, Exotics, Heavy Ion and other physics at CMS. An overview of the RDMS CMS physics tasks and RDMS CMS computing activities are presented in [5-10]. RDMS CMS computing support should satisfy the LHC data processing and analysis requirements at the running phase of the CMS experiment [11].

2. Current Status of RDMS CMS Activities

During the last few years, a proper grid-infrastructure for CMS tasks has been created at the RDMS CMS institutes, in particular, at the Institute for High Energy Physics (IHEP), Protvino, Joint Institute for Nuclear Research (JINR), Dubna, Institute for Theoretical and Experimental Physics (ITEP), Moscow, Institute for Nuclear Research (INR) of the Russian Academy of Sciences (RAS), Moscow, Skobetsyn Institute for Nuclear Physics (SINP), Moscow, Petersburg Nuclear Physics Institute (PNPI) of RAS, St. Petersburg, P.N. Lebedev Physical Institute (LPI), Moscow and National Scientific Center, Kharkov Institute of Physics and Technology, Kharkov, Ukraine (NSC KIPT). Within the CMS global grid-infrastructure these RDMS CMS sites operate as CMS centers of Tier 2 level with the following names: T2_RU_IHEP, T2_RU_JINR, T2_RU_ITEP, T2_RU_INR, T2_RU_SINP, T2_RU_PNPI, T2_UA_KIPT. The very solution that CERN will serve as a T1 center for RDMS serves as a strong basement to provide the RDMS CMS computing model requirements.The RDMS CMS computing model should provide a valuable participation of RDMS physicists in processing and analysis of CMS data. At the running phase of the experiment, the CMS basic requirements to the CMS Tier2 grid-sites for the physics group hosting are:

- persons responsible for site operation at each CMS T2 site;
- site visibility in the WLCG global grid-infrastructure (BDII);
- availability of CMSSW actual version;
- satisfactory execution of regular file transfer tests;
- certified links with CMS T1- and T2 grid-sites;
- regular CMS Job Robot (JR) testing;

[1] These activities are partially supported by a grant of the Russian Foundation for Basic Research (RFBR) and the Ukrainian Academy of Science (10-07-90400-Ukr)

- disk space of 150-200 TB for: central space (~30 TB), analysis space (~ 60-90 TB), Monte Carlo space (~ 20 TB), local space (~ 30-60 TB) and local CMS users space (~1 TB per user);
- CPU resources ~ 3KSI2K per 1 TB disk space, 2GB memory per job.

During 2008-2009 the RDMS CMS took part in CMS computing testing, cosmic run data processing and analysis, large MC samples production. An example of the successful participation in the STEP'09 (Scale Testing for the Experimental Program in 2009) activities in June, 2009 is given on Fig. 1.

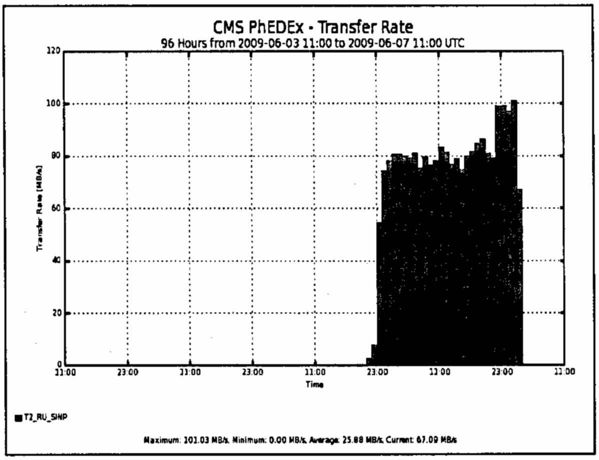


Fig. 1: Transfer rates (up to 100 MB/s) at the SINP T2-site during STEP'09

The network links testing and also the massive CMS jobs submission gave a possibility to detect and avoid bottle-necks in the RDMS CMS grid-sites configuration. For example, at the JINR site, it leads to a necessity to reorganize the internal network and Storage Element configurations. As a result of the creation of the dedicated sub-network for disk pools, computing farm and a number of NFS-servers, it became possible to increase significantly the efficiency of CMS jobs execution at the JINR grid-site. Finally these works on the reconfiguration have increased the JINR grid-site efficiency on the whole.

Starting in the 2008 year, the RDMS Tier2 is associated with the CMS Exotics Physics Analysis Group and the CMS Muon Physics Object Group (both groups are hosted at the JINR site), the CMS Heavy Ion Physics Analysis Group (hosted at the MSU site) and the JetMet/HCAL Physics Object Group (hosted at the ITEP). Special tests show that the RDMS Tier2 satisfies all the requirements for such a hosting, including the additional requirements for certification of data transfer links between RDMS sites and other Tier-2 centers associated also with the same CMS Physics Groups. In general, RDMS CPU resources are sufficient for the analysis of first data expected after the LHC start and for simulation.

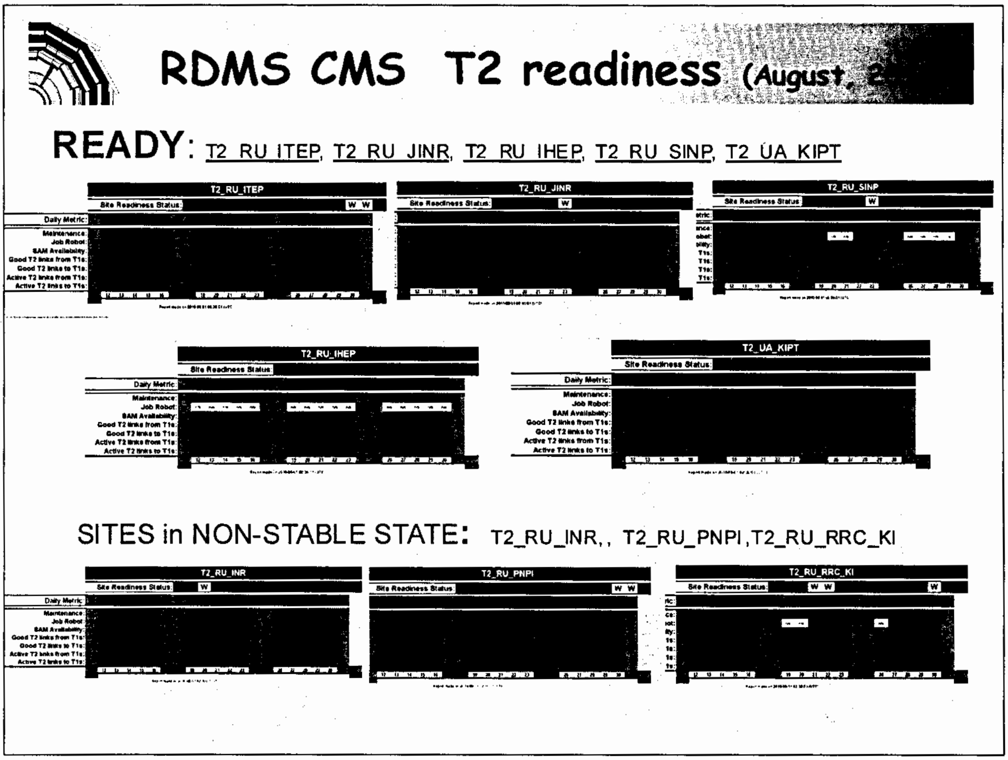
In October 2009, a special global test of the CMS Computing and Data model ("October Exercise") was performed. All stages of data processing and physics analysis spreading at the Tier-0/Tier-1/Tier-2 were stressed. The goals were check-up of the CMS readiness for the first data taking and analysis at the end of 2009. The RDMS took part in the test within two Physics Groups (Exotica and Muon) associated with JINR sites. During the exercise accessibility and stability of data transfer links, disk resources, job slots, core and CMS software were tested again and again. To check up the reconstruction and analysis procedures, about 80 TB both MC data (RECO and AOD) and cosmic test data (RAW) were transferred. The final step was on-line processing of transferred data by RDMS physicists and publication of obtained results in the CMS Discovery Data Base.

The RDMS Computing infrastructure was tested in the first LHC collisions on 0.9 TeV and 2.36 TeV in 2009. In 2010, the RDMS Tier-2's sites provide all the required facilities to perform data processing and analysis within 24-houres during 7 TeV CMS Run.

In 2010, CMS T2 sites (computing centers) were considered in the context of the CMS computing requirements as "ready" for the running phase of the experiment in the case of:

- site visibility and CMS virtual organization (VO) support;
- availability of disk and CPU resources;
- daily SAM tests availability > 80%;
- daily JR efficiency > 80%;
- commissioned links TO Tier-1 sites ≥ 2;
- commissioned links FROM Tier-1 sites ≥ 4.

Readiness of the RDMS CMS sites (by August, 2010) is shown on Fig. 2.



RDMS CMS T2 readiness (August, 2

READY: T2_RU_ITEP, T2_RU_JINR, T2_RU_IHEP, T2_RU_SINP, T2_UA_KIPT

SITES in NON-STABLE STATE: T2_RU_INR,, T2_RU_PNPI,T2_RU_RRC_KI

Fig. 2: A state of the RDMS CMS sites readiness by August, 2010 (for a current situation see *http://lhcweb.pic.es/cms/SiteReadinessReports/SiteReadinessReport.htm*)

In total about 112 TB were transferred to the RDMS Tier-2's in the period of LHC operation phase starting on March 30th to August, 2010 (Fig. 3). A maximum transfer rate to RDMS Tier-2 was ~ 58.57 MB/s, an average one ~ 12 MB/s (Fig. 4).

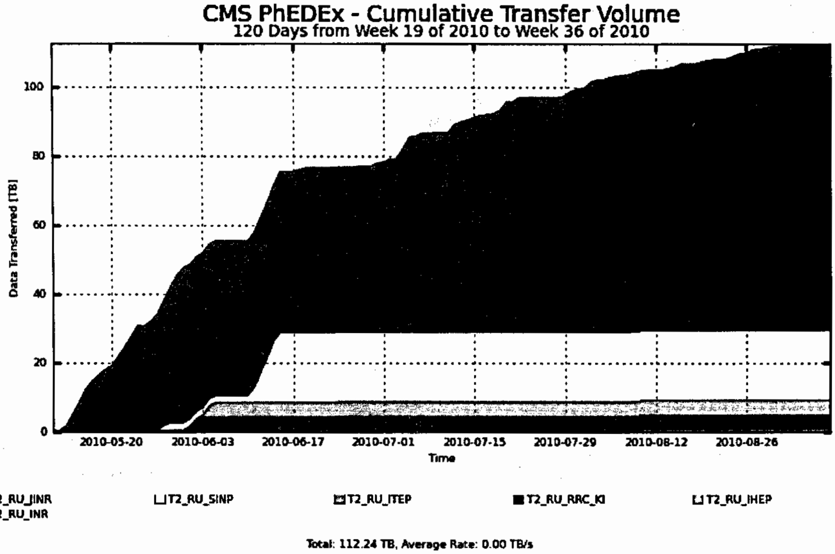


Fig. 3: The cumulative transfer volume for the RDMS T2-sites during the LHC operations in 2010 (from March to August, 2010)

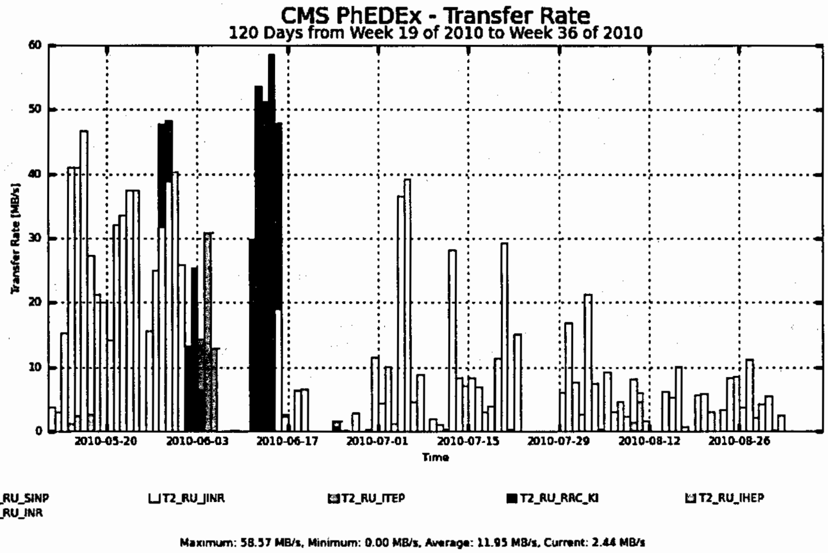


Fig. 4: Transfer rates (up to 59 MB/s) at the RDMS T2-sites during the LHC operations in 2010 (from March to August, 2010)

The RDMS CMS T2 sites are actively used by the CMS collaboration: 1,837,392 jobs of the virtual organization CMS were submitted to the RDMS CMS T2 sites from September 2008 to December 2009 and for the same period of 2009-2010 – 2,685,718 jobs. The RDMS sites were involved intensively in data processing and analysis. The Fig.5 shows that since the LHC 7 TeV start-up 24 % of CPU time was spent on CMS tasks to operate with 29 % of all Russia Tier-2 jobs.

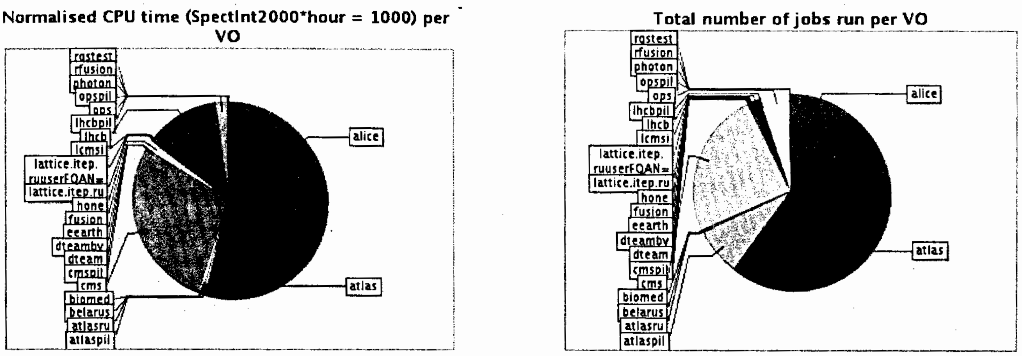


Fig.5: The normalized CPU time (lefit) and the total number of jobs (right) per virtual organization during the LHC operations in 2010 (from March to August, 2010)

Now, in the context of the CMS computing requirements for the running phase of the experiment, the RDMS CMS grid-sites provide:

- the computing and data storage resources in a full volume;
- centralized installation of actual versions of CMS specialized software (CMSSW);
- VOBOX grid-services for CMS with Phedex server installed to provide data transfers between the CMS grid-sites with the usage of the FTS grid-service;
- SQUID proxy-servers for the CMS conditions DB access;
- certification of network links at the proper data transfer rates between JINR and CMS Tier1 and Tier2 centers;
- daily massive submission of CMS typical jobs by the CMS Job Robot system;
- CMS data replication to the JINR data storage system in the accordance with RDMS CMS physicists' requests;
- participation in the CMS Monte-Carlo physical events mass production in accordance with the RDMS CMS physicists' scientific program.

A group of RDMS CMS specialists takes an active part in the CMS Dashboard development (grid monitoring system for the CMS experiments) (/http://dashboard.cern.ch/cms) [12]

The JINR CMS Remote Operation Center (ROC) was founded in 2009 to provide participation in CMS operations of a large number of RDMS CMS collaborating scientists and engineers; the dedicated CMS remote worldwide-distributed centers (ROC) were built in different scientific organization [13]. The JINR CMS ROC has been designed as a part of the JINR CMS Tier 2 center and provides the following functions:

- monitoring of CMS detector systems;
- data monitoring and express analysis;
- shift operations;
- communications of the JINR shifters with personal at the CMS Control Room (SX5) and CMS Meyrin centre;
- communications between JINR experts and CMS shifters;
- coordination of data processing and data management;
- training and information .

In 2010, the CMS ROC was founded and certified at the SINP MSU to provide similar functions for CMS participants in Moscow.

RDMS CMS physicists work within the WLCG environment, and now we are having about 30 members of CMS Virtual Organization (VO). To help them work in the WLCG environment, a number of training courses for CMS users have been organized [14].

3. Summary

The RDMS CMS computing centers have been integrated into the WLCG global grid-infrastructure providing a proper functionality of grid services for CMS. During 2008-2010 a significant modernization of the RDMS CMS grid-sites has been accomplished. As a result, the computing performance and reliability have been increased. In frames of the WLCG global infrastructure the resources of the both computing centers are successfully used in a practical work of the CMS virtual organization. A regular testing of the RDMS CMS computing centers functionality as grid-sites is provided.

All the necessary conditions for the CMS data distributed processing and analysis have been provided at the RDMS CMS computing centers (grid-sites). It makes possible for RDMS CMS physicists to take a full-fledged part in the CMS experiment at its running phase.

References

[1] Matveev V., Golutvin I. Project: Russia and Dubna Member States CMS Collaboration / Study of Fundamental Properties of the Matter in Super High Energy Proton-Proton and Nucleus-Nucleus Interactions at CERN LHC , 1996-085/CMS Document, 1996.

[2] CMS Collaboration, Technical Proposal, CERN/LHCC, 94-38, 1994, http://cmsinfo.cern.ch

[3] http://public.web.cern.ch/Public/Content/Chapters/AboutCERN/CERNFuture/WhatLHC/WhatLHC-en.html

[4] http://www.cern.ch

[5] Gavrilov V. et al. RDMS CMS Computing Model // Proc. of the Int. Conf. "Distributed Computing and Grid-Technologies in Science and Education", Dubna, 2004. P. 240.

[6] Gavrilov V. et al. RDMS CMS Computing // Proc. of the 2nd Int. Conference "Distributed Computing and Grid-Technologies in Science and Education", Dubna, 2006. P. 61.

[7] Oleinik D.A. et al. RDMS - CMS Data Bases: Current Status, Development and Plans // Proc.of the XX Int. Symp. on Nuclear Electronics and Computing, Dubna, 2006. P. 216.

[8] Gavrilov V. at al. Current Status of RDMS CMS Computing // Proc. of the XXI Int. Symposium on Nuclear Electronics and Computing, Dubna, 2008. P. 203-208.

[9] Oleinik D.A. at al. Development of the CMS Databases and interfaces for CMS experiment // Proc. of XXI Int. Symp. on Nuclear Electronics & Computing (NEC`2007), ISBN 5-9530-0171-1, 2008. P. 376-381.

[10] Gavrilov V. at al. RDMS CMS Computing activities before the LHC startup // Proc. of 3rd Int. Conference "Distributed Computing and GRID-technologies in Science and Education, Dubna, 2008. P. 156-159.

[11] CMS Collaboration, The Computing Project, Technical Design Report, CERN/LHCC-2005-023, CMS TDR 7, 2005.

[12] Andreeva J. et al. Dashboard for the LHC experiments. CERN-IT-NOTE-2007-048, presented at CHEP'2007 and published in J.Phys.Conf.Ser.119:062008, 2008.

[13] Golunov A.O. et al. The JINR CMS Remote Operation Centre", in these proceedings.

[14] http://www.egee-rdig.ru/rdig/user.php

THE JINR CMS REGIONAL OPERATION CENTER

A. O. Golunov, N. V. Gorbunov, V. V. Korenkov,
S. V. Shmatov, A. V. Zarubin

Joint Institute for Nuclear Research, Dubna, Russia

The dedicated remote center of CMS Experiment at the LHC was founded in JINR (Dubna). The main mission of the center is operational and efficient monitoring of the CMS detector systems, its working efficiency including the measurements of performance parameters during the prompt data analysis, monitoring of data acquisition and quality data.

For the Compact Muon Solenoid experiment (CMS) [1] at the LHC the remote operations plays an important role in detector operations, monitoring and prompt data analysis. To provide participation in CMS operations of large number of collaborating scientists and engineers the dedicated CMS remote worldwide-distributed centers (ROC) were built in different scientific organization (Fig.1).

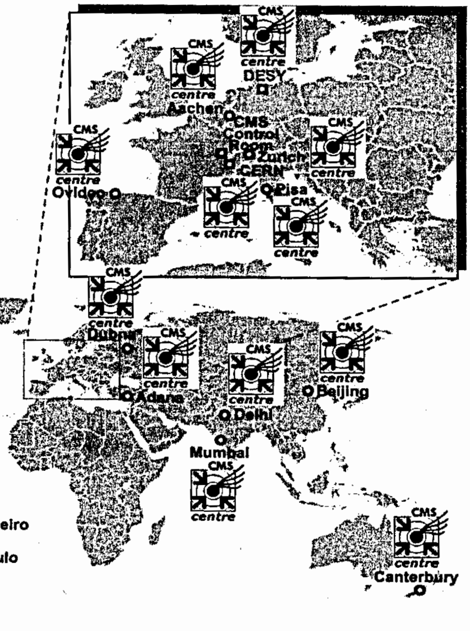


Fig.1: Locations of operating or planned CMS Centres Worldwide

The purpose of the worldwide centers is to help specialists working on CMS contribute remotely their expertise to commissioning and operations at CERN [2].

One of these centers was founded in Joint Institute for Nuclear Research (JINR). This center is located in Laboratory of High Energy Physics of JINR (LHEP). JINR ROC is focused on operations and monitoring of inner endcap detectors, where collaboration of Russia and Dubna Member States Institutions (RDMS) bears a full responsibility on Endcap Hadron Calorimeters (HE) [3] and First Forward Muon Stations (ME1/1) [4]. It should provide effective facilities to support activities which are associated both with the main CMS Center on the CERN main site in Meyrin and in part with the CMS Control Room at the LHC interaction point 5 in Cessy. The JINR ROC has following main primary functions:

1. *Monitoring* of the detector systems including data acquisition (DAQ) system, detector control system (DCS), i.e. slow control system for both high voltage and low voltage systems, temperature monitoring etc, and data quality monitoring (DQM).
2. On-the-fly *Data Analysis Operations*, in particular detector performance measurements, display events, *Calibrations* of sub-detector systems, etc.
3. Effective participation in remote shifts as well as prompt access to experts to experimental information. Communication of shifter with system experts during data taking and CMS systems operations.
4. Offline Computing Operations for coordinating the processing, storage and distribution of real CMS data, MC data, their transfers at RDMS Tier-1 and JINR Tier-2.
5. Training and outreach

The important point is to have secure access to information that is available in control rooms and operations centers at CERN.

The JINR ROC consists of the file and graphic server, monitoring and analysis system, user workstations, video-conferencing system. The scheme of JINR ROC is given Fig.2. The main ROC room is shown on the left with the server and three monitoring workstations while the right plot depicts user workstations (a total of nine stations) which are located outside the main room. The conference system includes a screen, a projector and high-quality Tandberg 550MXP video-conferencing system.

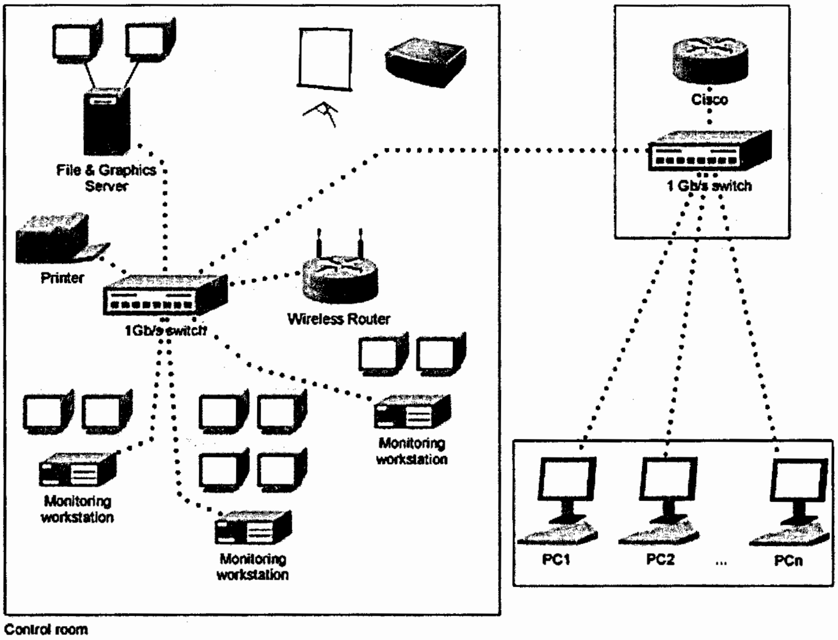


Fig.2: The scheme of the JINR Remote Operations Center

The monitoring system of JINR ROC includes the SLC Linux graphic and file server, two 40′ LCD screens (*information displays*) mounted on the wall and connected to the server and three monitoring workstations (*working places*) (Fig.3).

The server is used for express-analysis, storage of files with monitoring information and results of data analysis. The server is based on Dual Xeon 2.53Ггц processor, 16GB RAM, and 6TB HDD with 2 gigabit Ethernet cards and 9600GT dual-head video card.

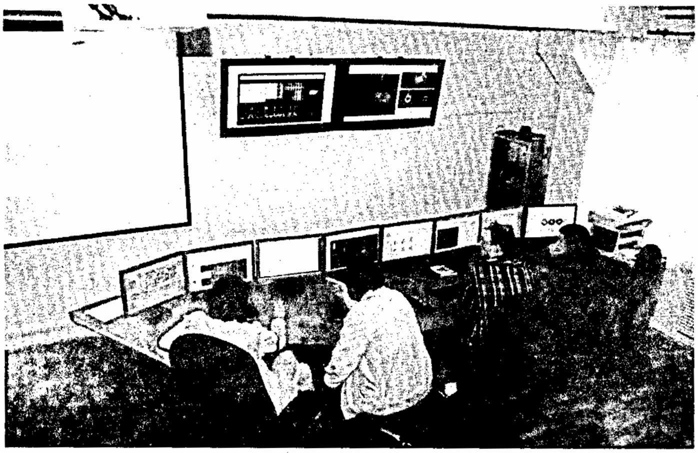
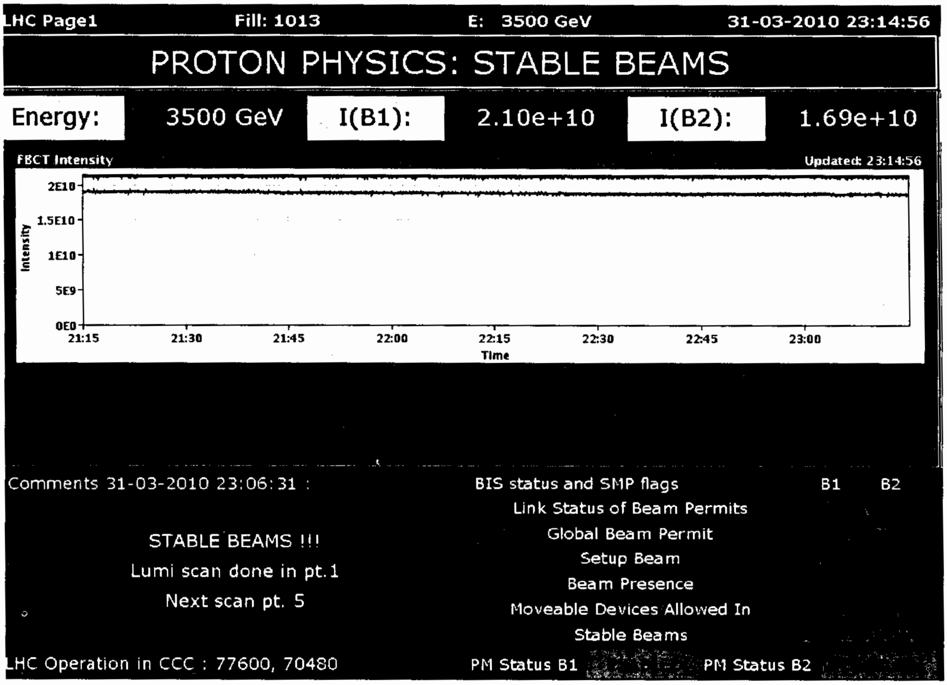


Fig.3: The JINR Remote Operations Center

One of *information displays* shows the LHC page 1 [5] with information on the LHC beam status (fig. 4). The second screen maps the CMS data taking including the DAQ system (fig.5), DCS, and event displays [6].



LHC Page1 Fill: 1013 E: 3500 GeV 31-03-2010 23:14:56

PROTON PHYSICS: STABLE BEAMS

Energy: 3500 GeV I(B1): 2.10e+10 I(B2): 1.69e+10

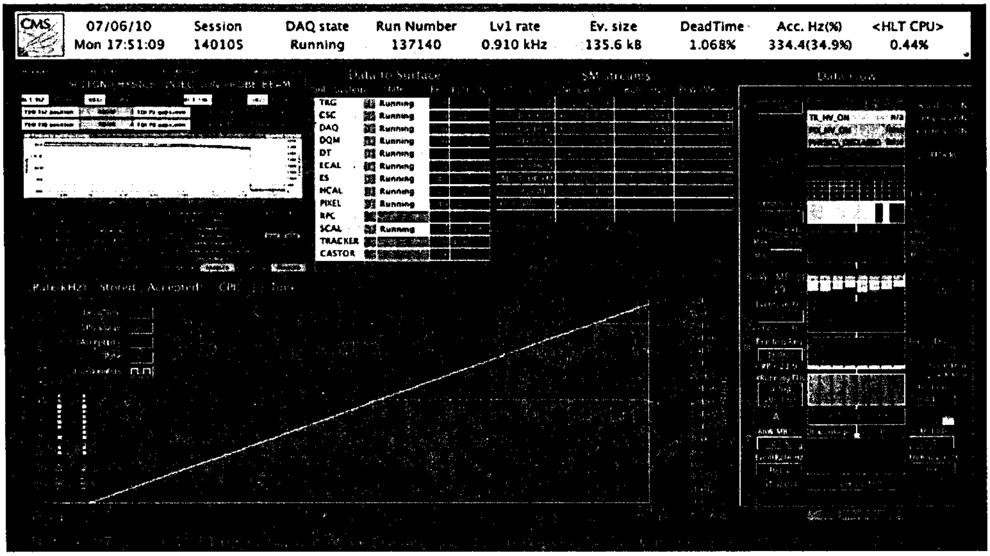
Comments 31-03-2010 23:06:31 :

STABLE BEAMS !!!
Lumi scan done in pt.1
Next scan pt. 5

BIS status and SMP flags B1 B2
Link Status of Beam Permits
Global Beam Permit
Setup Beam
Beam Presence
Moveable Devices Allowed In
Stable Beams

LHC Operation in CCC : 77600, 70480 PM Status B1 PM Status B2

Fig.4: The LHC Page 1 (LHC Beam Status)



CMS	Session	DAQ state	Run Number	Lvl rate	Ev. size	DeadTime	Acc. Hz(%)	<HLT CPU>
07/06/10 Mon 17:51:09	140105	Running	137140	0.910 kHz	135.6 kB	1.068%	334.4(34.9%)	0.44%

Data to Surface

TRG	Running
CSC	Running
DAQ	Running
DQM	Running
DT	Running
ECAL	Running
ES	Running
HCAL	Running
PIXEL	Running
RPC	[illegible]
SCAL	Running
TRACKER	[illegible]
CASTOR	[illegible]

Fig.5: The CMS Page 1 (CMS DAQ Status)

Two of three *working places* are equipped with an interactive console SLC Linux PC's (Intel CoreDuo 3Ггц, 4GB RAM, 750GB HDD, 2 dual-head video card) and three 20′ LCD screens each. They are assigned to operations of two CMS detector sub-systems (HE and ME1/1). The main functions include monitoring of detector operations and data quality. One screen provides information on status of parameters of the sub-detector control system – low and high voltage, cooling system etc, as well as information on a run number, run type, number of stored events, monitoring of data taking. The second one allows to monitor data quality and to display results of express-analysis. The third screen provides access to the e-log book of shifts, shifter manuals and also serves for another interactive works.

The third working place has two LCD screens. It is for shift leader and/or offline computing operations.

Each of working place is equipped by local communication tools (web-cameras and headphones) to enable connections when needed between shifter and experts.

The center is managed by the special control server placed in the single room (ROC Control Room). The gigabit router, network switch and a wireless network are used to link the ROC parts (server, monitoring workstations, network printer, and videoconferencing system) to each other. The ROC Control Room is connected to the 16-port gigabit switch in the main server room of LHEP JINR. This switch provides links between the JINR ROC and JINR Tier-2 located in Laboratory of Information Technologies of JINR. The user workstations are also linked directly with the LHEP Server Room.

Experimental data are transferred from CERN (Tier-0/RDMS Tier-1) as well as from other CMS Tier-1 sites to the JINR Tier-2 site. The CMS GRID transfer system (PHEDEX) [7] is used for the bulk transfer. Then data is processed at Tier-2 and analyzed in JINR ROC. The small part of data can be transferred directly to the JINR ROC for prompt processing and detailed analysis. The CMS individual file transfer system (FileMover) [8] is applied for this case.

The high definition H.323 point-to-point features of Tandberg videoconference system allows to involve center in the CMS TV outreach events for public overview of LHC and CMS Status, Live Event Displays, etc. The video-conference system uses a software-based video system (EVO) [9] to help to coordinate shifters and experts, and makes easy weekly meetings.

The JINR ROC was tested in cosmic tests and the first LHC collisions on 0.9 TeV and 2.36 TeV in 2009. The sub-system data quality monitoring, online and offline global data quality monitoring, slow control systems, DAQ monitoring system were in use remotely. In 2010 the JINR ROC centre provides three shifter working place for participation in data taking and analysis within 24-houres during 7 TeV CMS Run.

References

[1] CMS Collaboration: R. Adolphi et al., "The CMS experiment at the CERN LHC", JINST 3:S08004, 2008

[2] Lucas Taylor and Erik Gottschal, "CMS Centres Worldwide: a New Collaborative Infrastructure", Proc. Of CHEP'09, 21–29 March, 2009, Prague, J. of Phys: Conf. Series, 219 (2010) 082005; L. Taylor et al., "CMS centres for control, monitoring, offline operations and prompt analysis." Proc. of CHEP '07, 2.–7 Sept. 2007, Victoria; J. of Phys: Conf. Series, 119 (2008).

[3] CMS HCAL Collaboration: G. Baiatian et al.,"Design, performance, and calibration of CMS hadron endcap calorimeters", CERN-CMS-NOTE-2008-010, Mar 2008. 36pp.

[4] Erchov Yu.V. et al., "ME1/1 Cathode Strip Chambers", CERN-CMS NOTE-2008-026, Part. Nucl. Lett. **№4 (153)** (2009) 566.

[5] http://op-webtools.web.cern.ch/opwebtools/vistar/ vistars.php?usr=LHC1

[6] http://cmsdoc.cern.ch/cmscc/cmstv/cmstv.jsp?channel=2&frames=yes

[7] Egeland R., Wildish T., and Huang Ch.-H. "PhEDEx data service", J.Phys.Conf.Ser. **219** (2010) 062010; R. Egeland *et al.*, "Data transfer infrastructure for CMS data taking", Proceedings of Science, PoS (ACAT08) **033** (2008); L. Tuura *et al.*, "Scaling CMS data transfer system for LHC start-up", J.Phys.Conf.Ser, **119** (2008) 072030.

[8] Bockelman B. and Kuznetsov V. "CMS FileMover: one click data", CHEP 2009.

[9] http://evo.caltech.edu/evoGate/Documentation/

AN APPROACH TO DEVELOPING BUSINESS PROCESSES WITH WEB SERVICES IN GRID

R. D. Goranova[1], V. T. Dimitrov[2]

Faculty of Mathematics and Informatics,
University of Sofia"S. Kliment Ohridski", 1164, Sofia, Bulgaria
[1]radoslava@fmi.uni-sofia.bg, [2]cht@fmi.uni-sofia.bg

In this approach, g-Lite Grid middleware site accounting functionality is exposed as Web Services. In the essence of the approach, Web Services are registered in IBM WebSphere Service Registry and Repository Server. The last one supports UDDI. Business processes are described and developed in WebSphere Business Modeler and WebSphere Integration Developer. The business process orchestrator - WebSphere Process Server is outside of the Grid environment, but can manage processes composed of web services from the middleware.

Introduction

g-Lite [1] is a Grid middleware, which is designed and implemented for EGEE [2] Grid infrastructure and is tightly specified for the need of the project. The middleware provides Grid services for resource brokering, job computing and data storage. On Fig. 1 are shown g-Lite services, grouped logically into five groups:

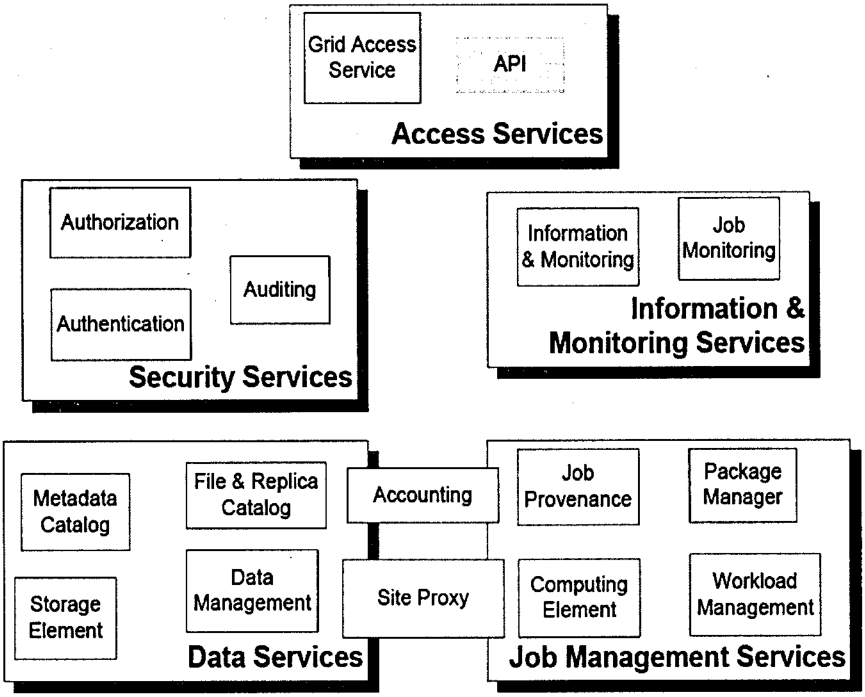


Fig. 1: g-Lite Services

Security services provide mechanism for user or service identification, authorization, verification of user permissions and log information for auditing. The monitoring and information services provide mechanism for task monitoring, resource discovery and retrieval of service information. The job management services include computational resource for job submission and execution, tasks scheduler, job tracking system and accounting information. The data services includes storage element for access to storage resources, file catalog and file transfer services.

The middleware provides services for collecting accounting information. This information includes data for the number of submitted jobs, for users who submit the jobs and for virtual organization to which these users belong.

The most of the services which g-Lite middleware provides are not service-oriented. The last one requires implementation of architectural principles [3] as: contract, abstraction, reusability, composability and discoverability. For clearness, we will describe them bellow:

- The contract is a document describing how a service can be programmatically accessed;
- Abstraction claims that services expose only the logic defined in service contract and hide implementation from the client;
- Reusability guaranties that services can be reused more than once and from multiple clients;
- Composability provides opportunity service to be grouped in composite services and execute as processes;
- Discoverability provides a standard mechanism for services discovery.

For example the securities services, which g-Lite provide haven't standard description. The middleware does not provide composition service. The discovery mechanisms that the middleware provide are not standard. There is information system, where the information for available services can be found, but there is not information how this services can be invoked. Another disadvantage is the lack of centralized registry, where all services can be published. However, as it is defined in SOA [4], the standard service description, service discovery, reusability and composition are main features of a service-oriented environment.

As we mentioned in the beginning, g-Lite is a Grid middleware tightly specified for the need of the EGEE project. The aim of this project is to develop and deploy new grid infrastructure for scientific research. The project has two priority scientific directions, serving the needs of biomedical experiments and needs of High Energy Physics (HEP).The last one is an area with complex business processes. By business process, we mean set of services, ordered in common schema for execution. The processes of HEP can include not only services form the Grid infrastructure, but also components from specific software for data simulation and analysis.

The goal of our research is to outline the approach for developing of business processes with Web services in Grid environment. More precisely, we are interested in business processes in HEP and their execution in g-Lite middleware. The security issues are not subject of our research.

IBM SOA Foundation

IBM SOA Foundation [5] encompasses tools, programming model, methodologies and techniques for capturing and implementing business design and the middleware infrastructure for hosting that implementation. The SOA Foundation is a comprehensive set of technologies, and practices that address all SOA features, such as flexibility, dynamicity, easier integration and reuse. Flexibility allows the business process to be changed without major efforts, after its deployment. Dynamicity at runtime allows a service from the process to be changed with another service, implemented with different technology, programming language or in different runtime environment. Service reuse means that services can be used in other applications.

The SOA approach breaks down the underlying software and information technology into reusable components (services). These services can be combined and recombined into complex processes. SOA allows the services, to talk to each other using the open standards. The SOA life cycle [6] includes four phases (Fig. 2). They can be summarized as follows:

- *Model* - During the modeling phase are gathered requirements and processes are designed. An IBM SOA Foundation tools for business analysts, modeling and simulation of business processes is IBM WebSphere Business Modeler;
- *Assemble* – During the assemble phase, processes are developed, assembled and tested in integration environment. This environment provides services for transport and mediation and control capabilities, for flow management and services interactions. An IBM SOA Foundation tool for workflows and data modeling and system interactions is IBM WebSphere Integration Developer;
- *Deploy* – During deployment phase are integrated people, process and information;

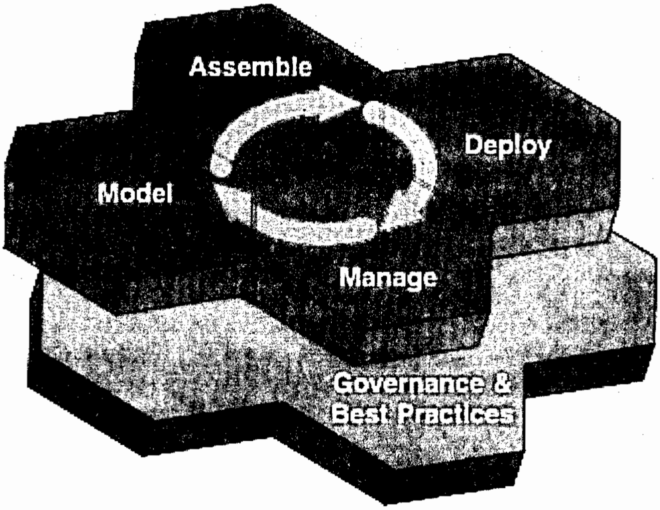


Fig. 2: SOA lifecycle

- *Manage* – During management phase, applications and services are monitored. An IBM SOA Foundation tools for business monitoring is IBM WebSphere Business Monitor.

Governance and best practices support the life cycle through the use of information technology alignment and process control. Service registry that allows users to manage SOA life cycle from development through deployment is IBM WebSphere Service Registry and Repository.

Approach Specificity and Implementation

The approach, we proposed, is based on programming model and methodologies defined in IBM SOA Foundation. It is service-oriented and is based on the next steps:

- Web service development,
- Web service registration,
- Process modeling,
- Process assembling, deployment and testing.

For approach realization, we use the WebSphere Business Modeler to define the process. WebSphere Integration Developer for application assembly. The WebSphere Process Server built into the WebSphere Integration Developer for deployment and testing. And the WebSphere Service Registry and Repository for governance, service metadata and reuse.

Web service development

The business processes modeling is not possible without services. They are the main components of the process. As we mentioned in the beginning, the high energy physics is an area with complex business processes. They can include not only services form the Grid infrastructure, but also components from specific software for data simulation and analysis. In order to achieve the goal of our research, services for software of HEP have to be developed. A problem is that g-Lite is not service-oriented. The environment does not provide standard for service description and mechanisms for service discovery. For modeling more complex business process in Grid environment, we also have to develop services for job submission, based on provided services in the g-Lite middleware.

In order to demonstrate the approach, we develop site statistic service, which provides the following functionality:

- userTaskCount – returns information for number of jobs for given user;
- userFailedJobs – returns information for number of failed jobs for given user;
- voTaskCount – returns information for number of jobs for given VO;
- voFailedJobsCount – returns information for number of failed jobs for VO;
- userCPUTime – returns information for used CPU time for given user;
- voCPUTime – returns information for used CPU time for given VO;
- siteTaskCount – returns information for number of jobs submit on a site;
- siteFailedJobs – returns information for number of failed jobs for a site;
- siteCPUTime – returns information for used CPU Time on a site;
- drawPieChart – returns URL of image pie chart for given data.

For web service implementation, we used Eclipse Platform, JDK 1.6 and Axis 2. The service was deployed on Tomcat 6 application server (Fig. 3).

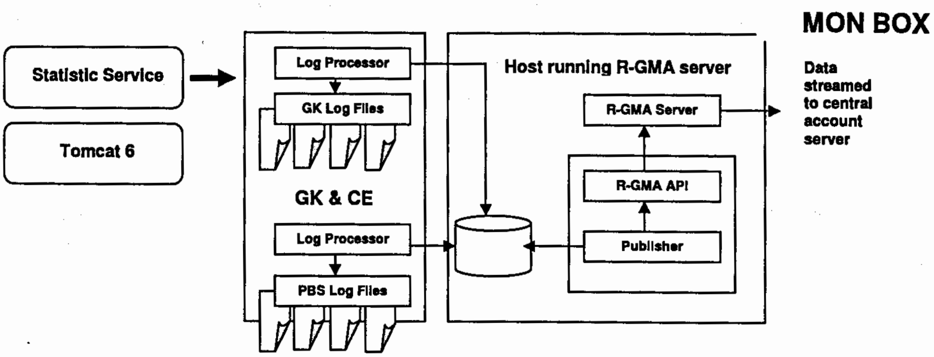
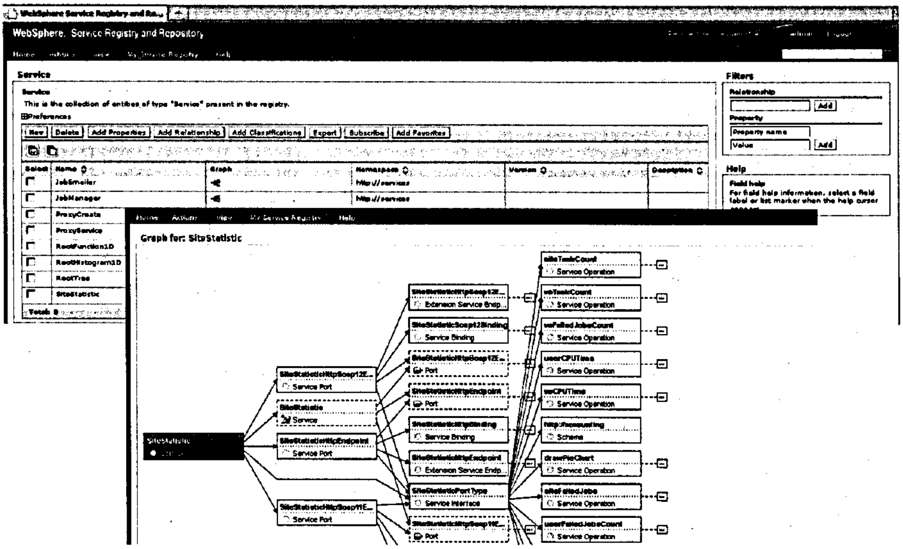


Fig. 3: Accounting web service development [7]

Web service registration

For service metadata and reuse, we use WebSphere Service Registry and Repository (WSRR) [8]. WSRR is a service registry, which provides suitable interface for service definition and registration. Another feature is WSSR plug-in for Eclipse that allows services to be registered into the repository form within the Eclipse environment. WSRR supports UDDI and provide Web Browser for service registration. Registered services can be browsed as shown on Fig. 4.



WebSphere. Service Registry and Repository

Service

Service

This is the collection of entities of type "Service" present in the registry.

⊞Preferences

Select	Name	Graph	Namespace	Version	Description
□	JobEmailer		http://services		
□	JobManager		http://services		
□	ProxyCreate				
□	ProxyService				
□	RootFunction1D				
□	RootHistogram1D				
□	RootTree				
□	SiteStatistic				

Filters

Relationship

Property

Help

Field help

For field help information, select a field label or list marker when the help cursor

Graph for: SiteStatistic

Fig. 4: Web service registry and service graph

Process modeling

We develop example process to demonstrate the approach. We have to mention, that process is modeled, assembled and deployed without writing a single line of code. The reasons are two:

- All services which are part of the process are developed to expose their functionality by using only simple types. For example, all operation of statistic service get as input simple data types – string, integer, etc. and return simple data types – string;
- IBM SOA Foundation provide good framework for business process specification, generation and execution.

For process modeling, we use WebSphere Business Modeler 7.0. On the process bellow (Fig. 5), we use the developed statistic service and two of its functionality voTaskCount and drawPieChart. The aim of the process is to show statistic for the number of jobs which different VOs submit to a site.

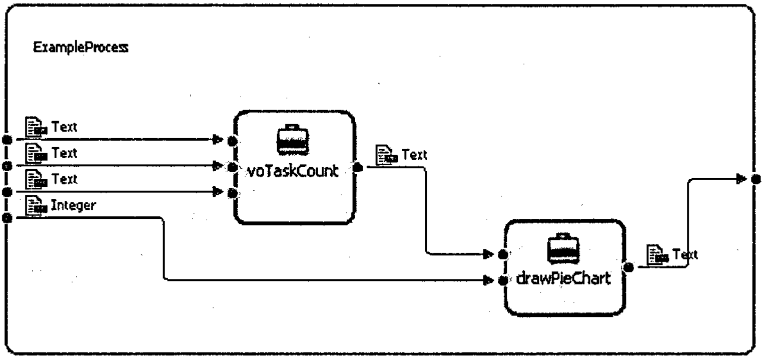


Fig. 5: Example process for site statistic

We have to mention that WebSphere Business Modeler can be also integrated with WSSR plug-in, which allows services to be included directly from registry into the process. As input the process gets the name of a VO and the date range for the desired period of time. If VO name is not specified, statistics is returned for all VOs. The result is URL of image, located on Tomcat server and is accessible from the Internet. On Fig. 6 we show the development of more complex process.

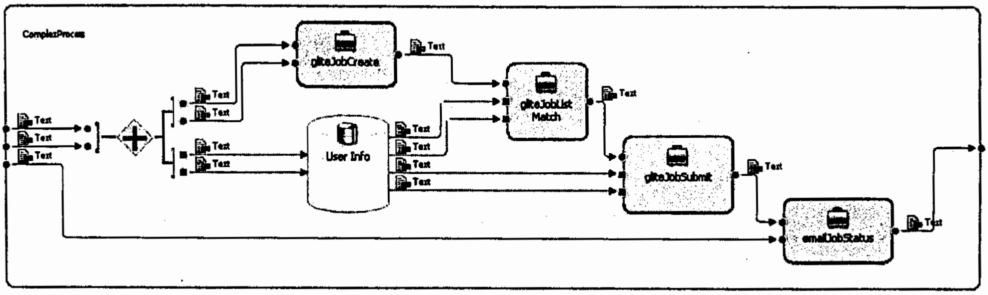


Fig. 6: Process for job submission in g-Lite environment

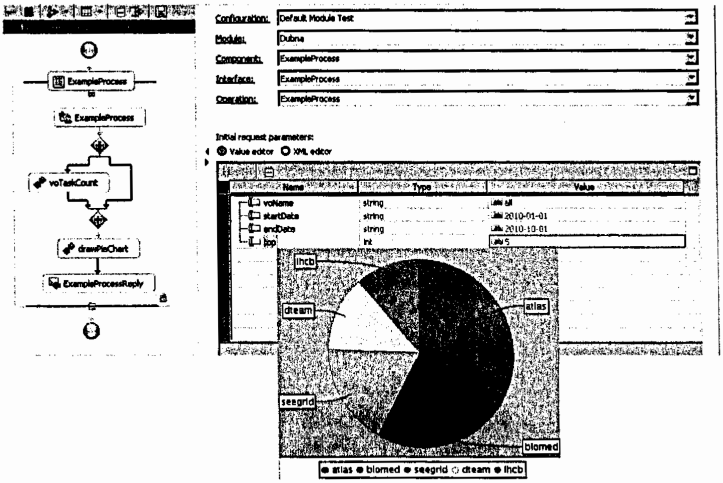
This is a process based on services for job submission into g-Lite environment. More complex service composition is possible, but in order to do that more services have to be developed. Currently, we are working on implementation of the following services:

- ROOT web services – will exposing legacy ROOT functionality as services;
- Job manager services – provide functionality for job submission into g-Lite environment, by exposing existing g-Lite functionality;
- Proxy services – provide functionality for proxy certificate management by exposing existing g-Lite functionality;

All of developed services are designed according principles of service-orientation; they are published into repository and can participate into more complex sequences of tasks – processes.

Process assemble, deployment and test

The example process, we modeled above, was assembled and deployed into Process Server. For process assembling, deployment and testing, we used WebSphere Integration Developer 7.0. On Fig. 7 are shown the process as it looks into integration development environment, and example test and result.



Configuration: Default Module Test
Module: Dubna
Component: ExampleProcess
Interface: ExampleProcess
Operation: ExampleProcess

Initial request parameters:
● Value editor ○ XML editor

Name	Type	Value
voName	string	all
startDate	string	2010-01-01
endDate	string	2010-10-01
top	int	5

Fig. 7: Process deployment, testing and result

Conclusion

The approach, we present, outlines framework for business processes specification, development and execution into g-Lite Grid middleware. The advantages of this approach are dynamic flexibility and loose-coupling. However business process specification of HEP is a subject of future investigation.

Acknowledgement

This work was supported by University of Sofia SRF under Contract N163/2010.

References

[1] Programming the Grid with g-Lite, http://cdsweb.cern.ch/record/936685/files/egee-tr-2006-001.pdf

[2] EGEE, http://public.eu-egee.org/

[3] Service-Oriented Architecture: Principles of service design, T. Erl, Prentice Hall (2007).

[4] Service-Oriented Architecture: Concepts, Technology, and Design, T. Erl, Prentice Hall, 2005.

[5] IBM SOA Foundation: Providing what you need to get started with SOA, ftp://ftp.software.ibm.com/software/solutions/pdfs/SOA_g224-7540-00_WP_final.pdf

[6] IBM SOA Foundation: An Architectural Introduction and Overview, http://download.boulder.ibm.com/ibmdl/pub/software/dw/webservices/ws-soa-whitepaper.pdf

[7] Apel User Guide, http://www.egee.cesga.es/EGEE-SA1-SWE/accounting/guides/apel-user-guide-glite.pdf

[8] IBM WebSphere Service Registry and Repository Handbook, http://www.redbooks.ibm.com/redbooks/pdfs/sg247386.pdf

WEB-BASED USER INTERFACE FOR GRIDNNN[1]

A. P. Gulin, A. K. Kiryanov, N. V. Klopov, S. B. Oleshko, Yu. F. Ryabov

Petersburg Nuclear Physics Institute, Russia, 188300, Gatchina, Orlova Roscha.
Tel.: +7 (81271) 46197, fax: +7 (81271) 35270
globus@pnpi.nw.ru

One of the core components of the GridNNN infrastructure is a user interface that allows users to prepare and run their jobs as well as to retrieve and analyze the results. The basic requirements for this interface are:

- Easy access to GridNNN from different platforms and environments.
- Single login point for job management and data access.
- State persistence across different sessions from different locations.

It was decided to use modern Web 2.0 technologies like DHTML and AJAX to provide users with a web-based interface accessible from a wide range of browsers on various platforms.

GridNNN Web interface is built as a client-server application where client part is written in JavaScript and run in a context of a web browser while the server part is a CGI application written mostly in Perl. Client-server data exchange is performed via secure HTTPS connections using AJAX API (fig. 1).

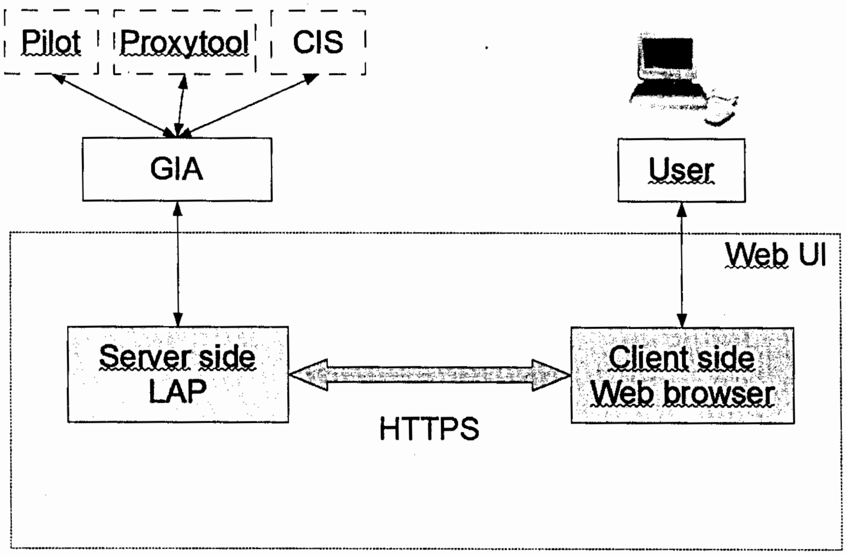


Fig. 1: Web UI general structure

Installation requirements for the server part are as follows:

- Linux (RHEL 5/CentOS 5 recommended)
- Apache + mod_ssl
- Perl + Python

[1] This work was partially supported by the Russian Ministry of Education and Science, Contract No.01.647.11.2004

- Globus GridFTP server
- GridNNN Pilot CLI
- GridNNN ProxyTool CLI

Server part has a modular architecture. All interaction with GridNNN middleware is done via Grid Interface Adaptor (GIA) modules, which understand a well-defined set of abstract commands. These commands are executed by GIA using Pilot, ProxyTool or CIS (Central Information System) interfaces. Such two-layer structure allows easy modification or even replacement of underlying middleware stack without any changes to the client or main server part of Web UI.

Client part runs in a context of a user's web browser and provides the following functionality:

- Proxy certificate management, VOMS role management – X.509 proxy certificates are used for user authentication on all GridNNN resources. Initial certificate is fetched from Myproxy repository hosted by GridNNN Security Service. VOMS roles may be added to the proxy certificate if user needs additional privileges for his actions.
- Visual editors for jobs and tasks (single job is represented as a DAG of tasks) (fig. 2) – Directed Acyclic Graphs (DAG) are natively supported by GridNNN workload management. Web UI provides an easy way to edit them right in a browser window. Pre-created DAGs may also be uploaded and edited.

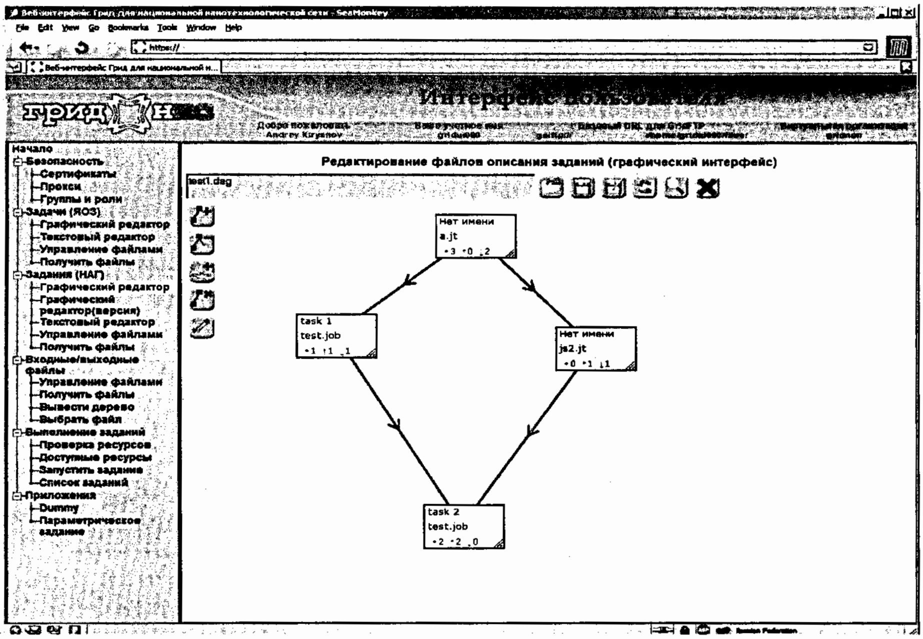
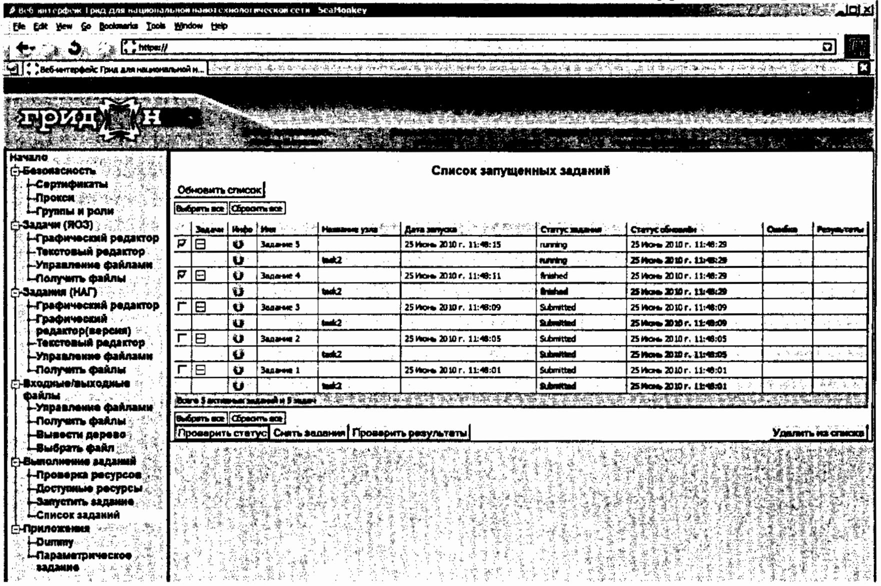


Fig. 2: Visual DAG editor

- File management with upload/download capability – Web UI provides some storage space for users where they can store job definition files and reasonable amounts of input and output data. User may browse, upload and download files right from the browser as well as using GridFTP client.
- Workload management (job control) – convenient visual representation of lists of available resources and submitted jobs (fig. 3). Group operations are supported for easy management of jobs. Detailed status information for each job and task may be displayed at user request.

- Integration with application software via plug-ins – special mechanism inside client part of Web UI allows specific application software to be integrated right into UI's main window. It allows easy creation of job and task descriptions based on application's parameters.



Список запущенных заданий

Обновить список

Выбрать все | Сбросить все

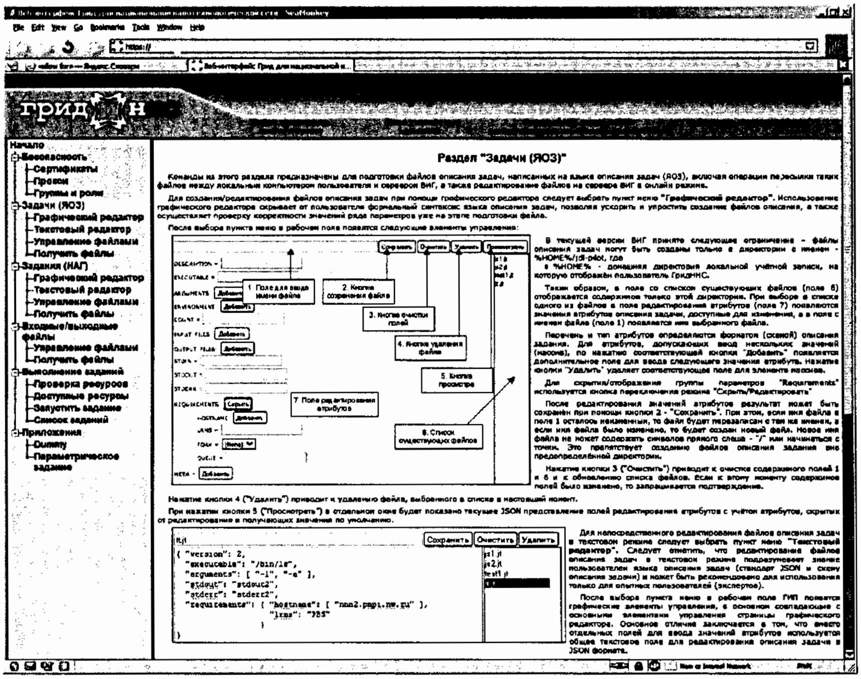
	Задачи	Инфо	Имя	Название узла	Дата запуска	Статус задания	Статус обновл.	Ошибки	Результаты
☑	⊟	●	Задание 5		25 Июнь 2010 г. 11:48:15	running	25 Июнь 2010 г. 11:48:29		
		●		task2		running	25 Июнь 2010 г. 11:48:29		
☑	⊟	●	Задание 4		25 Июнь 2010 г. 11:48:11	finished	25 Июнь 2010 г. 11:48:29		
		●		task2		finished	25 Июнь 2010 г. 11:48:29		
□	⊟	●	Задание 3		25 Июнь 2010 г. 11:48:09	Submitted	25 Июнь 2010 г. 11:48:09		
		●		task2		Submitted	25 Июнь 2010 г. 11:48:09		
□	⊟	●	Задание 2		25 Июнь 2010 г. 11:48:05	Submitted	25 Июнь 2010 г. 11:48:05		
		●		task2		Submitted	25 Июнь 2010 г. 11:48:05		
□	⊟	●	Задание 1		25 Июнь 2010 г. 11:48:01	Submitted	25 Июнь 2010 г. 11:48:01		
		●		task2		Submitted	25 Июнь 2010 г. 11:48:01		

Всего 5 активных заданий и 5 задач

Выбрать все | Сбросить все

Проверить статус | Снять задания | Проверить результаты | Удалить из списка

Fig. 3: List of jobs and tasks



Раздел "Задачи (ЯОЗ)"

Команды из этого раздела предназначены для подготовки файлов описания задач, написанных на языке описания задач (ЯОЗ), включая операции пересылки таких файлов между локальным компьютером пользователя и сервером ВИГ, а также редактирование файлов на сервере ВИГ в онлайн режиме.

Для создания/редактирования файлов описания задач при помощи графического редактора следует выбрать пункт меню "Графический редактор". Использование графического редактора скрывает от пользователя формальный синтаксис языка описания задач, позволяя ускорить и упростить создание файлов описания, а также осуществляет проверку корректности значений ряда параметров уже на этапе подготовки файла.

После выбора пункта меню в рабочем поле появятся следующие элементы управления:

В текущей версии ВИГ принято следующее ограничение - файлы описания задач могут быть созданы только в директории с именем - %HOME%/jdl-pilot, где

- %HOME% - домашняя директория локальной учётной записи, на которую отображён пользователь ГридННС.

Таким образом, в поле со списком существующих файлов (поле 6) отображается содержимое только этой директории. При выборе в списке одного из файлов в поле редактирования атрибутов (поле 7) появляются значения атрибутов описания задачи, доступные для изменения, а в поле с именем файла (поле 1) появляется имя выбранного файла.

Перечень и тип атрибутов определяются форматом (схемой) описания задания. Для атрибутов, допускающих ввод нескольких значений (массив), по нажатию соответствующей кнопки "Добавить" появляется дополнительное поле для ввода следующего значения атрибута. Нажатие кнопки "Удалить" удаляет соответствующее поле для элемента массива.

Для скрытия/отображения группы параметров "Requirements" используется кнопка переключения режима "Скрыть/Редактировать"

После редактирования значений атрибутов результат может быть сохранён при помощи кнопки 2 - "Сохранить". При этом, если имя файла в поле 1 осталось неизменным, то файл будет перезаписан с тем же именем, а если имя файла было изменено, то будет создан новый файл. Новое имя файла не может содержать символов прямого слеша - "/" или начинаться с точки. Это препятствует созданию файлов описания заданий вне предопределённой директории.

Нажатие кнопки 3 ("Очистить") приводит к очистке содержимого полей 1 и 6 и к обновлению списка файлов. Если к этому моменту содержимое полей было изменено, то запрашивается подтверждение.

Нажатие кнопки 4 ("Удалить") приводит к удалению файла, выбранного в списке в настоящий момент.

При нажатии кнопки 5 ("Просмотреть") в отдельном окне будет показано текущее JSON представление полей редактирования атрибутов с учётом атрибутов, скрытых от редактирования и получающих значения по умолчанию.

t.jt

```
{ "version": 2,
  "executable": "/bin/ls",
  "arguments": [ "-l", "-a" ],
  "stdout": "stdout2",
  "stderr": "stderr2",
  "requirements": { "hostname": [ "nnn2.pnpi.nw.ru" ],
                    "lrms": "PBS"
                  }
}
```

Для непосредственного редактирования файлов описания задач в текстовом режиме следует выбрать пункт меню "Текстовый редактор". Следует отметить, что редактирование файлов описания задач в текстовом режиме подразумевает знание пользователем языка описания задач (стандарт JSON и схему описания задачи) и может быть рекомендовано для использования только для опытных пользователей (экспертов).

После выбора пункта меню в рабочем поле ГИП появятся графические элементы управления, в основном совпадающие с основными элементами управления страницы графического редактора. Основное отличие заключается в том, что вместо отдельных полей для ввода значений атрибутов используется общее текстовое поле для редактирования описания задачи в JSON формате.

Fig. 4: Built-in help

Most modern browsers on all available platforms are supported. Data channel between client and server is always encrypted making it virtually impossible for the third party to interfere with it.

Web interface serves as a single entry point for a user in GridNNN, where he can store files, job and task descriptions. Job state information is retained between browser sessions making it easy for a user to control his jobs from different locations.

Client authentication is done by Apache's mod_ssl using X.509 certificate loaded in client's browser. After first-time successful authentication client is allocated an account from a pool and automatically gets a GridFTP access to its home directory.

Web interface is supplied with built-in help pages available from the main menu (fig. 4). Multiple Grid back-ends are supported by the Web interface architecture. During development, it was successfully used with gLite, Unicore, Gridway and Pilot (GridNNN workload manager).

While still being in active development Web UI may already be used to access real GridNNN resources. Some new features will be added before the end of the project including:

- DAG execution visualization
- Native support for iterative tasks
- Integration with GridNNN monitoring

References

[1] GridNNN: http://ngrid.ru/

DESKA: TOOL FOR CENTRAL ADMINISTRATION OF A GRID SITE[1]

T. Hubik, L. Kerpl, M. Krejcova, J. Kundrat

Institute of Physics of the AS CR, Prague, Czech Republic

Running a Tier-2 WLCG site is a demanding task, especially given the ever increasing number of machines, auxiliary hardware and accompanying software packages. System administrators have always tried to find ways to reduce their efforts while not compromising systems' security and reliability. In this paper, we focus on reducing the duplication of information describing a complete datacenter, including the relations between various components. We provide details about the Deska project, an in-house CLI appliance for managing these metadata.

1. Introduction

Recent developments in the hardware area are bringing unmatched challenges to datacenter operators. The ever increasing computational power density, along with advancements in semiconductor engineering and mass production, enables integrating many hundreds of CPU cores into a single rack while maintaining reasonable TCOs. Machines which formerly occupied several rack units are now produced in tiny blade form factors. That all leads to a rapid rise of a number of services which have to be managed and taken care of.

This rapid rise is, however, usually unmatched by a corresponding growth in the size of the IT department. A concrete example about the situation in Prague is presented in a site report given by Tomas Kouba [1]. Therefore, system administrators seek ways to mitigate many of the common problems they are facing. One way of possible problem mitigation scenario is making sure that all information one might need is conveniently accessible, kept up to date and actually used in production.

The design of such a system has to fulfill certain requirements, from being easy to use to scaling well with growing number of nodes. Most of the existing hardware inventory systems were deemed unusable for production use at Prague, for reasons provided later in the article. Therefore, a new tool called Deska was designed, and its architecture and design considerations are provided in this document. The goal was to provide a tool which would integrate with the existing infrastructure as smoothly as possible, yet be universal enough to keep up with future demands. We have to merge best practices from traditional relational database design with data versioning, conflict resolution and change set merging.

When the Deska project is mature and tried in production, we have to focus on possible ways to further integrate it with day-to-day operation. As an example, a reasonable way of putting the data to good use is enabling a one-action deployment or re-installation of new machines or using the Deska's knowledge of physical features of the machines to visualize the server room and provide power usage estimates.

2. The way to automation

Since its establishment in 2002, the Prague Tier-2 WLCG site has grown from 32 single-core machines into the 2010's biggest site in the Central Europe, offering their users more than 2600 modern CPU cores. This capacity, hosted by the Institute of Physics of the AS CR, is nowadays dedicated to performing heavy-duty HPC computations, supporting both in-house users from particle and solid-state physics, as well as to taking part in international collaboration for experiments in FNAL, USA and CERN.

[1] The work on the Deska project was partially funded by the Institute of Physics of the AS CR, v.v.i. The author of this poster can be reached at kundratj@fzu.cz.

Such a rapid rise in computing capacity was not achieved just by hardware advancements, but is also a result of deploying more and more machines. In fact, the Prague data center occupation was risen several times, from the original 32 machines to its current status of 336 computing nodes and tens of auxiliary or service nodes.

This increase in the computing capacity was not, however, compensated by a matching grow in staffing operators. While the original size was handled by three full-time employees, the current situation is actually a bit worse. The total contracted capacity is slightly more than three full-time-equivalents (FTE), but only two people are dedicated to day-to-day operations, and the rest consists of three students, each with 0.25 to 0.5 FTE employments. Some assistance is provided by other section from the Institute, but it should be obvious that the amount of work and responsibility that each of the employees has to bear has risen considerably over the years.

A naive manager might consider such situation a pleasant one - the people should not be slacking while at work, after all. However, due to the nature of the IT industry, it is in fact desired not to operate the staffers at full utilization levels all the time. When they do not have time for reading newspapers and are kept busy with day-to-day operation, these duties will inevitably suffer when an emergency arises. Therefore, the administrators simply *have* to have free time in order to handle unexpected situations.

In their quest to make their life easier, people at the Institute have deployed many tools which were supposed to automate the day-to-day operation. At first, in-house scripts were used for performing changes to the nodes, typically in the form of a shell for loop. This approach did not scale at all, and required manual supervision for completion. It also caused trouble when some nodes were offline at the time of the change, perhaps because of a hardware trouble. When such host got powered up again, the change wasn't performed on it, and there was nothing to remind the responsible people about that.

Therefore, yet more in-house scripts were written to look after the above mentioned scripts' execution. The scalability problems were not mitigated at all, and there were many corner cases not really fully understood, nor properly addressed. Therefore, a way to migrate to an "intelligent solution" was clearly needed.

We evaluated numerous tools, from Cfengine version 2 and Puppet to specialized applications developed inside the HEP community like Quattor. Ultimately, we decided that Cfengine would fit our goals in the best way, and therefore chose it. A proper tool for automation should ideally deal with random host changes and just check the final state to see if it matches the specified contract. For example, a correct way of converting a machine to use an LDAP database for user accounts, as done by Cfengine, is the following:

```
{ /etc/nsswitch.conf
  BeginGroupIfNoLineMatching ^passwd:([[:space:]])+files([[:space:]])+ldap$"
        HashCommentLinesStarting "passwd:"
        Append "passwd: files ldap"
  EndGroup
}
```

When compared to a more-usual sed magic, it is obviously more verbatim, but also arguably much more readable. Using Cfengine properly certainly takes some time and effort, but the results should pay off pretty soon - the configuration can be stored in a single place which could easily be put under a version control system (VCS) like Subversion or Git, and Cfengine's design ensures that a change, or rather a description of a desired *state* specified once will be verified for eternity, and if it can't be reached, an alarm would be risen.

Using Cfengine along with LDAP for centralized management of user accounts and Kick-start/Anaconda for new hosts deployment cut a big part of the burden from the sysadmins' shoulders, but still left much to be dealt with. The most annoying problem, as perceived by the Institute's staffers, is the need to specify the same thing on several places.

3. Information duplication

In order to better illustrate what we mean with duplication of information, let us consider a typical use case of new machine deployment. After a procurement is finished, the technicians from the winning bidder arrive on site along with the actual hardware. The machines and supporting technology get physically installed, cables are laid out and the electricity is switched on. Staffers have to write down certain information, from rack locations to part numbers and warranty information. After that is done, the network switches have to be properly configured, the MAC addresses previously gathered have to be entered into the DHCP server's database, IP addresses and matching hostnames have to be assigned and registered in the DNS system etc. When the machines are turned on for the first time, a proper Kickstart file has to be delivered over the PXE environment, which has to be set up properly. When the operating system is installed, a configuration management system, Cfengine in our case, has to be told which role the machine is going to fulfill, so that it can configure everything accordingly. Last but not least, various monitoring appliances have to be made aware of the new machine's existence, so we get to know about possible problems before we suffer availability or reliability loss, or our users notice.

Therefore, the information about a host's existence is needlessly duplicated into several places. In the case of the Prague's Tier-2 site, some of these places are the following:

- HW inventory DB;
- Warranty & issue tracking;
- Switch port configuration;
- DHCP server;
- DNS;
- Cfengine roles;
- Torque's CPU multipliers;
- MRTG & RRD network graphs;
- Nagios;
- Ganglia;
- Munin;
- ...

It is clear that this duplication will only lead to troubles over time. It is not entirely obvious that an innocent operation like warranty replacement of a motherboard shall result in changing the HW inventory database, the warranty database, the switch configuration, the DHCP server, and also the network intrusion detection system. What is needed here is a *central place* to store all host metainformation.

When such a place is ready, we have to put it to real use, too. Patching existing services to query this place, or a database, is likely not an option, so tools have to be written for pushing this information to places like network switches or DHCP servers' configuration. A slight delay has to be anticipated, but in reality it does not complicate things much. Distribution of these configuration files is outside of scope of the Deska project, and will usually be handled by standard tools like Puppet or Cfengine.

4. Requirements

Based on their experience with another in-house attempt at a central inventory database, a web-based tool, the staffers decided to assign the biggest importance to the ease of use of the new tool. As all of the staffers are also heavy Linux users, they tend to prefer the command-line interface to more traditional web-based ones.

Numerous situations have shown that having a textual dump of a database is of much experience. There are simply times when everything is down, perhaps as a result of fatal electricity incident or on-location fire. When the network is offline, having a full copy of the database, perhaps without much history, but at least with all the data, is a hard requirement. If such a dump resides on each administrator's laptop's hard drive, chances are that people will be able to consult up-to-date information when performing critical services recovery, reducing the possibility of making fatal mistakes and causing more damage by accident.

The Cfengine experience illustrated, among other things, the usefulness of keeping track of history of changes. When it is possible to identify what changes got performed by whom and what was their reason, people tend to make less mistakes, and, what is most important, they are less likely to repeat them.

Finally, while most web-based tools would work in an "online" mode where all changes get pushed to the production repository immediately, or perhaps after pressing the "go live" button, a two-phase commit with manual review of changes is much more common in the realm of version control systems like Subversion or Git. Under such system, changes are permanently copied only after you have seen them one more time, usually as a diff, a difference report between the former and the new version. In the case of the inventory database, it is worthwhile to display changes in the database along changes in various services' configuration.

5. Deska design and operation

The design of the Deska's UI was inspired by the way datacenter Ethernet switches are configured, most notably by the Cisco IOS shell. Therefore, the user is presented with a command line interface sporting intelligent tab-completion, and they are supposed to navigate through a hierarchy of objects, similar to what she would do when configuring a port of the switch.

These objects model a real-world entities which are found in a typical machine room, as well as abstract entities representing concepts like a "brand" or a "model" of a particular machine. A general rule of thumb is that before one can instantiate a "physical object", a matching "type" has to be defined - for example, before one adds fifty Altix XE320 WNs, the xe320 hardware model has to be defined.

To date, the following *abstract* entities are supported:

boxmodel A type of a physical box, like a room, universal rack, blade enclosure or a twin server chassis,

hwmodel A type of a HW box, from generic types like a 1U server to well-specified entities like "HP DL360".

Matching the above, the corresponding *instances* of a specified kind are:

rack An instance of a boxmodel. Contrary to what the name might suggest, this statement could represent anything from a machine room to physical rack, one concrete instance of a blade enclosure or even one of many twin server chassis which are deployed in a real-world rack,

host An instance of a physical computer. This computer is always a physical device, and runs exactly one operating system image. Support for virtualized machines is available, but not discussed in this document.

In order to better illustrate the above concepts, and to provide a real-life example about how they are used, consider the following snippet of the configuration of the Prague site:

```
# At first, we define a type representing plain room in a building
boxmodel room
        outer width 1000000
        outer height 1000000
        outer depth 1000000
end

# Next, we define a template for a regular rack:
boxmodel generic-rack
        # ...it is supposed to be placed in a room defined above
        in room

        # Because the outer element, ie. ``room" in this case, does not define any bays,
        # we specify physical dimensions:
```

```
        outer width 700
        outer depth 1300
        outer height 2000

        # Contrary to ordinary rooms, a rack is usually divided into areas with well-defined boundaries.
        # Let's call them ``bays" here and specify how they are organized:
        bay height 44 order bottom-up start-at 1
        # ...which is syntactically equivalent to:
        # bay
        #        height 44
        #        order bottom-up
        #        start-at 1
        # end
        # ...and after adding default values, to this:
        # bay
        #       height 44
        #       width 1
        #       depth 1
        #       order bottom-up
        #       start-at 1
        # end
end
```

We have seen that there are many syntactic ways to express the same concept; we hope that such an approach is not confusing.

In the real world, certain models are a *specialization* of other items. We have already defined a generic rack with some default dimensions. Suppose a new delivery comes in racks with slightly different dimensions:

```
# A slightly bigger rack...
boxmodel APC-rack
        # ...is a specialization of the generic-rack...
          template generic-rack

        # ...slightly taller...
           outer height 2300

        # ...and all other properties, including the bay organization, are inherited from
        # its parent, the "generic-rack"
          bay height 48
end
```

This specialization is called a **template.** A template works simply by inheriting all properties from the parent item, with the possibility to override each and every one of them.

The reason for differentiating between a *type* of an object and *properties* of one particular instance is to force some consistency on how the operators work with the database. If it was allowed to change arbitrary properties on the *instance* level, people would likely abuse that possibility to avoid using constructs like the boxmodel altogether.

Now that we have defined our abstract building blocks, let us see how to instantiate the real objects which are found in the machine room:

```
# Start with the "room", which is not derived from anything
rack machine-room
      model room
end

# Add real racks now
```

```
rack L01
        model APC-rack
        in machine-room # a reference to a "real object" defined above

        # "machine-room" is of instance "room" which does not provide any bays,
      # therefore we have to fall back to absolute positioning:
      position x 10 y 20 z 0
end

# As a further example of enclosure nesting, suppose this blade chasses:
rack hp-enc-p-1
        model hp-blade-p-chassis # ...this would have to be defined somewhere

         # This instance is a blade enclosure which fits to a rack. We also know its
         # dimension in rack units (ie. we know the number of occupied "bays in a
         # rack" at this point), we only have to place it somewhere. The placement is
         # done via a "significant corner", which is the bay number of the lowest,
         # left-most and front-most bay occupied by the box.
         in L01 bay 10
end
```

We have already demonstrated how to define a boxmodel and a rack. Matching counterparts for the physical machines are hwmodel and a host. We'll start by showing how to use the hwmodel, along with yet another template demonstration, and follow by a host definition:

```
hwmodel 1u
        in generic-rack
        # no bay occupation indicated -> default to "1 bay"
end

hwmodel dl360g5
        template 1u
end

hwmodel dl140
        template 1u
        benchmark hepspec 8
        cpu ‘'Intel Xeon 3.06 GHz”
        sockets 2
        cores-per-socket 1
        logical-cores-per-physical-core 2
        disk “ATA 80GB”
        power 110W
end

host ha1
        hw dl360g5
        serial 1234567890
        rack L01 bay 0
end
```

As we can see, even though certain vital information, like the network layout, is still missing from the table, we already have all the data needed for drawing a "map" of where machines are located.

In order for the database to be more valuable and to be able to be used as a source of data for various services, it is crucial to introduce network definitions, too. It is done in the following way:

```
network wn-nat
        vlan 172
        ip 172.16.0.0/16
```

```
end
```

The purpose of maintaining these data is to be able to perform certain checks on data validity, like verifying that a host's IP address belongs to the assigned range. Now let us introduce a first real server:

```
host hypericum10
        # Housekeeping records
        # Serial No. of the whole box
        # serial J019MF6C2J
        # Warranty No. & expiration date
        warranty KE1421631007 expires 2006-12-12
        purchased 2004-05-03

        # Physical dimension, architecture, location and interconnects
        hw dl140
        rack L04 bay 40
        kvm unit 4 port 7
        interface eth0 mac 00:0f:20:7a:e7:9c net wn-nat ip 172.16.4.10 switch swL041 port 5

        # Logical roles
        role wn
end
```

This concept of lists of object of well-known types can be extended to include other entities commonly found in a datacenter, should the need arise.

6. Comparison to existing tools

To the best of our knowledge, the Deska project is the only tool which focuses on a CLI interface. There are indeed numerous other applications, from the "industrial standard" OCS Inventory to less known projects like the RackMonkey, as well as appliances developed inside the WLCG community, like Quattor fabric management tool or the Smurf database.

In addition to a missing CLI, the mainstream way of tackling the inventory management problem is usually focused on automatic discovery of existing properties. This is a very useful approach when working with desktop computers, which often change outside of the network manager's approval, but could be less optimal when working in a well-controlled datacenter environment. In addition, while it could be desirable to automatically pick certain changes in the environment and propagate them to the database, at least under certain conditions, the Deska tool is meant to provide an *authoritative source* of information - what is in the database is the desired situation, representing the sysadmins' decisions, and should be implemented in the real world and not vice versa.

7. Future work

The design described so far presented the database scheme which was pretty much set in stone. We recognize that each site's requirements are different, and have ourselves struggled with providing a good approximation of the real situation in Prague and translating it to the database structure. An intermediate solution for a request for the possibility of customization is to allow the administrator to specify certain additional attributes to attach to existing types of entities in the database. However, such an approach would make expressing constraints rather difficult, at least due to the extension fields' unspecified structure.

The considerations above led us to another approach within the Deska application. In future versions, we are going to make the DB structure modifiable by the administrator. We will retain the basic idea that the database consists of lists of objects of a well-defined type, and that each such object has a list of attributes, or properties, of one of a few predefined types. Such objects are entities like the rack or a boxmodel defined above. The definitions of such objects, along with a list of supported attributes and their data types, should be easily provided by the administrator. The Deska project will of course ship with a predefined set of objects which are common in the datacenter.

In consistency with the Deska's motto of reducing duplication of information, we are seeking ways to define the DB scheme at exactly one place, along with all possible constraints. We do not have any motivation to invent yet another language for describing the database, but plan to use SQL DDL as the way to define the DB structure, along with some rules for coding conventions. Our goal is not to create an "¨uber Database" capable of describing each and every situation, but to come up with reasonable limits allowing the administrators to describe their infrastructure accurately.

8. Conclusion

We have described some of the current griefs which the staffers at the Institute of Physics of the AS CR are going through, along with a proposed solution which could reduce this burden. We have learned valuable lessons when trying to design a proper database structure, using the Prague Tier-2 WLCG site as an example. The database structure which we picked is centered on lists of objects of a few predefined types.

During the implementation works, we have discovered that it makes a lot of sense to allow the administrators to extend the database structure. Therefore, most of the hard-coded limitations of the original Deska design were lifted. The Deska will now read the database structure at the startup and attempt to reconstruct the valid object types and their attributes. In practice, such an approach to the problem helps to eliminate a lot of boilerplate code. What should be done, however, is defining a reasonable set of constraints for the administrators to follow.

References

[1] Kouba, T. et al. The Prague WLCG Tier-2 Report. June 2010, Grid'2010 Conference, JINR Dubna.

ON AUTOMATION OF MONTE CARLO SIMULATION IN HIGH ENERGY PHYSICS

D. Kekelidze[1], S. Belov[1], L. Dudko[2], A. Sherstnev[2,3]

[1]Joint Institute for Nuclear Research, Dubna, Moscow region, Russia
[2]Scobeltsyn Institute of Nuclear Physics, Moscow State University, Moscow, Russia
[3]R. Peierls Centre for Theoretical Physics, University of Oxford, Oxford, UK
Dmitry.Kekelidze@cern.ch

Monte Carlo simulation and its common problems

Every experiment in high energy physics requires Monte Carlo simulation to properly design detectors and process obtained data in future. The chain of simulation usually consists of ME generators (partonic level), SH generators (showering, hadronic level) and detector simulators (GEANT, etc.). Monte Carlo process becomes technically more complicated, so the process of generation requires a lot of time and human resources and is exposed to mistakes due to human factor. Also, tuning and usage of some of the generators require expert knowledge. The way to avoid all those problems is to properly store and document the simulated events, so they can be used in future.

Another problem is that different groups of physicists all over the world need the same simulated events very often. Making the stored samples publicly available helps to speed up the checking of theoretical models or detector configuration.

This is the goal of the MCDB project [1,2] in a compartment with HepML language. It allows to share event samples already generated by experts between different groups of physicists. Events stored in LCG MCDB [1,2] are properly described as a separate articles, so they can be easily found and used.

Conception of completely automated Monte Carlo simulation chain

Firstly events are generated with a Matrix Element MC generator, stored in knowledge database, processed with a Shower and Decay MC generator, and then passed to the Detector simulation software. Handling all the data in automatic way allows a user to avoid most of human-related errors and can save researcher's time and keep simulated event files in order.

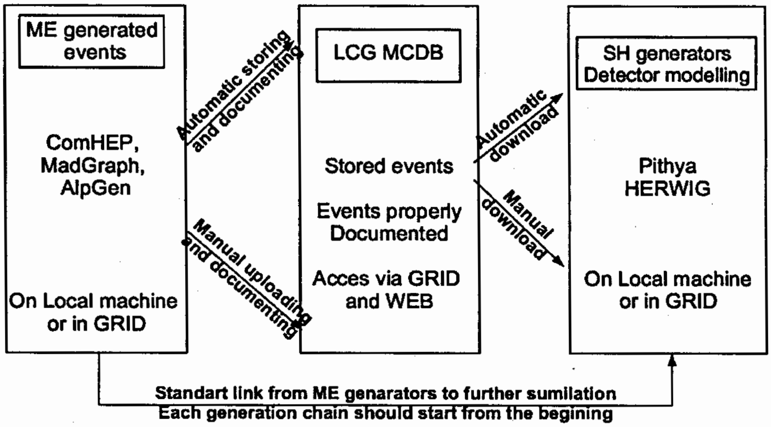
Firstly, data from a Matrix Element generator are kept in the standard LHE format. Detailed sample's description in HepML format is included to the Les Houches Event Format (LHEF) [3] header. After this step MC generator provides self documented event sample with the full description of simulation inside it. At the moment, CompHEP [4] generator can provides extended information in the form of HepML code inside the standard LHEF header. The next step is to store the sample in a public place. LCG MCDB allows automatic upload and documentation (for several types) of such event files. Then, the events can be taken via a Grid interface or directly from CASTOR at CERN or through the web interface.

After getting the files, they could be transferred to the next generation level, Showering and Hadronization, along with the full meta-data set. Then, stored, for example, in the HepMC format, the data can be processed by Detector Simulation software.

Brief description of MCDB

The facilitated communication in the LHC collaborations between experts and authors of MC generators and users of the programs has been created with LCG MCDB. Now MCDB provides a flexible way for storing, bookkeeping and accessing the MC simulated events. It has Web-based interface, which can be used both for storing and describing new samples and accessing already stored ones. Also, there is a way to store simulated events using perl scripts. The stored event files are accessible in numerous ways, such as AFS, CERN CASTOR, Grid. The interface also allows to store graphical files - products of Pythia [5] diagrams for example.

The LCG MCDB helps to automatize the simulation chain. The events, which have to be simulated using ALPGEN [6], CompHEP, or MadGraph [7] each time they are required, now stored in the MCDB and properly documented. The amount of information in event's description makes them easy to find. As they can be accessed via Web or Grid, they can be used for further simulation avoiding the passage of the entire simulation chain from the very beginning. Availability of automatic access to stored events also helps saving user's time. On the Fig. 1 is shown simplified scheme of the MCDB part in simulation chain.



ME generated events
ComHEP, MadGraph, AlpGen
On Local machine or in GRID
Automatic storing and documenting
Manual uploading and documenting
LCG MCDB
Stored events
Events properly Documented
Acces via GRID and WEB
Automatic download
Manual download
SH generators Detector modelling
Pithya HERWIG
On Local machine or in GRID
Standart link from ME genarators to further sumilation
Each generation chain should start from the begining

Fig. 1: Bottom line shows standard way of simulation. Above MCDB block is between ME and SH generators

HepML markup language

To unify the way of stored events description a new markup language required to be designed. As the base an XML [8] markup language has been selected. Beyond the benefits of such choice is availability to use sachems, so the stored information can be easily parsed using any XML parser. As the result an HepML [9] language has been created. It provides a flexible way to store all information of MC generated events and generator setup information. Using schemes also provides an easy way to extend the possible data to be stored. Now HepML description header is an allowed part of LHEF file format.

External library to work with files in HepML format

In addition to on-line access to stored events, LCG MCDB provides an external library to create, parse and modify HepML descriptions of MC event files. The library wrote in C++ language and can be used in C++, C and Fortran programs. The main purposes of it is availability to create HepML description or get parsed one as a C++ objects, so they can be used in further processing. Another ability to use *libhepml* [9] is mixing few HepML files into a new one. Fig. 2 shows *libhepml* place in simulation chain.

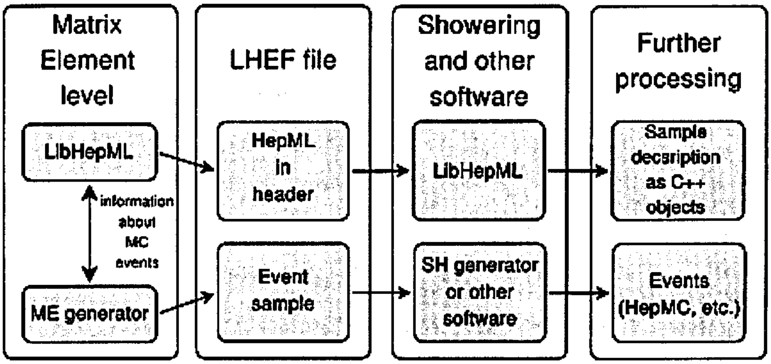


Fig. 2: Place of *libhepml* in simulation chain. HepML header in LHEF file can be parsed or created with *libhepml*

Current usage of the LCG MCDB

Now LCG MCDB is actively used by CMS experiment on the LHC. Uploaded to MCDB files are accessing via CASTOR and then processing by the internal CMS software called CMSSW.

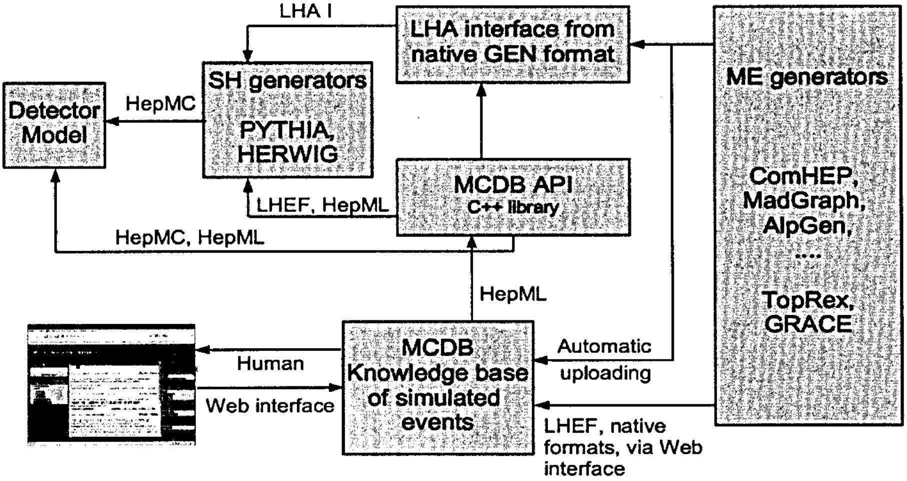


Fig. 3: Usage of LCG MCDB by CMS internal software

Conclusion

The LCG MCDB in a compartment with HepML language provides a way to automatize MC simulation chain by storing simulated event files with proper description. Access to stored files via Web and Grid provides a flexible way of using them in further simulation. Such automation could give prominent acceleration for gaining of scientific results and helps in saving human resources and their time.

References

[1] LCG MCDB project site, http://mcdb.cern.ch

[2] Belov S. et al. LCG MCDB - a Knowledgebase of Monte Carlo Simulated Events, Computer Physics Communications, Volume 178, Issue 3, 1 February 2008. P. 222 [hep-ph/0703287]

[3] Alwall J., et al., Comput. Phys. Comm. 176 (2007) 300, arXiv:hep-ph/0609017.

[4] Boos E. et al. CompHEP Collaboration, Nucl. Instrum. Methods A 534 (2004) 250, arXiv:hep-ph/0403113.

[5] Sjostrand T. arXiv:0809.0303 [hep-ph].

[6] Mangano M.L., Moretti M., Piccinini F., Pittau R., Polosa A.D. JHEP 0307 (2003) 001, arXiv:hep-ph/0206293.

[7] Maltoni F., Stelzer T. JHEP 0302 (2003) 027, arXiv:hep-ph/0208156.

[8] Bray T. et al., http://www.w3.org/TR/REC-xml

[9] Belov S., Dudko L., Kekelidze D., Sherstnev A. HepML, an XML-based format for describing simulated data in high energy physics, Computer Physics Communications, 2010. doi:10.1016/j.cpc.2010.06.026.

TESTING OF PRODUCTION CEs IN GLITE MIDDLEWARE SUITE

A. K. Kiryanov, N. V. Klopov, Yu. F. Ryabov

Petersburg Nuclear Physics Institute, Russia, 188300, Gatchina, Orlova Roscha.
Tel.: +7 (81271) 46197, fax: +7 (81271) 35270, e-mail:knv@omrb.pnpi.spb.ru

Introduction

The aim of this work was to estimate a performance and to compare the results for two major production CEs currently available in gLite middleware suite: LCG-CE and CREAM-CE [1, 2]. Performance was measured in terms of registered jobs per second on CE during a simulation of a real-life usage pattern. Testing was performed on PNPI testbed with the following configuration: WMS – Core 2 Duo, 4 GB RAM; CE – Core 2 Duo, 2 GB RAM; 2 WNs – Pentium IV, 2GB RAM.

Test description

We estimated the performance of the CE in term of throughput (number of registered on CE jobs/s). To do so, we submitted 1000 identical jobs to CE using the credentials of one or ten different users (each user submits 100 jobs). Tests were run in two different modes:

- WMS submission of 1000 jobs from one or ten users to CREAM-CE and LCG-CE,
- Direct submission of 1000 jobs from one or ten users to CREAM-CE.

After test run, timestamps of all important events about job life were retrieved from WMS or CREAM-CE (for direct job submission case) and test results in graphic and tabular forms were generated. The job registration rate was evaluated by analyzing the 'Transfer to LRMS' events and 'Registered' events when testing CEs through WMS and direct respectively (Fig.1).

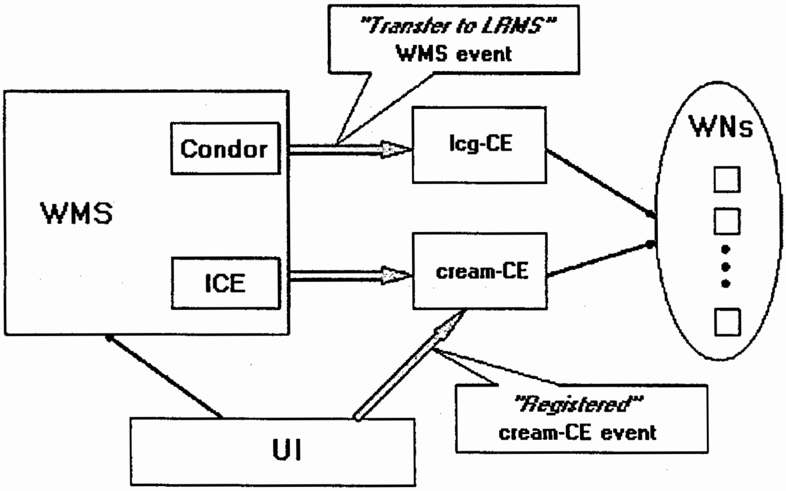


Fig.1: Test structure

The job submission strategy when testing was influenced by two following factors:

1. Submission of simple jobs in one stream cannot produce any significant load neither on WMS nor on CE, which led us to the choice of the using parametric jobs. Parametric job allows submission of a bunch of jobs in one shot. The result of the comparison of the job submission rate for

simple and parametric job submission demonstrates that the job submission rate for parametric job is in 8 times higher than for simple jobs (4.1 job/sec and 0.5 job/sec).

2. Usage of automatic proxy delegation (-a option in glite-wms-job-submit and glite-ce-job-submit) slows down job registration a lot. This fact led us to the choice of the using explicit proxy delegation when user explicitly delegates a proxy before the first job is submitted, and uses the same delegation ID for all subsequent submissions. The result of the comparison of the job registration rate for the automatic and explicit proxy delegation demonstrates the acceleration of the registration rate in 4 times (0.4 job/sec and 1.5job/sec).

Testing CREAM-CE and LCG-CE

Procedure of the testing CREAM-CE and LCG-CE include:

- Direct job submission to cream-CE from 1 and 10 users;
- Parametric job submission to cream-CE through WMS from 1 and 10 users;
- Parametric job submission to lcg-CE through WMS from 1 and 10 users;
- One-second sleep script was used as a test job for all submissions;
- Load averages of the cream-CE and lcg-CE were measured and compared;
- Job registration rates on CEs were measured and compared.

The results of the testing for the direct job submission to cream-CE for 1 and 10 users are presented in Tables 1-2.

Table 1. cream-CE: 1000 jobs from 1 user (cream-CE events)

Event	Performance (jobs/sec)	Time (sec)		
		Start	Finish	Total
Registered	**1.72**	0	582	582
Idle	1.72	5	587	582
RealRunning	1.15	8	874	866
Done	1.15	9	875	866

Table 2. cream-CE: 1000 jobs from 10 users (cream-CE events)

Event	Performance (jobs/sec)	Time (sec)		
		Start	Finish	Total
Registered	**1.55**	0	647	647
Idle	1.54	4	653	649
RealRunning	1.15	7	873	866
Done	1.15	8	874	866

The results of the testing for parametric job submission to cream-CE through WMS from 1 and 10 users are presented in Tables 3-4.

Table 3. cream-CE: 1000 jobs from 1 user (WMS events)

Event	Performance (jobs/sec)	Time (sec)		
		Start	Finish	Total
RegJob	**27.03**	**0**	**37**	**37**
Match	2.04	41	531	490
Transfer,LRMS	**1.58**	**45**	**676**	**631**
Run, LRMS	0.80	55	1300	1245
Run, LogMon	1.23	360	1375	815
Done, LRMS	0.80	58	1303	1245
Done, LogMon	1.23	561	1376	815

Table 4. cream-CE: 1000 jobs from 10 users (WMS events)

Event	Performance	Time (sec)		
	(jobs/sec)	Start	Finish	Total
RegJob	**4.12**	**0**	**242**	**242**
Match	1.22	14	835	821
Transfer,LRMS	**1.22**	**18**	**838**	**820**
Run, LRMS	0.80	25	1271	1246
Run, LogMon	0.83	153	1349	1196
Done, LRMS	0.80	27	1274	1247
Done, LogMon	0.84	155	1349	1194

Analysis of the "Registered" events for direct job submission and "Transfer, LRMS" for the WMS case show the following:

- The job registration rate slows down when jobs are submitted from 1 and 10 users;
- The job registration rate is higher for the direct submission in comparison with job submission through WMS.

The results of the testing for the parametric job submission to lcg-CE through WMS are presented in Tables 5-6.

Table 5. lcg-CE: 1000 jobs from 1 user (WMS events)

Event	Performance	Time (sec)		
	(jobs/sec)	Start	Finish	Total
RegJob	35.71	0	28	28
Match	4.02	80	329	821
Transfer,LRMS	**0.83**	**92**	**1304**	**820**
Run, LRMS	0.36	137	2902	1246
Run, LogMon	0.34	285	3203	1196
Done, LRMS	0.36	140	2904	1247
Done, LogMon	0.31	360	3607	1194

Table 6. lcg-CE: 1000 jobs from 10 users (WMS events)

Event	Performance	Time (sec)		
	(jobs/sec)	Start	Finish	Total
RegJob	5.46	0	183	183
Match	2.00	50	549	499
Transfer,LRMS	**0.72**	**67**	**1461**	**1394**
Run, LRMS	0.35	125	2989	2864
Run, LogMon	0.33	220	3292	3072
Done, LRMS	0.35	128	2991	2863
Done, LogMon	0.30	375	3718	3343

As well as in the previous case the registration rate depends on the number of users. Table 7 shows the registration rates for all the experiments.

Table 7. Job registration rates

	cream-CE		lcg-CE
N of users	Direct	WMS(ICE)	WMS(Condor)
1	1.72	1.58	0.83
10	1.55	1.22	0.72

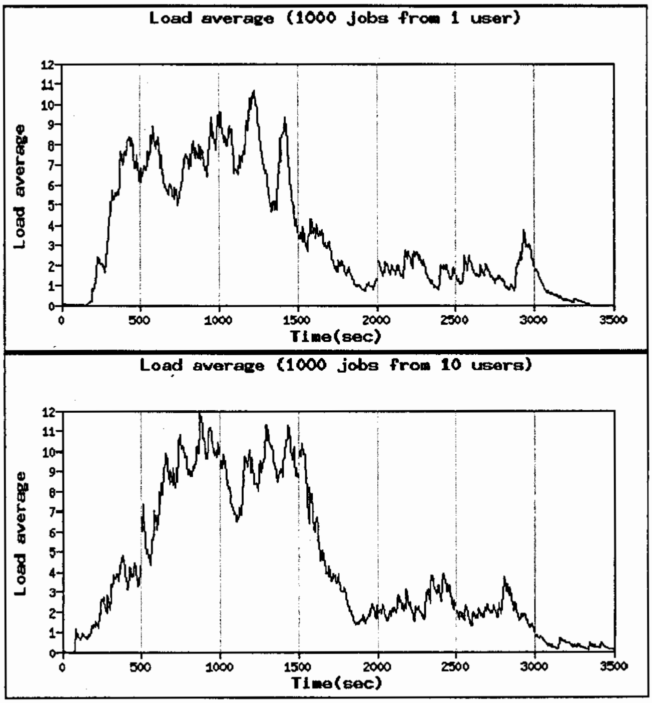
We observe that the registration rate slows down when jobs are submitted from different users for cream-CE and lcg-CE. Also on cream-CE the job registration rate ~2 time higher than on lcg-CE.

Monitoring of the CE load average

During the tests CEs were monitored for load average. A special script was run on CEs that recorded system load average every 5 seconds. Example of the load average monitoring is presented on Fig. 2. Table 8 shows the load average for all the experiments. We observe that the load average on cream-CE does not depend on the number of users whereas on lcg-CE it increased on 15% for the 10 users case.

Table 8. Max load average during testing

N of users	cream-CE		lcg-CE
	Direct	WMS (ICE)	WMS (Condor)
1	8	9	10
10	8	9	12


Load average (1000 jobs from 1 user)

Load average (1000 jobs from 10 users)

Fig 2: Load average for lcg-CE

Conclusion

The results of the performance testing of the CREAM-CE and LCG-CE currently available in gLite middleware suite shown that the job registration rate depends on the number of users to submitted jobs and it slows down when the number of users increased at the both CEs. Also the CREAM-CE shows better performance than the LCG-CE. It was also observed that load average of CREAM-CE does not depend on the number of users unlike on LCG-CE.

References

[1] gLite home page http://glite.web.cern.ch/glite/

[2] Aiftimiei C. et al. Future Generation Computer Systems, Volume 26, Issue 4, April 2010, Pages 654-667.

[3] Aiftimiei, P. et al. INFN Technical Note. INFN/TC_09/3. May 5, 2009.

GRID ACTIVITIES AT THE JOINT INSTITUTE FOR NUCLEAR RESEARCH

V. Korenkov

Joint Institute for Nuclear Research, Dubna, Russia

1. Introduction

For more than last ten years, staff members of the Joint Institute for Nuclear Research have been actively involved in the study, use and development of advanced grid technologies. The most important result of this work was the creation of a grid infrastructure at JINR that provides a complete range of grid services. The created JINR grid site (T2_RU_JINR) is fully integrated into the global (world-wide) WLCG / EGEE infrastructure. The resources of the JINR grid site are successfully used in the global infrastructure, and on indicators of the reliability, the T2_RU_JINR site is one of the best in the WLCG / EGEE infrastructure.

A great contribution is made by the JINR staff members to testing and development of the grid middleware, the development of grid-monitoring systems and organizing support for different virtual organizations. The only specialized conference in Russia devoted to grid technologies and distributed computing is organized and traditionally held in JINR. Constantly working to train the grid technologies, the JINR created a separate educational grid infrastructure. In the field of grid the JINR actively collaborates with many foreign and Russian research centers. Special attention is paid to cooperation with the JINR Member States.

2. JINR grid infrastructure

By September 2010 the JINR grid-infrastructure included a farm of 1104 processor slots (computers of 2 GB RAM and ~20GB scratch disk per core), more than 50 specialized servers, disk data storage system on the dCache basis of nearly 700 TB (SE of 550 TB with dCache usage and SE of 150 TB with XROOTD usage) and 5 interactive machines for users' work. The current version of the middleware is the gLite 3.2. In October of 2009 transition to 5.4 version of Scientific Linux operating system has been carried out. The name of the JINR grid site in the WLCG/EGEE infrastructure is JINR-LCG2.

The JINR in the WLCG/EGEE infrastructure provides both basic grid services: Berkley DB Information Index (top level BDII), site BDII, Computing Element (CE), Proxy Server (PX), Workload Management System (WMS), Logging&Bookkeeping Service (LB), RGMA-based monitoring system collector server (MON-box), LCG File Catalog (LFC), Storage Element (SE), User Interface (UI)), and specialized grid-services (ROCMON, 2 VOboxes for ALICE and 1 VObox for CMS, Storage Element of XROOTD type for ALICE). For the LHC virtual organizations the actual versions of the specialized software are supported: AliROOT, ROOT and GEANT for ALICE, atlas-offline and atlas-production for ATLAS, CMSSW for CMS and DaVinchi and Gauss is for LHCb. Putting into operation of the JINR-Moscow channel with throughput in 20 Gbit/s is considered as the major achievement of the JINR grid-infrastructure development in the year 2009.

The JINR grid site supports the computing activities of 10 Virtual Organizations (VO) (alice, atlas, biomed, cms, dteam, fusion, hone, lhcb, rgstest and ops) and also provides the grid resources for CBM and PANDA experiments. The VOs of the LHC experiments (ALICE, ATLAS, CMS и LHCb) are primary users of the JINR grid resources.

3. Participation in the grid projects

From 2001, after staring the EU Data Grid project on creation of grid middleware and testing the initial operational grid infrastructure in Europe, the JINR takes an active part in the international grid activities [1]. The JINR made a significant contribution both in the WLCG and EGEE projects.

The JINR is an active member of the Russian consortium RDIG (Russian Data Intensive Grid) which was set up in September 2003 as a national federation in the EGEE project [2].

The EGI (European Grid Infrastructure) project (http://www.egi.eu/) succeeded the EGEE project to coordinate the integration and interaction between National Grid Infrastructures (NGIs) and operate the European level of the production Grid infrastructure for a wide range of scientific disciplines to link NGIs. Russian NGI is e-ARENA. The national association of research and educational e-Infrastructures «e-ARENA» was established in August 2009 as a legal body for coordinating efforts of different organizations in the Russian Federation in creating and developing the e-infrastructures, including networking and grids, to serve science and higher education. The e-Arena Association is recognized by the Ministry of the Communications as a legal body for coordination of the e-infrastructure efforts at national level. In the scope of the EGI stream, the Russian NGI include five organizations, actively participated in the EGEE/EGEE-II/EGEE-III projects: RRC KI (Moscow), SINP MSU (Moscow), JINR (Dubna), PNPI RAS (Gatchina) and ITEP (Moscow).

During of the EGEE project (by April 2010) the JINR staff members took part in the following directions of this project: **SA1** – Grid Infrastructure Support; **SA3** – Integration, Testing & Certification; **NA2** – Dissemination and Outreach of Knowledge on Grid; **NA3** – Training and Induction; **NA4** – Application Identification and Support. These activities were carried out in the frames of collective participation of Russia in the EGEE project (http://www.egee-rdig.ru/).

The Joint Institute for Nuclear Research takes an active part in the WLCG project from its start. Participation of the JINR in the WLCG project is fixed by the Agreement signed by heads of the CERN, the JINR and several Russian institutes. In frames of the project works were carried out both at JINR and at CERN in following directions:

- support and development of the WLCG infrastructure for the high energy physics and for the LHC project [3-5];
- gLite, OMII and Globus Toolkit testing (in particular, creation of certification tests for FTS, LFC and gLite MPI in the 2009-2010 years);
- participation in Service Challenges and Data Challenges – global large-scale testing of WLCG grid-infrastructure[3-5];
- computing support and development for ALICE, ATLAS and CMS experiments [3-5];
- support and development of data storage based on the dCache system[6];
- development of grid monitoring systems [7-14] ;
- development of the MCDB system (WLCG Monte-Carlo Events Data Base)[15-17];
- support of the WLCG activity in the JINR member states;
- training of WLCG users and systems administrators[18,19].

The JINR T2 center participates in the Service, Data, Software and Analysis Challenges and MC Production for ALICE, CMS and ATLAS in coordination with LHC experiments and Tier1 centers at Karlsruhe (FZK), CERN (CERNPROD) and Amsterdam (SARA) [3-5].

A rich experience in designing and developing the grid monitoring and accounting systems has been accumulated at JINR:

- the creation and the support of the monitoring and accounting system for Russian consortium RDIG (http://rocmon.jinr.ru:8080) can be considered as the most significant work. RDIG is Russian grid-segment of the WLCG/EGEE global infrastructure and it comprises about 5 thousands computing slots and more than 3 PB of data storage. The system monitors the RDIG grid-infrastructure by many parameters and provides the statistical information with a high degree of detail and in a very visible form. The monitoring allows keeping an eye on parameters of Grid sites' operation in real time. The accounting shows resources utilization on Grid sites by virtual organizations and single users. The following values are monitored: number of CPUs (total/working/down/free/busy) and jobs (running/waiting), storage space (used/available), available network bandwidth. The accounting values are: a number of submitted jobs, CPU time used (total sum in seconds, normalized time with WNs productivity and average time per job), waiting time (total sum in seconds, average ratio waiting/used CPU time per job) and physical memory (average per job) [6, 7];

- the JINR staff members made a great contribution to the development of the monitoring system for the LHC virtual organizations, which is developed and supported in the department of the Information Technologies in the CERN (Dashboard) (http://dashboard.cern.ch) . These activities were supported by CERN-RFBR project "Grid Monitoring from VO perspective" and included, in particular, monitoring of the CMS Monte-Carlo production system and Condor-G job monitoring, CMS Job Monitoring and CMS job failures reporting, CMS Dashboard data repository maintenance, development of the interface to the CMS Dashboard database for GridMap monitoring tool (http://gridmap.cern.ch/gm); development of real time monitoring subsystem with the usage of Google Earth system (http://dashb-cms-job-devel.cern.ch/doc/guides/service-monitor-gearth/html/user/index.html);
- the monitoring system for the FTS data transfer service was developed and implemented in the CERN. The system enabled not only to monitor but also to improve the FTS service [11-13];
- the Job Monitoring system of the H1 experiment was supplemented with a statistical monitoring component [20];
- a local monitoring system was developed to analyze the dCache storage system (monitoring input/output traffic, requested and utilized space for both ATLAS and CMS experiments): http://litmon.jinr.ru/dcache.html [6].

During the year 2010 the works on development of the deletion service for ATLAS Distributed Data Management (DDM) system were started. The ATLAS DDM is responsible for the replication, access and bookkeeping of ATLAS data across more than 100 distributed grid sites. Deletion service is one of the most important DDM services. The works on the deletion service include a support of the current version of software and development of a new version of the deletion service (more stable and with extended monitoring system). The development comprises the building of new interfaces between parts of deletion service (based on the web service technology), creating new database schema, rebuilding the deletion service core part, development of extended interfaces with mass storage systems and extension of the deletion monitoring system [21].

ATLAS Remote Control Room in Dubna has been constructed at the JINR to monitor the detector at any time, provide a participation of the subsystem experts from Dubna in the shifts and data quality checks remotely and also train the shifters before they come to CERN [22].

The JINR's participation in the GridNNN (Russian National Nanotechnological Net) project includes development activities (especially for monitoring and accounting system), support of registration system of grid services and sites, user support service, support of the virtual organization for molecular dynamics calculations, adjustment of some JINR's applications to parallel execution in the GridNNN environment and creation of the infrastructure for applications development and testing.

Since 2004, different kinds of training courses on grid-technologies have been organized at JINR on the basis of the Laboratory of Information Technologies. These courses include introduction into the grid and the work in the gLite environment and training of system administrators on the organization of grid-infrastructures as well as acquaintance with specificity of work in the grid in the concrete virtual organizations. Course participants are the JINR staff members, their colleagues from Russia and from the JINR Member States as well as the students of the JINR University Center. Then a special infrastructure for training on grid-technologies was created and originally located on the dedicated servers at JINR. At the moment, the educational complex on the basis of the gLite environment consists of three grid-sites of the JINR, and also of the grid-sites of the Institute of High Energy Physics (IHEP, Protvino), of the Institute of Mathematics and Information Technologies of Academy of Sciences of the Republic of Uzbekistan (Tashkent), of the Sofia University (Sofia, Bulgaria), of the Bogolyubov Institute for Theoretical Physics (Kiev, Ukraine) and of the Kiev Polytechnic Institute (Kiev, Ukraine)[18.19].

Protocols and agreements for cooperation in the filed of grid technologies are signed between the JINR and Armenia, Belarus, Bulgaria, Moldova, Poland, Czech and Slovak. The JINR takes part in a number of joint grid projects (some of them are supported by different grants) with Czech, Slovak, Germany, South Africa, Belarus, Bulgaria, Ukraine and Romania, in particular:

- BMBF grant “Development of the Grid-infrastructure and tools to provide joint investigations performed with participation of JINR and German research centers”,
- “Development of Grid segment for the LHC experiments” was supported in frames of JINR-South Africa cooperation agreement,
- NATO project "DREAMS-ASIA“ (Development of gRid EnAbling technology in Medicine&Science for Central ASIA),
- JINR -FZU AS Czech Republic Project “The GRID infrastructure for the physics experiments”,
- NASU-RFBR project “Development and support of LIT JINR and NSC KIPT grid-infrastructures for distributed CMS (CERN) data processing during the first two years of the Large Hadron Collider operation”[23],
- project “Elaboration of distributed computing JINR-Armenia grid-infrastructure for carrying out mutual scientific investigations“,
- JINR-Romania cooperation Hulubei-Meshcheryakov programme,
- project "SKIF-GRID" (Program of Belarusian-Russian Union State).

Trainings for grid site administrators from Ukraine, Romania, Uzbekistan and Azerbaijan have been conducted at JINR during 2008-2010. Courses and practical training for students and users from Egypt and Bulgaria have been also organized. We provide a continuous support and consulting for specialists from Cuba, Georgia, Kazakhstan, Mongolia, Vietnam and Korea.

A new informational resource has been created at JINR (initially in Russian): the web-portal “GRID AT JINR” (http://grid.jinr.ru). The content includes a detailed information on the JINR grid-site and JINR’s participation in grid projects.

4. Summary

The resources of the JINR grid site are actively used by different virtual organizations and the JINR’s contribution into the resources provided by the consortium RDIG in 2009-2010 is the most significant one (see Fig. 1).

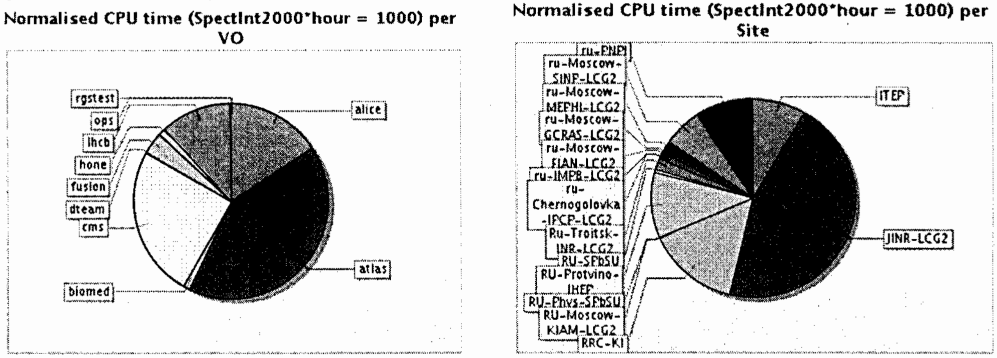


Fig.1: The diagram of the usage of the JINR’s grid resources by VOs (left) and the distribution of CPU time provided by the members of the consortium RDIG (right) during the period from July, 2009 to June, 2010

We shall continue our grid activities within the EGI project in frames of Russian NGI providing a continuous reliable support to a number of VOs including the LHC VOs. A special attention will be paid to the grid deployment of new applications from the fields of nanotechnology, industry, medicine and engineering.

The further development of the JINR Grid-environment in 2010-2016 comprises:

- at the network level: links between Moscow and Dubna on the basis of state-of-the-art technologies DWDM and 10Gb Ethernet;
- for the JINR Local area network: JINR high-speed backbone construction (10Gbps);
- at the resource level: to reach effective processing and analysis of the experimental data, further increase in the JINR CICC performance and disk space is needed.

References

[1] Belov S.D. at al. Joint Institute for Nuclear Research in the WLCG and EGEE projects // Proc. of NEC'2009, Dubna, JINR, 2010. P.137-142.

[2] Ilyin V.A., Korenkov V.V., Soldatov A.A. RDIG (Russian Data Intensive Grid) e-Infrastructure: status and plans // Proc. of NEC'2009, Dubna, JINR, 2010. P.150.

[3] Bogdanov A. et al. RDIG ALICE computing just before the first LHC data // Proc. of the 3rd Int.Conference GRID.2008, Dubna, JINR, 2008. P. 164-168.

[4] Demichev M. et al. Readiness of the JINR grid segment to process the first ATLAS data // Proc. of NEC'2009, Dubna, JINR, 2010. P. 111.

[5] Gavrilov V. et al. RDMS CMS computing activities to satisfy LHC data processing and analysis // Proc. of NEC'2009, Dubna, JINR, 2010. P.129.

[6] Trofimov V., Dmitrienko P. The approach to monitoring and optimization of Storage Element based on the dCache system // Scientific Report of the Laboratory of Information Technologies (LIT) 2008-2009, JINR, Dubna, 2009. P.41-43 (in Russian).

[7] Belov S.D., Korenkov V.V. Experience in development of Grid monitoring and accounting systems in Russia, in Proc. of NEC'2009, Dubna, JINR, 2010. P.75.

[8] Belov S.D., Korenkov V.V. Experience in the development of grid monitoring and accounting systems // Proc. of the 3rd Int. Conference GRID'2008, Dubna, JINR, P.189-192.

[9] Sidorova I. Job monitoring for the LHC experiments // Proc. of NEC'2009, Dubna, JINR, 2010. P. 243.

[10] Andreeva J. et al. Job monitoring on the WLCG scope: Current status and new strategy. J. Phys.: Conf. Ser. 219, 2010. 062002.

[11] Andreeva J. et al. Dashboard for the LHC experiments, J. Phys. Conf. Ser.119: 062008, 2008.

[12] Uzhinski A., Korenkov V. Monitoring system for the FTS data transfer service of the EGEE/WLCG project, Calculating Methods and Programming. V.10. P.96-106, 2009 (in Russian).

[13] Uzhinski A., Korenkov V. Data transfer service architecture in grid, Open Systems, N2, 2008 (in Russian).

[14] Uzhinski A., Korenkov V. Statistical analysis of data transfer errors in the global WLCG/EGEE infrastructure. P11-2008-82. JINR Preprint (in Russian).

[15] Belov S. et al. HepML, an XML-based format for describing simulated data in high energy physics, Computer Physics Communications, In Press, Accepted Manuscript.

[16] Belov S. et al. LCG MCDB – a Knowledgebase of Monte Carlo Simulated Events, Computer Physics Communications, V. 178, I. 3, 1 February 2008. P. 222-229.

[17] Belov S. et al. LCG MCDB and HepML, next step to unified interfaces of Monte-Carlo simulation // Proc. are published by Proceedings of Science, PoS ACAT08:115, 2008.

[18] Belov S.D., Korenkov V.V., Kutovskiy N.A. Educational grid infrastructure: status and plans // Proc. of NEC'2009, Dubna, JINR, 2010. P.81.

[19] Korenkov V.V., Kutovskiy N.A. Educational grid infrastructure, Open Systems, N. 10, 2009 (in Russian).

[20] Mitsyn S., Lobodzinski B. H1 Monte Carlo Production on the LCG grid-job monitoring // Proc. of the 3rd Int. Conference GRID'2008, Dubna, JINR, 2008. P.369-362.

[21] Oleynik D., Petrosyan A. ATLAS Deletion Service, http://indico.cern.ch/conferenceDisplay.py?confId= 76895#2010-07-14

[22] Kotov V.M., Rusakovich N.A. Development of the System remote access real time (SRART) at JINR for monitoring and quality assessment of data from the ATLAS LHC experiment (Concept and architecture of prototype SRART at JINR) // Proc. of NEC'2009, Dubna, JINR, 2010. P.162.

[23] Bunetsky O.O. et al. Preparation of the LIT JINR and the NSC KIPT (Kharkov, Ukraine) grid-infrastructures for CMS experiment data analysis, P11-2010-11, 2010, Dubna, JINR. 12 pages (in Russian).

DISTRIBUTED TRAINING AND TESTING GRID INFRASTRUCTURE

V. V. Korenkov[1,2], N. A. Kutovskiy[1,3]

[1]*Laboratory of Information Technologies, Joint Institute for Nuclear Research, 141980, Dubna, Russia*
[2]*Dubna International University of Nature, Society, and Man, 141980, Dubna, Russia*
[3]*National Scientific and Educational Centre of Particle and High Energy Physics of the Belarusian State University, 220040, Minsk, Belarus*

Grid-technologies have already became standard tools used by scientists in different fields and first of all in high-energy physics. The associated steep learning curve may be alleviated within a dedicated education and training process. To this purpose, a distributed training and testing grid-infrastructure (t-infrastructure for short) has been set up with core services at the Laboratory of information technologies (LIT) of JINR and integrating resources of several organizations from the JINR Member States. It is used for educating and giving practical tutorials to students of University Centre (UC) of JINR, University «Dubna», JINR and its member states colleagues as well as for performing obligations in different Grid related activities of local and international projects.

During the last 2 years the educational grid infrastructure has been intensively developed and the spectrum of the tasks that should be used for has been grown up as well. Currently the list of possible activities looks as follows:

- user, system administrator and developer trainings,
- grid-oriented application development and porting,
- middleware functionality testing and certification.

In order to be able to perform these tasks, the training infrastructure (t-infrastructure for short) needs to be autonomous i.e. independent of any of external services of production grid infrastructures (for example EGEE\WLCG one). Taking into account the non-production nature of the activities, the gLite (http://glite.web.cern.ch/glite/) based t-infrastructure is used for the following principles of its deployment can be considered.

1. Certification authorities (CAs) of all organizations whose grid sites are the part of the t-infrastructure can be self-signed and trust each other without root ("top-level") CA's signature as it is done in EGEE\WLCG production infrastructure.
2. There is no need in VOBOX service on sites which acts as a gateway for VO-specific software since each VO conducts special courses for their users on production infrastructure.
3. The number of grid sites in t-infrastructure is much less than in production one (up to 10 in comparison with few hundred). Thus its list can be maintained manually instead of hosting Grid Operations Center Database (GOCDB) services which is used by top-level BDIIs for grid site list retrieval.
4. All organizations having their grid sites as part of t-infrastructure are equal and its use is free of charge. Thus there is no need in accounting i.e. in such services of EGEE\WLCG production infrastructure as MON-box, APEL и R-GMA Registry.
5. Since the production quality of service of t-infrastructure is redundant the simplified monitoring of its services can be used. It means there is no need in SAM, GStat, GridView, Gridmap services but information from top-level BDIIs can be utilized instead.
6. A few WNs per each CE are enough since jobs run during trainings are simple and short in time.

7. A small size of transferring files and absence in its reliable transfers let avoid the deployment of File Transfer Service (FTS).
8. Since the performance is not a critical point for the training infrastructure, all its services can be deployed on virtual machines.
9. There is no need in MyProxy server because users' jobs are much shorter than the default lifetime of user proxy certificate.
10. All communications among site administrators and t-infrastructure management as well as technical support can be done via email, skype, icq, etc. Thus there is no strong need in such administrative services as CIC, ROC, GOC what EGEE\WLCG projects have.

The educational grid infrastructure was built following the mentioned principles. Initially all its services were located at JINR (see [1] for more details). Since then new grid sites of some organizations from the JINR Member-States were integrated into it thus making that infrastructure geographically distributed. Apart from that two more grid sites with MPI enabled CE were deployed at JINR and integrated into t-infrastructure: 1) LCG-CE with three worker nodes (WNs) run on 32-bits platform and 2) CREAM based CE with three WNs run on 64-bits platform. A schema of the current state of the distributed training and testing grid infrastructure is shown on Figure 1.

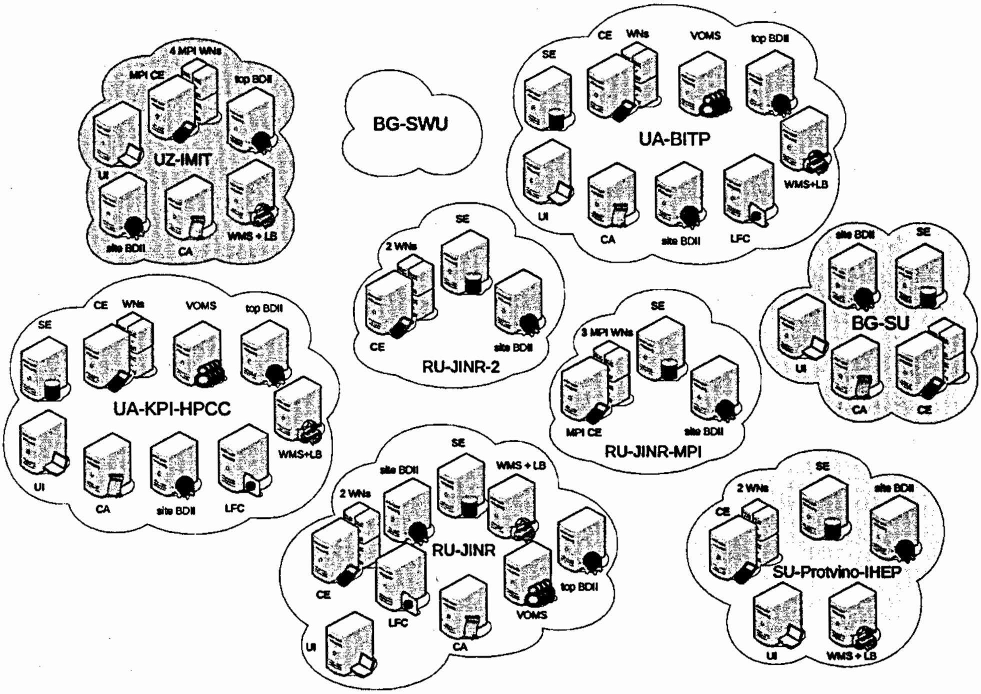


Fig. 1: The schema of the distributed training and testing grid infrastructure based on gLite middleware

This gLite based t-infrastructure consists of the grid sites and services listed in Table 1.

Table 1. List of the gLite based t-infrastructure services and hosting organizations

site name	hosting organization	services
RU-JINR	JINR	User Interface (UI), LCG Computing Element (LCG-CE) with two Worker Nodes (WNs), Disk Pool Manager Storage Element (DPM SE), LCG File Catalogue (LFC), Workload Management System (WMS), Logging&Bookkeeping Service (LB), site BDII (sBDII), top BDII (tBDII), Virtual organizations management service (VOMS)
RU-JINR-2	– // –	LCG -CE + 2 WNs, DPM SE, sBDII;
RU-JINR-MPI	– // –	MPI enabled LCG-CE + 3 WNs, MPI enabled CREAM based CE + 3 WNs, DPM SE, sBDII;
BG-SU	Sofia University “St. Kliment Ohridski”, Sofia, Bulgaria	UI, LCG-CE + 4WNs, DPM SE, sBDII;
BG-SWU	South-West University “Neofit Rilski”	no running services yet
SU-Protvino-IHEP	Institute of High-Energy Physics, Protvino, Moscow region, Russia	UI, LCG-CE + 2 WNs, dCache SE, WMS, LB, sBDII;
UZ-IMIT	Institute of Mathematics and Information technologies of Academy of Science of Republic of Uzbekistan, Tashkhent, Uzbekistan	UI, MPI enabled LCG-CE + 4 WNs, WMS, LB, sBDII, tBDII;
UA-BITP	Bogolyubov Institute for Theoretical Physics, Kiev, Ukraine	UI, LCG-CE + 8 WNs, DPM SE, LFC, WMS, LB, sBDII, tBDII;
UA-KPI-HPCC	National Technical University of Ukraine "Kyiv Polytechnic Institute", Kiev, Ukraine	UI, LCG-CE + 8 WNs, DPM SE, LFC, WMS, LB, sBDII, tBDII.

Apart from gLite environment the Globus Toolkit 5.0.x – GT5 (http://globus.org) based testbed was deployed at JINR. There are plans to extend user trainings by giving a tutorial on working in GT5 as well as making a new course for developers in that grid environment.

CA, VOMS, MyProxy can be used by services of both testbeds: gLite and GT5.

A schema of JINR part components of t-infrastructure is shown at Figure 2.

Besides grid services there are also java application server for custom java-based applications deployment, web- server hosting the web-portal (https://gridedu.jinr.ru) containing information about the t-infrastructure, some administrator guides, instructions on how to integrate grid site into the t-infrastructure, shell access to the gLite user interface via web.

All services are run on virtual machines. As it was already mentioned in [1], OpenVZ (http://openvz.org) is used as a software for virtualization.

For the moment t-infrastructure has been used for the following activities:

1) semestral educational course in Grid for students of UC and University “Dubna” (4 years of successful experience, more than 300 students);
2) user and system administrator trainings for colleagues from the JINR Member-States and partner countries (Egypt, Ukraine, Romania, Belarus, Uzbekistan, Azerbaijan, North Korea and Republic

of South Africa);

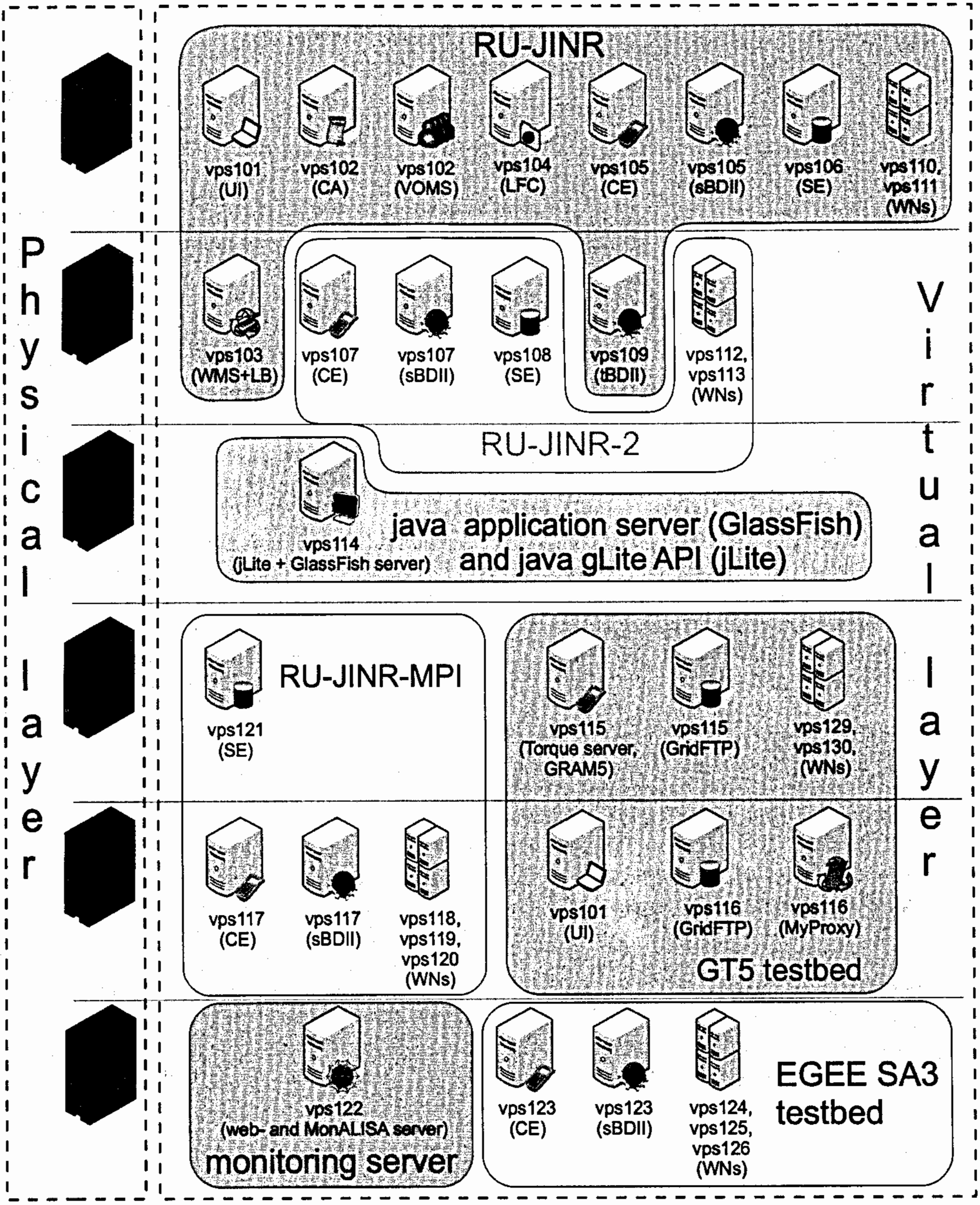


Fig. 2: The schema of JINR part components of the t-infrastructure and services distribution over hosts

3) research and development projects:
   i) grid-oriented applications development on the basis of SOA:
      (1) grid-oriented application for data quality processing (parsing, cleansing, standardization, enrichment etc.) on a large volumes of data,
      (2) grid service for minimum spanning tree computation;
   ii) molecular dynamics simulations (DL_POLY package) and molecular electronic structure calculations with high accuracy (molpro package) on grid CEs with MPI support;
   iii) computer-aided engineering (CAE) simulations;
   iv) image and video processing;
4) testing and certification of gLite components in the framework of EGEE SA3 activity and some other ones.

Conclusion

As one can see there is a strong demand in different grid related activities what requires a separate autonomous infrastructure. Following that necessity, the training and testing grid infrastructure was deployed with core services at LIT JINR. It is already successfully used for a wide spectrum of tasks and there are some in plans.

It is planned to run in testing mode and after in production one the educational web-portal which aims to provide a possibility for distance learning grid technologies. It includes a detailed user guide for work with educational grid infrastructure, lectures, methodical materials, interactive tests for each topic, full-featured web access to the t-infrastructure.

Apart from that the future plans comprise evaluation of modern virtualization technologies and migration of the JINR t-infrastructure to one of them; new grid sites integration; making courses on Globus Toolkit and grid applications and services development; innovative grid projects support.

Besides, there is a plan in cooperation with DEGISCO project (http://degisco.eu) team to set up BOINC server (http://boinc.berkeley.edu/), connect it to the t-infrastructure, port some applications to that environment and run them on idle resources of desktop computers. On the basis of the gained experience, a course on desktop grid computing can be developed and added into educational programs of different groups of trainees.

All t-infrastructure related work in 2010 is supported by the JINR grant for young scientists and specialists.

References

[1] Belov S.D., Korenkov V.V., Kutovskiy N.A. “Educational Grid infrastructure at JINR” // Proceedings of the 3rd International conference “Distributed Computing and Grid-Technologies in Science and Education” (GRID'2008), Dubna, 2008. P.341-342.

[2] Belov S.D., Korenkov V.V., Kutovskiy N.A. “Educational grid infrastructure: status and plans” / Proceedings of XXII International Symposium on Nuclear Electronics & Computing (NEC`2009), Dubna, 2010, P.81-83.

[3] Kutovskiy N.A. “Educational, training and testing grid infrastructure” // Proceedings of XIV conference of young scientists and specialists (OMUS'2010), Dubna, 2010, P.70-73

VIRTUAL LABORATORY AND SCIENTIFIC WORKFLOW MANAGEMENT ON THE GRID FOR NUCLEAR PHYSICS APPLICATIONS

V. V. Korkhov[1], D. A. Vasyunin[1,2], A. S .Z. Belloum[2], S. N. Andrianov[1], A. V. Bogdanov[1]

[1] St.Petersburg State University, Universitetsky pr. 35, Peterhof, 198504, St.Petersburg, Russia
[2] University of Amsterdam, Science Park 107, 1098 XG, Amsterdam, the Netherlands
vladimir@csa.ru

This paper discusses the ways to implement accelerator physics applications as distributed applications on the Grid. The proposed solution is based on virtual laboratory approach which makes use of scientific Grid workflow management systems, in particular WS-VLAM. An overview of WS-VLAM is given, decomposition of generic accelerator physics application as a workflow is presented.

1. Introduction

Recent advances in Internet and Grid technologies have greatly enhanced processes in scientific experiments; not only computing and data intensive tasks become feasible, but also large scale collaborations between resources and users are now possible. Scientific workflows are becoming an increasingly popular approach to develop and execute complex scientific experiments and computer simulation that require large number of compute and data resources. With appropriate virtualization, workflows hide the complexity of underlying computing resources and data systems, so that domain scientists can focus on the logic of the experiments without going into low-level details. This approach allows creating a Virtual laboratory that provides means to run legacy applications, manages the e-Science experiments, and automates large-scale computations.

The focus of the workflow research is currently on studying and developing novel approaches to improve support for the design of scientific workflow and increase reusability of workflows among scientists. In essence, scientific workflow management systems aim at the automation of scientific processes based on data dependencies and their control, as well as at the abstraction of the usage of the necessary underlying resources to help scientists focus on their own research.

An initiative to apply scientific workflow management for creating complex modelling applications for nuclear physics has started at the St.Petersburg University recently. WS-VLAM workflow system, developed at the University of Amsterdam [1], has been selected as the core integration and workflow management software. WS-VLAM is developed using WS-RF concepts based on Globus 4.1 toolkit, it provides means to compose, execute and monitor data-flow based workflows on the Grid with capabilities of task farming for parameter sweeps. The current work of integration accelerator software as a Grid workflow is inspired by the Unified Accelerator Library (UAL) [2], an object oriented programming toolkit for developing distributed accelerator software. Shifting the paradigm towards e-Science and Grid computing, workflow management systems can support and extend the capabilities for execution, interconnection and providing unified access to various accelerator software (e.g. MAD, COSY) on the Grid.

The paper is structured as follows: Section 2 gives an overview of virtual laboratory concepts; Section 3 presents the WS-VLAM workflow management system, a virtual laboratory environment developed at the University of Amsterdam; Section 4 describes an application from the accelerator physics domain that can be decomposed and executed as a Grid workflow, and Section 5 concludes the paper.

2. Virtual Laboratories on the Grid

Grid environment allows coordinated resource sharing and problem solving among groups of trusted users within Virtual Organizations. Such environments enable global distributed collaborations involving large numbers of people and large scale resources, and make data and computing intensive scientific experiments feasible. One of the important research topics in e-Science [3] is to develop effective Grid enabled Problem Solving Environments (PSE), also called Virtual laboratories, for different scientific domains. Organizing software utilities (e.g. simulators, visualization and data analysis tools) as a meta experimental environment, PSE allows a scientist to plan and conduct experiments at high level of abstraction [4].

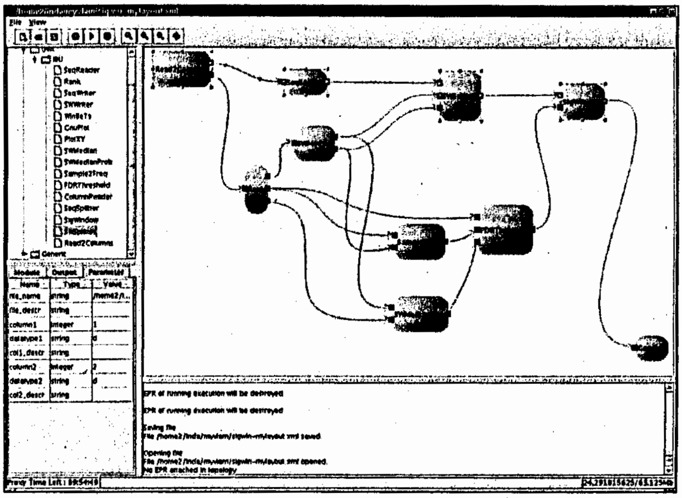
A problem solving environment (PSE) provides a complete integrated computing environment for composing, compiling, running, controlling and visualizing applications. It incorporates many features of an expert system and provides extensive assistance to users in formulating problems and integrating program codes, processes, data, and systems in distributed computer environments [5]. An important PSE flavor is a Scientific Workflow Management System (SWMS) [6, 7]. A SWMS explicitly models the dependencies between experiment processes, and orchestrates the runtime behavior of involved resources according to a flow description.

3. WS-VLAM Workflow Management System

The aim of the WS-VLAM (Virtual Laboratory AMsterdam) system developed at the University of Amsterdam is to provide and support coordinated execution of distributed Grid-enabled components combined in a workflow. This system combines the ability to take advantage of the underlying Grid infrastructure and a flexible high level rapid prototyping environment. On the high level, a distributed application is composed as a data driven workflow where each component represents a process or service on the Grid. Processes are activated only when the data is available on their input ports. The significant difference from other similar systems is the support for simultaneous execution of co-allocated processes on the Grid which enables direct data streaming between the distributed components: traditional batch processing of grid jobs and workflow execution based on input/output files exchange between the components is not suitable for many use case scenarios. This feature is highly required for semi-realtime distributed applications e.g. in the bio-medical domain or in online video processing and analysis.

WS-VLAM is a workflow management system, which coordinates the execution of distributed Grid-enabled software components. WS-VLAM workflow management system is developed following the OGSA/WSRF standards. It has a set of client-side applications that allow scientists to design and monitor the execution of the workflows with intuitive interfaces, and provides also server-side applications, including a workflow engine that schedules and executes the workflow on Grid enabled resources [1, 8].

WS-VLAM provides a composition editor for graphical creation of workflows (Fig. 1). This tool is used for two main tasks: composition and monitoring. It supports developing workflow components in a number of programming and scripting languages (Java, C++, R, etc.), allows accessing to web services, establishes job farming and parameter sweep requirements, and is platform independent. Once the workflow has been submitted to execution, scientists are able to follow up the results of the experiments.



Name	Type	Value
file_name	string	/home2/i...
file_descr	string	
column1	integer	1
datatype1	string	d
col1_descr	string	
column2	integer	2
datatype2	string	d
col2_descr	string	

```
EPR of running execution will be destroyed

EPR of running execution will be destroyed

Saving file
File /home2/inda/myvlam/sigwin-mlayout.xml saved.

Opening file
File /home2/inda/myvlam/sigwin-mlayout.xml opened.
No EPR attached in topology
```

Fig. 1: WS-VLAM graphical workflow composer

Monitoring the execution allows follow the effects and outcomes of each particular atomic (or composite) execution process. WS-VLAM is a grid enabled workflow management system and it allows of co-existence of different types of grid execution models within a single workflow. This feature is achieved by abstracting a particular Grid execution model to an intermediate common representation. In WS-VLAM the workflow is composed not from particular Grid jobs or services but from components having special interface. These components are called modules; they are the core entities of the WS-VLAM workflow. Thus a module can represent a specially developed application, which uses WS-VLAM native libraries, a standard web service or a third party application (legacy application).

The runtime control of the execution of a distributed workflow provides the ability to monitor the execution and influences the behavior of workflow components. WS-VLAM supports several ways of runtime control: direct interaction with the user interface of a module (remote X GUI access) and module parameter control (reading flags and values set by a module and updating these values from outside the module). Monitoring delivers all the log data from remote modules to the WS-VLAM user interface thus all the issues in module execution can be tracked centrally.

Intensive distributed data processing might take a long time. To facilitate the handling of the executing workflow, the system is capable of closing the user interface, detaching from the workflow engine and reattaching later on during runtime.

4. Parallel and Distributed Computing in Accelerator Physics

The natural parallel and distributed structures of beam physics problems allow the use of parallel and distributed computer systems. But the usual approaches based on traditional numerical methods demand using the resources of supercomputers. This leads to the impossibility of using such multiprocessing systems as computational clusters. In this paper some examples of beam physics problems are discussed from the computational point of view using clustered systems.

There are two classes of problems in beam physics which demand very extensive computer resources. The first class includes long-time evolution problems, the second is concerned with the computer realization of optimization procedures for beam lines. Examples of the first type of problem include multiturn injection and extraction of the beam in circular accelerators. Usually, these problems do not consider space charge effects. For advanced applications it is essential to study beam dynamics in high-intensity accelerators. Such machines are characterized by large beam currents and by very stringent uncontrolled beam loss requirements. An additional difficulty of numerical simulation is connected with long-time beam evolution that requires the computation of hundreds of thousands or millions of turns. It requires the use of high-performance computers for beam evolution study. The problems of similar multi-turn evolution such as transverse stability with nonlinear space charge, uncontrolled beam losses due to space-charge-induced halo generation, etc. can also be mentioned. These problems are peculiar to modern high-intensity machines and require careful investigations of long-time evolution effects. From the computational point of view there are some problems related to the choice of models for beams with space charge, the presentation form of the beam propagator, and so on.

Currently, there are many efforts [9, 10] devoted to the application of parallel computation to beam physics problems. But most of these efforts are dedicated to the creation of parallel algorithms for space-charge forces and multi-particle dynamics computing.

In one of previous papers we addressed the role of parallel and distributed computing in beam physics [12]. The generic scheme of the decomposition of a beam physics application and mapping it to distributed resources has been proposed. In short it is illustrated here in figures 2 and 3.

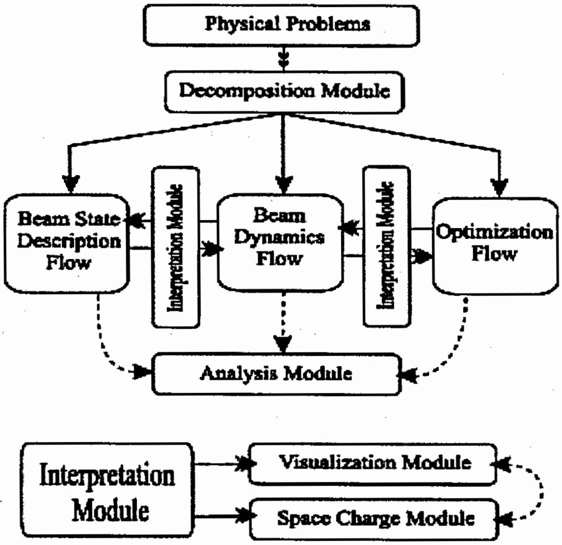
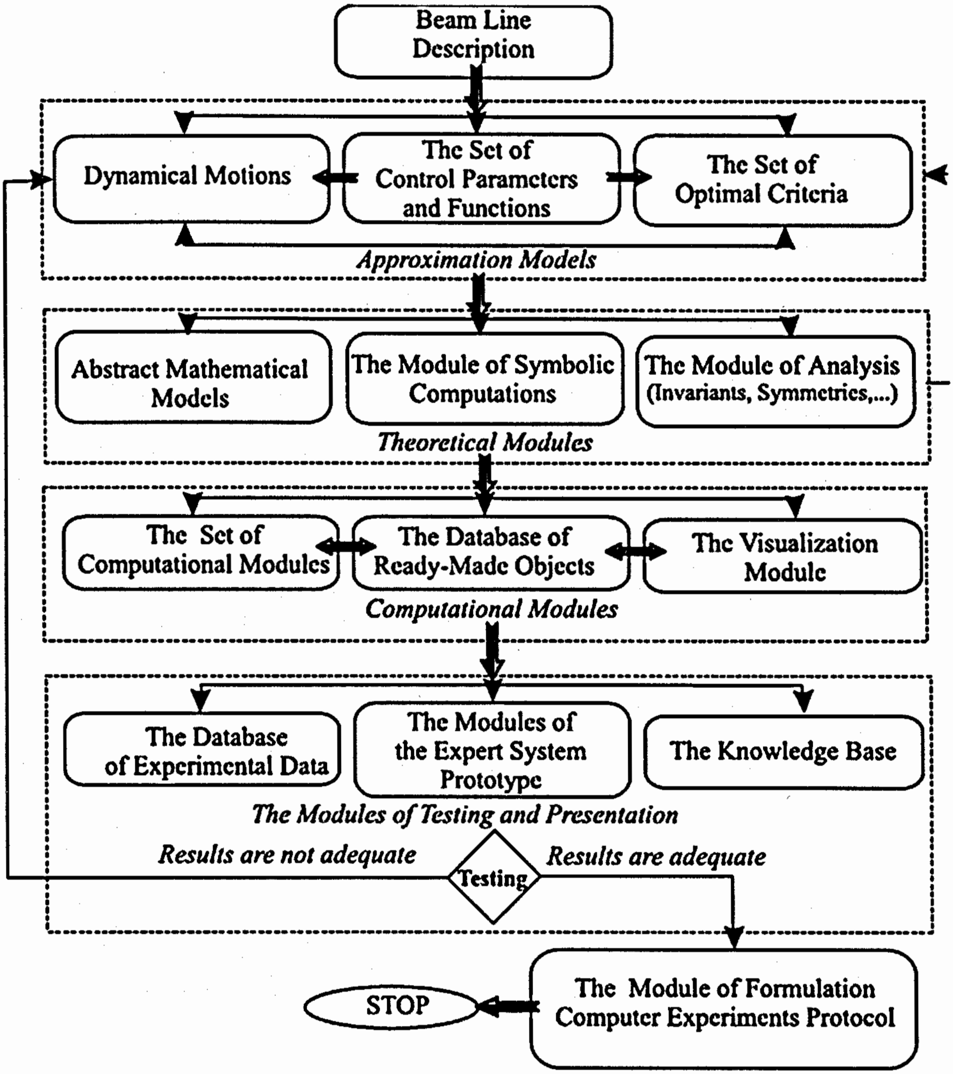


Fig. 2: The types of basic computational flows

The applied approach is based on the use of the matrix formalism for Lie algebraic methods developed in a previous work [13]. The choice of a matrix formalism as the basic tool for the beam evaluation process allows the use of databases of matrix objects prepared in symbolic and/or numerical modes. Moreover, this aids the construction of effective numerical and symbolic codes for computational experiments. In this paper we will distinguish two concepts: parallel and distributed computing. The first type of computational process involves the implementation of homogeneous operations on a set of homogeneous processors. In the second, operations having different structure are computed using a heterogeneous set of processors. This requires us to distinguish two types of computational operations: the first of them corresponds to matrix operations and the second - to computational flows. This separation allows us to distribute a computational experiment over several clusters. Every cluster solves the problems intrinsic to one of the flows. This approach has a bottleneck problem connected with the synchronization of these flows. This problem can be solved using a base of homogeneous mathematical tools—the matrix formalism for Lie methods. The matrix form of practically all required information allows us to realize parallel computing in a natural way. First of all, parallel computing is realized in the numerical stage when the matrix presentation of the current map is built. The second parallelization process is connected with the phase beam portrait construction stage. For this stage there are several possible approaches.

Additional computational flows are connected with the next two procedures. The first of them is devoted to visualization of all necessary information including auxiliary procedures (for example, analysis of images using differential geometry methods) and space-charge force computing (see Fig. 2). Here we rely upon the methods proposed in our previous publications [11, 12].



Beam Line Description
Dynamical Motions
The Set of Control Parameters and Functions
The Set of Optimal Criteria
Approximation Models
Abstract Mathematical Models
The Module of Symbolic Computations
The Module of Analysis (Invariants, Symmetries,...)
Theoretical Modules
The Set of Computational Modules
The Database of Ready-Made Objects
The Visualization Module
Computational Modules
The Database of Experimental Data
The Modules of the Expert System Prototype
The Knowledge Base
The Modules of Testing and Presentation
Results are not adequate
Testing
Results are adequate
STOP
The Module of Formulation Computer Experiments Protocol

Fig. 3: The total cycle of the computational experiment

Conclusion

In this paper we discussed the approach to implement a generic accelerator physics application as a Grid workflow. The work is ongoing at the Faculty of Applied Mathematics and Control Processes of St. Petersburg State University. The core component of the system is the Virtual laboratory based on the WS-VLAM Grid workflow management system, and the application part is built on longstanding faculty members' expertise in beam physics.

References

[1] Korkhov V., Vasyunin D., Wibisono A., Guevara-Masis V., Belloum A., de Laat C., Adriaans P., Hertzberger L.O. WS-VLAM: Towards a Scalable Workflow System on the Grid. Proc. 16th IEEE High Performance Distributed Computing, WORKS07, pp 63-68, ISBN 978-1-59593-715-5, Monterey Bay, California, USA, June 25-29, 2007.

[2] Malitsky N. and Talman R. Unified Accelerator Libraries, AIP 391, 1996.

[3] Hey T. and Trefethen A. E. The UK e-science core programme and the grid. Future Generation Computer System, 18(8):1017–1031, 2002.

[4] Efstratios Gallopoulos J. R. R., Elias Houstis. Computer as thinker doer: Problem-solving environments for computational science. IEEE Computational Science and Engineering, 2:13–23, 1994.

[5] Gallopoulos E., Houstis E., Rice J.R. Computer as Thinker Doer: Problem-Solving Environments for Computational Science, IEEE Computational Science and Engineering, vol 2, pp 13-23, 1994.

[6] Chin J. G., Leung L. R., Schuchardt K., and Gracio D. New paradigms in problem solving environments for scientific computing. In Proceedings of the international conference of Intelligent User Interface, San Francisco, 2002.

[7] McClatchey R. and Vossen G. Workshop on workflow management in scientific and engineering applications report. SIGMOD Rec., 26(4):49–53, 1997.

[8] Korkhov V., Vasyunin D., Wibisono A., Belloum A.S.Z., Inda M.A., Roos M., Breit T.M., and Hertzberger L.O. VLAM-G: Interactive Dataflow Driven Engine for Grid-enabled Resources, Scientific Programming 15(3), pp.173-188 (2007).

[9] Qiang J., Ryne R.D., Habib S. Beam halo studies using a 3-dimensional particle-core model, Proceedings of the 1999 Particle Accelerator Conference, New York, 1999, pp. 366–368.

[10] Giovannozzi M. Proceedings of the 1998 European Particle Accelerators Conference, Stockholm, Sweden, 1998, pp. 1189–1191.

[11] Andrianov S.N. Parallel computing in beam physics problems, Proceedings of the Seventh European Particle Accelerator Conference—EPAC'2000, Vienna, Austria, 2000, pp. 1459–1461.

[12] Andrianov S. Role of parallel and distributed computing in beam physics, Nuclear Instruments and Methods in Physics Research A 519 (2004) 37–41.

[13] Andrianov S.N. A matrix representation of the lie algebraic methods for design of nonlinear beam lines, Proceedings of the 1996 Computational Accelerator Physics Conference, Williamsburg, VA, USA, AIP Conf. Proc. 391 (1997) 355–360.

PRAGUE WLCG TIER-2 REPORT

T. Kouba, L. Fiala, J. Chudoba, M. Lokajicek, J. Svec

Institute of Physics, Academy of Sciences of the Czech Republic

The Prague site participates in various grid activities such as EGEE/EGI and WLCG. The computing infrastructure serves for several high energy physics experiment, for example D0, Atlas, Alice, Star, Auger. In this report, we would like to present the recent changes in our infrastructure ranging from upgrade to 10Gb interconnect to installation of new water cooling system. We will also present our storage infrastructure and decisions made to ensure high capacity and throughput for DPM grid software.

1. Introduction

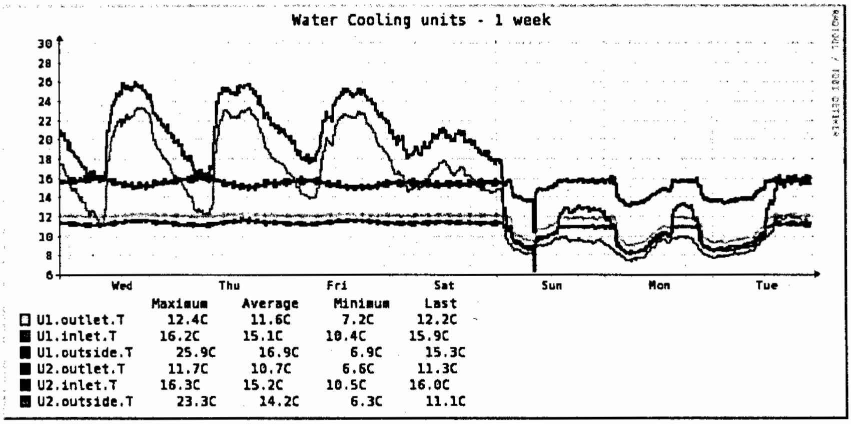
The Prague site for computing in high energy physics was established in 2002 under the name "Golias" or in the grid world under the name "praguelcg2". In 2004 a new server room was built from scratch. The server room is used for all information technology needs of the Institute of Physics (e.g. web servers, mail servers, firewalls) but the major part of the computing power and data storage is dedicated for computing in High Energy Physics. Since the beginning the main consumer of the computing cycles has been the experiment D0. But with the start of LHC production users from other experiments (mainly Atlas and Alice) have become more active.

2. Recent hardware challenges and changes

2.1. Cooling

In the last two years the number of computing servers has been significantly increased. Also the total power consumption and the waste heat has increased despite the fact that computer manufacturers improved the efficiency in computing power per watt. This brought problems in two areas: cooling and emergency power supply.

Our server room is equipped with two units of air conditioning Liebert-Hiross with cooling power 56kW per unit. This was sufficient until spring 2009. In this year we purchased new iDataPlex system with 84 IBM x340 nodes. These new 672 CPU cores more than doubled computing capacity of the Golias farm. Another system was purchased at the beginning of 2009 - Altix ICE 8200. Waste heat from these two systems summed with the other systems in the server room got over the cooling capacity of Liebert-Hiross system. We decided to install a water cooling solution for these two new systems with extra capacity for future new systems.



	Maximum	Average	Minimum	Last
□ U1.outlet.T	12.4C	11.6C	7.2C	12.2C
■ U1.inlet.T	16.2C	15.1C	10.4C	15.9C
■ U1.outside.T	25.9C	16.9C	6.9C	15.3C
■ U2.outlet.T	11.7C	10.7C	6.6C	11.3C
■ U2.inlet.T	16.3C	15.2C	10.5C	16.0C
■ U2.outside.T	23.3C	14.2C	6.3C	11.1C

Fig.1: Graph of temperatures

The new water cooling consists of two STULZ CLO 781A units that deliver 88kW cooling power each (graph of temperatures in this system can be seen on Fig. 1). They are placed on the roof of the building and connected with rear doors of the computer racks of iDataPlex and Altix ICE. The rear doors contain a heat exchanger (aka radiator) where the flowing hot air is cooled. It means the air flow in the server room is still very important. The IBM has created an air flow model (figure 2). This model showed that it is very important to ensure that the hot air is effectively sucked in the air conditioning and that the cold air is not mixed with hot air. Based on this model, we have implemented the following two decisions:

- cold aisle covered with a roof,
- big fans installed on top of the racks so the hot air is pushed to the air condition input.

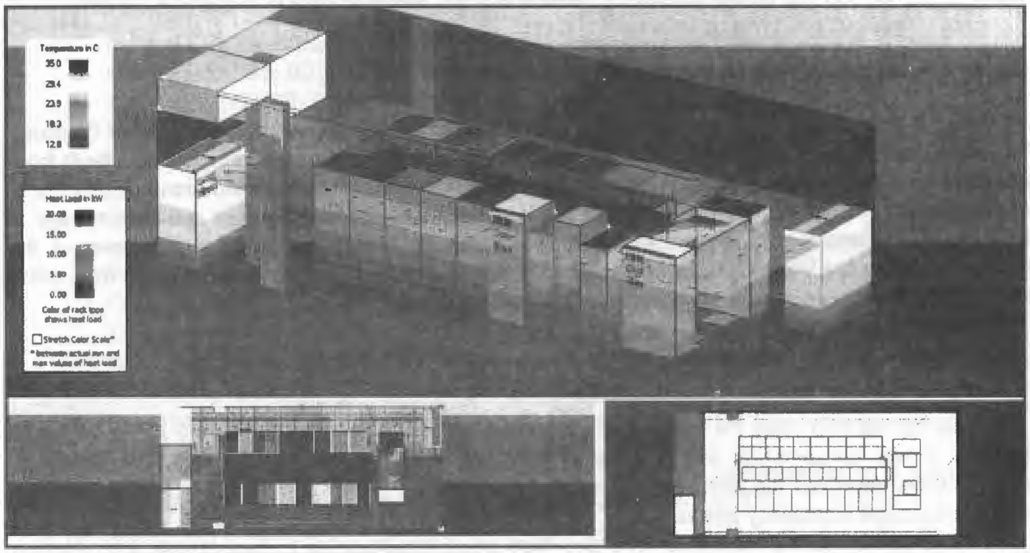


Fig. 2: Air flow model

At the beginning of 2010 we have purchased another two racks with water-cooled rear doors (one iDataPlex and one general-purpose rack mainly for 1U twin nodes). It means that currently we have 271 worker nodes (2688 cores) in water-cooled racks. The rest (service nodes, disk arrays, and about 130 older worker nodes) is still in open racks and cooled by air in the traditional way.

2.2 Emergency power supply

Since the start of operation, our server room was equipped with 200kVA UPS. This was sufficient till 2009 when the total power consumption got over the recommended 80% of the UPS capacity. An extension to the UPS system was found to be too expensive so the new worker nodes are currently not backed up in terms of power supply. We try to keep only service nodes “covered” by the UPS.

2. 3. Network

With the procurement of the new hardware another two factors related to network became a bottleneck in effective resource utilization:

- number of jobslots (or even HEPSpecs) per uplink ,
- number of TB per uplink.

The term “per uplink” means network path to the central router in our server room. The first case can be shown on an example that compares hardware from 2004 and iDataPlex from 2008, both placed in dedicated racks:

- 2004: 42 nodes with 8 HEPSpecs each, connected to one switch with 1Gb uplink - 336HEPSpecs/Gb,
- 2008: 84 nodes with 70 HepSpecs each, 2 switches with 1Gb uplink would be 2940HepSpecs/Gb.

The second is more straightforward, as the size of disk arrays grow the uplink must be strengthen up as well. We count this in terms of terabytes of disk space per one gigabit of bandwidth.

In 2009 these factors lead us to a redesign of the network topology in 2009 and a procurement of a 10Gbit Force10 S2410 switch. It is connected via 20Gbit interconnect to the main router (Cisco C6506). All new worker nodes are connected to switches with 10Gbit uplink and all new disk arrays with more than 20TB are required to contain 10Gbit NIC.

The schema of the topology of our network infrastructure is shown on Fig. 3.

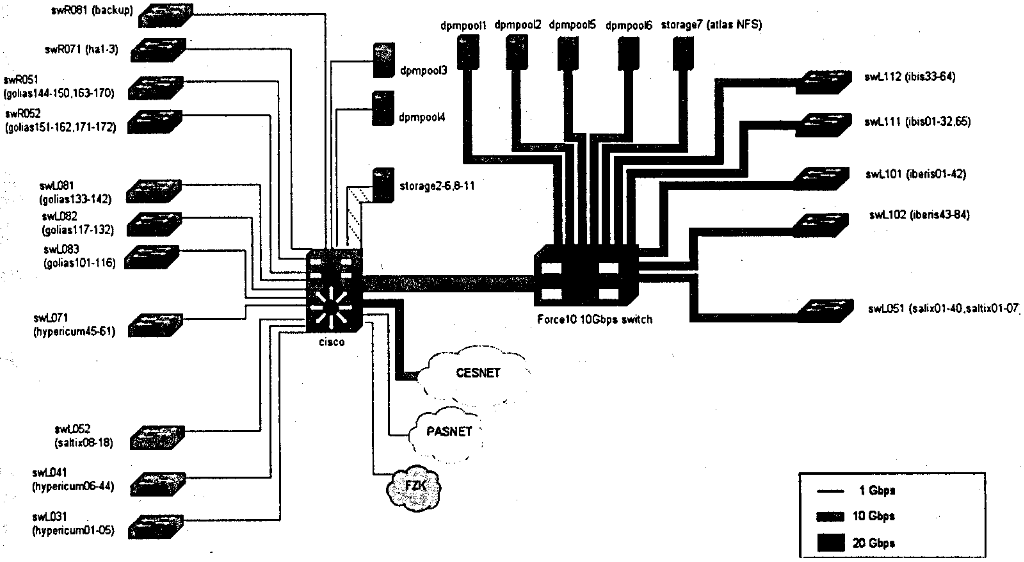


Fig. 3: The schema of the topology of network infrastructure

2.4. Summary of recent hardware changes

In 2009 we have installed:

- 84x IBM iDataPlex node x340,
    - 2x Xeon E5440 => 8 cores,
- 20x Altix XE 310 (twin nodes),
    - 2x Xeon E5420 => 8 cores,
- 3x Overland Ultamus 4800 (48TB raw each) SAN with 2x IBM X3650 as frontend servers,
- New backup solution,
    - 1x Overland Ultamus 1200 (12TB raw) with X3650 as frontend server,
    - Tape library NEO 8000,
- 2x VIA based NFS storage,
- 10 Gb switch Force 10 S2410,
- First decommission of computing nodes,
    - 29xHP lp1000 (Pentium III).

In 2010 we have installed:

- 65x IBM iDataPlex x360 nodes,

    - 2x Xeon E5520 with HT => 16 Cores,
- 9x Altix XE 340 (twin nodes),
    - 2x Xeon E5520 without HT => 8 cores,
- 3x Nexsan SataBeast (84TB raw each) SAN with 2x IBM X3650 M2 as frontend servers,
- 3x MSI-based storage nodes.

The rise in number of computing servers brought management challenges as well. We have improved our installation and configuration procedures. This is described in the next section.

We also insist on remote manageability of the servers for every procurement of our hardware.

3. Automatic installation, configuration and management

Currently there are 336 worker nodes with 2630 cores and approx. 20500 HEPSpecs, all dedicated to WLCG project. This part describes how we install and configure this cluster.

The Altix ICE system stands aside and it is managed and installed with SGI proprietary tools.

The most tedious work is the installation of worker nodes. We do this in the classical linux/redhat way:

1. Node is powered up and asked to boot from the network (it is a great help if the Baseboard Management Controller is able to change boot source remotely),
2. DHCP server provides the IP address,
3. According to the IP address the tftp server provides the correct installation kernel and parameters (mainly URL of the kickstart),
4. The node is then installed by the kickstart script,
5. At the end of installation the kickstart script tells the tftp server (via http request) that this node is reinstalled and so it will not be reinstalled again after reboot,
6. It also installs cfengine package and downloads basic cfengine configuration.

When the machine is installed with only basic operating system, the cfengine comes to play. It is run for the first time and so it downloads all the configuration files needed. After that it configures the node so it fulfills all its roles and duties. The detailed description of how this is achieved has been presented at NEC2009 and is available at [1].

4. Monitoring

Monitoring is a crucial thing in ensuring that problems with services are promptly discovered and fixed. We use several tools for detecting problems and visualization of the resource utilization:

4.1. Munin

Very simple system with rich set of sensors. It consists of a server that actively (once per 5 minutes) polls all the monitored nodes and stores performance data into rrd database. It requires a munin agent to run on the client side. This system provides neat graphs, but it does not scale well as it polls every monitored service on every monitored node. Munin does not send any notifications about detected problems. It is very simple to write a new sensor, it can be a script in any language and there are only few requirements on the output of the sensor.

At Golias farm we have developed several munin plugins mainly for temperature and fan sensors in servers.

4.2. Ganglia

Ganglia is a monitoring system with many built-in sensors oriented on large systems. It can automatically discover new hosts and their services. Ganglia presents the collected results in pretty rrd graphs. The scalability of Ganglia is great - it separates the presentation web and collecting agents.

Ganglia does not send any notifications and there is no concept of error state at all. Writing new sensors is more complicated and we did not develop any extension for Ganglia at Golias farm.

4.3. Nagios

State of the art in cluster monitoring. Easy to install and setup. Nagios community is very broad and there are many custom sensors and scripts for various hardware. Nagios implements sophisticated notification system to limit false positives when reporting problems (e.g. it tests the service multiple times in a row, it does not send notifications in downtime period etc.) We use Nagios as the main monitoring tool and we have developed many sensors to check the status of the special hardware (UPS, air conditioning). Nagios is also capable of passively receiving results of checks performed outside of Nagios. This way we insert SNMP traps from disk arrays into Nagios and we check system logs on the central syslog server for suspicious records.

Nagios does not automatically discover hosts nor services and it has got similar problems with scalability as munin has. Currently we solve the scalability issues by careful setting the checking period for every service type. Usual checks on worker nodes are scheduled every 20 minutes. Some long term checks (e.g. check if the running kernel is up-to-date) are scheduled to run only once a day.

There is a technical solution for scalability problems: check_mk. It is an extension that exploits the possibility to submit passive checks into Nagios and so it collects data on the monitored node and sends them all in one tcp connection (similar to ganglia agent). We are currently in the phase of testing this monitoring approach.

5. Other useful tools

There are two other tools developed at our farm that helps us to manage hardware and software services:

5.1. Hardware database

Every hardware component is recorded in our hardware database. It contains important data about hardware ownership (date of purchase, warranty dates), hardware operation (server location, power consumption, network connection) and system operation (DNS name, operating system installed, HEPSpec measured, history of hardware problems etc.).

5.2. Farmevents

Farmevents is a log for administrators with web and email interface. It features full text search, tagging and email notifications for every new record. You can see the screenshot of the service in Fig. 4.

In addition to the above, a new inventory database is currently under development. Please see the paper "Deska: Tool for Central Administration of a Grid Site" in these proceedings for details.

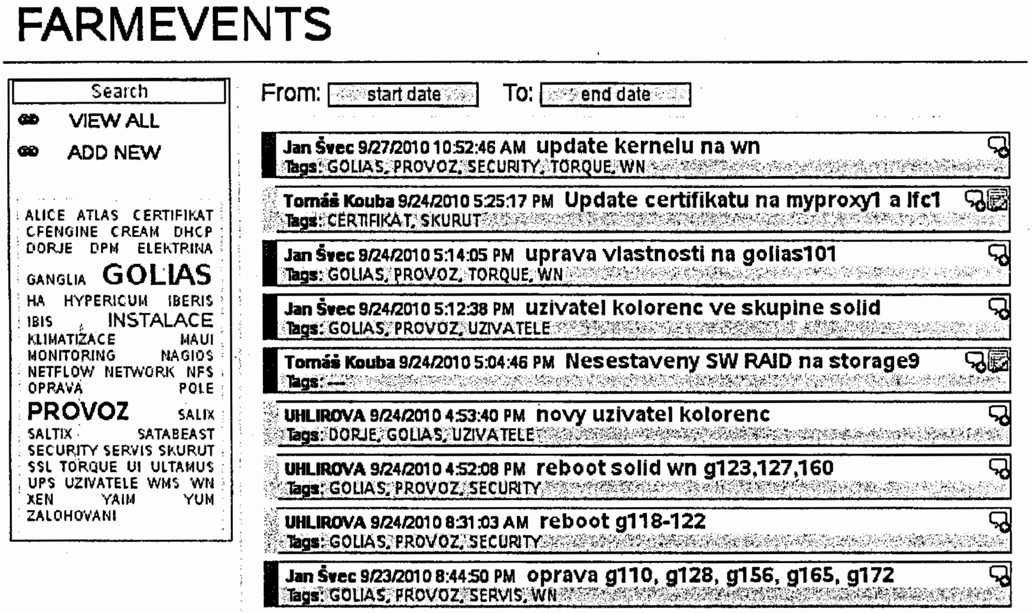
6. Network connection to outer world

Our external connectivity is delivered by CESNET (fig.5) - Czech network research institute. CESNET is part of the GEANT infrastructure and it provides the following network infrastructure for Golias farm:

- 1Gb connection to the Internet,
- 1Gb dedicated connection to FZK – Karlsruhe (our Tier 1 site in the Atlas experiment),
- 10 Gb connection to CESNET,
    - CESNET then provides several dedicated 1Gb lines (FNAL, BNL, ASGC, Czech tier 3 sites).

7. Participation in the EGI project

Praguelcg2 takes part in European grid projects from the early beginning. We participated in European data grid and all EGEE projects. When the EGEE became EGI this summer, our site became the biggest site in newly born NGI_CZ (Czech National Grid Initiative). Our staff is the main part of Czech grid operations team and we are responsible for monitoring, accounting and first line support of NGI_CZ.



FARMEVENTS

Search

VIEW ALL

ADD NEW

ALICE ATLAS CERTIFIKAT
CFENGINE CREAM DHCP
DORJE DPM ELEKTRINA
GANGLIA GOLIAS
HA HYPERICUM IBERIS
IBIS INSTALACE
KLIMATIZACE MAUI
MONITORING NAGIOS
NETFLOW NETWORK NFS
OPRAVA POLE
PROVOZ SALIX
SALTIX SATABEAST
SECURITY SERVIS SKURUT
SSL TORQUE UI ULTAMUS
UPS UZIVATELE WMS WN
XEN YAIM YUM
ZALOHOVANI

From: start date To: end date

Jan Švec 9/27/2010 10:52:46 AM update kernelu na wn
Tags: GOLIAS, PROVOZ, SECURITY, TORQUE, WN

Tomáš Kouba 9/24/2010 5:25:17 PM Update certifikatu na myproxy1 a lfc1
Tags: CERTIFIKAT, SKURUT

Jan Švec 9/24/2010 5:14:05 PM uprava vlastnosti na golias101
Tags: GOLIAS, PROVOZ, TORQUE, WN

Jan Švec 9/24/2010 5:12:38 PM uzivatel kolorenc ve skupine solid
Tags: GOLIAS, PROVOZ, UZIVATELE

Tomáš Kouba 9/24/2010 5:04:46 PM Nesestaveny SW RAID na storage9
Tags: —

UHLIROVA 9/24/2010 4:53:40 PM novy uzivatel kolorenc
Tags: DORJE, GOLIAS, UZIVATELE

UHLIROVA 9/24/2010 4:52:08 PM reboot solid wn g123,127,160
Tags: GOLIAS, PROVOZ, SECURITY

UHLIROVA 9/24/2010 8:31:03 AM reboot g118-122
Tags: GOLIAS, PROVOZ, SECURITY

Jan Švec 9/23/2010 8:44:50 PM oprava g110, g128, g156, g165, g172
Tags: GOLIAS, PROVOZ, SERVIS, WN

Fig. 4: The screenshot of the service

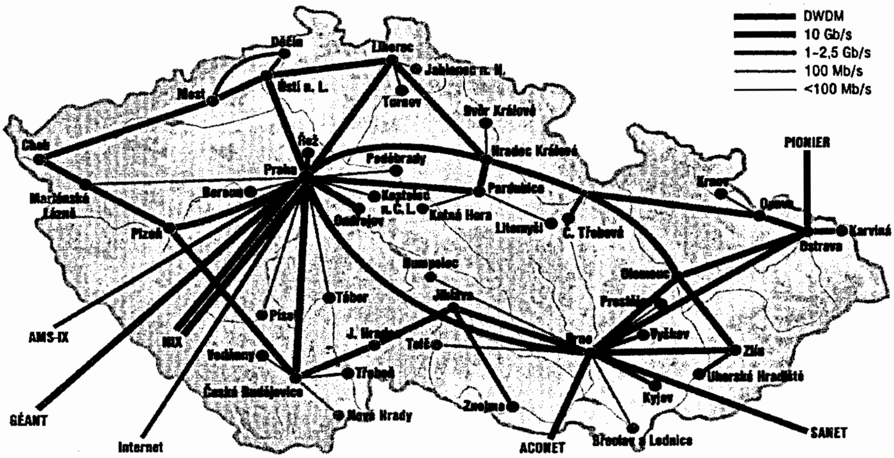


Fig. 5: CESNET topology

References

[1] http://nec2009.jinr.ru/docs/4/nec2009_kouba_cfengine.pdf

EXPANDING SCIENTIFIC COMPUTATIONAL INFRASTRUCTURES WITH DESKTOP GRIDS

R. Lovas[1], A. H. L. Emmen[2]

[1]*MTA SZTAKI, Kende u. 13-17, Budapest, 1111, Hungary*
rlovas@sztaki.hu
[2]*AlmereGrid, James Stewartstraat 248, 1325JN, Almere, The Netherlands*
ad@almeregrid.nl

The project 'Desktop Grids for International Scientific Collaboration' (DEGISCO) [1] is aimed at supporting and expanding Desktop Grids in order to provide more computational resources for research infrastructures, and to link these systems to scientific Grids maintained by the European Grid Infrastructure (EGI) and other initiatives. The proposed methods and solutions in DEGISCO are based on a generic grid-to-grid bridging technology that has been developed in the framework of EDGeS project [2]. This paper describes the main approach and some results from various points of view; infrastructure expanding, application porting, and community building.

1. Introduction

Desktop Grids consist of computers and other devices, including desktop PCs and notebooks that are used for general purposes but having unused computational and storage capacities. These distributed infrastructures can be formed inside research institutes and universities (local Desktop Grids) or by citizens that voluntarily donate unused computing time to science (volunteer Desktop Grids). Both types of Grids collect the underutilized resources and can offer them for scientific simulations or other applications. To be more useful for researchers and students, Desktop Grids have to be integrated into scientific workflows on a regular basis; the bridge between Desktop Grids and traditional service Grids, and the appropriate application development methodology can foster this integration.

The DEGISCO project transfers the knowledge concerning this combined European distributed infrastructure towards other countries by supporting the creation of new Desktop Grids for e-Science in the partner countries (see Section 2). The project members (including ISA RAS) connect the different types of Grids using the bridge technology, support the production level combined Grid infrastructure, assist in porting applications, as well as disseminate, promote and provide training about the Grid and its usage. Several scientific applications with large user communities are already available but new scientific applications are to be ported (see Section 3) in order to benefit of the infrastructure.

The International Desktop Grid Federation (see Section 4) has been set-up to exchange experience about the usage of Desktop Grid technology to expand scientific infrastructures, and in order to bring together Grid operators, application developers, and other key players.

2. Infrastructure

The EDGeS infrastructure with more 100.000 CPUs has been built up from several Desktop and Service Grids [2], and can be exploited in several ways by the scientists:

1. Desktop Grid and Service Grid operators can connect their grids to others through the EDGeS infrastructure. This increases the amount of computing resources available to the users. Grid operators can also take advantage of the applications that are ported to the Grid, and offer them to their scientific user communities.
2. Virtual Organisation (VO) managers of gLite-based Grids [3], or one of the other connected service Grids, can connect their Grids VO to the EDGeS VO and add resources and applications.

3. Grid application users on an EDGeS connected Grid can use more computing resources than are available on their 'own' Grid. When they use an application that is ported to several Grids, it will also automatically run on those Grids and jobs need to be submitted only on the owned Grid. Because of that, there is no need to learn the peculiarities of other Grids.
4. For people who want to donate their unused computing time to science, just need to connect to one of the Grids in the EDGeS infrastructure, and they become part of the largest computing Grid in the world.

The project (at the beginning) focuses mostly on the first two cases, and provide information for system administrators to understand and join the EDGeS integrated EGEE-DG infrastructure.

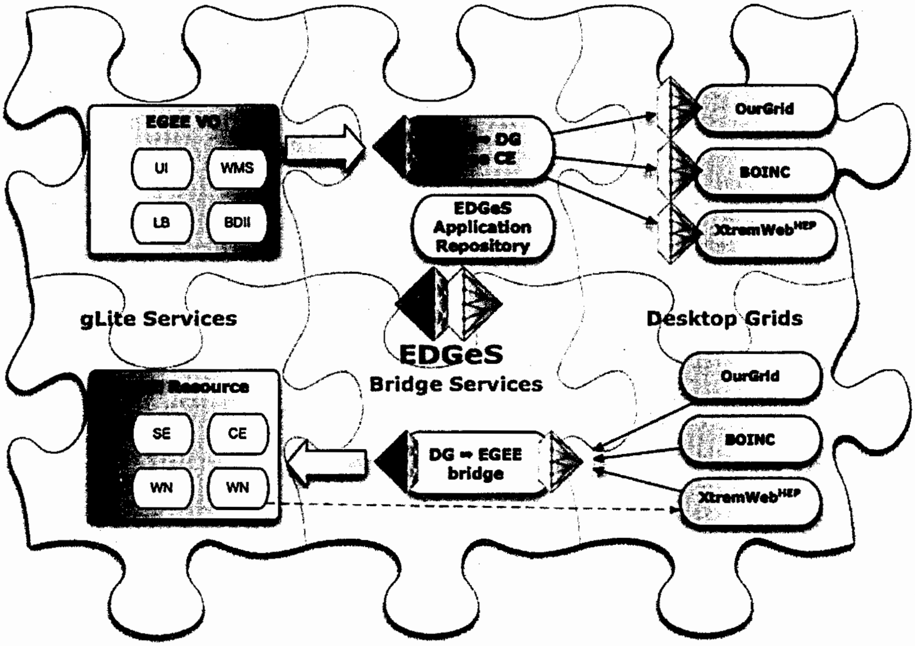


Fig. 1: Generic approach for bridging Grids in DEGISCO

The EDGeS production infrastructure is based on the following major components (see fig.1):

- The EDGeS bridge services provide bridges to connect EGEE VOs to Desktop Grids based on BOINC [4], XtremWeb-HEP [5], OurGrid [6][7] and vice versa. There are 4 kinds of bridges:
    1. **EGEE ⇒ DG bridge** which acts like a gLite CE [3] but instead of Worker Nodes (WNs) it can connect a BOINC, or a XWHEP, or an OurGrid desktop grid to any EGEE VO. There are two flavours of this bridge: one that uses an lcg-CE and another that uses a CREAM CE.
    2. **BOINC ⇒EGEE bridge** which acts like a BOINC client but executes downloaded WUs in an EGEE VO.
    3. **XWHEP ⇒EGEE bridge** which can connect WNs from an EGEE VO to an XWHEP desktop grid (in a way similar to pilot jobs) in order to execute desktop grid jobs.
    4. **OurGrid ⇒EGEE** bridge which allows OurGrid jobs to be submitted through the OurGrid user interface and be executed in WNs from any EGEE VO.

- An EGEE VO named `desktopgrid.vo.edges-grid.eu` which is operated by the DEGISCO project solely for executing jobs coming from desktop grids via the BOINC ⇒ EGEE, XWHEP ⇒ EGEE, and OurGrid ⇒ EGEE bridges.
- Several desktop grids that are connected to EDGeS and support EGEE applications including the three flavours (BOINC, XWHEP, and OurGrid) of EDGeS@home that are dedicated to run EGEE applications and allow volunteers to contribute to EDGeS and EGEE.

There are different installation and configuration instructions available depending on what kind of component is to be deployed, the detailed guideline with policies is available at the project website [1].

3. Applications

As one the most important objective, DEGISCO ports existing applications to the integrated infrastructure and provides a seamless job execution mechanism among the interconnected Service and Desktop Grid systems.

3.1. EDGeS Application Development

Fig. 2 illustrates the stages of the applied EDGeS Application Development Methodology [8], and names the participants and expected outputs of every stage. As it is shown on the figure the EADM aids the developers through the whole lifecycle of application porting, from identification of potential applications to providing support and upgrades for end-users. Here we only refer those stages that have been relevant and crucial at the beginning of DEGISCO project.

The current work addresses the first three stages of the EADM: *Analysis of current application, Requirements analysis, and Systems design.*

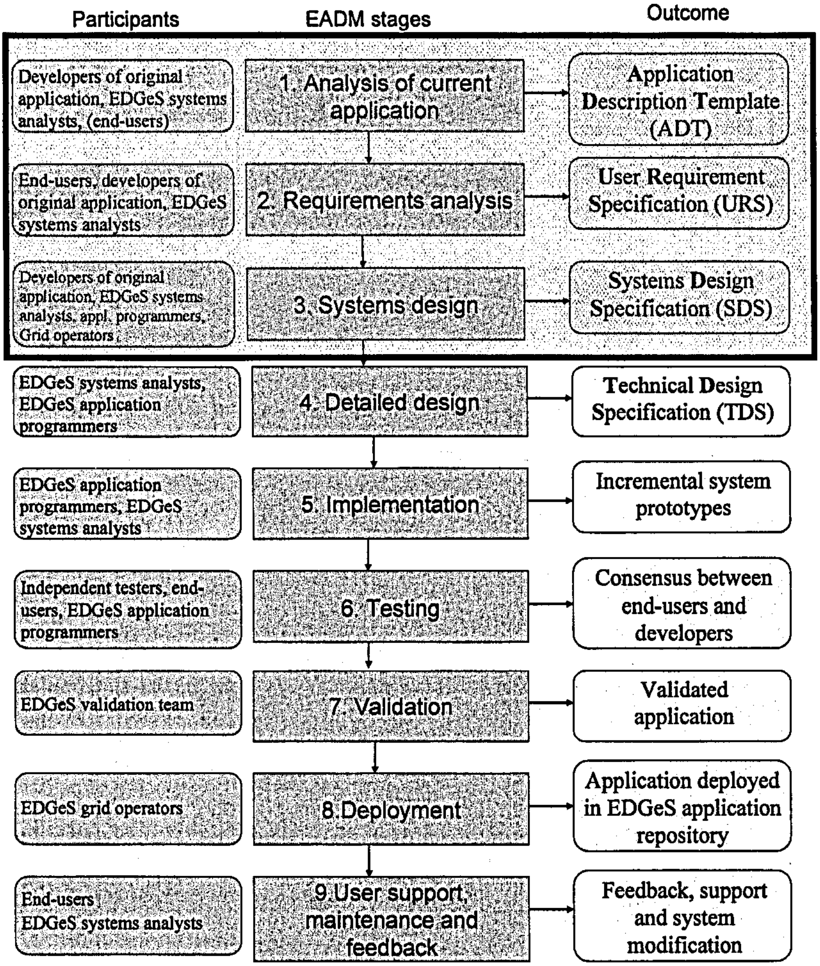
The aim of the *Analysis of current application* stage is to describe the currently existing application in detail. The EADM does not deal with the development of new applications but it provides a methodology describing how to port existing applications to the combined SG/DG platform. Therefore, the EADM assumes that the application is already exploited by the target user community. The aim of the porting process is to improve the usability of the application and to extend the target user community. The outcomes of this stage are summarised in an Application Description Template (ADT). The ADT describes the currently existing application and answers questions related to the identification of the target user community and problem domain, type of computing platform, parallelism, data access and functionalities currently offered. Other factors, such as licensing issues, security solutions and ethical and gender issues are also described.

It is defined during the *Requirements analysis* stage how the target user community will benefit from porting the application to the SG/DG platform. End-user involvement at this stage is crucial to capture their requirements towards the final ported application. The requirements towards the ported application concerning efficiency of execution and data access are analysed from a user perspective. The outcome of this stage is a User Requirement Specification (URS) document. The user requirements may have to be refined later on based on the output of technical investigations and limitations. Therefore, revisiting this phase may be required.

The aim of the *Systems design* stage is to answer the major questions of systems design principles concerning the ported application. What will be the target platform for execution and how will it be accessed? What level/type of parallelism will be utilised? What data access mechanisms will be applied? The outcome of this stage is a Systems Design Specification (SDS) document that may have to be modified according to technical constraints identified during the forthcoming stages.

As it was stated earlier, the applications described in this document have gone through these first three stages of the EADM and the above three documents have been produced for each of them. However, these documents are subject to change based on the results of further investigation or any constraints discovered later regarding the implementation.

At the beginning of the project, BNB-GRID (see the next section) is one the application candidates has gone through these above described steps.



Participants
EADM stages
Outcome
Developers of original application, EDGeS systems analysts, (end-users)
1. Analysis of current application
Application Description Template (ADT)
End-users, developers of original application, EDGeS systems analysts
2. Requirements analysis
User Requirement Specification (URS)
Developers of original application, EDGeS systems analysts, appl. programmers, Grid operators
3. Systems design
Systems Design Specification (SDS)
EDGeS systems analysts, EDGeS application programmers
4. Detailed design
Technical Design Specification (TDS)
EDGeS application programmers, EDGeS systems analysts
5. Implementation
Incremental system prototypes
Independent testers, end-users, EDGeS application programmers
6. Testing
Consensus between end-users and developers
EDGeS validation team
7. Validation
Validated application
EDGeS grid operators
8. Deployment
Application deployed in EDGeS application repository
End-users
EDGeS systems analysts
9. User support, maintenance and feedback
Feedback, support and system modification

Fig. 2: EADM stages, participants, and outputs

3.2. BNB-GRID

This section describes an application that has been identified by ISA RAS (DEGISCO project partner from the Russian Federation), which has been currently investigated regarding their portability to the target SG/DG platform. The section gives a short summary of the work carried out so far and also the proposed work that will be carried out as part of the project. Detailed EADM documentation for this application (among others) is downloadable from the DEGISCO website [1].

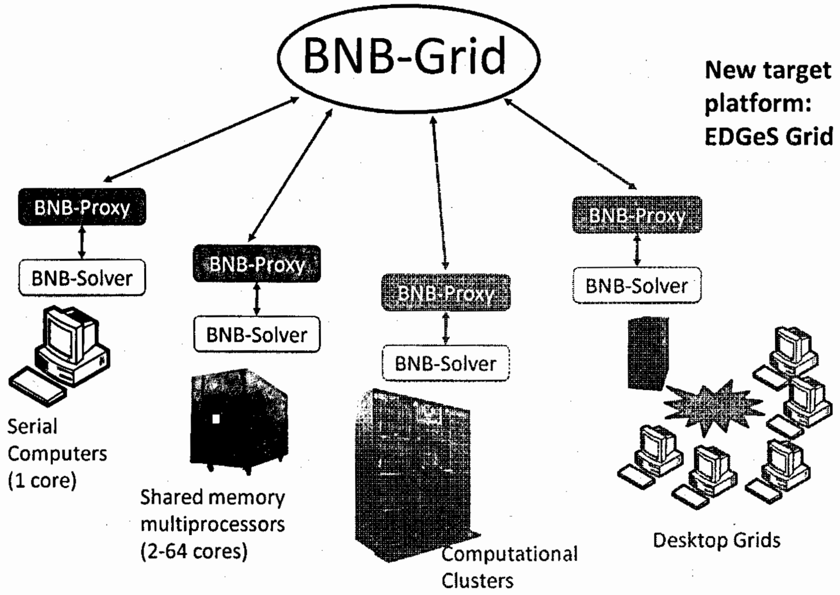


Fig. 3: BNB-GRID main architecture and supported platforms

BNB-Grid [9] is a generic framework for implementing optimization algorithms on distributed systems developed by ISA RAS. Current status is that the BNB-Grid tool can harness the consolidated power of computing elements collected from service Grids, desktop Grids and standalone resources to solve hard optimization and combinatorial problems. Adding new type of computational resource is available; currently the tool supports SSH, Unicore service Grid, and BOINC desktop Grid system. BNB-Grid central manager submits applications to different computing elements and organizes their interaction via specially designed protocol and uses hierarchical two-level work distribution scheme. The top-level distribution is done by a central manager: it sends or requests work chunks from running BNB-Solver applications via BNB-Proxy components. BNB-Solver instances run on individual nodes, supercomputers or in the Grids (see Fig. 3). The work chunk consists of several workunits where the notion of unit depends on the running optimization method.

BNB-Grid has already been tested in a desktop Grid environment and in the EDGeS test infrastructure. As the next step, BNB-Grid will be deployed on the new ISA RAS Desktop Grid, and the BNB-Grid project will be available for the Service Grid part of the EDGeS distributed computing infrastructure as well.

Then BNB-Grid will be tested on different optimization problems of different scale to study the performance and reliability of the developed application in the desktop Grid environment. The performance will depend on the number of desktops attached to the project. ISA RAS is going to start with a small institutional DG and then enlarge this DG by adding computers from other institutions (mainly from Universities and Academic institutions) and volunteer PCs.

To enable access for a large operational research community ISA RAS will establish and maintain a Web-portal for interactive submission of BNB-Grid tasks, so the optimization research community will be able to exploit BNB-Grid easily by using methods currently implemented or by extending the BNB-Grid with new methods/problems (by implementing new methods based on provided generic skeletons).

The results will be promoted and supported via the International Desktop Grid Federation (see the next section) as well.

4. The International Desktop Grid Federation

Desktop Grid operators and developers today often work in isolation. This means that each encounters the same difficulties, has to avoid the same pitfalls, and cannot take advantage of experiences made elsewhere. That is why DEGISCO, together with its sister project EDGI [10], founded the International Desktop Grid Federation (IDGF). The IDGF is a platform that brings together operators of Desktop Grids, and developers for these types of infrastructures. IDGF is setup as a membership organisation.

The International Desktop Grid Federation (IDGF) is also a corner stone of the sustainability plan for DEGISCO. The DEGISCO project is a two-year project with the goal of advancing Desktop Grid computing and embedding Desktop Grids in eScience infrastructures. This work is not finished after two years; the IDGF will assure the sustainability of the DEGISCO work. By setting up the IDGF as a membership the organisation that right from the start, and doing all DEGISCO dissemination, support, training and documentation activities under the brand name 'International Desktop Grid Federation' we prepare for sustainability right from the start of the DEGISCO project.

From a technology point of view, the International Desktop Grid Federation has the goal of promoting Desktop Grid technologies, advancing Desktop Grid technologies and advancing the usage of Desktop Grid technologies. Desktop Grid technologies are to be understood in a broad sense, i.e. not only stand-alone volunteer computing Grids, but also interconnections of Desktop Grids with other infrastructures, especially when they use the EDGeS Bridge technology. The International Desktop Grid Federation brings together organisations (institutes, companies, universities) that operate Desktop Grids or infrastructures that incorporate Desktop Grids, and organisations or groups that run and develop applications on these infrastructures. Hence the International Desktop Grid Federation is set up as a member organisation.

The main members of the IDGF are companies, universities, institutes, etc. The Federation will advance Desktop Grid technology and use, because of exchange of experience between members and collaboration between members. The EDGI and DEGISCO projects will provide initial support to the International Desktop Grid Federation. The International Desktop Grid Federation is set up to be long-lived, i.e. to continue after the EDGI and DEGISCO projects are finished. Probably IDGF will continue as an independent organisation, but not necessarily. We try to align the IDGF as much as possible with existing e-Infrastructure organisations, such as EGI, so it could also be possible that (part of) the Federation could become a user group in one of these e-Infrastructures organisations.

The International Desktop Grid Federation will consist of two chapters:

- The European Desktop Grid chapter,
- The International Desktop Grid chapter.

The European Desktop Grid chapter will organise the Grid operators and Application developers in the European Union. The International Desktop Grid chapter will organise the Grid operators and Application developers outside the European Union, starting with those in the International Cooperation Partner Countries (ICPC). This division also helps alignment with for instance EGI and the activities of the EGI and other European e-Infrastructure initiatives.

The members of the IDGF will also be organised according to activity and interest:

- General interest group on Desktop Grids Operations. This group will discuss general topics and issues when operating a Desktop Grid.
- General interest Group on application porting and running. This group will discuss general topics and issues when porting applications to a Desktop Grid.
- Integrating Desktop Grids with (existing) Service Grid Interest Group. This Group will look into setting up Desktop Grids as part of a larger e-

Infrastructure (such as EGI). The Desktop Grid is the main entry into the e-Infrastructure. We expect this to become true in a number of ICPC countries

- Interest Group on Integrating Desktop Grids with new type of Service Grids. A group for discussions where Desktop Grids are used mainly as accelerators for Service Grids. (Supported by the EDGI project)
- Interest Group on integrating Clouds and Desktop Grids. This group will look into the newly developed EDGI technologies (Supported by the EDGI project).

For each of these Interest Groups, a section of the portal has been setup, with membership list, special web pages, etc. The persons working in the EDGI and DEGISCO projects on the topic of an Interest Group, will join the appropriate Interest Group and start the activity.

5. Conclusions and Future Work

The DEGISCO project is unique in bringing scientific and technological knowledge to all corners of the world. The importance is not so much the developments that DEGICO makes itself, but scientific discovery enabling it will have. Making massive amounts of computing capacity available, will allow researchers from all kinds of disciplines to do new science. In this paper we reported one such example from ISAS RAS.

Also, setting up an organisation, IDGF, that right from start of the project is intended is a unique feature of DEGISCO. Most European projects start with a vague idea of what to do after the project has finished, but by the end of the project there is not enough time left to really set-up a sustainable follow-up.

IDGF also offers the opportunity for countries or regions to set-up local chapters that can help with harnessing otherwise unused Desktop Computing power. We are now investigating for instance the formation of a Russian chapter.

DEGISCO will investigate how green Desktop Grids are, and in what circumstances does it make sense to implement a Desktop Grid from an energy efficiency point of view [11].

6. Acknowledgements

The research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007- 2013) under grant agreement n° 261561 (DEGISCO).

References

[1] The DEGISCO project, http://degisco.eu

[2] Urbach E., Kacsuk P., Farkas Z. et.al. EDGeS: Bridging EGEE to BOINC and XtremWeb. Journal of Grid Computing, Vol 7, No. 3, 2009. P. 335-354.

[3] Burke S. et al., GLITE 3.1 USER GUIDE. EGEE - Manuals Series, December 18, 2009.

[4] Anderson D.P. BOINC: A System for Public-Resource Computing and Storage// In Proceedings of the 5th IEEE/ACM International GRID Workshop, Pittsburgh, USA, 2004.

[5] Fedak G. et al. XtremWeb: A Generic Global Computing Platform// In Proceedings of 1st IEEE International Symposium on Cluster Computing and the Grid CCGRID'2001, Special Session Global Computing on Personal Devices. IEEE Press, 2001. P. 582-587.

[6] Abmar Grangeiro de Barros, Adabriand Andrade Furtado, Francisco Brasileiro. Bridging OurGrid-based and gLite-based Grid Infrastructures// Proceedings of the Second EELA-2 Conference, Choroní, Venezuela, 2009.

[7] Cirne W. et al. Labs of the World, Unite!!!. Journal of Grid Computing Vol. 4. No. 3. Springer, 2006. P. 225-246.

[8] Kiss T. et al. Porting Applications to a Combined Desktop Grid/Service Grid platform using the EDGeS Application Development Methodology, INGRID 2009 workshop, Sardinia, Italy, 2009.

[9] Posypkin M. Solving hard practical global optimization problems in a distributed computational environment. 3rd International Conference 'Distributed Computing and Grid-technologies in Science and Education', GRID 2008, Dubna, Russia, 30 June - 4 July, 2008. P. 257-260.

[10] The EDGI project, http://edgi-project.eu

[11] Schott B. Green methodologies in Desktop-Grids// Proceedings of the 6th Workshop on Large Scale Computations on Grids and 1st Workshop on Scalable Computing in Distributed Systems, Wisla, Poland.

TOWARDS MOLECULAR SPINTRONICS: NRG METHOD AND DISTRIBUTED COMPUTING TO STUDY TRANSPORT PROPERTIES OF SINGLE-MOLECULE MAGNETS

M. Misiorny, I. Weymann, G. Musial, J. Barnas

Faculty of Physics, Adam Mickiewicz University, ul. Umultowska 85, 61-614 Poznan, Poland

We present results of numerical calculations of transport properties of single molecule magnets with the use of numerical renormalization group (NRG) method. We show that Wilson's idea of renormalization applied in energy space within the DM-NRG code manifests a good scalability. The NRG method is applied to the regime of strong coupling between the molecule and external ferromagnetic (in general) electrodes. From the calculated spectral function we get the zero bias conductance. The presented results reveal the Kondo anomaly which appears due to screening of the molecules spin by conduction electrons of the leads. Exchange coupling of the internal magnetic core of molecular magnets and conduction electrons in the transport orbital suppress the Kondo anomaly.

1 Introduction and motivation

In recent years, with the advent of novel experimental techniques allowing to study transport properties of single molecules, a number of molecule-based electronic devices have been proposed and constructed. In terms of possible applications, especially interesting seem to be single-molecule magnets (SMMs). Due to a large spin number and significant magnetic anisotropy, they exhibit an energy barrier for their spin reversal [1, 2] - especially at low temperatures. It has been suggested that spin-polarized current can be employed to modify magnetic state of a molecule [3-5]. The current-induced magnetic switching of SMMs may become a key mechanism to be utilized in future devices, as it does not require application of an external magnetic field for manipulating the molecule's spin.

From the computational point of view, among various theoretical aspects of transport through SMMs, especially interesting seems to be the problem of transport when the coupling between a molecule and electrodes is strong. In such a case, electrons cannot be considered as tunnelling between electrodes *via* pure molecular states, which is a standard approach for the limit of weak coupling. As the strength of the coupling grows, the energy spectrum of the molecule, initially characterized by a set of discrete energy levels, becomes modified. The interaction between the localized electron state of the molecule and extended electron states of electrodes results then in gradual broadening of the original molecular states. Furthermore, when the mixing of the states becomes significant, molecular states have to be substituted with new hybrid states, which take into consideration some degree of electron delocalization between electrodes and the molecule.

An interesting situation arises when the transport orbital of a molecule is occupied by a single electron. Since the strong coupling limit means that electrons can somewhat freely tunnel back and forth between electrodes and the molecule's transport orbital, such tunnelling processes lead to fluctuations of the unpaired electron's spin. This in turn results in an additional resonance in the spectral density (density of states) at the Fermi level of electrodes, known as the Rado-Suhl resonance. A further consequence of the resonance in density of states is the Kondo anomaly in electrical conductance of the molecule.

The key numerical problem is therefore to find a way how to derive the hybrid states and the corresponding energies. One of the most effective ways to achieve this objective is the numerical renormalization group (NRG) method [6]. In the present communication we thus address some main numerical aspects concerning calculations of SMM's transport properties with the use of the NRG approach. Especially, we examine the scalability of the problem and the resulting DM-NRG code as it is of exponential complexity.

2 Theoretical model

The model to be considered consists of an individual SMM embedded between two metallic, ferromagnetic electrodes. Transport of electrons through the molecule is then assumed to occur *via* the lowest unoccupied molecular orbital (LUMO) level. Electrons in the LUMO level are exchanged coupled to the inner magnetic core of the molecule. The Hamiltonian of the molecule HSMM(ε, *U, D, D_1*, D_2, *J)* in principle depends on four variables: the energy of the LUMO level *ε*, the energy *U* of the Coulomb interaction between two electrons of opposite spin in the LUMO level, the uniaxial anisotropy *D* (with small corrections *D_1* and *D_2* accounting for the influence of the molecule's reduction state on the anisotropy), and the energy *J* of exchange interaction between an electron in the LUMO level and the SMM's core spin. The explicit form of *H_{SMM}* can be found for instance in Ref. [4]. The electrodes, on the other hand, are described by noninteracting, itinerant electrons.

The above described model is based on some simplification and therefore its applicability is also limited. Anyway, it is a model which captures all basic features of SMMs and electronic transport. In more accurate descriptions one should include also coupling between magnetic molecule's core and electrodes, Coulomb interaction of electrons in the LUMO level and in the magnetic core, and others. Anyway, we will show that the model can be use to describe basic features of electronic transport in the strong coupling regime.

3 Numerical renormalization group method

In order to calculate the transport properties of SMMs strongly coupled to external leads we use the Wilson's numerical renormalization group method. The NRG is known as an essentially exact, very versatile and powerful method to address quantum impurity problems in general, and in particular transport through quantum dot and molecular systems coupled to leads [7]. The key idea of NRG consists in a logarithmic discretization of the conduction band and mapping of the system on a semi-infinite chain (the so-called Wilson chain) with the impurity sitting at the end of the chain. By diagonalizing the Hamiltonian at consecutive sites of the chain and storing the eigenvalues and eigenvectors of the system, one can calculate the finite-size energy spectrum, static quantities, as well as correlation functions and their temperature dependence.

From numerical point of view, the main steps of the NRG algorithm are: the construction of the new basis, Hamiltonian and respective operators at each iteration, diagonalization of the Hamiltonian, unitary transformation of operators, calculation of spectral functions. In a typical NRG calculation for single orbital level coupled to metallic leads (single-channel Anderson model), the initial Hamiltonian is a 4 × 4 matrix and at each iteration one needs to add 4 local states. This immediately implies that the size of the Hilbert space is 4^{k+1}, where *k* is the iteration index, and for large *k* the problem cannot be solved exactly. Therefore, one introduces a truncation scheme by keeping only fixed number of low-energy states at each iteration and discarding the other states. The discarded states, on the other hand, are then used to construct the complete basis of the Wilson chain and to define the density matrix of the full system (DM-NRG). Especially, using non-Abelian symmetries increases the efficiency of calculations, which is crucial in the case of more complex models such as multi-channel Kondo models.

From the above discussion follows that for more complex systems the calculation is even more costly numerically and one needs to truncate the Hilbert space at earlier iterations, though still trying to keep enough states for the calculation to be correct. For example, for a spin-full n-channel calculation one adds 4^n local states at each iteration and the Hilbert space grows extremely fast. It is therefore very important to use the symmetries that the considered Hamiltonian possesses in order to speed up and improve the calculation. In calculations we have thus used the free-access Flexible DM-NRG code which can tackle with arbitrary number of both Abelian and non-Abelian symmetries [8]. Especially, using non-Abelian symmetries increases the efficiency of calculations, which is crucial in the case of more complex problems such as various multi-channel Kondo or multi-orbital models.

As the most time and memory consuming steps of the whole algorithm one can consider the diagonalization of the Hamiltonian and the unitary transformation of operators at each iteration.

Therefore, for these steps we introduce parallelization of computing. In addition, one usually also needs to read and write a lot of data during the run, so that very efficient and fast hard drive system is desirable. We have thus obtained some support from the SPINLAB project for the respective multicomputer. For multiprocessors we use the OpenMP directives to parallelize processing whereas for multicomputers we exploit message passing and the MPI library. Parallelization of processing and data distribution enables one to consider larger systems and/or to get better precision of the results.

4 Numerical results and discussion

Some exemplary results obtained by the application of the NRG procedure are presented in Fig. 1, where the spectral density is presented as a function of the exchange coupling between magnetic core and electrons in the conducting (LUMO) level. Antiparallel configuration of electrodes' magnetic moments has been assumed there, so the system (for symmetric coupling to the left and right lead) behaves similarly to a nonmagnetic one. First, when the exchange coupling between magnetic core and electrons in the transport orbital vanishes, one finds typical behavior of a quantum dot, with a Kondo peak at the Fermi level. This Kondo peak in the spectral function (density of states) gives rise to a zero bias Kondo peak in conductance through the system.

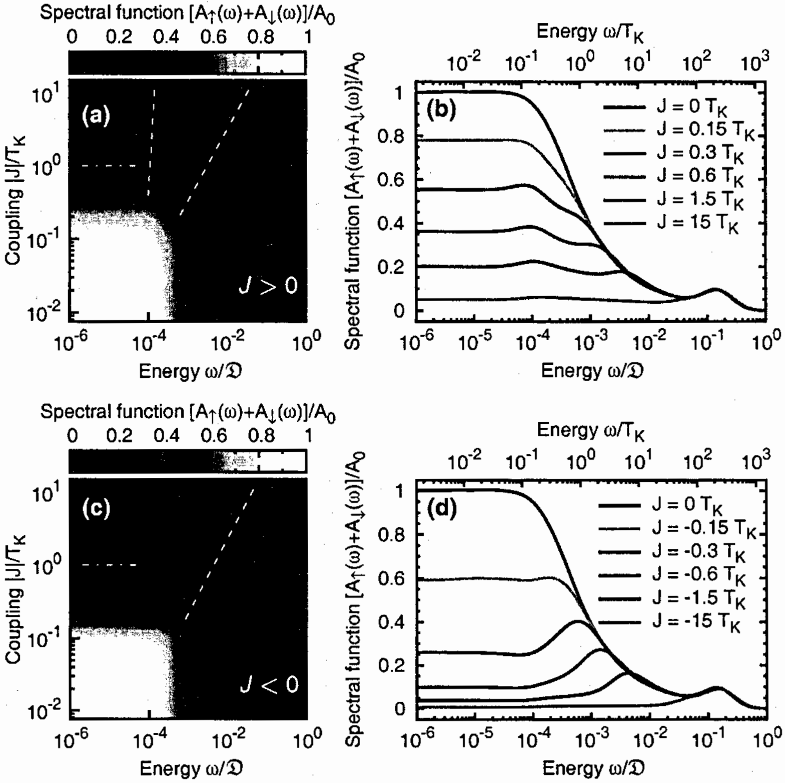


Fig. 1: The normalized total spectral function $A(\omega) = \sum_{\sigma} A_{\sigma}(\omega)$ of the LUMO level as a function of the coupling constant J for the antiparallel magnetic configuration of electrodes. Plots (b) and (d) represent cross-sections of the plots (a) and (b), respectively, for indicated values of J. Note that all spectral functions are normalized to $A_0 \equiv A(0)$ for a given J. Here, T_K refers to the Kondo temperature estimated here to be $T_K \approx 0.00066$ in units of $\mathfrak{D}$ ($k_B \equiv 1$). The parameters describing the molecule are as follows: $D = 5 \cdot 10–^{5}$, $D_1 = – 5 \cdot 10–^{6}$, $D_2 = 2 \cdot 10–^{6}$, $\varepsilon = −0.1$ and $U = 0.3$ (all given in units of $\mathfrak{D}$).

The system behaves differently for nonzero values of J, as shown in Fig.1 for both $J > 0$ and $J < 0$. When the absolute value of the parameter J increases, the Kondo anomaly in the spectral function becomes gradually suppressed. When $|J| \gg T_K$, the Kondo anomaly practically disappears. Thus, exchange coupling between the internal magnetic core and conduction electrons tunnelling through the LUMO level suppress the Kondo zero-bias anomaly in the conductance of a molecule attached to external leads.

When magnetic configuration of the leads' magnetizations is parallel, the Kondo effect is additionally suppressed by ferromagnetism of the electrodes. Accordingly, the Kondo resonance in spectral function (not shown) is suppressed even for $J = 0$. However, it can be restored by an external magnetic field.

5 Conclusions

We conclude that Wilson's idea of renormalization is applied in energy space within the DM-NRG code manifests a good scalability. Calculation of one curve obtained for hypothetical molecule with the small total spin $S = 2$ takes about 200 hours with 60 to 80 iterations in the DM-NGR code. As the algorithm is of exponential complexity, we need to run a code in parallel with hundreds of parallel processes to consider the real molecules with total spins S of order of 10 or even greater and to obtain the results in a realistic time.

The results seem to be encouraging for further studies of SMMs and to utilize them as elements of spintronic devices. Nevertheless, the ideas are still far from experimental realization, as the temperatures in which these effects are currently observed are around several K. Thus enormous efforts are concentrated on synthesizing of new SMMs with higher blocking temperatures but characterized by a decent energy barrier for the spin reversal.

The next problem is to understand how the deposition of a molecule onto a surface changes properties of the molecule, the crucial ones for the electronic transport. The first experiments probably will be observed for a SMM on a metallic but nonmagnetic substrate and the scanning tunneling microscope with a magnetic tip. Thus, it is worth to understand how ballistic transport of electrons through a molecule affects its structure and magnetic state.

6 Acknowledgements

The authors acknowledge support by Program POIG.02.02.00-00-020/09-00 (under SPINLAB project), which enabled presentation of the results, and also will enable more extended numerical work in future *via* supported computing facility.

References

[1] Bogani L. and Wernsdorfer W. Molecular spintronics using single-molecule magnets, Nature Mater. 7 (2008), no. 3, 179.

[2] Gatteschi D., Sessoli R., and Villain J. Molecular nanomagnets, Oxford University Press, New York, 2006.

[3] Misiorny M. and Barnaś J. Magnetic switching of a single molecular magnet due to spin-polarized current, Phys. Rev. B 75 (2007), no. 13, 134425.

[4] Misiorny M. and Barnaś J. Switching of molecular magnets, Phys. Stat. Sol. B **246** (2009), no. 4, 695.

[5] Misiorny M., Weymann I., and Barnaś J. Spin effects in transport through single-molecule magnets in the sequentail and cotunneling regimes, Phys. Rev. B 79 (2009), 224420.

[6] Wilson K. G. The renormalization group: Critical phenomena and the Kondo problem, Rev. Mod. Phys. 47 (1975), no. 4, 773S.

[7] Bulla R., Costi T. A., and Pruschke T. Numerical renormalization group method for quantum impurity systems, Rev. Mod. Phys. 80 (2008), no. 2, 395.

[8] Legeza Ö., Moca C.P., Tóth A.I., Weymann I., and Zaránd G. Manual for the flexible DM-NRG code, arXiv:0809.3143v1 (2008). (the code is available at http://www.phy.bme.hu/dmnrg/).

DISTRIBUTED COMPUTING AND OPTIMIZATION ALGORITHMS FOR INTERPRETATION OF X-RAY SCATTERING BY CARBON NANOSTRUCTURES IN THE DEPOSITED FILMS FROM TOKAMAK T-10[1]

V. S. Neverov[1], V. V. Voloshinov[2], A. P. Afanasiev[2], A. B. Kukushkin[1], N. L. Marusov[1], I. B. Semenov[1], V. G. Stankevich[3], N. Yu. Svechnikov[3], A. S. Tarasov[2], A. A. Veligzhanin[3], Ya. V. Zubavichus[3]

[1]*Tokamak Physics Institute RRC "Kurchatov Institute", 123182, Moscow, Russia*
[2]*Institute for System Analysis RAS, 117312, Moscow, Russia*
[3]*Kurchatov Center for Synchrotron Radiation and Nanotechnology*
RRC "Kurchatov Institute", 123182, Moscow, Russia

1. Introduction

The x-ray scattering by the films deposited in the vacuum vessel of tokamak T-10, carried out at the Kurchatov Synchrotron Radiation Center (wavelength λ = 0.0464 nm) [1], has shown the presence of a wide peak at low scattering angles θ (namely, at $q \sim 10$ nm^{-1}, where $q = |\vec{k}_s - \vec{k}_i| = (4\pi/\lambda)\sin(\theta)$ is modulus of scattering wave vector) which corresponds to fluctuations of elementary scatter's density in the few nanometers range. This peak appeared to be not explainable by contributions of typical impurity polycrystals and most popular nanostructures like fullerenes or straight carbon nanotubes [2].

Interpretation of these results requested a numerical modeling of x-ray scattering by chaotic and regular ensembles of carbon nanostructures of various topology (isolated structures, including spheres, tubes, ellipsoids and toroids) over a broad range of their geometric sizes. Here we report on using (i) parallel computing (MPI + OpenMP) for these calculations and (ii) remote optimization services for solving an optimization problem of identification of possible topological contents of carbon nanostructures in the films analyzed.

Our goal is to determine possible topological structural composition in the sample, responsible for three peculiarities of the scattered intensity curve. We propose the following method:

1. Restricting the class of the nanostructures, possibly responsible for the wide peak at $q \sim 10$ nm^{-1} by comparing the calculated x-ray scattered intensity curves with experiment;
2. Identification of possible topological contents of isolated carbon nanostructures in the sample within the above selected class of structures by means of an optimization method;
3. Full interference modeling of x-ray diffraction (XRD) by the above optimal ensemble of nanostructures in amorphous medium to take into account previously neglected interference terms (nanostructure–nanostructure, nanostructure–ambient medium) and to estimate the accuracy of the above optimization and the domain of its applicability.

2. Numerical code

Numerical code has been developed for modeling the x-ray scattering intensity for various nanostructures (first of all, carbon ones). Main features of the code are as follows:

- ***Input.*** Whole ensemble can consist of unlimited number of structures. Each structure is represented by an elementary block (e.g., unit cell of a crystals). Elementary blocks may have

[1] Supported by the Russian Foundation for Basic Research (grant RFBR 09-07-00469) and the European project EGEE-III (Enabling Grids for E-sciencE).

unrestricted number of copies, that could be organized in a regular (crystals) or irregular structure. For regular structures, translation vectors and translation numbers are specified. It is also allowed to take into account possible defects in crystal structure;

- ***Output.*** The 1D curve for x-ray scattered intensity, calculated with Debye formula, with possible output of particular interference terms from interference of the waves scattered by different structures or the 2D XRD pattern with possible averaging over angles of incident wave vector;
- ***Interface.*** Simple XML interface is used for initial parameters definition. Such interfaces are suitable for an automated use, e.g., in GRID. For initial data generation, processing and visualization of the results, a number of python scripts are used;
- ***Computation.*** Code is written in C/C++ and parallelized both with MPI and OpenMP.

Here we consider the parallelization algorithms used. For 1D x-ray intensity curve the calculation with double summation over scattering centers (i.e. atoms) in the Debye formula (1) was distributed between MPI processes:

$$S(q)=\sum_{i=1}^{N_{at}}\sum_{j=1}^{N_{at}} f_i(q)f_j(q)\frac{\sin(qr_{ij})}{qr_{ij}}, \tag{1}$$

where q is the modulus of scattering wave vector, r_{ij} is the distance between i-th and j-th atoms, $f(q)$ is the atomic scattering form-factor. Simplified scheme is shown in the Table 1.

Table 1. Simplified scheme of atomic data (coordinates, etc.) distribution between MPI processes for 1D Debye case: p# is MPI process number, $N_p = k(k+1)/2$ is total number of processes, where k is non-negative integer

atoms		*i*			
		1 2 3 …			N_{at}
	1 2 3 …	p0	p1	p2	p3
j			p4	p5	p6
				p7	p8
	N_{at}				p9

Since the partial sums are computed independently, the synchronization is needed only before and after the computation. High parallel efficiency is achieved with a very simple realization. The only condition here is $N_p \ll N_{at}$, where N_{at} – the total number of atoms in ensemble, and it is always satisfied, because for small N_{at} the computation time is so small that no parallelization is needed. In addition, each point of discrete space of scattering vector modulus q is computed independently as an OpenMP thread.

In the 2D XRD pattern calculations, the intensity matrix $S(q,\varphi)$ is distributed between MPI processes, and each point (q,φ) is computed independently as an OpenMP thread.

3. Optimization algorithms

Determination of topology and size distribution of isolated nanostructures was carried out by minimizing the following errors:

$$Z_j(\mathbf{x},a,b,A) = S_{\exp}(q_j) - \sum_{i=1}^{N} S_i(q_j)x_i - a\left(f_C(q_j)\right)^2 - A\sum_{k=1}^{N_{im}} \alpha_k \left(f_k(q_j)\right)^2 - b, (j = 1:m), \quad (2)$$

where $S_{exp}(q_j)$ is experimental XRD intensity curve, $S_i(q_j)$ is the calculated XRD intensity curve for i-th carbon nanostructure divided by the number of atoms in the structure, j is the number of point in the discrete space of scattering vector modulus q, $A = \sum_{i=1}^{N} x_i + a$, x_i/A is probability of carbon atom to belong to the i-th nanostructure, N is total number of selected nanostructures, a/A is probability of carbon atom to belong to amorphous medium, $f_c(q_j)$ is XRD form-factor of single carbon atom, α_k is the ratio of the number of k-th impurity atoms (including hydrogen) to that of carbon atoms in the sample, taken from the experiment [3], N_{im} is total number of impurity atoms in the sample, $f_k(q_j)$ *is* XRD form-factor of the k-th impurity single atom, b is the value of possible constant background signal, 0.5 a.u.is the average error of experimental data, m is the number of points on the axis of the variable q.

The additional constraints are as follows:

$$S_{\exp}(q_j) - a\left(f_C(q_j)\right)^2 - A\sum_{k=1}^{N_{im}} \alpha_k \left(f_k(q_j)\right)^2 - b \geq -0.5, (j = 1:m) \quad (3)$$

$$x_i, a \geq 0(i = 1:N) \quad (4)$$

The optimization was carried out for three different criteria:

$$\sum_{j=1}^{m} \left|Z_j(\mathbf{x},a,b,A)\right| \xrightarrow[\mathbf{x},a,b,A]{} \min$$ – L_1: minimization of the sum of error's absolute values, (5)

$$\sum_{j=1}^{m} \left(Z_j(\mathbf{x},a,b,A)\right)^2 \xrightarrow[\mathbf{x},a,b,A]{} \min$$ – L_2: minimization of the sum of error's squared values, (6)

$$\max_{j=1:m} \left|Z_j(\mathbf{x},a,b,A)\right| \xrightarrow[\mathbf{x},a,b,A]{} \min$$ – L_{inf}: minimization of maximum absolute value of errors, (7)

The expression for the error $Z_j(\mathbf{x},a,b,A)$ is linear in the variables x_i. So for criteria L_1 and L_{inf} the GNU Linear Programming Kit [4] was used. In the case L_2, the standard least-squares method cannot be used because of additional inequality constraints (x_i, a >= 0). In this case, the non-linear optimization package MINOS, available for remote use on NEOS project servers [5], was used.

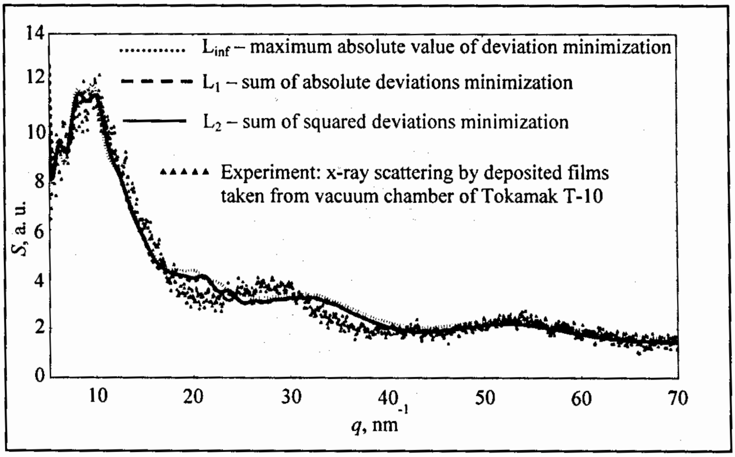


Fig. 1: Comparison of the curves, obtained by minimizing with three different criteria, with the experiment

4. Optimization results

Based on preliminary analysis, the structures with the following topologies and dimensions were selected:

- Tubes and half tubes: 0.4 nm ≤ R ≤ 0.6 nm,
- Ellipsoids and half-ellipsoids: 0.5 nm ≤ R_x ≤ 0.7 nm, $R_x \le R_y \le 2R_x$,
- Toroidal carbon nanotubes (toroids) [6,7] with elliptic cross-section and half-toroids: 0.85 nm ≤ R ≤ 1.25 nm, 0.3 nm ≤ R_x ≤ 0.4 nm, 0.5 nm ≤ R_y ≤ 0.9 nm.

The XRD intensity profiles $S_i(q)$ for N = 450 nanostructures were calculated in GRID. The fig. 1 shows optimization results for the criteria (5)-(7).The results of optimization procedure are shown in the Table 2.

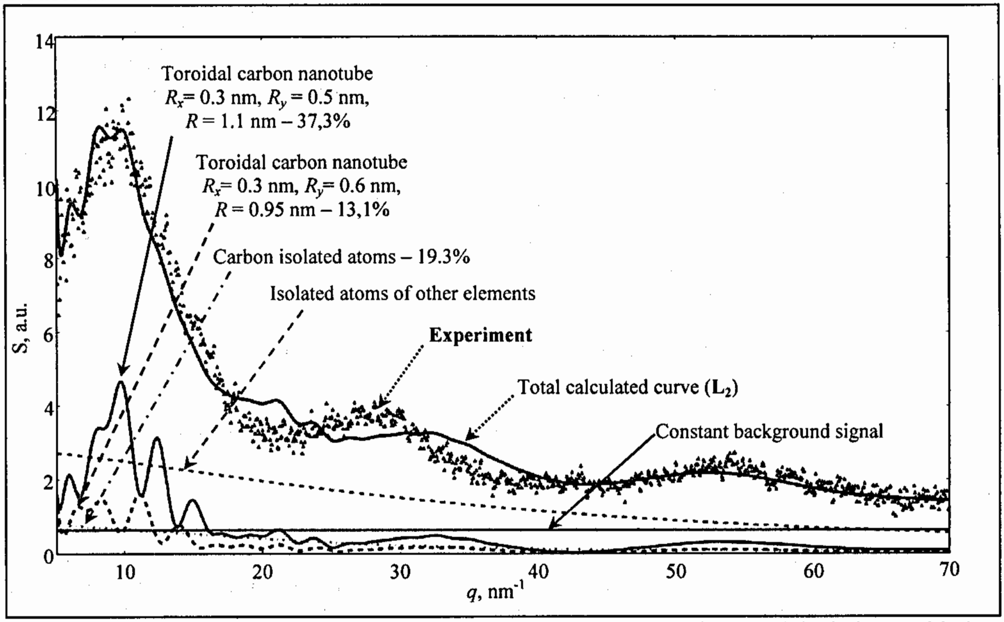


Fig. 2: Comparison of the curve, obtained by minimizing the sum of squared deviations, with the experiment. The most significant partial contributions are indicated

Atoms Num.	Topology	Dimensions			Probabilities		
		R_x, nm	R_y, nm	R, nm	L_1, %	L_2, %	L_{inf}, %
631	toroid	0.3	0.5	1	6.75	7.16	[illegible]
663		0.3	0.5	1.05	19.43	4.95	0
675		0.3	0.6	0.95	24.35	13.07	0
694		0.3	0.5	1.1	24.77	37.27	16.11
704		0.3	0.55	1.05	0	0	14.71
710		0.3	0.6	1	0	8.76	6.76
750		0.35	0.6	1	0	0	0.42
757		0.3	0.5	1.2	0	0	12.74
771		0.3	0.55	1.15	0	0	16.51
862		0.3	0.65	1.15	0.73	0	0
900		0.3	0.65	1.2	0	7.44	0
454	half-toroid	0.3	0.7	1.15	14.0	2.01	0
	tube			4.0	0	0	1.04
Carbon isolated atoms					9.96	19.34	21.44

Table 2. Probability of carbon atom to belong to nanostructure of given topology and dimensions (comparison of three different optimizations)

Structure–structure and structures–amorphous medium interference terms as well as atom–atom interference in medium has been neglected in the above optimization procedure. We carried out full-interference XRD modeling by the ensemble of nanostructures, previously determined by optimization procedures in amorphous medium, to estimate the accuracy of optimization-based identification and the conditions of applicability. The figure 3 shows the model volume element (~$1.3 \bullet 10^5$ atoms) of the sample with the contents of carbon nanostructures, close to that determined in the case of L_2 optimization, and the contents of other chemical elements, taken from experiment [3].

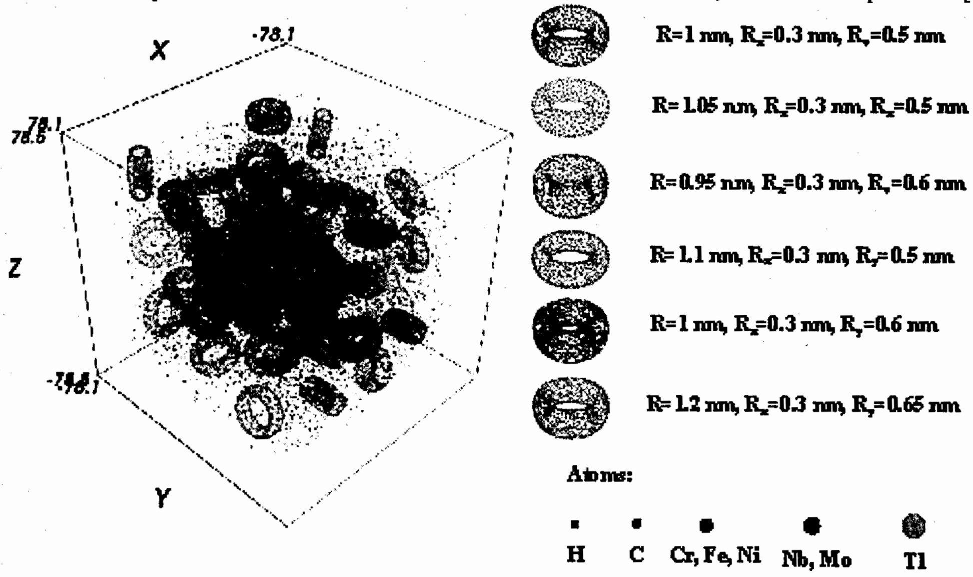

X
-78.1
78.1
78.5
Z
-78.1
Y
R=1 nm, R_x=0.3 nm, R_y=0.5 nm
R= 1.05 nm, R_x=0.3 nm, R_x=0.5 nm
R=0.95 nm, R_x=0.3 nm, R_y=0.6 nm
R= 1.1 nm, R_x=0.3 nm, R_y=0.5 nm
R=1 nm, R_x=0.3 nm, R_y=0.6 nm
R= 1.2 nm, R_x=0.3 nm, R_y=0.65 nm
Atoms:
H
C
Cr, Fe, Ni
Nb, Mo
Tl

Fig. 3: Model volume element (~$1.3 \bullet 10^5$ atoms) of the sample with contents of nanostructures close to that determined in the case of L_2 optimization.

5. Integration into MathCloud services

Now we are working on the integration of our numerical code into MathCloud framework [8], www.mathcloud.org, based on the RESTful services for e-science. The approach easily enables implementations of complete computing scenario of x-ray scattering data processing to determine the contents of a wide class of carbon nanostructures.

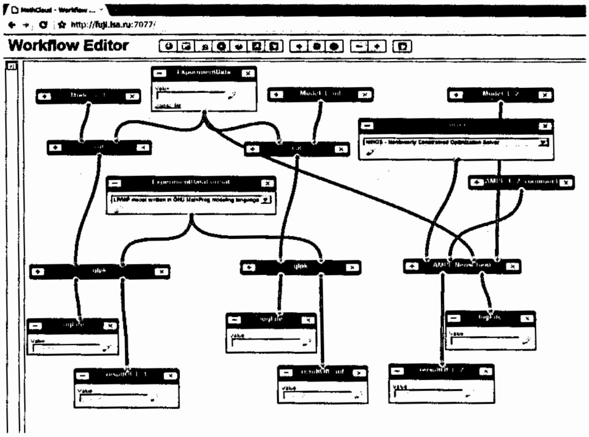


Fig. 4: Screenshot of MathCloud workflow editor page. A typical scenario of data processing is shown

This, we hope, will help other researchers in processing the results of x-ray diffraction and respective diagnostics of carbon-based nanomaterials. The figure 4 shows possible scenario of data processing in MathCloud workflow editor (a kind of Rich Web Application running in browser). Each block represents either a required input data, or a separate RESTful service (or an output data). For example, "ExperimentData" block is a loader for AMPL formatted data (www.ampl.com) composed of experimental profile, numerical modeling results and other data (noise, etc.); "glpk" is a wrapper for GNU Linear Programming Kit (for L_1 and L_{inf} criteria), and "AMPL-NeosClient" wraps remote access to nonlinear programming optimization solver for L_2 criterion (i.e. MINOS, see details on NEOS optimization solvers portal, neos.mcs.anl.gov); "cat" is just files concatenation service.

Conclusions

1. The developed code for modeling of x-ray diffraction characteristics of a wide range of carbon nanostructures and the proposed algorithms for the identification of the topological contents within a given class of nanostructures, can be used in automated systems for processing x-ray diffraction signals in the nanomaterial diagnostics with the modern computational tools, on the principles of parallel (MPI + OpenMP) and distributed (GRID and remote services) algorithms. Integration into MathCloud services is under development now.
2. Application of the developed algorithms to interpretation of x-ray scattering by the films deposed in tokamak T-10 shows that Unusual strong wide peak at $q \sim 10$ nm^{-1} peak may be caused by the nanostructures, formed by non-planar single-layer graphene sheets. Toroids (small toroidal carbon nanotubes with radii of the torus ~ 1 nm) are found to be the most likely candidates responsible for this peak. Full interference modeling of x-ray diffraction by ensemble of nanostructures in amorphous medium shows small deviation from the case without full interference. So our optimization algorithms are of 90% accuracy if the ratio of isolated carbon atoms to carbon atoms in nanostructures is less than 25%.

References

[1] Svechnikov N.Yu., Stankevich V.G., *et al.* Journal of Surface Investigation. X-ray, Synchrotron and Neutron Techniques, 2009. V. 3. N. 3. P. 420-428.

[2] Neverov V.S., Kukushkin A.B., Marusov N.L., Semenov I.B., Voloshinov V.V., Afanasiev A.P., Tarasov A.S., Veligzhanin A.A., Zubavichus Ya.V., Svechnikov N.Yu., Stankevich V.G. Modelling of x-ray diffraction by carbon nanostructures and determination of their possible topological contents in the deposited films from tokamak T-10// The problems of Atomic Science and Technology, Ser. Thermonuclear fusion, 2010. V. 1. P. 7-21 (in Russian, http://vant.iterru.ru/vant_2010_1/1.pdf).

[3] Svechnikov N.Yu., Stankevich V.G., Men'shikov K.A., et al. J. Surf. Invest., 2 (2008). P. 826-835.

[4] http://www.gnu.org/software/glpk/

[5] http://neos.mcs.anl.gov

[6] Jie Han. Toroidal Single Wall Carbon Nanotubes in Fullerene Crop Circles. NASA Technical Report 97-015, 1997. http://marsoweb.arc.nasa.gov/News/Techreports/1997 /PDF/ nas-97-015.pdf

[7] Satoshi Itoh, Sigeo Ihara. Phys. Rev. B, 1994. V. 49. N. 19. P. 13970-13974.

[8] Lazarev I.V., Sukhoroslov O.V. Implementation of Distributed Computing Scenarios in MathCloud framework. // Problems of distributed computing/ Ed. S.V. Emel'yanov, A.P. Afanas'ev. Proceedings of ISA RAS. V. 46. Moscow: KRASAND, 2009. P. 6-23 (in Russian).

GRID SYSTEM FOR REAL-TIME MANAGEMENT OF DISTRIBUTED DATABASES, WITH APPLICATION TO THE COORDINATION OF COMPLEX PROJECTS[1]

C. Placinta, I. Vasile, M. Dulea

Department of Elementary Particles and Information Technologies,
National Institute of R&D for Physics and Nuclear Engineering 'Horia Hulubei',
077125, Magurele, Romania

We report a Grid-based information system designed to improve the monitoring, management and reporting activities in the framework of complex collaborations.

The solution is based on the Globus Toolkit 4 [1] and the OGSA-DAI [2] software framework, and makes use of web services to share data between databases located in different domains. Also, web services are used to automatically synchronize the databases, by means of daemons.

The Grid environment provides the necessary security, and the management of distributed data is performed through OGSA-DAI. The system also offers tools for the online communication between the collaboration partners.

A case study is presented which targets the development of an integrated information system for the management and monitoring of research projects [3]. The system can easily be mapped on other similar case scenarios, such as the administration of the resources within an institution and all its subsidiaries, or the management of the human resources within research and education networks.

1. Introduction

The coordination of complex collaborations through digital means is highly desirable whenever an e-communication infrastructure that connects all the participants is available. This in particular is the case of the multi-partner projects in which institutional networks of independent organizations are created for longer periods of time in order to reach common objectives, when the partners share the same communication network. Examples are offered by European FP7 projects [4], international business projects, projects developed in educational and e-Health networks, national research projects, etc.

While the underlying communication networks (GEANT2 [5], Internet, NRENs [6]) are quasi-horizontal, the project-specific institutional networks are in most of the mentioned cases hierarchic, various organizations fulfilling specific roles within the same project. Accordingly, the participants in the EU projects can be coordinators, beneficiaries, or third parties, the business consortia are structured in parent companies, subsidiaries, regional representatives, sites, departments, etc.

The infrastructure of the information systems associated with these project-driven networks is inherently distributed due to the geographic dispersion of the participant resources, but the information flow reflects the hierarchical nature of the relationships between the partners. In particular, the selective and secure access of the users to the partners' databases must be ensured according to their identity and roles within the project. This is one of the main reasons why a Grid solution is indicated for the information management of the projects which use such distributed resources.

[1] Work partially supported from the project SIMPROC, contract PN2 11-043/2007, funded by the National Centre for Programme Management (CNMP), and the project PN 09 37 0104/2009, funded by the National Authority for Scientific Research (ANCS).

At present, most of the proposed solutions for the integrated informatisation of the project management are based on EPM (Enterprise Project Management) systems, which offer the users the possibility of monitoring and checking up of the activity within a single company/institution.

In this work we first design a Grid system for the real-time management and monitoring of the information flow in complex projects carried out by multiple partner institutions which are organized in a hierarchical structure. Then the general formalism is applied to the specific conditions of public financing of the national scientific research, which is performed through R&D projects, in accordance with the European Commission requirements.

In this context, the study proposes new procedures for the monitoring and reporting of the scientific results and economic management, aiming at improving the current management of the information and answering the needs for a better coordination between the research units and the contracting authorities.

2. Overview

The model

The system under investigation consists of a set of autonomous entities (organizations) which are connected together in institutional networks, within which they develop collaboration activities according to some commonly agreed activity plans that define what we'll generically call in what follows *projects*. Each entity (institute, university, company) has a specific role within a project, but it can fulfill different roles in different projects (e.g. as *partner* or *coordinator* - in the case of the national research projects).

In general, the projects are grouped together into programs according to some common goals and/or the source of funding. The entity that coordinates a program and monitors its projects will be called a *monitor*.

For instance, the 3-level hierarchical model corresponding to the national research projects is represented schematically in the image below where, for simplicity, the possibility of participation of the same entity (institute) in several projects was not shown.

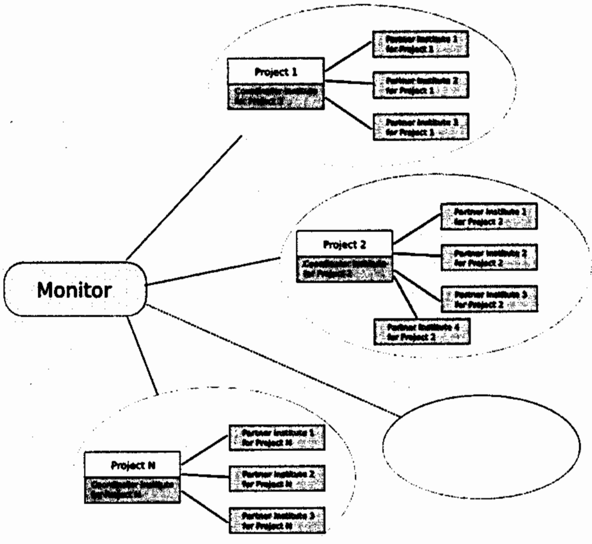

Project 1
Coordinator Institute for Project 1
Partner Institute 1 for Project 1
Partner Institute 2 for Project 1
Partner Institute 3 for Project 1
Project 2
Coordinator Institute for Project 2
Partner Institute 1 for Project 2
Partner Institute 2 for Project 2
Partner Institute 3 for Project 2
Partner Institute 4 for Project 2
Monitor
Project N
Coordinator Institute for Project N
Partner Institute 1 for Project N
Partner Institute 2 for Project N
Partner Institute 3 for Project N

Fig. 1

The project database

We assume that the participant entities store all the relevant data for projects in the relational databases of some informatic systems which are dedicated to the local management of the projects. These systems can be, for instance, based on EPM software, but their precise nature is not important as long as the database schema is similar for all the participants to the same project.

The first step towards the achievement of the global automation of the project management is the design of the project database (PDB), whose distributed structure is based on the local databases of the participants. This must be done in agreement with the rules of development of the project during its life cycle and with the various requirements of the system's users, which fulfill different roles (project responsible, project manager, accountant, evaluator, monitor, etc.)

The second step regards the data flow, and consists in the definition of general procedures for the automatic synchronization of the databases with the input information the lower branches were fed in the hierarchical structure. In this respect, it is to be noted that every entity, except the monitor, must be able to transfer some of the local management system data to the higher level entity, and this transfer must take place between different domains.

The third step is to establish the differentiated and secure access of the users to the PDB information, following authentication procedures which ascertain the identity, the membership of the users and the roles assigned in the project. Moreover, the information transfer between the databases and to the user interfaces must also be secure.

3. Technical description

The above-mentioned requirements can easily be satisfied if a Grid solution is chosen for the management of the informatic system. This can be achieved if each partner hosts a Grid node that communicates with the local database dedicated to the management of projects, and if the Grid nodes can securely and automatically transfer the database information along the hierarchical structure.
Also, the client interaction with the local database is made through a web interface, the user authentication being provided through Grid certificates.

The Grid environment was built making use of Globus Toolkit 4 (GT), which provides increased security, all the communications being made exclusively through GT authentication (SSL certificates). Moreover, GT provides means for application development and ensures the filtered synchronization of the databases, offering tools which are independent of the infrastructure, but dependent of the SQL structure of the databases.

The access to and the integration of the system's distributed data sources is provided by the OGSA-DAI toolkit [2], which is an extension of GT and contains a package of web services that work with Grid functionalities through GT.

Web services

The OGSA-DAI functions are called through web services (WS), which are developed with the help of the Globus Toolkit and deployed in GT and Apache Tomcat [7].

In the simplified case of a three-level hierarchical management structure (e.g. monitor, coordinator, partners) three web services are necessary:

- WS1 handles single resource requests (one-to-one queries of the gridnodes' databases);
- WS2 is designed to handle multiple resources and queries one-to-many the databases of the gridnodes;
- WS3 is similar to WS2, but it is configured to work at the monitor level.

The web service WS1 is developed to work with OGSA-DAI at the first level of the management hierarchy. In WS1 one must configure the database resources (DB Res) and then the whole system communicates through these resources with the database servers in a way which is transparent for the developer. Thus, for each node the database resource for the database management server must be configured. In order to create a secure link for sending the SQL request and to receive the results, one must define a pipeline in WS1. This pipeline is based on the DB Res mentioned above and looks like in Fig. 2 below.

Thus, whenewer a result is required, a SQL Statement is sent as a parameter to SQLQuery. SQLQuery is double-linked with TupleToWebRowSetCharArrays which converts one SQL tuple to strings. These strings are filtered with CharArraysResize, which groups the strings in packets of data of a certain size (specified as a parameter). In the same manner, the link TupleToWebRowSetCharArrays – CharArraysResize is double (in both ways).

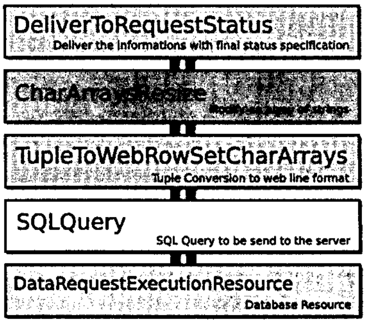

DeliverToRequestStatus
Deliver the informations with final status specification
CharArraysResize
TupleToWebRowSetCharArrays
Tuple Conversion to web line format
SQLQuery
SQL Query to be send to the server
DataRequestExecutionResource
Database Resource

Fig. 2

The last element of the pipeline is the double-link with DeliverToRequestStatus with assures that the SQL request is sent to DB Res and that it will return the result to the query. The double-link ensures the passing of the messages in both ways, to and from the Database Resource.

The project coordinating institution has an additional specific interface in which it calls a second web service - WS2 - which is configured so that it can read from several database resources as a result of a single command issued by the user. Thus, within this web service the data is read from several DB Res. Accordingly, the pipeline is modified such that the first level uses many Database Resources.

The data read from multiple partners can, for example, be the transferable parts of the economic databases. The pipeline's structure in this case is the following:

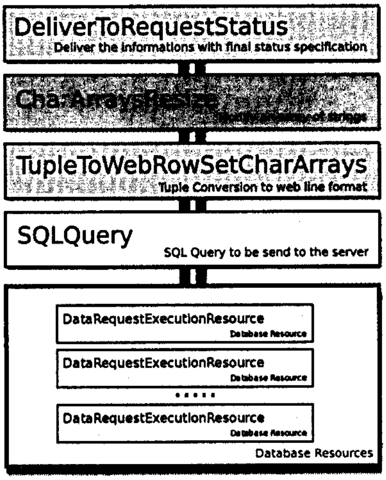

DeliverToRequestStatus
Deliver the informations with final status specification
CharArraysResize
TupleToWebRowSetCharArrays
Tuple Conversion to web line format
SQLQuery
SQL Query to be send to the server
DataRequestExecutionResource
Database Resource
DataRequestExecutionResource
Database Resource
DataRequestExecutionResource
Database Resource
Database Resources

Fig. 3

The functional relationships between institutions at the first and the second level of the hierarchy are represented in Fig. 4 below.

The monitoring institution possesses its own database that it is populated using a variety of daemon applications which run as client-server (just like the previous level), and a specific user interface using the third web service - WS3.

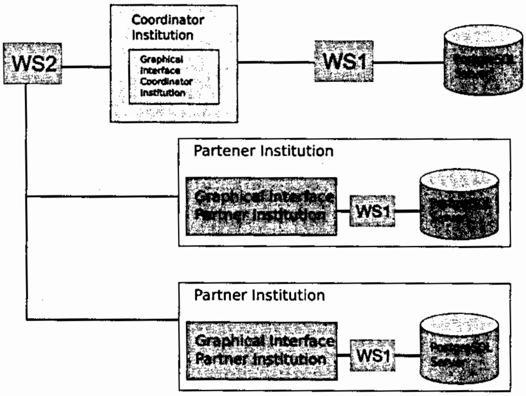
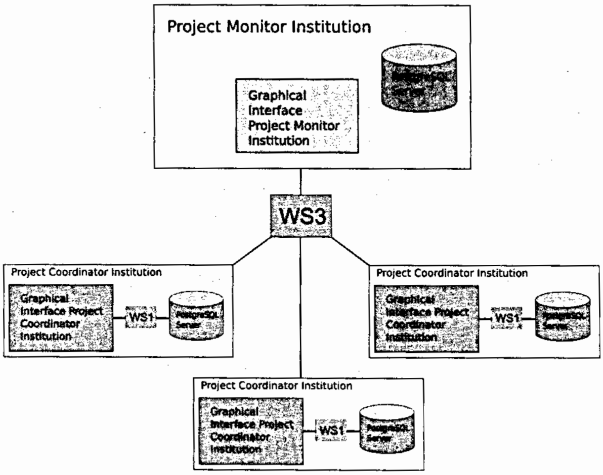


Fig. 4

Although web service WS3 is similar to WS2, it is configured differently: while in the case of WS2 the DB Res were owned by each partner institutes, in the case of WS3 the DB Res are owned by each coordinating institute.

The architecture of the third level is represented in Fig. 5.



Project Monitor Institution
PostgreSQL Server
Graphical Interface Project Monitor Institution
WS3
Project Coordinator Institution
Graphical Interface Project Coordinator Institution
WS1
PostgreSQL Server
Project Coordinator Institution
Graphical Interface Project Coordinator Institution
WS1
PostgreSQL Server
Project Coordinator Institution
Graphical Interface Project Coordinator Institution
WS1
PostgreSQL Server

Fig. 5

Synchronization daemons

The migration of the new data from the partners' databases to the coordinator, respectively from the coordinator's database to the monitor's database is performed through scheduled updates by means of daemon processes.

Thus, the synchronization between the database of the project coordinator and the database of any of the partners is realized using a daemon which is configured to work with WS2 through a client, reads the data from the partners and populates the coordinator's database using WS1.

Similarly, the synchronization between the monitoring institution and the coordinators is realized using another daemon configured to read the data, through a client working with WS2, and populates the monitoring database using WS1.

The action of the daemons is depicted below:

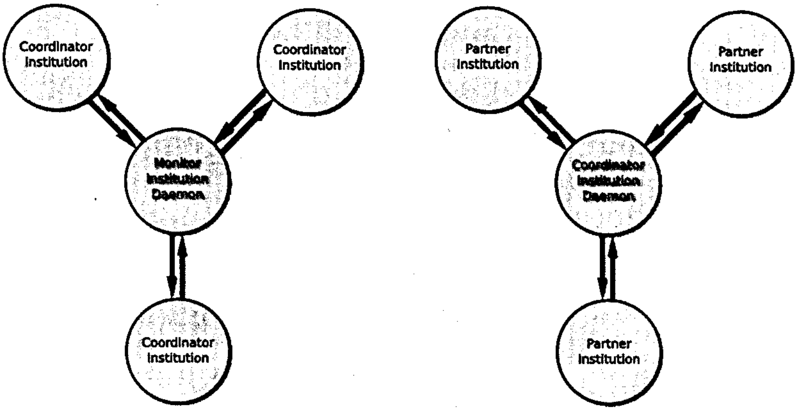


Fig. 6

4. Case study: the real-time management of the research projects

The implementation of the general formalism above for the execution of the national R&D projects was performed after the in depth analysis of the functioning and interconnection of the reporting and monitoring activities within the system. As a result, most of the subsequent work was dedicated to the development and implementation of the project database structure and user interface in agreement with the specific projects' requirements.

The management structure

The management structure of the national multi-partner R&D projects is based on a three-level hierarchy that includes the contracting/monitoring authority, the project coordinator, and the partner institutions which collaborate with the coordinator.

The information flows along the hierarchical structure: the partner institutions regularly report their results to the project coordinator, which in turn sends scheduled deliverables to the contracting authority.

Parts of this information are handled by the accountants, human resources managers, project responsible, project directors, evaluators, monitors, etc., whose access is restricted to their area of expertise.

These positions within the project define the roles of the system's users. Also, user requirements are important for defining the design of the interaction between the database and the user interface.

The general formalism described in the previous chapter can easily be adapted to the management of the national R&D projects. The software tools are described in what follows.

Implementation

As a minimal condition, the participants to the research project and the monitoring authority must have installed grid nodes with at least one reliable server, which interacts with one or more databases within that institution.

An open source architecture was chosen for the operation infrastructure. Each grid node has a Linux distribution installed, with a web-server Apache – PHP and a PostgreSQL database management system.

In our specific implementation we used the CentOS 5.3 distribution, the Apache 2.0 server, PHP 5.1.6 and PostgreSQL 8.1.11.

The Grid architecture is based on Globus Toolkit 4.0.5 (GT4) middleware, whose tools were used for developing the grid applications and web services. Additionally, OGSA-DAI (ver 3.1) was installed for the interaction with databases and Tomcat (ver 5.0.28) as a web-environment for the deployment and use of web services.

Globus Toolkit's Grid Security Infrastructure (GSI) was used for user authentication and authorization.

The graphical user interface

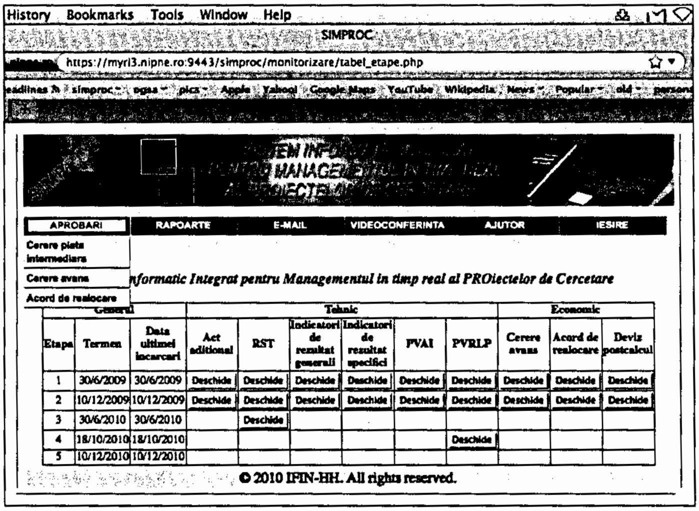
The web services described in Chap. 3 were programmed to ensure the communication between the GUI and the OGSA-DAI services which are required for interaction with the local databases of the partners.

The GUI is PHP based, deployed in Apache, and makes use of the above described web services to address SQL queries to the PostgreSQL databases.

For the ease of programming of the GUI, a specialized PHP library was developed.

The design of the GUI takes into account the different roles assigned to the users. The correspondence between the user identity (grid certificate), the projects to which he has access, and the roles the user plays within these projects is stored by the system administrator in the authentication database.

The user accesses the system portal, his grid certificate is read from the browser, its credentials are checked from the authentication database, and a specific web interface is presented in agreement with the credentials, providing access to the information the user is allowed to see and/or edit.



History Bookmarks Tools Window Help
SIMPROC
https://myrl3.nipne.ro:9443/simproc/monitorizare/tabel_etape.php
eadlines simproc ogsa pics Apple Yahoo! Google Maps YouTube Wikipedia News Popular old person
APROBARI
RAPOARTE
E-MAIL
VIDEOCONFERINTA
AJUTOR
IESIRE
Cerere plata intermediara
Cerere avans
Acord de realocare

nformatic Integrat pentru Managementul in timp real al PROiectelor de Cercetare

General			Tehnic						Economic		
Etapa	Termen	Data ultimei incarcari	Act aditional	RST	Indicatori de rezultat generali	Indicatori de rezultat specifici	FVAI	PVRLP	Cerere avans	Acord de realocare	Deviz postcalcul
1	30/6/2009	30/6/2009	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide
2	10/12/2009	10/12/2009	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide	Deschide
3	30/6/2010	30/6/2010		Deschide							
4	18/10/2010	18/10/2010						Deschide			
5	10/12/2010	10/12/2010									

© 2010 IFIN-HH. All rights reserved.

Fig. 7

The GUI presents customized interfaces for the roles of project responsible/director, accountant, legal representative of the entity, monitor, and system administrator.

The interface also provides means of accessing communication tools, videoconferencing between the users located in different domains being possible through Access Grid [8].

Deployment

The system was first implemented and tested for the real-time management of the SIMPROC project [3] and other national R&D projects.

Three Grid nodes were deployed in the domains of the institutions which participate to the SIMPROC project, i.e. the coordinator IFIN-HH, and the partners, the Institute for Space Sciences and the University 'Politehnica' from Bucharest. Each node is connected to a PostgreSQL DBMS whose tables were populated with the scientific and economic data of the project. The node of the monitoring institution was simulated by means of a different server, hosted in IFIN-HH.

The server and user certificates were issued by the national certification authority, RomanianGRID CA.

5. Conclusions

An integrated system for the management and monitoring of the data flow in distributed systems with hierarchical informational structure was designed making use of the OGSA-DAI extension of the Globus Toolkit and web services. The system was developed and implemented in the particular case of the management of national R&D projects, being deployed by the partners of the SIMPROC [3] project.

The implementation of the solution making use of Grid technologies ensured the necessary security, the user authentication, the access to and the management of the system's distributed database, and the scalability of the system.

The general design features of the project management system can easily be mapped on other similar case scenarios, such as the administration of the resources within a company and its subsidiaries, or the management of the activities performed in the framework of institutional networks such as the educational and e-Health project-based collaborations.

References

[1] The Globus Alliance, web site: http://www.globus.org

[2] Open Grid Service Architecture – Data Access and Integration project, web site: http://www.ogsadai.org.uk

[3] Integrated Information System for the Real Time Management of the Research Projects (SIMPROC) project, web site: http://proiecte.nipne.ro/pn2/index_en.php?id=1

[4] The Art of Networking. European Networks in Education, Bienzle H., Gelabert E., Jütte W., Kolyva K., Meyer N., Tilkin G., "die Berater" UmbH, Wien, 2007.

[5] The European high-bandwidth network for research and education, web site: http://www.geant2.net/

[6] National Research and Education Networks, TERENA NREN Compendium, web site: http://www.terena.org/activities/compendium/

[7] Apache Tomcat, web site: http://tomcat.apache.org/

[8] Access Grid, web site: http://www.accessgrid.org/

INTEGRATION OF LOCAL RESOURCE MANAGERS IN GRIDNNN: CLEO EXAMPLE[1]

N. V. Prikhodko[1], V. A. Abramovsky[2], A. P. Kryukov[3]

[1] Yaroslav-the-Wise Novgorod State University (NovSU),
173003, Velikiy Novgorod, Russia; niko2004x@mail.ru
[2] Yaroslav-the-Wise Novgorod State University (NovSU),
173003, Velikiy Novgorod, Russia; Victor.Abramovsky@novsu.ru
[3] Skobeltsyn Institute of Nuclear Physics Lomonosov Moscow State University (SINP MSU)
119991, Moscow, Russia; kryukov@theory.sinp.msu.ru

The project GridNNN [1] is aimed creation of a distributed computing environment for Russia National Nanotechnology Network. This goal requires integration GridNNN infrastructure with supercomputer resources which belong to various Russian universities and research centers. Since GridNNN software is partially based on GT4 (Globus Toolkit 4.2 [2]), computing resources which use widespread LRMS (local resource manager systems) such as PBS/Torque or Condor can be easily integrated in GridNNN. There are however some popular LRMS in Russia which integration is essential to the GridNNN. Among them are Cleo [3] and Slurm. For example, Cleo was designed in Lomonosov Moscow State University and is used on "Chebyshev" supercomputer. Slurm is used on Kurchatov Institute supercomputer.

There are a several requirements by GridNNN project related to developing bridge between GT4 and new LRMS:

- LRMS should be interfaced with GT4 in a way similar to PBS/Torque, Condor, LSF already supported by GT4;
- Bridge between LRMS and GT4 should be installed in the same way as other LRMSs bridges using GPT(Globus Packaging Toolkit);
- Additional LRMS capabilities which is essential to use cases should be supported in a matter consistent with GT4;
- GridNNN software and bridge between GT4 and LRMS should be installed on machine separated from cluster. Only NFS mounted /home and ssh access to central cluster node has be provided. Stack GT4 does not support this, even for PBS/Torque.

Cleo LRMS was developed in Moscow State University for purpose of parallel tasks execution control on computer clusters in one or more task queues. Essentially its architecture is common for all LRMS: client, server to control execution, optional agent for work nodes. Comparing to other LRMS Cleo support all standard and some non-standard features:

- tasks queues including node partitioning and subqueues;
- tasks scheduling;
- all MPI implementations are supported, most other parallel environments are supported too;
- can transparently use few MPI implementations on the same set of cluster nodes;
- task processes terminating on compute nodes;
- automatic dead nodes blocking;
- programmable free nodes distribution between tasks;
- cluster usage policy control;
- controllable user limits (max used CPUs, task work time, etc.);
- customizable graphical or text queue and tasks info (via qs-web package);
- custom modules for CPUs distributors and schedulers;

[1] The work is supported by MON Contract No. 01.647.11.2004 .

- execution profile allows to configure task execution in any way possible;

From user point of view Cleo has two CLI programs: 'mpirun' and 'tasks'. Despite the name 'mpirun' the program is a wrapper, which emulates standard mpirun and understand some additional options. However it does not limit user to running MPI programs. 'tasks' program allows to queue and destroy submitted tasks.

Common use cases for Cleo require support for all standard PBS/Torque like features from GridNNN. However execution profile parameter should also be supported since it is commonly used. Moreover Cleo ability to transparently use few MPI implementations rely on execution profile parameter. For example MSU Chebyshev cluster has following execution profiles:

- single - allows to run single CPU job;
- zingle - allows to run single CPU job with full node allocation;
- mpich - allows to run MPI task using mpich;
- openmpi - allows to run MPI task using openib;
- intelrdma - allows to run MPI task using RDMA;
- mcs - allows to run .Net executable with mono (just mpirun -np 1 task.exe);

Since GT4 is very modular bridging between new LRMS and GT4 requires rewriting only few components. GT4 execution subsystem is well documented and consist of three main components (fig. 1):

- SA (Schedule adapter) - transforms external request to GRAM into requests to LRMS (main component JobManager.pm);
- SEG (Scheduler Event Generator) - provides asynchronous notification of GRAM about job state changes;
- SP (Schedule Provider) - output current LRMS state (query status etc, this is not execution management component but it is essential for meta-scheduling)

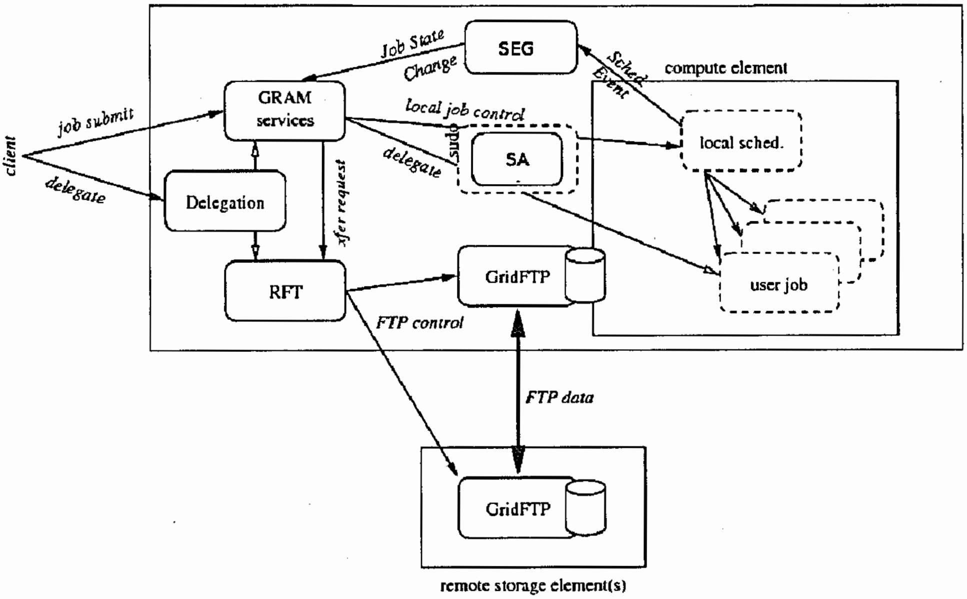


Fig.1: Globus Toolkit 4.2 execution management subsystem

However components of GT4 execution management subsystem do not have good documentation about how to develop bridge which integrate GT4 with LRMS. In practice, this task is not trivial and requires many tricks. There are lots of legacy pieces inherited from previous versions of GT4 which complicate such undertaking. If LRMS have capabilities which is not available in PBS/Torque bridge implementation will be even more tricky.

Analysis of GT4 shows that there are five GPT components inside GT4 source tree related to particular LRMS (PBS/Torque, Condor, LSF)::

- globus_gram_job_manager_setup_LRMS – gram/jobmanager/setup/LRMS;
- globus_scheduler_event_generator_LRMS – ws-gram/job_monitoring/LRMS/c/source;
- globus_scheduler_event_generator_LRMS_setup – ws-gram/job_monitoring/LRMS/c/setup;
- globus_scheduler_provider_setup_LRMS – ws-gram/service/java/setup/LRMS;
- globus_wsrf_gram_service_java_setup_LRMS – ws-gram/service/java/setup/LRMS;

All this GPT components have a lot of code which seems boilerplate and it is not oblivious which parts should be written for new LRMS and which can simply be borrowed. Moreover existing GT4 bridges for PBS/Torque, Condor, LSF have quite complex (but as experiments shows essential) 'conventions' for mixing upper/lower case LRMS name and '_', '-' usage in LRMS related names.

It is evident that GPT components for new LRMS can be manually forked from similar component for other LRMS. However to simplify future development we developed simple script create_gpt_component.pl [4] to create necessary components with essential boilerplate code and proper GT4 naming conventions. Just run with mixed case in LRMS name: create_gpt_component.pl -n Lrms. All GPT components will be created from internal templates with files having proper content. Some essential code which should be written will be pointed out. Write only useful parts.

Generated components contain lots of boilerplate code which is common to all components:

- doxygen - empty directory;
- pkgdata/pkg_data_src.gpt.in - GPT component description file;
- bootstrap - script which runs autoconf/automake to create configure etc;
- dirt.sh - empty file for time stamp;
- configure.in - stub file forconfigure;
- Makefile.am - stub file forMakefile.in;

and some mostly boilerplate code which is component specific:

- globus_gram_job_manager_setup_LRMS
    - setup-globus-job-manager-LRMS.pl - script to create LRMS.pm from LRMS.in;
    - globus_gram_job_manager_LRMS.dox - doxygen documentation file for setup-globus-job-manager-LRMS.pl;
    - find-LRMS-tools.in - ac-file which describes how to transform LRMS.in into LRMS.pm;
- globus_scheduler_event_generator_LRMS
    - pkgdata/MyFilelists.pm - Perl module which subclass Grid::GPT::MyFilelists and redefines few methods to correctly install corresponding shared library;
- globus_scheduler_event_generator_LRMS_setup
    - setup-seg-LRMS.pl - create configuration file $GLOBUS_LOCATION/etc/globus-LRMS.conf (in practice checks where LRMS log-files are placed)
- globus_scheduler_provider_setup_LRMS
    - setup-globus-scheduler-provider-LRMS - shell script to run setup-globus-scheduler-providerLRMS.pl;
    - setup-globus-scheduler-provider-LRMS.pl - Perl script to create globus-scheduler-provider-LRMS.pl from in-file (for given LRMS mostly find where CLI utilities, log-files etc resides);
    - find-LRMS-provider-tools.in - ac-file which describes transformation from in-file;
- globus_wsrf_gram_service_java_setup_LRM
    - setup-gram-service-LRMS - simply runs $GLOBUS_LOCATION/setup/globus/setup-grammanager.pl with parameters.

There are of cause some real code which should be written.

Template globus_gram_job_manager_setup_LRMS/LRMS.in is used to generate Perl module LRMS.pm which contain definitions for methods submit, cancel and poll to submit job, cancel job and poll job state. The methods have $self->{JobDescription} for input data and output have form ever {JOB_ID =>$job_id, JOB_STATE => Globus::GRAM::JobState::* } or Globus::GRAM::Error::*. There are useful things to know about submit, cancel and poll:

- In all LRMS submit method create script in .globus sub directory which calls LRMS CLI utilities.
- In all LRMS cancel method just calls 'job destroy' using LRMS CLI.
- Poll method doesn't needed for WS-GRAM.
- Job identifier $job_id should be unique identifier inside LRMS.
- If it is unique only inside particular LRMS query use '$queue-$job_id' as LRMS job identifier.
  - Do not use ';' in job identifier (this break WS GRAM event stream parser).
  - Use identifier in form which allows easy job event identification in LRMS log-files.
- If you want to pass some strange parameters to LRMS use extensions section in job description.
  - Standard Globus::GRAM::ExtensionsHandler work fine in most cases.
  - If you need to pass some heavy structured parameters subclass Perl module Globus::GRAM::ExtensionsHandler into Globus::GRAM::ExtensionsHandler::LRM.

globus_scheduler_event_generator_LRMS/seg_LRMS_module.c contain C source code for SEG shared library and is used to parse LRMS log-files and emit events related to job (submitted, queues, deleted etc) in common GT format. In seg_LRMS_module.c it is essential to write several functions:

- globus_l_LRMS_find_logfile - find current log-file. Usually parse $GLOBUS_LOCATION/etc/globus-LRMS.conf} and just returns file name. In most LRMS however does more complex things if log-files are rotated, date splinted etc.
- globus_l_LRMS_read_callback - read lines from log-files into buffer and calls globus_l_LRMS_parse_events to parse buffer. If log-files rotated or date splinted calls globus_l_LRMS_find_logfile after reading current log-file. It is better to just borrow this part of code from other LRMS and change parts related to globus_l_LRMS_find_logfile.
- globus_l_LRMS_parse_events - parse buffer with line from LRMS log-file and generate event using standard functions globus_scheduler_event_STATE;
  - But in reality it just writes string '001;TIMESTAMP;JOB_ID;STATUS;0' into stdout. WS-GRAM just parse it. For some reason IPC was not used.
  - Output could be viewed. Just run globus-scheduler-event-generator -s LRMS -t FROM_TIMESTAMP. Useful for debugging.
  - For easy debugging insert SEGLRMSDebug(TYPE, STRING) into code. Debugging mode can be activated by non zero value in environment variable SEG_LRMS_DEBUG.
  - Use globus_strptime function to parse date in log-files.

Template globus_scheduler_provider_setup_LRMS/globus-scheduler-provider-LRMS.in is used to create Perl program globus-scheduler-provider-LRMS.pl. globus-scheduler-provider-LRMS.in has no direct input and just writes XML file which describes LRMS state (fig 2.). Mostly uses LRMS CLI utilities for data acquisition. Mostly trivial to write but not so trivial to understand why related WS-MDS does not work properly:

- In case of multiple queues of proper info will not be shown in information system in most cases. Publishing component of WS-MDS is broken in few places and require patching.
- There are some mismapping between scheduler provider and WS-MDS;
  - totalnodes, freenodes from scheduler provider transforms into freeCPUs, totalCPUs in WS-MDS.
  - Neither variant is not nodes nor CPUs. Just some abstract quantity for CPU resources;

```
<scheduler
  xmlns="http://mds.globus.org/batchproviders/2004/09"
  xmlns:ce="http://mds.globus.org/glue/ce/1.1"
  xmlns:cfg="http://mds.globus.org/2005/09/cluster-config"
  <Info ce:LRMSType="LRM" ce:LRMSVersion="VERSION"
        ce:GRAMVersion="GRAM_VERSION" ce:HostName="HOSTNAME"
        ce:TotalCPUs="TOTALCPU" ce:FreeCPUs="FREECPU">
  <Queue name="NAME">
    <totalnodes>TOTALNODES</totalnodes>
    <freenodes>FREENODES</freenodes>
    <maxtime>MAXTIME</maxtime>
    <maxCPUtime>MAXCPUTIME</maxCPUtime>
    <maxCount>MAXCOUNT</maxCount>
    <totalJobs>TOTALJOBS</totalJobs>
    <runningJobs>RUNNING</runningJobs>
    <maxReqNodes>MAXREQNODE</maxReqNodes>
    <maxRunningJobs>MAXRUNNINGJOBS</maxRunningJobs>
    <maxJobsInQueue>MAXJOBSINQUEUE</maxJobsInQueue>
    <maxTotalMemory>MAXTOTALMEMORY</maxTotalMemory>
    <maxSingleMemory>MAXSINGLEMEMORY</maxSingleMemory>
    <whenActive>WHENACTIVE</whenActive>
    <status>STATUS</status>
    <dispatchType>DISPATCHTYPE</dispatchType>
  </Queue>
</scheduler>
```

Fig.2: globus-scheduler-provider-LRMS.pl dump example

All resulting GPT components could be installed in standard GPT way: creating and installing GPT bundle:

- cd GPT_COMPONENT
- ./bootstrap
- ./configure
- or ./configure --with-flavor=FLAVOR in case globus_scheduler_event_generator_LRM
- make dist (../GPT_COMPONENT-VERSION.tar.gz will be created)
- cd ..
- gpt-build GPT_COMPONENT-VERSION.tar.gz
- gpt-postinstall

Looking back at Cleo integration with GridNNN first and second requirement for bridge will be automatically provided by create_gpt_component.pl script. Since Cleo requires support for execution profile parameter it should be passed in 'extension' section of job description. Since execution profile is essentially 'flat' parameter no Globus::GRAM::ExtensionsHandler modifications are required to pass it to SA. Resulting SA was modified to pass execution profile parameter to Cleo 'mpirun'. Since only NFS mounted /home and ssh access to central cluster node was provided for bridge Cleo 'mpirun' and 'tasks' is called from central cluster node by means of ssh-wrapper (to call Cleo CLI using 'ssh -t' on central cluster node). Moreover resulting SEG was modified to tolerate when Cleo log-files is destroyed and recreated (log-files are received to GT4 node using rsync from central cluster node since mounting /var is not acceptable). Bridge between Cleo and GT4 was successfully tested for "Chebyshev" supercomputer Moscow State University which now belongs to production part of GridNNN.

References

[1] Grid for National Nanotechnology Network; http://ngrid.ru/ngrid

[2] Globus Toolkit 4.2; http://globus.org

[3] Cleo; http://parcon.parallel.ru/cleo.html

[4] work in progress but mostly useful; request authors for code

INSTALLATION AND SETUP PROOF CLUSTER ON GRID SITE

S. S. Sayzhenkova

Institute for High Energy Physics, Russia
142280, Moscow region, Protvino, Pobeda st. 1
Sofia.Sayzhenkova@ihep.ru

PROOF - Parallel ROOT Facility

ROOT is an object-oriented framework aimed at solving the data analysis challenges of high-energy physics.

The Parallel ROOT Facility, **PROOF**, is an extension of ROOT enabling interactive analysis of large sets of ROOT files in parallel on clusters of computers or many-core machines. More generally **PROOF** can parallelize the class of tasks the solution of which can be formulated as a set of independent sub-tasks [1].

PROOF is primarily meant as an alternative to batch systems for Central Analysis Facilities and departmental work groups (Tier-2's and Tier-3's) in particle physics experiments. However, thanks to a multi-tier architecture allowing multiple levels of masters, it can be easily adapted to a wide range of virtual clusters distributed over geographically separated domains and heterogeneous machines (GRIDs) [2].

Apart from the pure interactive mode, **PROOF** has also an interactive-batch mode. With interactive-batch the user can start very long running queries, disconnect the client and at any time, any location and from any computer reconnect to the query to monitor its progress or retrieve the results. This feature gives it a distinct advantage over purely batch based solutions, that only provide an answer once all sub-jobs have been finished and merged [1]. The **PROOF** system implements a multi-tier architecture as shown in the figure 1.

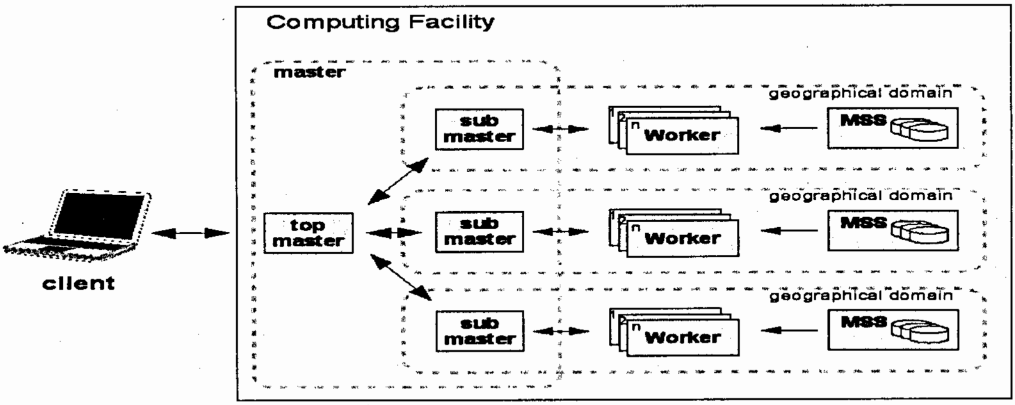


Fig 1: Multi-Tier Master-Worker Architecture

The client is the user that wants to use the resources to perform a task. The master is the entry point to the computing facility: it parses the client requests, it distributes the work to the workers, and it collects and merges the results. The master tier can be multi-layered. This allows, for example, federating geographically separated clusters by optimizing the access to auxiliary resources, like mass storages. It also allows distributing the distribution and merging work, which may become the bottleneck in the case of many workers.

The purpose of our work is to enable PROOF on cluster of machines. Enabling **PROOF** on a cluster of machines means to configure and start a dedicated daemon on each machine running as master / worker, hereafter referred to as the servers.

The dedicated daemon - which is implemented as a plug-in for the multi-purpose xrootd daemon - processes **PROOF**-related requests on server nodes, which may come from the client or from a master on behalf of a client. The daemon is running on the PROOF server machines accepting connections on port 1093 (assigned by IAAA) [3]. It performs two tasks:

- Authenticates the requests: it checks that the request makes sense and comes from an authorized entity; the strength of the checks depends on the configuration settings;
- Starts a ROOT session and puts the client in connection with it.

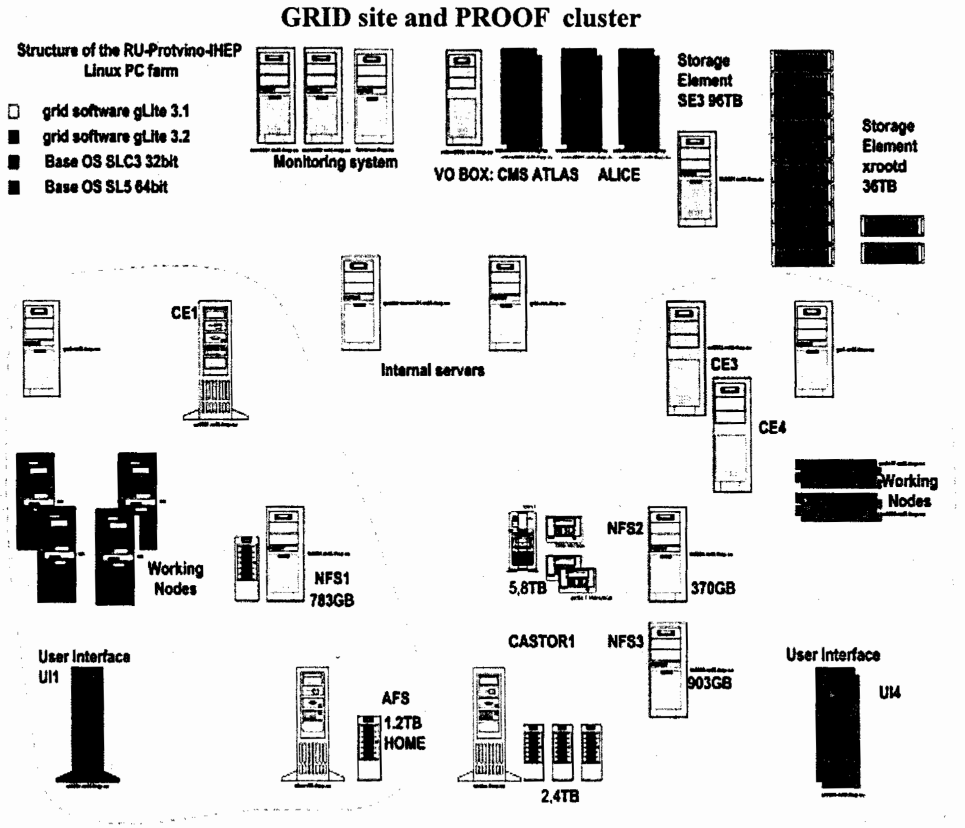


Fig 2: Structure of site RU-Protvino-IHEP

Server side: the master is the entry point to the computing facility. There are not CPUs on master.

Server side: there are 60 workers with one CPU on each on IHEP GRID site.

Client side: User Interface.

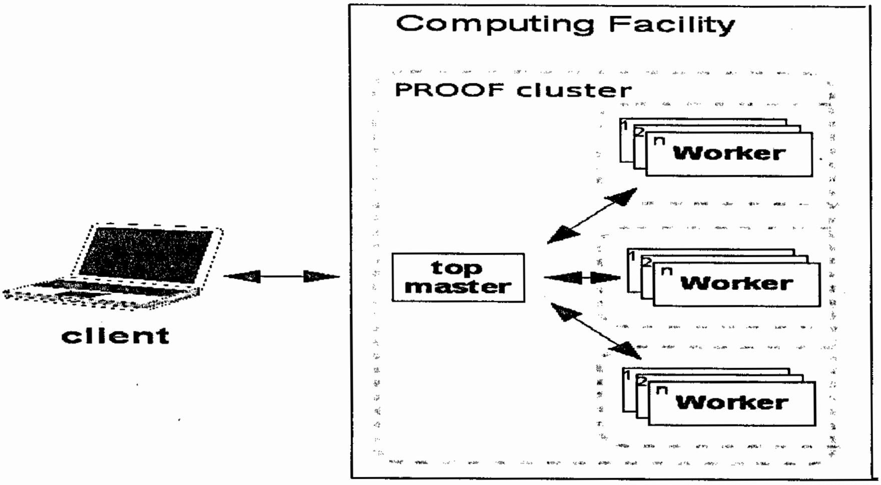


Fig 3: PROOF architecture in IHEP

Step by step install and configure PROOF cluster on GRID site

It is necessary to download source codes from an official site root.cern.ch: root_<version> .source.tar.gz. Then unpack archive, change configuration files and choose a way of installation. After archive unpacking we edit configuration files.

In the config file (proof.conf) we add master machine and all working nodes:

master <hostname>
worker <hostname>
worker <hostname>
....

In the second config file xpd.cf in part one "data serving" we add some libraries such as libXrdProofd.so and libXrdOfs.so. In third part "enable PROOF serving" we show way to config file proof.conf.

In the instruction xpd.schedparam we specify type of queue which we wish to apply and quantity of simultaneous sessions *queue:fifo mxrun:1*. In IHEP GRID site queue fifo (first input first output) and one simultaneous session is used.

In daemon xrootd start script we must specify some variables, such as:

export ROOTSYS=<place_where_root_is_installed>
export PATH=$PATH:$ROOTSYS/bin
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:$ROOTSYS/lib/root/5.26

Then we choose a way of installation.

There are two main methods of installing ROOT. The first one is installation from source. As an alternative, you can build either a set of Debian GNU/Linux or Redhat Linux packages. We decided to use rpm packages. GRID site contains many nodes and on each we need install this software. It was necessary to automate this process.

Operating system on master and all nodes is Scientific Linux SL release 5.3 (Boron) (x86_64).

In a folder with source codes we execute a command:

make redhat

This will create a RPM spec file in the source tree: root.spec.

We do some changes in a file root.spec such as: specify release version, directories in which rpm-packages will be gathered and where to be setup and choose features with which proof should be configured.

If you do not have system privileges, you should set up a build area by having the file ~/.rpmmacros with a contents like:

%_topdir <some where you can write>/redhat
%_sourcedir %{_topdir}/SOURCES
%_specdir %{_topdir}/SPECS
%_builddir %{_topdir}/BUILD

Then you should make the appropiate directories:

mkdir <some where you can write>/SOURCES
mkdir <some where you can write>/BUILD
mkdir <some where you can write>/RPMS
mkdir <some where you can write>/SRPMS
mkdir <some where you can write>/SPECS

and finally copy the source tar-ball and spec file:

cp root_v<version>.source.tar.gz \ <some where you can write>/SOURCES
cp root.spec \ <some where you can write>/SPECS

Whether you have system privileges or not, you can now build the RPM packages by issuing:

rpmbuild -ba <some where you can write>/SPECS/root.spec

Finally in a folder <some where you can write>/RPMS we have two rpm-packages:

libroot-static-<version>-<release>.x86_64.rpm
root-system-<version>-<release>.x86_64.rpm

These two packages should be installed on master and every node in cluster. First **libroot-static-<version>-<release>.x86_64.rpm**, second **root-system-<version>-<release>.x86_64.rpm.**

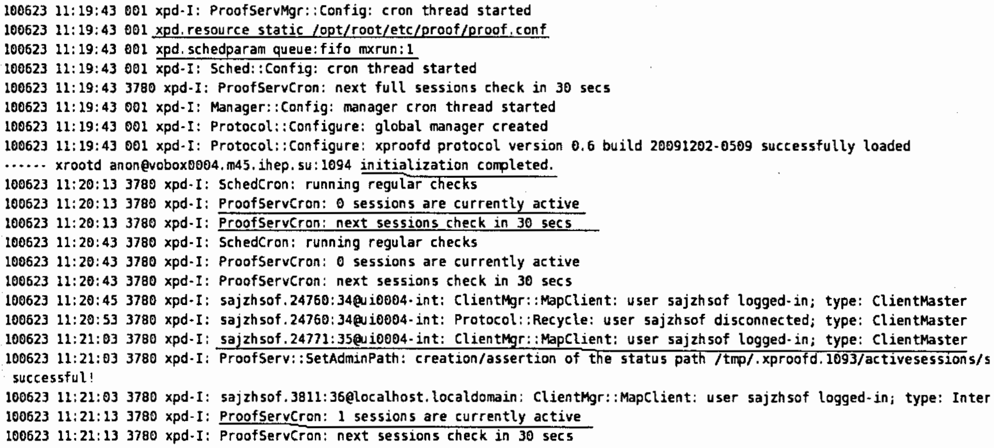
Then on master and all nodes we start a daemon xrootd.

/etc/init.d/xrootd status
xrootd (pid 9885) is running...

At successful start in log file there should be a following message: *initialization complete.*

When user connect with PROOF cluster one can see message in log file: *<user> logged in; 1 session are currently active.*

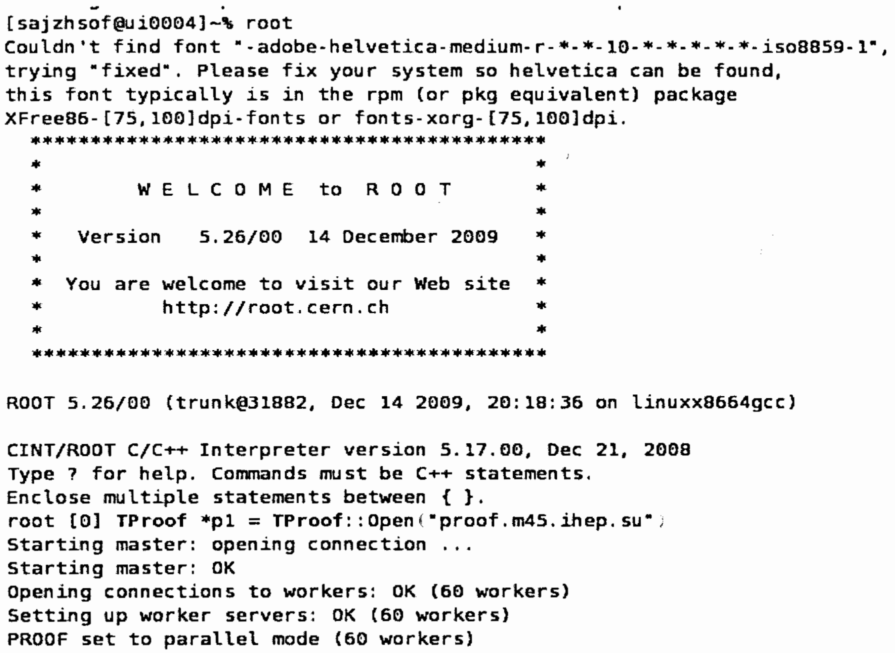
Xrootd daemon check active sessions every 30 seconds.



```
100623 11:19:43 001 xpd-I: ProofServMgr::Config: cron thread started
100623 11:19:43 001 xpd.resource static /opt/root/etc/proof/proof.conf
100623 11:19:43 001 xpd.schedparam queue:fifo mxrun:1
100623 11:19:43 001 xpd-I: Sched::Config: cron thread started
100623 11:19:43 3780 xpd-I: ProofServCron: next full sessions check in 30 secs
100623 11:19:43 001 xpd-I: Manager::Config: manager cron thread started
100623 11:19:43 001 xpd-I: Protocol::Configure: global manager created
100623 11:19:43 001 xpd-I: Protocol::Configure: xproofd protocol version 0.6 build 20091202-0509 successfully loaded
------ xrootd anon@vobox0004.m45.ihep.su:1094 initialization completed.
100623 11:20:13 3780 xpd-I: SchedCron: running regular checks
100623 11:20:13 3780 xpd-I: ProofServCron: 0 sessions are currently active
100623 11:20:13 3780 xpd-I: ProofServCron: next sessions check in 30 secs
100623 11:20:43 3780 xpd-I: SchedCron: running regular checks
100623 11:20:43 3780 xpd-I: ProofServCron: 0 sessions are currently active
100623 11:20:43 3780 xpd-I: ProofServCron: next sessions check in 30 secs
100623 11:20:45 3780 xpd-I: sajzhsof.24760:34@ui0004-int: ClientMgr::MapClient: user sajzhsof logged-in; type: ClientMaster
100623 11:20:53 3780 xpd-I: sajzhsof.24760:34@ui0004-int: Protocol::Recycle: user sajzhsof disconnected; type: ClientMaster
100623 11:21:03 3780 xpd-I: sajzhsof.24771:35@ui0004-int: ClientMgr::MapClient: user sajzhsof logged-in; type: ClientMaster
100623 11:21:03 3780 xpd-I: ProofServ::SetAdminPath: creation/assertion of the status path /tmp/.xproofd.1093/activesessions/s
 successful!
100623 11:21:03 3780 xpd-I: sajzhsof.3811:36@localhost.localdomain: ClientMgr::MapClient: user sajzhsof logged-in; type: Inter
100623 11:21:13 3780 xpd-I: ProofServCron: 1 sessions are currently active
100623 11:21:13 3780 xpd-I: ProofServCron: next sessions check in 30 secs
```

Fig 4: PROOF log file

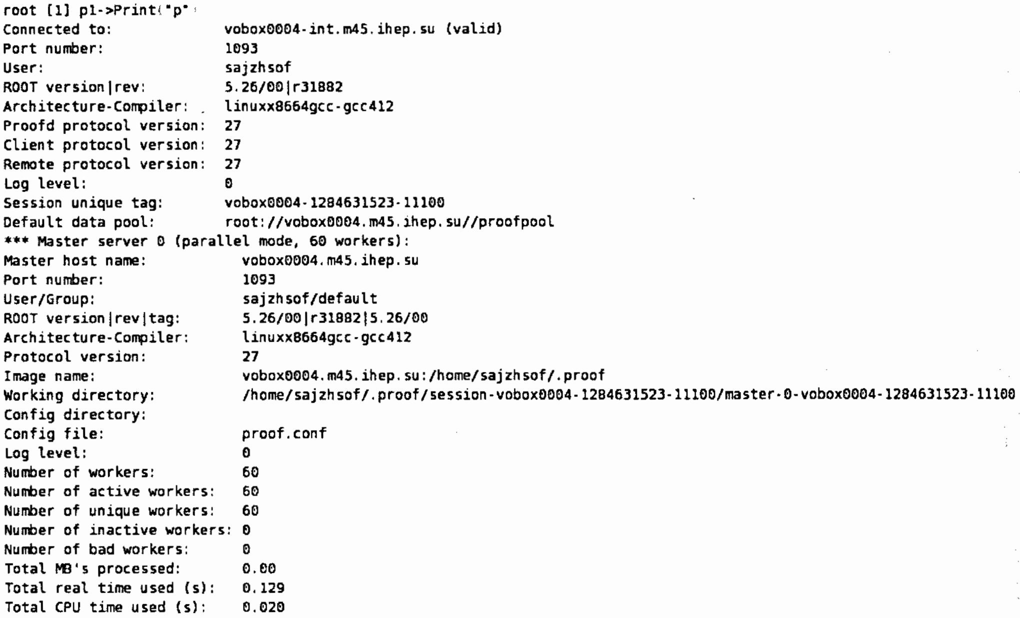
When user starts proof connection, he can see useful short information (fig. 5).



```
[sajzhsof@ui0004]~% root
Couldn't find font "-adobe-helvetica-medium-r-*-*-10-*-*-*-*-*-iso8859-1",
trying "fixed". Please fix your system so helvetica can be found,
this font typically is in the rpm (or pkg equivalent) package
XFree86-[75,100]dpi-fonts or fonts-xorg-[75,100]dpi.
  *******************************************
  *                                         *
  *        W E L C O M E  to  R O O T       *
  *                                         *
  *   Version   5.26/00  14 December 2009   *
  *                                         *
  *  You are welcome to visit our Web site  *
  *          http://root.cern.ch            *
  *                                         *
  *******************************************

ROOT 5.26/00 (trunk@31882, Dec 14 2009, 20:18:36 on linuxx8664gcc)

CINT/ROOT C/C++ Interpreter version 5.17.00, Dec 21, 2008
Type ? for help. Commands must be C++ statements.
Enclose multiple statements between { }.
root [0] TProof *p1 = TProof::Open("proof.m45.ihep.su")
Starting master: opening connection ...
Starting master: OK
Opening connections to workers: OK (60 workers)
Setting up worker servers: OK (60 workers)
PROOF set to parallel mode (60 workers)
```

Fig 5: Start PROOF

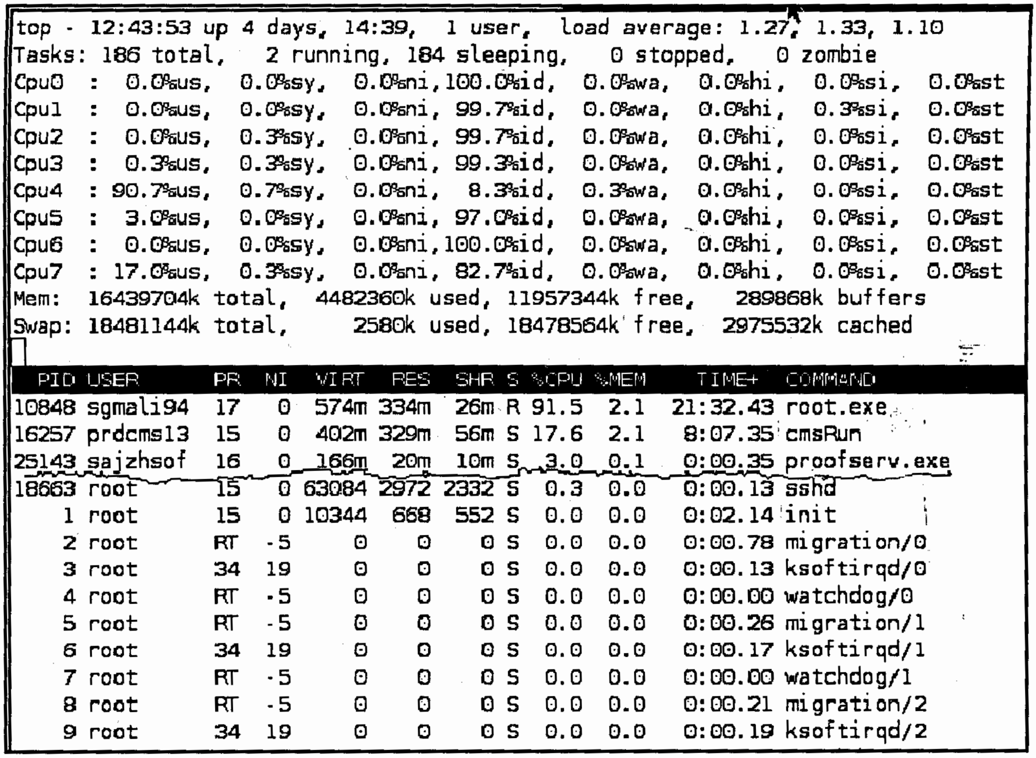
More detail information you can see during PROOF session (fig. 6).



```
root [1] p1->Print("p")
Connected to:             vobox0004-int.m45.ihep.su (valid)
Port number:              1093
User:                     sajzhsof
ROOT version|rev:         5.26/00|r31882
Architecture-Compiler:    linuxx8664gcc-gcc412
Proofd protocol version:  27
Client protocol version:  27
Remote protocol version:  27
Log level:                0
Session unique tag:       vobox0004-1284631523-11100
Default data pool:        root://vobox0004.m45.ihep.su//proofpool
*** Master server 0 (parallel mode, 60 workers):
Master host name:           vobox0004.m45.ihep.su
Port number:                1093
User/Group:                 sajzhsof/default
ROOT version|rev|tag:       5.26/00|r31882|5.26/00
Architecture-Compiler:      linuxx8664gcc-gcc412
Protocol version:           27
Image name:                 vobox0004.m45.ihep.su:/home/sajzhsof/.proof
Working directory:          /home/sajzhsof/.proof/session-vobox0004-1284631523-11100/master-0-vobox0004-1284631523-11100
Config directory:
Config file:                proof.conf
Log level:                  0
Number of workers:          60
Number of active workers:   60
Number of unique workers:   60
Number of inactive workers: 0
Number of bad workers:      0
Total MB's processed:       0.00
Total real time used (s):   0.129
Total CPU time used (s):    0.020
```

Fig 6: PROOF session detail information

PROOF and GRID jobs can be executed together on the same GRID site (fig.7).



```
top - 12:43:53 up 4 days, 14:39,  1 user,  load average: 1.27, 1.33, 1.10
Tasks: 186 total,   2 running, 184 sleeping,   0 stopped,   0 zombie
Cpu0  :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu1  :  0.0%us,  0.0%sy,  0.0%ni, 99.7%id,  0.0%wa,  0.0%hi,  0.3%si,  0.0%st
Cpu2  :  0.0%us,  0.3%sy,  0.0%ni, 99.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu3  :  0.3%us,  0.3%sy,  0.0%ni, 99.3%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu4  : 90.7%us,  0.7%sy,  0.0%ni,  8.3%id,  0.3%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu5  :  3.0%us,  0.0%sy,  0.0%ni, 97.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu6  :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Cpu7  : 17.0%us,  0.3%sy,  0.0%ni, 82.7%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st
Mem:  16439704k total,  4482360k used, 11957344k free,   289868k buffers
Swap: 18481144k total,     2580k used, 18478564k free,  2975532k cached

  PID USER      PR  NI  VIRT  RES  SHR S %CPU %MEM    TIME+  COMMAND
10848 sgmali94  17   0  574m 334m  26m R 91.5  2.1  21:32.43 root.exe
16257 prdcms13  15   0  402m 329m  56m S 17.6  2.1   8:07.35 cmsRun
25143 sajzhsof  16   0  166m  20m  10m S  3.0  0.1   0:00.35 proofserv.exe
18663 root      15   0 63084 2972 2332 S  0.3  0.0   0:00.13 sshd
    1 root      15   0 10344  668  552 S  0.0  0.0   0:02.14 init
    2 root      RT  -5     0    0    0 S  0.0  0.0   0:00.78 migration/0
    3 root      34  19     0    0    0 S  0.0  0.0   0:00.13 ksoftirqd/0
    4 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/0
    5 root      RT  -5     0    0    0 S  0.0  0.0   0:00.26 migration/1
    6 root      34  19     0    0    0 S  0.0  0.0   0:00.17 ksoftirqd/1
    7 root      RT  -5     0    0    0 S  0.0  0.0   0:00.00 watchdog/1
    8 root      RT  -5     0    0    0 S  0.0  0.0   0:00.21 migration/2
    9 root      34  19     0    0    0 S  0.0  0.0   0:00.19 ksoftirqd/2
```

Fig 7: PROOF and GRID jobs

The most important conclusion is PROOF compatible with GRID site. PROOF cluster easy to install. It has not limit worker nodes. Can be install by rpm or deb (multiplatform).

In the future it is planned to do authentication by krb5 (now by uid, gid) and PROOF monitoring system.

References

[1] http://root.cern.ch

[2] Data Analysis with PROOF - G Ganis, J Iwaszkiewicz, F Rademakers. Proceedings of ACAT 2008 Conference.

[3] Dynamic Setup for Clusters with Multi-Master Architecture - K. Opocenska, Master Thesis, Faculty of Mathematics and Physiscs, Charles University in Prague, Czech Republic, February 2010.

THE ALICE GRID OPERATION

G. Shabratova *(on behalf of ALICE)*

Laboratory of High Energy Physics,
Joint Institute for Nuclear Research, 141980, Dubna, Russia

Since the start of the LHC physics program in October 2009 the ALICE experiment at CERN has collected a significant amount of data. To record, process and analyze the data, ALICE has developed over the last 8 years a distributed computing model and Grid middleware AliEn and is using resources deployed in the framework of the WLCG project. Presently, the ALICE Grid is operating on more than 90 sites worldwide, with about 30K CPU cores and 6PB of disk and tape storage. ALICE implements the different Grid services provided by the gLite middleware into the experiment computing model. During the period 2008-2009 the WLCG project deployed new versions of services which are used by ALICE, for example the gLite3.2 VOBOX, the CREAM-CE and the gLite3.2 WMS. As the current LCG-CE, still widely used at many sites, is about to be deprecated in benefit of the new CREAM CE, ALICE has been an early adopter and tester of this middleware. The experiences and issues found during the initial operations have served as a basis for the service improvements and presently CREAM is fully in production and is deployed at every site used by ALICE.

ALICE has been also a leader in testing and implementing other generic services - the gLite3.2 VOBOX and WMS before their full deployment. In this talk we present a summary of the ALICE experiences with these new services also including the test results of the other LHC experiments. In addition to the workload management, the talk will also present the ALICE approach to a fully distributed storage, based on the xrood/scalla suite.

Finally we will discuss the PROOF-based parallel interactive analysis facilities.

Introduction

The ALICE [1] experiment at the CERN Large Hadron Collider (LHC) will accumulate data at a unprecedented speed and volume. The yearly estimate is 1.5PB of raw data from the experimental setup and additional 1.5PB of reconstructed data and Monte-Carlo simulations. The data management and processing is done on the Worldwide LHC Grid [2, 3] (WLCG), encompassing hundreds of computing centers with many thousands of CPUs and PB scale disk and mass storage systems. The data volumes and distributed computing environment present a set of unique challenges for the reconstruction and analysis software and the ways the physicists perform the data analysis.

However, this computing model has an intrinsic delay in providing results from the data that are processed. In the AliEn model, the Grid is a large distributed batch system where use jobs typically process very large data volumes and must pass through several steps. They are submitted to a central queue, then split and assigned to computing farms with suitable resources, scheduled within the local batch systems and executed on the worker nodes. Outputs are finally stored on the user's sandbox. This approach is not efficient for code prototyping where tasks have generally short execution time and need several iterations over a data set. Typical examples are the tuning of cuts during the development of an analysis as well as calibration and alignment. One of the most important aspects of data analysis is the speed with which it can be carried out. Fast feedback on the collected data and publication of results is essential for the success of the experiment.

The ALICE experiment offers its users a cluster for quick interactive parallel data processing called the CERN Analysis Facility [4] (CAF) which runs the PROOF [5] software (Parallel ROOT Facility).

So the software infrastructure created by ALICE in the past ten years had to respond to the first LHC data taking. The stable and secure processing of ALICE data is managed by sure operation of all components of this infrastructure: GRID ALICE environment – AliEn, WLCG services together with perfect support and operation across more than 60 sites all over the world. Success in reconstruction and analysis of data has been and will demonstrate an adequate construction of this infrastructure to the goal of the most modern data development and analysis. Bellow the main components of this infrastructure will be considered.

1. ALICE GRID environment - AliEn

AliEn[6] is a set of middleware tools and services that implement a GRID structure. Since 2005 AliEn has been used both for data production and end-user analysis. AliEn has a very successfully served goal of hiding the complexity and heterogeneity of the underlying GRID services from the end user. The basic AliEn components will be considered. The most modern development of these components will be presented that allows one to automatize the process of production, data processing and end-user analysis.

1.1. File catalog with metadata capabilities

The file catalogue is the principal component of the AliEn system. Unlike real file systems, it does not own the files but provides the mapping between the visible to the end user Logical File Name (LFN) and one or more Physical File Names (PFN). Multiple PFNs are remotely stored replicas of the same file. The PFN entries in the catalogue describe a physical location of files by indentifying the name of the AliEn storage element and the path to the local file.

LFN can be manipulated by user. Moreover, two different LFNs can point to the same PNF. To prevent duplicate file entries, each LFN is associated to Globally Unique Identifier (GUID) entry.

The interface to the file catalogue like UNIX file system has a hierarchical structure of files and directories. The catalogue provides also tools to associate user-defined meta information to any entry.

The file catalogue is used by almost all components of AliEn. It contains information on all data and software packages used for data production and analysis. The job output is also registered in the catalogue as well the metadata definitions and various triggers used for automatic data management or processing.

For user the catalogue behaves as a single entry. However, the internal structure is divided between LFN and the GUID catalogue, which are kept in an independent database.

1.2. Data management and storage

AliEn practices a global and transparent file access. Every file can be accessed remotely in any storage element if necessary. Jobs are scheduled close to the storage location of the file in order to keep the file access the most optimal. Files are referenced via a LFN or GRID and storage end points. PFNs are resolved by AliEn file catalogue taking into account the client location. The policy is to return the closest replica of a requested file, unless a specific storage element is explicitly requested by job. The closeness of the storage element is defined in a central configuration and by storage domain names.

AliEn distinguishes between batch and interactive file access and scheduled file transfers. Interactive file access is done using the xrootd protocol by the scalla software suit [7]. Xrootd protocol is well integrated into the ROOT software framework [8] used by ALICE. It offers specific optimizations for a remote file access in the experimental data analysis:

a) connection multiplexing,
b) vector read request,
c) plugin architecture to integrate enhanced file system and security features,
d) fault tolerance with automatic retry mechanism, asynchronous server responses and server redirection.

xrootd offers a copy client (xrdcp) and posix interface for a partial file access. ROOT has special plugin for xrootd access (TXNetFile) to optimize file access using vector read etc.

Scheduled data transfer is handled by AliEn FTD (File Transfer Daemon). Depending on the capability of storage elements involved in the transfer, the protocol can be configured individually. If supported by the storage system, wide area transfers are done using common GRID middleware services like gLite FTS (File Transfer Service)[9] which itself used GLOBUS GridFTP[10] as a WAN (Wide Area Network) optimized transfer protocol and a SRM interface[11].

The data management system today is based on four storages technologies:

i) CASTOR2,
ii) dCache,
iii) DPM,
iv) Scalla (xrootd).

CASTOR2 [12] and dCache [13] are used by ALICE in Tier-0 and Tier-1 centers as they provide a tape backend. DPM [14] and Scalla are used mainly in Tier-2 centers as disk pool managers. There are an ongoing development to enable xrootd protocol on CASTOR2, dCache and DPM, which allow application to use only xrootd as a global protocol for data access to any storage. These three storage elements provide also a SRM interface and a GridFTP protocol for scheduled WAN file transfers.

1.3. Workload management system

One of the main concepts applied in the workload management is based at JobAgent (JA) schema. During submit of JA on a worker node, it executes a set of sanity checks on the environment and if successful, sends its description to the Job Broker. These checkers include the available disk space, memory, platform architecture and OS, application packages currently installed or that could be installed and its time of live. If this description satisfies the requirements of a job waiting in the Task Queue, the Job Broker assigns the job to the JA. Once the user job terminates inside the JA, the latter registers its output in the catalogue and requests another job to execute. If there are no more jobs to execute, the JA exits.

Job Agents usage improves the workload system in multiple ways. First of all, it eliminates the possibility of job failure due to problems with a local batch system or on the worker nodes themselves. It reduces also the time between the submission and execution of user jobs. As JA can execute several jobs, it reduces the load on the local batch system. In the same time functionality of JAs is far out weights the drawbacks. In the current operational model of ALICE any overload of the Job Broker with more than 6000 jobs running in parallel is not observed.

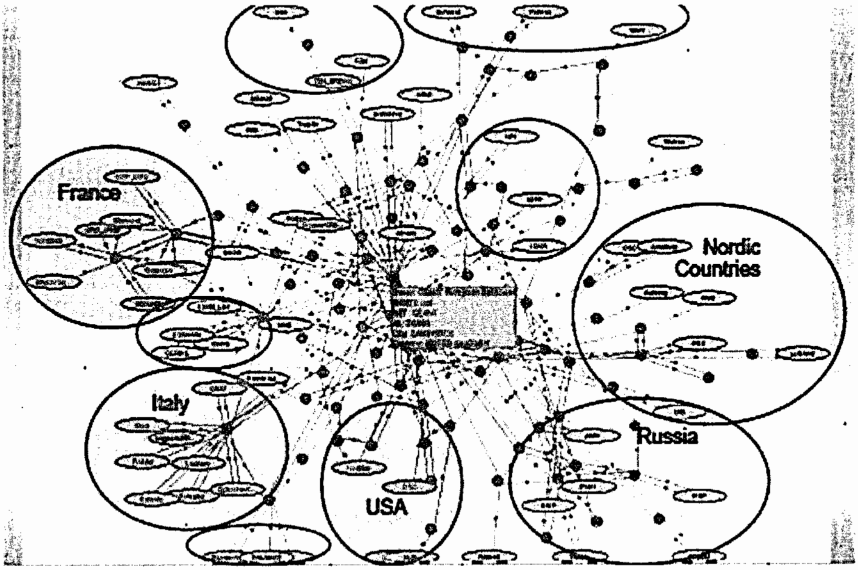


Fig. 1: Map of SEs used by ALICE

The version of AliEn used in the first data taking v2.18 has provided implementation of a large number of features intended to escalate the number of concurrent jobs for fundamental ALICE activities: reconstruction of Pass1 and Pass2, calibration, MC production and user analysis. There are two new features important for the GRID sites and the end users:

1. Implementation of job and file quotas:
    a) limit on the available resources per user,
    b) jobs: #jobs, CPU cost, running time,
    c) files: #files; total size, including replicas.
2. Automatic storage elements discovery:
    a) finding the closest working SEs of a QoS for optimal configuration. The discovery is based on MonALISA monitoring information (topology, status, etc.);
    b) sorting replicas for reading from the closet available one;
    c) simplifying the selection of SE and adding more options in case of special needs.

The algorithms of closed SE based at discover and derived network topology (see Fig. 1) which is taking into account each SE associated with a set of IP addresses: VOBox IP, IPs of xrootd redirector & nodes. MonALISA performs tracepath/traceroute between all VOBoxes, recording all routers and the RTT of each link, SE nodes status & bandwidth between sites. Group the routers is taking in the respective Autonomous Systems (AS) computes the distance (RTT) between sites Distance(IP, IP) hierarchy v.s.: same C-class network, common domain, same AS, same country, same continent , use known distance between ASes et al.

Results of the Storage discovery in AliEn could be expressed in:

1. Flexible storage configuration:
    - *QoS tags are all that users should know about the system;*
    - *We can store N replicas at once;*
2. Maintenance-free system:
    - *Monitoring feedback on known elements, automatic discovery and configuration of new resources;*
3. Reliable and efficient file access:
    - *No more failed jobs due to auto discovery and failover in case of temporary problems;*
    - *Use the closest working storage element(s) to where the application runs.*

1.4. Interoperability

AliEn has interfaces to other GRID flavors. At present, these interfaces allow ALiEn to interoperate with g Lite, ARC [15] and OSC [16] GRIDs. Interfaces can be deployed at different hierarchy levels. It would be possible to make the whole GRID behave as a single CE through a gateway deployed by the GRID service provider. The interface with ARC is geared toward this model. Another approach is to associate each computing element from a GRID with an AliEn computing element. This is used in the gLite interface.

There is also a possibility to interface any number of Virtual Organizations running AliEn to each other. This interface makes a whole AliEn VO appear as a single computing element for another VO. The interface uses the Job Agent model, where one VO is submitting Jas to the resources of other VO.

1.5. Monitoring

Monitoring in AliEn is based on the MonALISA framework [17]. The monitoring architecture consists of a hierarchical structure with a selective aggregation of the monitoring information sent upward to the next level (see Fig. 2). The aggregation is a determinant factor in reducing the overall volume of information, while preserving important details.

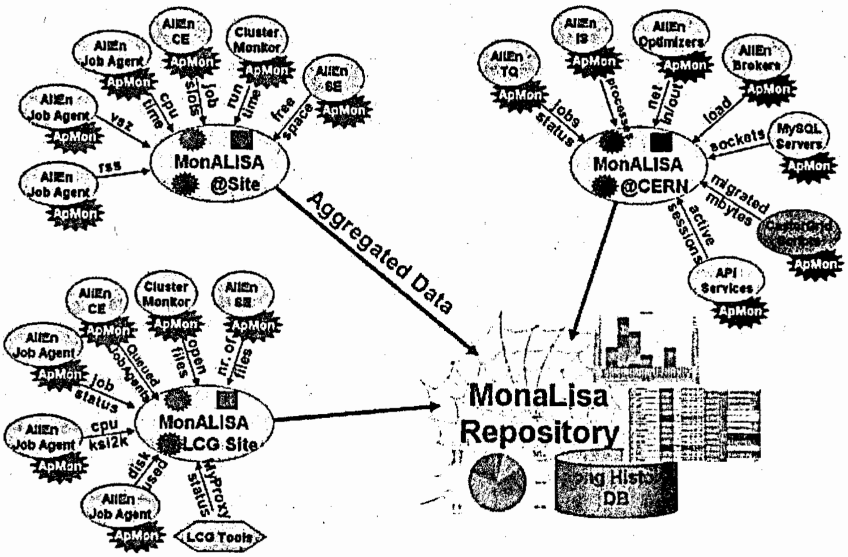


Fig. 2: Information flow from the monitored entries

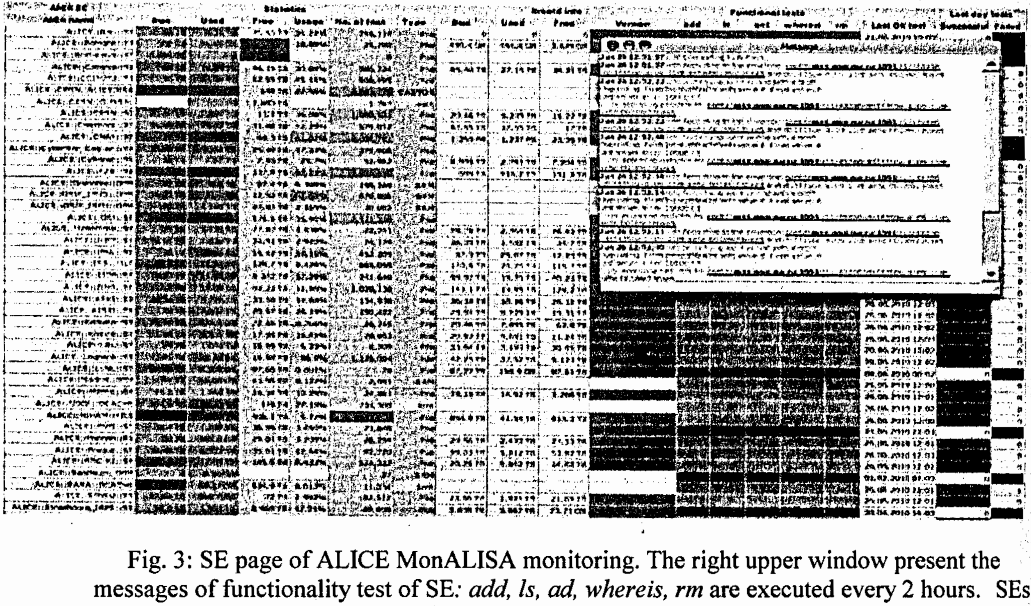


Fig. 3: SE page of ALICE MonALISA monitoring. The right upper window present the messages of functionality test of SE: *add, ls, ad, whereis, rm* are executed every 2 hours. SEs failing the functionality tests are removed from the storage matrix

All AliEn services and clients are instrumented with ApMon[18], which transmits general job and host monitoring information and also allows services to send specific parameters. Extensive information is collected about CPU and memory usage, consumed and wall CPU time, open files and networks traffic. Each site of the AliEn GRID has a dedicated node (VO-Boc) where the Alien site services run. On this nod, MonALISA also monitors the health of the AliEn services through periodical functional tests. Each MonALISA service has a short-time temporary buffer for highly detailed history data. It performs the filtering and aggregations of data and sends it to a central repository. Fig. 3 presents the monitoring SE (Storage Element) page which presents an allocated and used space of each SE in a real time and also health of SEs.

The MomALISA repository [19] subscribes to general interest data and stores them into a PostgreSQL database. It offer both near real time and history views, with different levels of detail – ranging from general overviews to details about individual user jobs. Currently the MonALISA repository for ALICE is keeping the history for 40000 different data series for the last 18 months. The data stream is in average 2500 values per minute with a current database size of 140 GB (~1.4 billion data points).

In addition to presenting the information to users and site administrators, the repository also has the task of taking the automated decisions based on the monitoring information received. The example of such usage has been presented upper in the description of the SE close decision algorithm.

2. WLCG Services of ALICE sites

For the stable and successful operation of ALICE software in GRID activity of sites there is a necessity of installation and support of the following WLCG [20] services at these sites: VObox [21], CE front-end job submission directly to the local batch system, Xrootd based Storage system

2.1. VO box – Virtual Organization box

The VO-box is a type of node where experiments can run specific agents and services to provide a reliable mechanism to accomplish various tasks. It is provided as an interim solution in order to allow experiments to provide their own services whenever the middleware still does not provide a required functionality. The access to the VO-box (or VO node) is restricted to the Software Group Manager (SGM) of the Virtual Organization (VO). Note that there is not a unique description about "the" gLite VO-box as each VO provides their own requirements. At the time when the VO-box has been deployed, the VOs have completed the "LCG VOBox Operations Recommendations and Questionnaire" which describes, for instance, the requirements on the hardware (e.g., "any modern dual CPU system would be sufficient"), the preferred operating system (Scientific Linux 5 last year), or the requirements on the backup policy. The VO software running on the VO-box, and the open ports, are described in the "VO-Box Security Recommendations and Questionnaire" (VOBOX-SRQ). Templates of the questionnaires have been provided by the Joint Security Policy Group. VO-box WIKI for the templates and the current versions of the completed questionnaires could be found in https://twiki.cern.ch/twiki/bin/view/LCG/VoBoxesInfo.

What daemons are running at VO box:

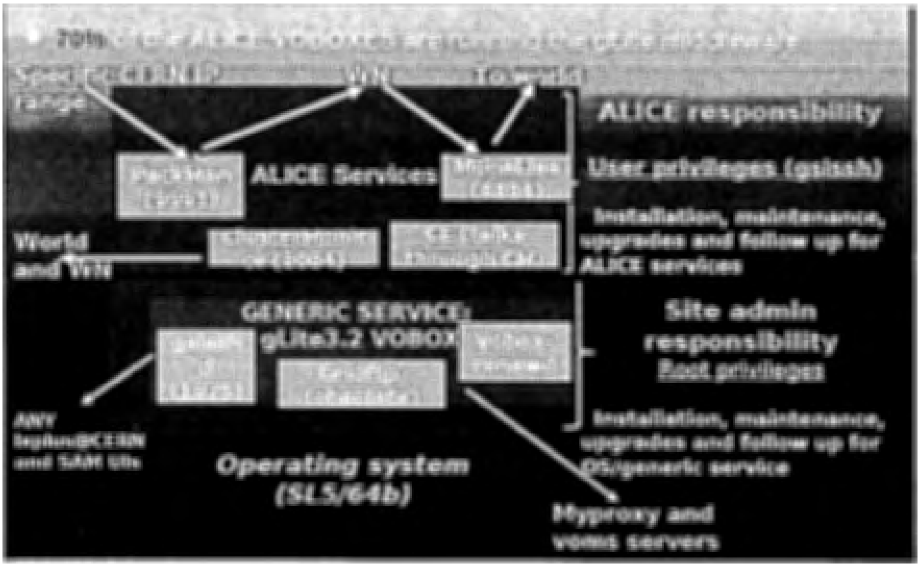
- a GSISSH server (running by default on port 1975) which allows *ssh* connection authorized through X509 proxies and proxy delegation (**/opt/globus/sbin/sshd –p 1975**),
- a GRIS (registering to the site GIIS) which publishes the GSISSH service and port (usually **/usr/sbin/slapd –f /opt/bdii-slapd.conf –b ldap://localhost:2171 –u edguser**),
- a Proxy Renewal Service (together with a user level tool) to ensure automatic refresh of user creditals(**/opt/lcg/sbin/vobox-renewd**).

Modern VO box at the layer of gLite 3.2 is a gLite User Interface (UI) with two added features:

1. automatic renewal of user proxy, what provide the long-lived proxy in myproxy.cern.ch,
2. provide the direct access to the experimental software area with this access restriction to the **VOMS**, only ***lcgadmin*** has a such access. So VO box is unique service not accepting pool accounts, **ONLY** single local account is valid.

gLite-VOBOX operation is shared between site and experiment. Participation of the site in this operation is minimal, only installation and configuration. VO box service is one of the most stable services. The experiment support side is in support of the whole ALICE site at any moment by the CERN team. Let us remind that gLite3.2 VO box has been created by the ALICE support team in collaboration with the WLCG. All VO boxes are monitored now via NAGIOS. A few words about alarm VO box system:

- T0 VOBOXEs are monitored via Lemon with an alarm system behind,
- T1 sites can be notified via GGUS alarm tickets,
- T2 sites will be notified on best effort base.



ALICE responsibility

ALICE Services

User privileges (gsissh)

Installation, maintenance, upgrades and follow up for ALICE services

World and WN

GENERIC SERVICE: gLite3.2 VOBOX

Site admin responsibility

Root privileges

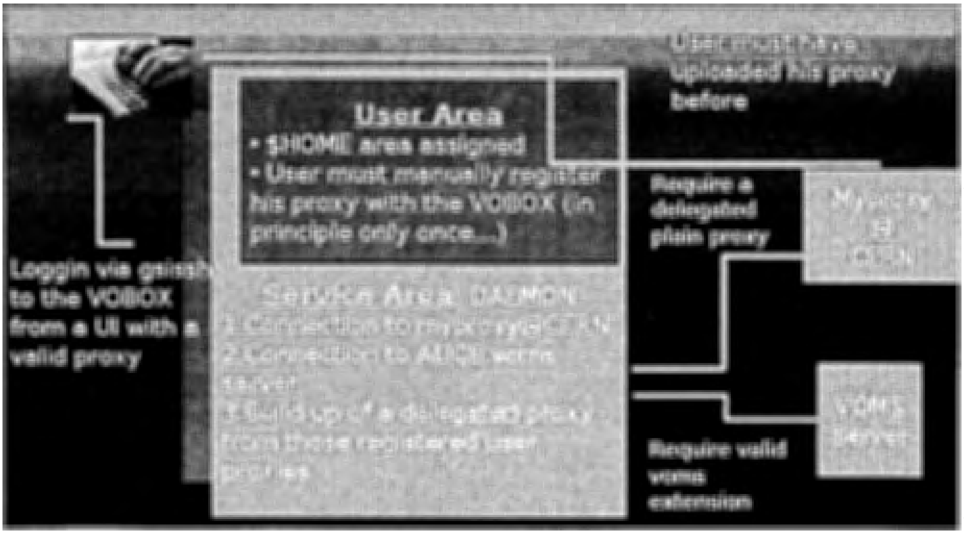
Installation, maintenance, upgrades and follow up for OS/generic service

ANY lxplus@CERN and SAM UIs

Operating system (SL5/64b)

Myproxy and voms servers

Fig. 4: The structure of VO box



User Area

- $HOME area assigned
- User must manually register his proxy with the VOBOX (in principle only once...)

Loggin via gsissh to the VOBOX from a UI with a valid proxy

Require a delegated plain proxy

Require valid voms extension

Fig. 5: The Proxy renewal mechanism at gLite3.2 VO box

In Fig. 4 one can see VO box components and share of support of these components between the site and the experiment and Fig. 5 is illustrated the proxy renewal mechanism.

2.2. The CREAM Service

The job management component is a Grid component which is used to submit, cancel, and monitor jobs for execution on a suitable computational resource, called a Computing Element (CE). A CE is the interface to a large farm of computing hosts managed by a Local Resource Management System (LRMS), such as LSF or PBS. Moreover, a CE provides additional features to those of the underlying batch system, such as Grid-enabled user authentication and authorization, accounting, fault tolerance and improved performance and reliability.

New Computing Element components, called the Computing Resource Execution and Management (CREAM) and CEMonitor [22], are designed to manage a CE in the gLite Grid middleware. CREAM provides a service for job operations. It exposes an interface based on Web Services, which enables a high degree of interoperability with clients written in different programming languages. CEMonitor is a general-purpose asynchronous notification engine, which can be coupled with CREAM in order to notify clients when job status changes occur. Such a notification mechanism is particularly important for those clients which need to handle a large amount of jobs.

CREAM is a job submission service: users can submit jobs, described via a classad-based Job Description Language (JDL) expression [23], to CREAM CEs. The JDL is a high-level notation based on Condor classified advertisements (classads) [24] for describing jobs and their requirements. CREAM supports the execution of batch (normal) and parallel (MPI) jobs. Normal jobs are single or multithreaded applications requiring one CPU to be executed; MPI jobs are parallel applications which usually require a larger number of CPUs to be executed, and which make use of the MPI library for inter process communication. CREAM is a Java application which runs as an Axis servile inside the Tomcat application server. The CREAM interface exposes a set of operations which can be classed **in three groups** (see [25] for details).

The first group of operations (Lease Management) allows the user to define and manage leases associated with jobs. The lease mechanism has been implemented to ensure that all jobs get eventually managed, even if the CREAM service loses connection with the client application due to network partitioning. Each lease defines a time interval, and can be associated with a set of jobs. A lease can be renewed before its expiration; if a lease expires, all jobs associated with it are terminated and purged by CREAM.

The second group of operations (Job Management) is related to the core functionality of CREAM as a job management service. Operations are provided to create a new job, start execution of a job, suspend/resume or terminate a job. Moreover, the user can get a list of all owned jobs, and it is also possible to get a status of the set of jobs.

Finally**, the third group** of operations (Service Management) deals with the whole CREAM service. It consists of two operations, one for enabling/disabling new job submissions, and one for accessing general information about the service itself. Note that only users with administration privileges are allowed to enable/disable job submissions.

Authentication (implemented using a GSI based framework) is properly supported in all operations. Authorization on the CREAM service is also implemented, supporting both Virtual Organization (VO) based policies and policies specified in terms of individual Grid users.

A Virtual Organization is a concept that supplies a context for operation of the Grid that can be used to associate users, their requests, and a set of resources. CREAM interacts with a VO Membership Service (VOMS) [26] service to manage VOs; VOMS is an attribute issuing service which allows high-level group and capability management and extraction of attributes based on the user's identity. VOMS attributes are typically embedded in the user's proxy certificate, enabling the client to authenticate as well as to provide VO membership and other evidence in a single operation.

The CREAM-CE [28] is the latest gLite service created to replace the current LCG-CE [29]. The CREAM-CE is a lightweight service for job management operations at the CE level, which includes two submission modes:

- Submission to the CREAM-CE via the gLite WMS [27], [29] service,
- Direct submission via generic clients.

The ALICE experiment has expressed their interest in the CREAM-CE system since summer 2008. ALICE was interested in the direct submission mode of the CREAM-CE with a major goal: replace the use of the gLite WMS that was showing large instabilities in the ALICE production by using the direct submission mode of the CREAM-CE. In addition, the experiment had negotiated two years ago with the Security Team of CERN a voms extension of 48 hours; time enough for any production or analysis job.

The first test phase of the CREAM-CE was performed at FZK-LCG2 Tier1 (Germany) from June to August 2008. The WLCG GDB advice (testing the CREAM-CE in parallel to the LCG-CE) is translated into the ALICE computing model in a second VOBOX deployment per site. The VOBOX service is deployed at all sites providing access to the ALICE experiment following the experiment computing model and it is responsible for the job agent submissions. Each VOBOX can submit to a single backend, LCG-CE or CREAM-CE. Therefore a second VOBOX was necessary to ensure the CREAM-CE/LCG-CE duality A simplified schema of the ALICE workload management components is presented in Fig. 6.

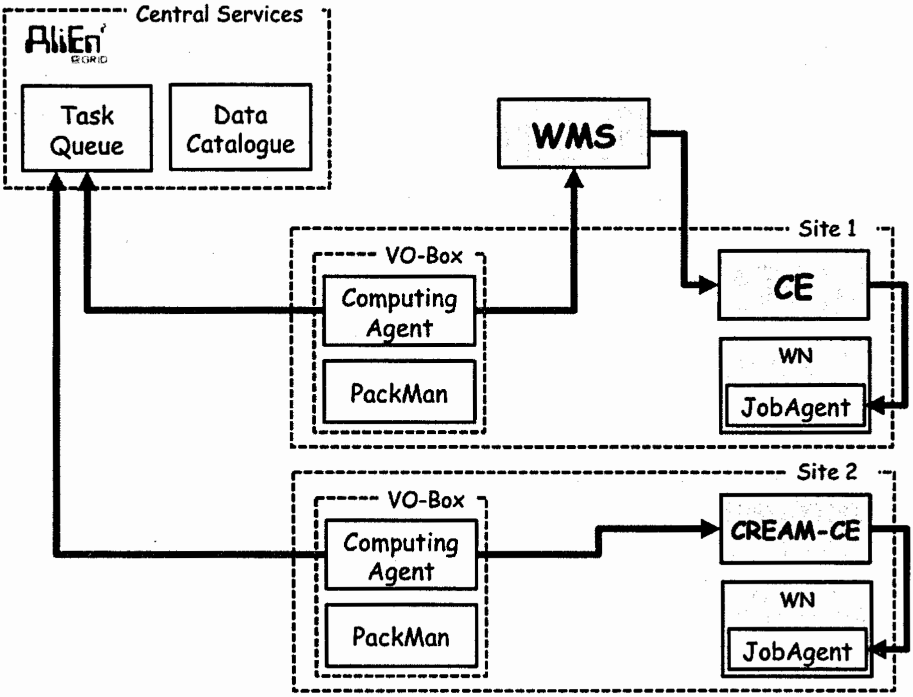


Fig. 6: A simplified schema of the ALICE workload management components in the parallel processing of CE-LCG and CREAM-CE

In the second test phase, starting from February 2009, ALICE asked several sites to deploy a CREAM-CE and second VO box, submitting to the sites production queues in parallel with existing LCG-CE. Even if some sites reported problems in installation and configuration the CE, after entering production the CREAM-CEs were remarkably stable; this of course and because of obvious shortcut in the submission chain. In less than one month ALICE achieved more than 67000 jobs executed through the CREAM-CE services of all these mentioned sites. Fig. 7 shows the comparison of three sites load by jobs processing via CREAM-CE and LCG-CE in the same 4 months.

From the beginning of 2010 CREAM has been fully in production and deployed at every site used by ALICE.

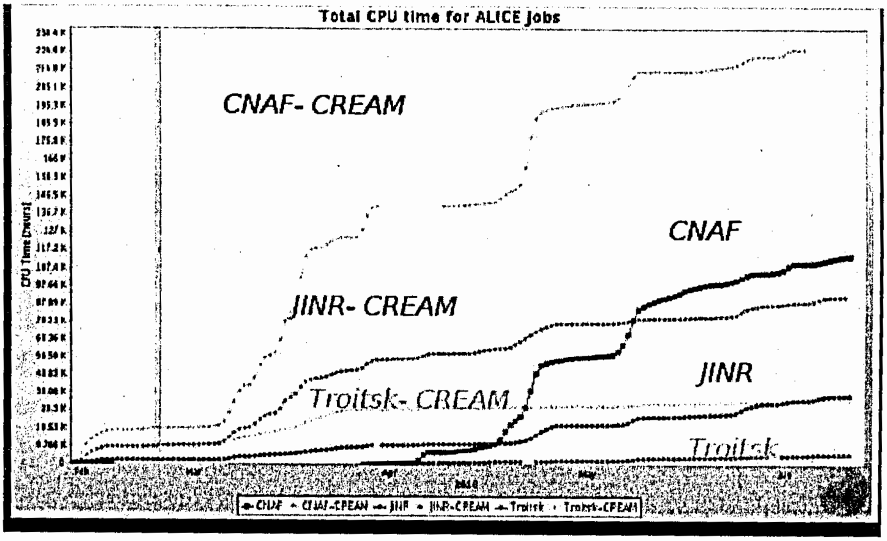


Fig. 7: Distribution of Done jobs in four months operation of three ALICE sites (CNAF, JINR and Troitsk) via CREAM-CE and LCG-CE

3. ALICE Analysis Facility - AAF

The ALICE experiment at CERN LHC intensively uses the PROOF cluster for fast analysis and reconstruction. PROOF enables an interactive parallel processing of data distributed on clusters of computers or multi-core machines. **PROOF** (Parallel ROOT Facility) [30] allows an interactive parallel analysis on a local cluster. Interactive means that you see the results right away (contrary to a batch job where you have to wait for the job to finish before you see the results). Parallel means that several nodes execute sub sets of your data at the same time. You connect to a PROOF system from your usual ROOT prompt. Using PROOF is aimed to be transparent, that means you can execute the same analysis code locally and on a PROOF system. The CERN Analysis Facility (CAF) [31] is a cluster at CERN running PROOF. It can be used for a prompt analysis of pp data as well as selected PbPb data. Furthermore calibration programs can be run on the CAF. The CAF will run PROOF for ALICE. Simulated data and measured data, once ALICE starts data taking, is available on the CAF. It is used also to perform analysis and calibration. The aim of the CAF is conception ally different from analysis on the Grid. The CAF will not make it possible to analyze all the data taken by ALICE because its space is limited. However, it is possible to run an analysis and see results after a few minutes or even seconds, thus allowing very fast development cycles.

3.1. Data distribution and staging

The CAF disk space is principally used as a cache space for data imported from the Grid storage systems. It is not meant for permanent storage because data for user analysis might change quickly and, moreover, only a subset of the data produced by the experiment may be significant for fast interactive parallel analysis. The CAF provides an automatic mechanism to stage files stored in the AliEn SEs upon user request. Presently about 110000 files have been staged, corresponding to 23 million p+p events with a total size of 8TB. The concept of dataset is used to group and process files describing the input for analysis as well as for data staging from the Grid. Datasets have replaced the initial publication of available files in the form of a text file. Data staging is performed through datasets. Datasets are lists of files registered by users for processing with PROOF. They may share the same physical file, i.e. files that are in several datasets are stored only once, allow one to keep the file information consistent and take care of disk quotas.

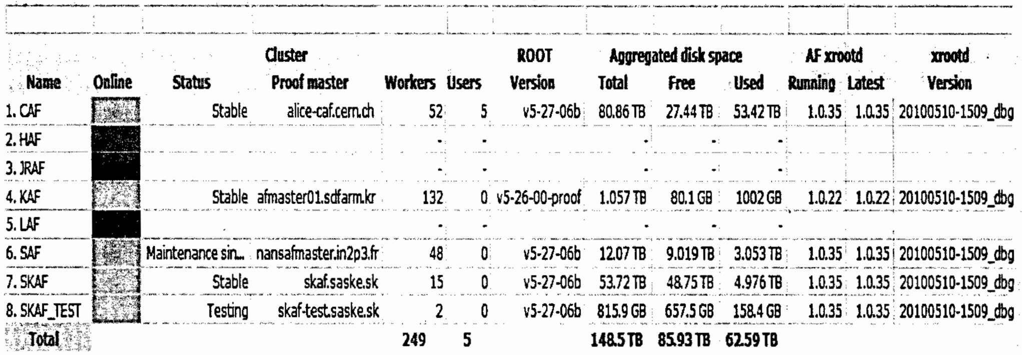
Data staging from AliEn is performed by a datastager script that is plugged into the cmsd service on each disk server. Cmsd (cluster management service daemon) is part of the Scalla/xrootd software suite. A correspondence between AliEn datasets and PROOF datasets exists, i.e. a PROOF dataset can be created from an AliEn dataset. As a user registers a new dataset, the xrootd redirector selects the suited xrootd disk servers and forwards the request to stage each single file. The disk servers send the request to their datastager which performs the staging. When the disk usage reaches a high-water mark (90%), a garbage collector is triggered to delete files with the oldest access time that have not been accessed for a certain period, e.g. one day. This condition preserves the consistency of the datasets. The garbage collection stops when a low-water mark (80%) is reached.

3.2. Authentication

User authentication is based on the Globus Security Infrastructure (GSI) and uses X509 certificates and an LDAP-based configuration management. Grid certificates are used to authenticate the users to the system. In this way, the same mean of authentication is used either for Grid and the CAF. An additional advantage is the possibility to access directly Grid files from the CAF workers. The framework for fast parallel reconstruction of raw data has been recently developed relying on this new feature.

Last year in ALICE a new schema was developed to use the local PROOF clusters as a distributed PROOF cluster having the same management of application software and staging the data information. This distributed PROOF cluster has been named as AAF (Alice Analysis Facilities). It is a common setup of PROOF clusters using xrootd setup. A detailed information on ALICE Analysis Facilities (AAF) and CAF is available at the AAF web site: aaf.cern.ch. The list of the local clusters combined to AAF is presented in Fig. 8.

Fig. 9 shows some statistics in the last time operation of AAF as to a number of workers, a number of jobs, wall time et al for different analysis groups of user.



		Cluster				ROOT	Aggregated disk space			AF xrootd		xrootd
Name	Online	Status	Proof master	Workers	Users	Version	Total	Free	Used	Running	Latest	Version
1. CAF	▨	Stable	alice-caf.cern.ch	52	5	v5-27-06b	80.86 TB	27.44 TB	53.42 TB	1.0.35	1.0.35	20100510-1509_dbg
2. HAF	■			-	-		-	-	-			
3. JRAF	■			-	-		-	-	-			
4. KAF	▨	Stable	afmaster01.sdfarm.kr	132	0	v5-26-00-proof	1.057 TB	80.1 GB	1002 GB	1.0.22	1.0.22	20100510-1509_dbg
5. LAF	■			-	-		-	-	-			
6. SAF	▨	Maintenance sin...	nansafmaster.in2p3.fr	48	0	v5-27-06b	12.07 TB	9.019 TB	3.053 TB	1.0.35	1.0.35	20100510-1509_dbg
7. SKAF	▨	Stable	skaf.saske.sk	15	0	v5-27-06b	53.72 TB	48.75 TB	4.976 TB	1.0.35	1.0.35	20100510-1509_dbg
8. SKAF_TEST	▨	Testing	skaf-test.saske.sk	2	0	v5-27-06b	815.9 GB	657.5 GB	158.4 GB	1.0.35	1.0.35	20100510-1509_dbg
Total				249	5		148.5 TB	85.93 TB	62.59 TB			

Fig. 8: List of PROOF clusters of AAF

Conclusion

The infrastructure of LHC computing developed and tested by ALICE during the long period of the detector construction demonstrated its full adequacy to data development and analysis during first expositions at the LHC beams.

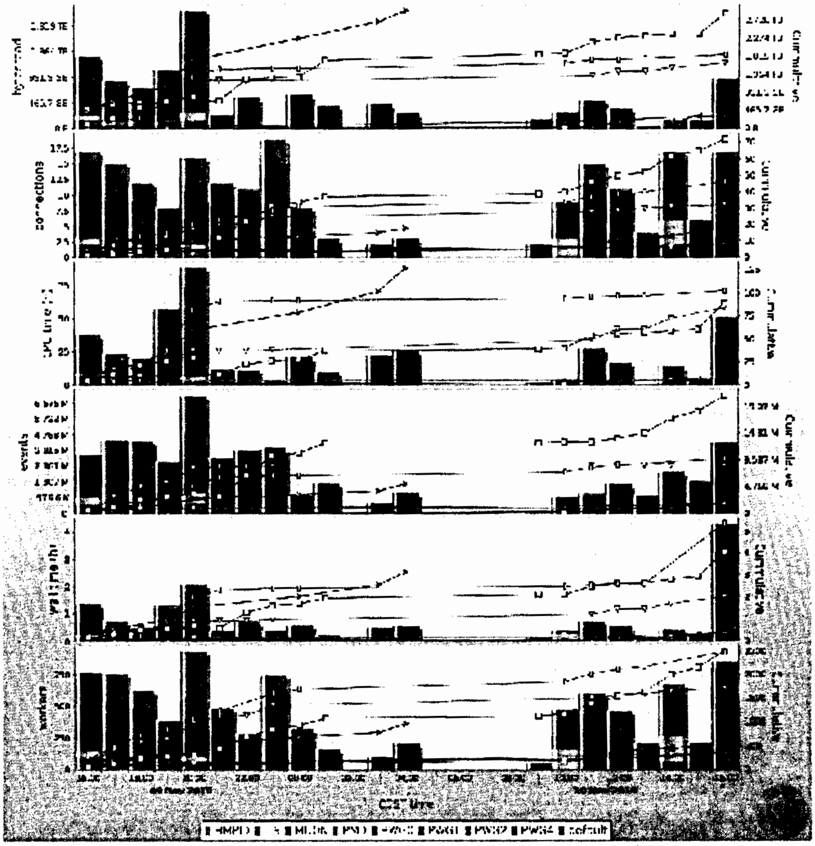


Fig. 9: Few parameters of AAF operation for different analysis group of users

References

[1] ALICE collaboration 1995 Technical Proposal for a Large Ion Collider Experiment at the CERN LHC CERN/LHCC/95-71.

[2] Foster I., Kesselman C. The Grid: Blueprint for a New Computing Infrastructure. /Morgan Kaufmann. San Francisco, 1999.

[3] http://lcg.web.cern.ch/LCG

[4] Grosse-Oetringhaus J.F. 2008 J. Phys. Conf. Ser. 119 072017.

[5] Ballintijn M., Roland G., Brun R., Rademakers F. The PROOF distributed parallel analysis framework based on ROOT // Proc. Conf. for Computing in High-Energy and Nuclear Physics (CHEP), La Jolla, California, 24-28 Mar, 2003.

[6] Saiz P, et al., AliEn –ALICE environment on the GRID, Nucl. Instrum. Meth., A502 (2003) 437.

[7] The Scalla Software Suit, http://xrootd.slac.stanford.edu

[8] ROOT-An Object-Oriented Data Analysis Framework, http//root.cern.ch

[9] gLite – Lightweight Middleware for Grid Computing, http//glite.web.cern.ch/glite

[10] GRIDFTP, http//www.globus.org/grid_software/data/gridftp.php

[11] The Storage Resource Manager Interface Specification, http://sdm.lbl.gov/srm.wg/doc/SRM.v2.2.html

[12] CASTOR-CERN Advanced Storage management, http://castor.web.cern.ch/castor

[13] dCache, http://www.dcache.org

[14] Disk Pool Management, http://www.gridpp.ac.uk/wiki/Disk_Pool_Manager

[15] ARC, The NorduGrid middleware, http://www.nordugrid.org/middleware

[16] OSG, http://www.oprnsciencegreid.org

[17] Legrand I.C., Newman H.B., Voicu R., Cirstoiu C., Grigoras C., Toarta M., Dobre C. MonALISA: An Agent based, Dynamics Service System to Monitor, Control and Optimize Grid base Applications // CHEP 2004, Interlaken. Switzerland.

[18] User Guide for ApMon, http://wiki.egee-see.org/index.php/User_guide_for_ApMON_%28Application_Monitoring_API%29

[19] http://monalisa.cacr.caltech.edu/monalisa.htm

[20] Worldwide LHC Compiting GRID, http://lcg.web.cern.ch/LCG/public/

[21] gLite-VObox, https://twiki.cern.ch/twiki/bin/view/EGEE/GLiteVOBOX

[22] EGEE User's Guide: CREAM Service. EGEE-JRA1-TEC-595770.

[23] Sgaravatto M. CREAM Job Description Language Attributes Specification for the EGEE Middleware document Identier EGEE-JRA1-TEC-592336, 2005. Available online at https://edms.cern.ch/document/592336

[24] Raman R. Matchmaking Frameworks for Distributed Resource Management Ph.D. thesis, 2001.

[25] Aiftimiei C., Andreetto P., Bertocco S., Dalla Fina S., Dorigo A., Frizziero E., Gianelle A., Marzolla M., Mazzucato M., Sgaravatto M., Traldi S. and Zangrando L. Design and implementation of the gLite CREAM job management service Technical Note INFN/TC 09/3 Istituto Nazionale di Fisica Nucleare, 2009.

[26] Aleri R., Cecchini R., Ciaschini V., dell'Agnello L., Frohner A., Loentey K. and Spataro F. Future Generation Computer Systems 21 549{558 ISSN 0167-739X}. 2005.

[27] Garzoglio G., Levshina T., Mhashilkar P. and Timm S. Grid Computing // International Symposium on Grid Computing (ISGC 2007) / ed Lin S. C and Yen E. (Springer). P. 89-98. 2009.

[28] van Engelen R. gSOAP 2.7.6 user guide. 29 Dec. 2005.

[29] Andreetto P. et al. // Proceedings of CHEP'04 (Interlaken, Switzerland), 2004.

[30] Ballintijn M., Roland G., Brun R., Rademakers F. The PROOF distributed parallel analysis framework based on ROOT // Proc. Conf. for Computing in High-Energy and Nuclear Physics (CHEP), La Jolla, California, 24-28 Mar, 2003.

[31] Grosse-Oetringhaus J.F. J. Phys. Conf. Ser. 119 072017, 2008.

GRIDNNN JOB EXECUTION SERVICE: A RESTFUL GRID SERVICE[1]

L. Shamardin, A. Demichev, A. Kryukov, V. Ilyin

Lomonosov Moscow State University Skobeltsyn Institute of Nuclear Physics (MSU SINP),
Russian Federation, 119991, Moscow, GSP-1, Leninskie gory 1(2),
shamardin@theory.sinp.msu.ru

In this work we demonstrate a possibility of using REST architecture style for building grid services as an alternative to traditional methods, such as WSRF services. We demonstrate approaches to implementation of the basic functionality of a grid service and grid resource management. The proposed methods were used in a grid job execution service «Pilot» which was developed for the GridNNN project. This service allows to execute workflow-style jobs and provides a simple but powerful REST API for client applications. The main goal for the service is to automate computations with multiple stages since they can be expressed as simple workflows. This service is deployed and used in GridNNN production infrastructure.

1 Introduction

In this work we describe the architecture and interfaces of the grid job execution service for the GridNNN project [1] which is called «Pilot». In contrast to many modern grid services which are implemented as WSRF web services [2], it is built as a RESTful grid service [3, 4]. A RESTful Web service provides access to a collection of resources using standard HTTP methods like GET, PUT, POST or DELETE and representations which are agreed with the service consumers. It is very important here that the logical operations with the resource conceptually match the HTTP method used for the operation. Grid service built on REST architecture differs from a web service by taking special care of resource lifetime management and resource creation process. This allows to design a simple but powerful API while preserving all properties of a grid service. In this work we describe the purpose of the service, the notion of workflow jobs and tasks which can be executed and discuss the implementation of the Pilot grid service.

2 Jobs and Tasks

Computations carried out in the GridNNN grid infrastructure are represented as jobs, which are composed of tasks. A grid job, or simply a job is a directed acyclic graph composed from tasks. Each node of the graph corresponds to a task, edges define the order of tasks execution. A task is a minimal executable element of a job which is a submission of an executable to some GridNNN Computing Element (CE) which is generally a host running a Globus WS-GRAM grid service. Each task has a set of resource requirements attached defining which grid resources are compatible with a task. These requirements may specify minimal amounts of available memory and storage, number of CPUs and so on. Implicit requirements are satisfied by any CE in the grid.

The job execution service accepts a job from a user and executes this job on a set of grid resources. It will track the execution progress of each task and will try to run any task for which the prerequisites are met on the best resource matching the task resource requirements. It is possible to specify the criteria to determine the success of completion for each task based on its exit code, and to stop the execution of the entire job if some task fails, or only part of the job which will not be able to complete due to failed task.

Listing 1 Example job definition *(Job.js)*.

```
{ "version": 2,
```

[1] This work was partially supported by RFBR (grant 10-07-00332), and Russian Ministry of Education and Science (Contracts No. 01.647.11.2004, No. 02.740.11.0388 and Nsh-4142.2010.2).

```
  "description": "test job",
  "default_storage_base": "gsiftp://tb01.ngrid.ru/tmp/staging/",
  "tasks": [ { "id": "a",
               "description": "task #1",
               "filename " :    "task_a.js " ,
               "children":    ["b"]
             },
             { "id":    "b",
               "definition " :    {
                    "version":    2 ,
                    "executable":    "/bin/cat",
                    "arguments":    ["-n"],
                    "stdin":    "test.log",
                    "stdout":"gsiftp://tb05.ngrid.ru/home/john/examples/results.txt"
               }
             }]
}
```

2.1 Syntax

Jobs and Tasks definitions are written as JSON documents, there is a JSON Schema definition for the jobs/tasks formats [5, 6]. A job definition consists of a list of tasks, indications of task dependencies and some common options for the whole job, and default resource requirements for all tasks. The definitions for tasks may be done inline in the job definition file, or stored in separate files referenced from the job definition. Listing 1 provides an example of a job definition. This definition corresponds to a simple job with two tasks, *a* and *b,* and structure of *a*→*b*. The task *a* is defined in a separate file *task_ a.js,* provided on listing 2. The definition of the task *b* is contained in the job definition.

These examples demonstrate some of the features of the jobs and tasks definitions for the Pilot service:

- Task dependencies; the task *b* is started only after completion of the task *a*.
- Staging of standard input/output as well as arbitrary other files to and from external storage.
- Using paths relative to `default_storage_base` instead of absolute URLs in files specifications.
- Setting environment variables prior to task execution.

There are many other features supported by the jobs and tasks syntax that are not covered in this introductory description. Complete documentation on the job, tasks and resource requirements syntax, as well as a number of examples, is available at the GridNNN project web site [7] under Pilot service documentation section (http://www.ngrid.ru/sw/pilot/docs/).

Listing 2 Example task definition *(task_a.js).*

```
{ "version": 2,
  "description": "example task defined in a separate file",
  "executable": "worker.sh",
  "input.files": {
   "worker.sh": "gsiftp://tb05.ngrid.ru/home/john/examples/worker.sh"
   },
   "stdout": "test.log",
   "environment": { "CFLAGS": "-O2" }
}
```

2.2 Resource requirements

Resource requirements for tasks may be specified both in job and in task definitions. The requirements keys specified in the task override the values from the job. Pilot supports all of the following:

- Node and operating system parameters: CPU speed and architecture, available memory and disk storage, SMP size, operating system name and release.

- Batch system parameters like time limits, queue lengths.
- Requirements for preinstalled software.

GridNNN information system publishes information about the software preinstalled on the CEs by system administrators and virtual organizations, including software versions. Pilot support specifying software requirements optionally based on expressions with software versions. For example software requirements of «`mvapich, abinit > 6, gcc == 3.5.5`» will be satisfied by any CE which has any version of `mvapich` package, `abinit` with any version greater than 6 and `gcc` of the specified version 3.5.5.

3 Pilot

Pilot is a grid service which can run jobs and tasks as defined in the section 2. The service is developed as a part of the GridNNN project. It is designed as a RESTful grid service [3, 4]. All requests to service are done through authenticated HTTPS connections with peer authentication using user X.509 certificate or proxy certificate [8]. Pilot uses VOMS extensions [9] to determine the virtual organization which the user belongs to. Requests with a certificate without VOMS extensions are allowed only to access existing data owned by the user on the service.

All requests to the service which have payload except the request headers are sent in JSON format. Most of the service replies are also JSON documents, however a client may specify alternative representations in Accept header, and the service will try to satisfy such requests if possible. This allows, for example, to produce a human-friendly information about the job status as an HTML document if the job URI is opened in a web browser.

3.1 Service API

Pilot service API is outlined in the table 1. User jobs are accessible through the */jobs/* collection and each job itself is a collection of job's tasks. To create a job a user may submit a POST request to */jobs/* service URI creating the job resource and all dependent tasks resources. The job and tasks then may be modified using PUT requests. Jobs are not started automatically after their creation, they are started only after the corresponding operation request from the user, which is done through the PUT request to the job URI.

Table 1: Pilot API summary

GET	PUT	POST	DELETE
/jobs/			
List user's jobs (URIs)	N/A	Submit a new job (non-idempotent interface)	N/A
/jobs/<jobid>/			
Job status information and task URIs	- Modify job definition. - Perform an operation with job. - Submit a new job (idempotent interface).	N/A	Cancel and delete the job
/jobs/<jobid>/<taskid>/			
Task status information	Modify task definition.	N/A	N/A
/accounting/period/<st art >-<stop>			
Accounting information for the specified time span	N/A	N/A	N/A

Task resources are created and deleted only implicitly based on the job definition. It is not required to define and submit all tasks to the service while creating a job. Any task definition may be modified if it has not been started yet. This allows to define tasks at virtually any moment of time, even after the job has been started. This allows the user to make corrections to job tasks without interrupting the job workflow execution.

Pilot records all events relevant to any job or task processed by the service. Job's and task's individual history is stored in corresponding resource documents.

Accounting information is represented as a dynamic read-only collection which URI determines the time span requested. Besides the JSON representation, accounting information may also be returned as a CSV document if client requests this format. The amount and scope of accounting events depends on the certificate used for request authentication. Pilot can provide full accounting information access to a limited number of users specified in the configuration file (this is useful for external accounting aggregation services). A user has access to accounting information for his own jobs, and a VO manager (recognized by VOMS role) has access to accounting information for his virtual organization.

3.2 Implementation

Pilot service is written in Python language. It uses M2Crypto library [10] for the SSL/X.509 functions and libvomsc libraries from the gLite project for parsing VOMS attributesfll]. SQLAlchemy object relationship manager is used for the database interface. Currently Pilot is compatible with PostgreSQL and SQLite databases. MySQL support is not available due to the lack of under second precision in date-time fields. PostgreSQL is recommended for production installs, SQLite may be used for low-load testing.

The service is split into two daemon processes (see figure 1). The first process, *pilot-httpd,* handles all HTTPS authentication and authorization, and processes the requests to the Pilot service. It is implemented as a Pylons WSGI application running in a custom service container supporting X.509 proxy certificates with VOMS extensions. The *pilot-httpd* daemon handles most jobs and tasks operations by storing to and retrieving from the database all of the corresponding information. Globus WS-Notification [12] events are accepted by the *pilot-httpd* and passed to the *pilot-spooler* daemon. For job matchmaking requests *pilot-httpd* makes queries to the matchmaker service, which is available as an internal *pilot-spooler* RESTful HTTP service.

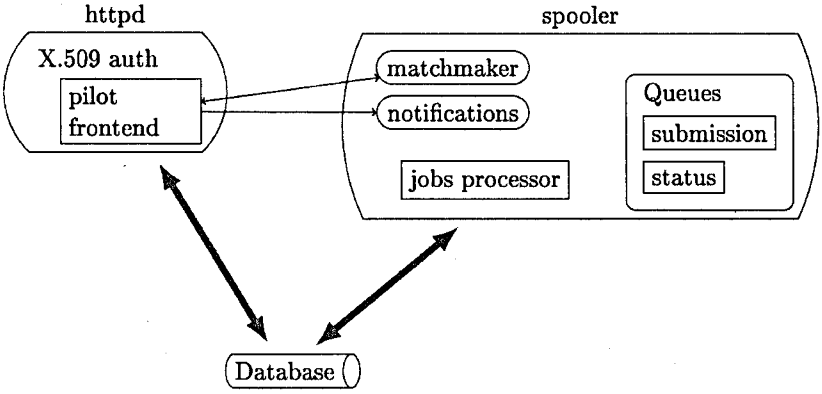


Fig.1: Pilot service structure

The second daemon, *pilot-spooler,* contains the following parts:

- Job processor: analyzes the jobs stored in the database, schedules tasks for the execution.
- Task submission queue: submits the tasks to the resources where they are executed. Current implementation supports only Globus WS-GRAM resources.
- Task status queue: polls the status information for the running tasks which did not receive status notifications for a long time; sends status notifications based on poll results to the notifications service.
- Notifications service: processes task status notifications coming from status queue and from

grid resources via WS-Notification protocol.

- Matchmaker service: returns lists of preferred resources based on requirement sets, virtual organization information. This service is also responsible for querying the aggregated informational system or individual resources for up to date information on the grid resources availability and properties.

For the ease of installation Pilot service and its dependencies are available as an RPM package for CentOS 5 Linux distribution.

3.3 Command Line Interface

There is a command line interface package available for the Pilot service. It provides a set of batch-like commands pilot-*something,* including:

- pilot-job-submit, pilot-job-status, pilot-job-info (contrary to pilot-job-status also includes information for all job tasks), pilot-job-cancel;
- pilot-task-status;
- pilot-job-matchmaker (gives a list of resources compatible with a task for each task of the job and an overall estimation if the job can be executed on current grid resources);
- pilot-query-jobs (lists all jobs owned by the user).

The CLI is available as an RPM for the CentOS 5 Linux distribution and requires a minimal number of dependencies.

4 Conclusions

The workflow jobs described in this work are useful because they allow to automate multistage computations. In cases where some of the stages are independent, workflows may speed up the computations by running corresponding tasks in parallel. In this work a simple syntax for workflow jobs with computational tasks, based on JSON language, was described. The grid service «Pilot» for executing such jobs was developed. It was designed as a RESTful grid service. The outline of the service API was described, and some implementation details are given. The Pilot service is running in production on GridNNN project testbed resources.

References

[1] Ilyin V., Dobretsov V., Kryukov A., Korenkov V., and Ryabov Yu. Design and Development of Grid-infrastructure for National Nanotechnology Network. In these proceedings.

[2] Foster I., Frey J., Graham S., Tuecke S., Czajkowski K., Ferguson D., Leymann F., Nally M., Storey T., Vambenepe W., et al. Modeling stateful resources with web services. Globus Alliance, 2004.

[3] Demichev A., Kryukov A., and Shamardin L. Restful grid: using restful web-services in grid. Software Products and Systems, (4), 2009.

[4] Demichev A., Ilyin V., Kryukov A., and Shamardin L. Design of the application programming interface for a Pilot RESTful grid service. Numerical Methods and Programming, 11:62-65, 2010.

[5] Crockford D. The application/json Media Type for JavaScript Object Notation (JSON). Technical report, IETF Network Working Group, July 2006. RFC4627.

[6] Zyp K. A JSON Media Type for Describing the Structure and Meaning of JSON Documents. Technical report, IETF Network Working Group, March 2010. draft-zyp-json-schema-02.

[7] GridNNN project web site, http://www.ngrid.ru/

[8] Tuecke S., Welch V., Engert D., Pearlman L., and Thompson M. Internet X.509 Public Key Infrastructure (PKI) Proxy Certificate Profile. Technical report, IETF Network Working Group, June 2004. RFC3820.

[9] Groep D. The VOMS Attribute Certificate Format. OGF Draft, artf6312, April 2010. [10] M2Crypto - python crypto and SSL toolkit.http://chandlerproject.org/Projects/MeTooCrypto

[10] Virtual Organization Membership Service, http://glite.web.cern.ch/glite/

[11] Graham S., Hull D. and Murray B. Web Services Base Notification 1.3 (WS – Base Notification). Technical report, OASIS Standard, October 2006.

ARC NOX AND THE ROADMAP TO THE UNIFIED EUROPEAN MIDDLEWARE[1]

Oxana Smirnova

NDGF / Lund University (on behalf of the NorduGrid Collaboration)
oxana.smirnova@hep.lu.se

ARC Nox release came out in the end-2009 as a result of several years of development by NorduGrid and several other projects. It represents a major step towards a truly modular Grid middleware solution, which can be easily extended with new services and functionalities, while keeping standard interfaces. As such, it is ready to become one of the key contributing middlewares to the European Mddleware Initiative, which in turn will provide components to the future Unified Middleware Distribution to be deployed by the European Grid Initiative project.

1 Introduction

During the last decade, Grid computing evolved from an innovative academic exercise to an everyday reality for many researchers. This progress was largely driven by the needs of physicists working at the Large Hadron Collider at CERN. Despite the diverse international nature of the CERN collaborations, the necessity to work together lead to creation of several distinct Grid infrastructures, each powered by a different middleware flavour.

It is possible to argue that a single infrastructure would have been the ultimate goal, however, just like with any infrastructure, geopolitical considerations tend to define boundaries. And just like with any infrastructure, adherence to common basic standards, policies and guidelines is the key to successful cooperation.

It is by now clear that modern Grid infrastructures are defined not that much by network topologies or hardware resources, but by the underlying middleware. There are attempts of creating multi-middleware infrastructures, but so far they rely on additional layers of application-specific solutions, such as e.g. the Worldwide LHC Computing Grid (WLCG) [1]. Either way, Grid infrastructures are defined by software. This paper presents the current state and plans of the Advanced Resource Connector(ARC) [2] - one of such infrastructure-defining middlewares.

It is not a secret that many middlewares were originally created for the purposes of establishing infrastructures addressing quite specific user and provider requirements. ARC was developed originally for the Nordic Grid computing infrastructure, which was initially set up with the goal to provide computing and data storage services to the physics experiments at CERN. Specifics of this Nordic infrastructure is such that while the compute and storage resource providers are highly diverse, independent and are not controlled by the researchers, there is a strong tradition of coming with a collective effort, and the idea of a common Nordic Tier1 center for CERN is one of such initiatives. Similar conditions exist in other countries, therefore ARC usage spreads well beyond Scandinavia, leading to creation of distinct national and international Grid infrastructures. Nordic DataGrid Facility (NDGF) [3] is the largest and most complex of them, serving a large variety of users since 2006, in particular, providing Tier1 services to ALICE and ATLAS experiments at CERN. NDGF is established by four countries, Denmark, Finland, Norway and Sweden, as a body that coordinates service deployment, operations and maintenance, and produces necessary software solutions when necessary. NDGF is a key contributor to development of various softwares: ARC, dCache [4] and SGAS [5], and provides substantial input to other middlewares, most notably, Globus Toolkit [6]. An important difference of NDGF from other similar projects is that it

[1] This work was supported in parts by the Information Society and Media Directorate-general of the European Commission through the KnowARC (Contract No. 032691) and EMI (Contract No. 261611) projects, as well as by the Nordic Council of Ministers through the NDGF and Nordunet3 projects.

does not own or control computational or storage resources and thus can not affect local policies. Portability, flexibility and fault-tolerance of chosen middleware solutions is thus of ultimate importance. Figure 1 shows an overview of the NDGF re-spources as of fall 2009; the computing and storage capacity is being constantly increased.

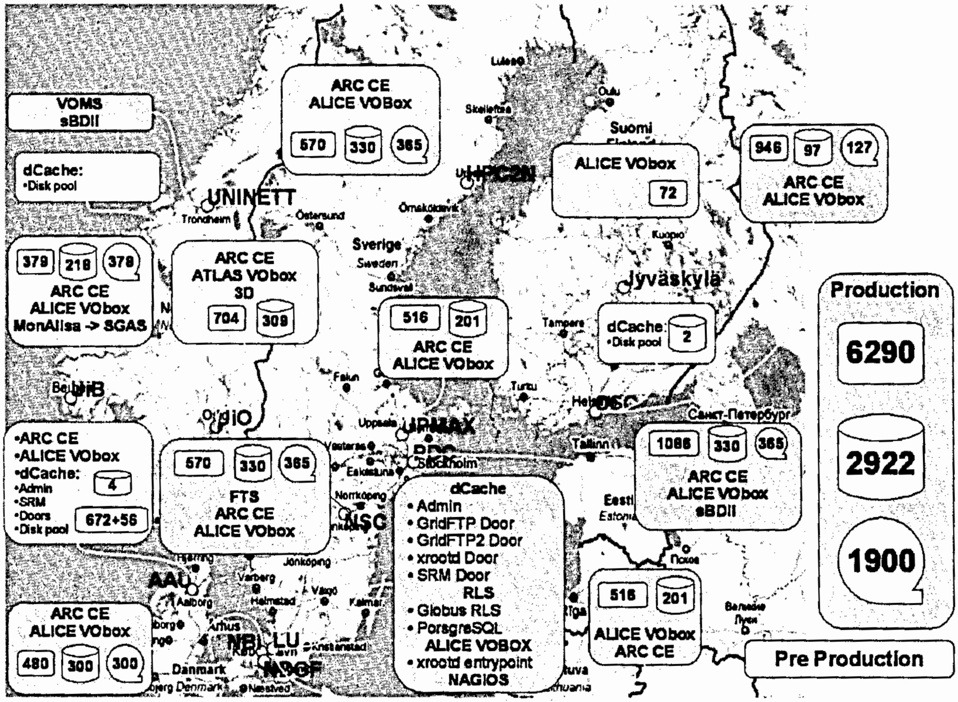


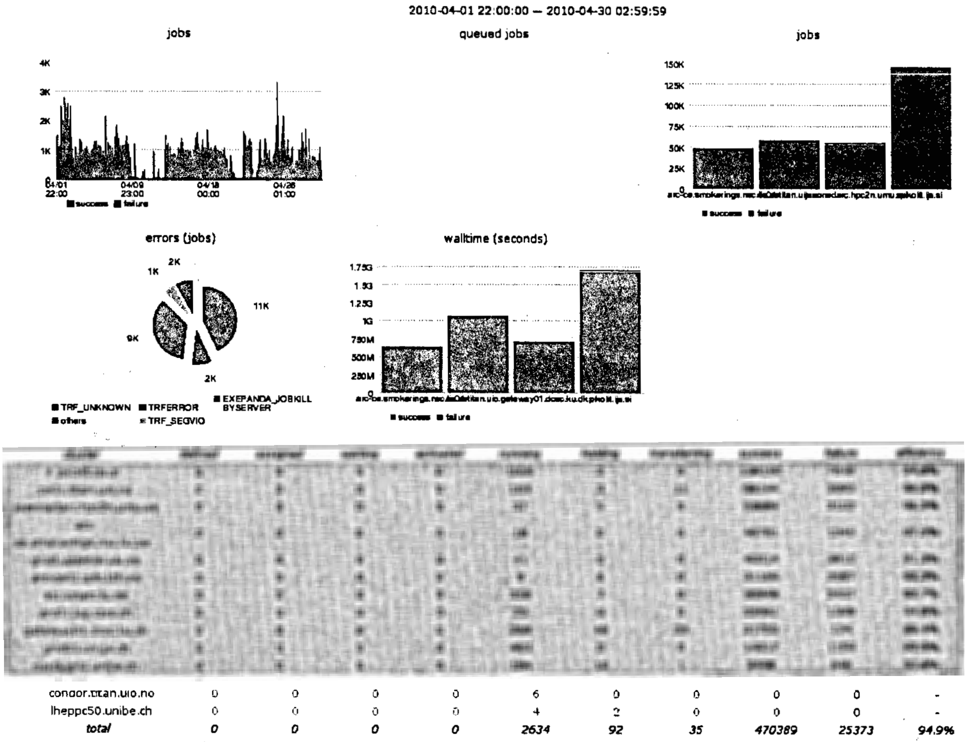
Fig. 1: Overview of the NDGF resources as of fall 2009

In what follows, this paper summarizes current experience with using ARC in the NDGF-based infrastructure (Section 2), proceeds to the overview of the latest ARC release and new developments (Section 3), and gives an outlook to the future of European middlewares in Section 4 before concluding.

2 Usage of ARC

ARC is a fairly generic middleware solution, suitable for a large variety of applications. Having no centralized operational structure for ARC customers, it is impossible to keep detailed track of ARC usage; however, from accounting records of NDGF, national infrastructures and circumstantial evidence obtained from the ARC Grid Monitor [7], it is possible to conclude that well above two dozen research groups in various fields rely on ARC. Historically, the largest user community relying on ARC are the LHC research teams. All LHC-specific application environments are integrated with ARC (albeit to a different extent). Since ARC-based solutions are comparatively lightweight and highly adaptable, smaller communities also enjoy ARC-based services provided by NDGF and smaller national Grid infrastructures.

The largest ARC-based service offered to the scientists is the NDGF's Tier1 centre. This Tier1 is quite unique: traditionally, and until now, all Tier1 centres, except of the Nordic one, are confined within one country, and in most cases -within one computing centre. Individual facilities in Nordic countries could not host such a massive resource, thus a distributed Tier1 was created, spanning 4 countries, and ARC is one of the key software that made it possible - the other being dCache, adapted by NDGF for management of distributed storage.



condor.titan.uio.no	0	0	0	0	6	0	0	0	0	-
lheppc50.unibe.ch	0	0	0	0	4	2	0	0	0	-
total	*0*	*0*	*0*	*0*	*2634*	*92*	*35*	*470389*	*25373*	*94.9%*

Fig. 2: Snapshot of the ATLAS production system dashboard showing status of the NDGF-based "cloud"

Every Tier1 center in the WLCG infrastructure has affiliated Tier2 centers: while Tier1 centers are focused on custodial primary data storage and processing, Tier2 centers are used for simulations and derived data storage and processing. The NDGF Tier1 has such affiliated Tier2 facilities, and these include countries outside Scandinavia that chose to deploy ARC. Operation of the resulting ARC-based infrastructure is a common effort by NDGF and national Grid initiatives, with a different degree of involvement. Major share of computing and storage resources is provided by national scientific computing centers, and the rest is provided by individual research groups. Despite the very different scales (from few CPUs cluster to a top-500 site), the chosen approach provides a reliable, smoothly integrated multi-purpose Grid infrastructure.

The largest customer of NDGF is ATLAS Collaboration at CERN. Figure 2 shows a recent snapshot of the ATLAS production system dashboard [8] with the NDGF-based *cloud* status. ATLAS production infrastructure within WLCG is organised as clouds (not to be confused with the Cloud technology) centered around Tier1s [9], and the NDGF cloud brings together ARC-powered resources contributing to ATLAS computing. The figure shows computing jobs executed on various Linux clusters, and it can be seen that the success rate and efficiency of this cloud in terms of computing services is very high. It also can be seen that the load is quite substantial, with about half a million jobs processed during one month.

Figure 3 presents an overview of various ATLAS computing activities in the ARC-based NDGF cloud over past year. Before LHC data taking resumed in fall 2009, main focus was on simulation tasks, and it can be seen that the cloud can sustain the load of about 10 thousand simultaneous tasks, with peaks up to 30 thousand. With the start of data taking, analysis jobs became more prominent; though the number of these jobs is less than the simulation ones, analysis jobs in general are more challenging and have unpredictable and hence often suboptimal workflows, often leading to lower efficiencies.

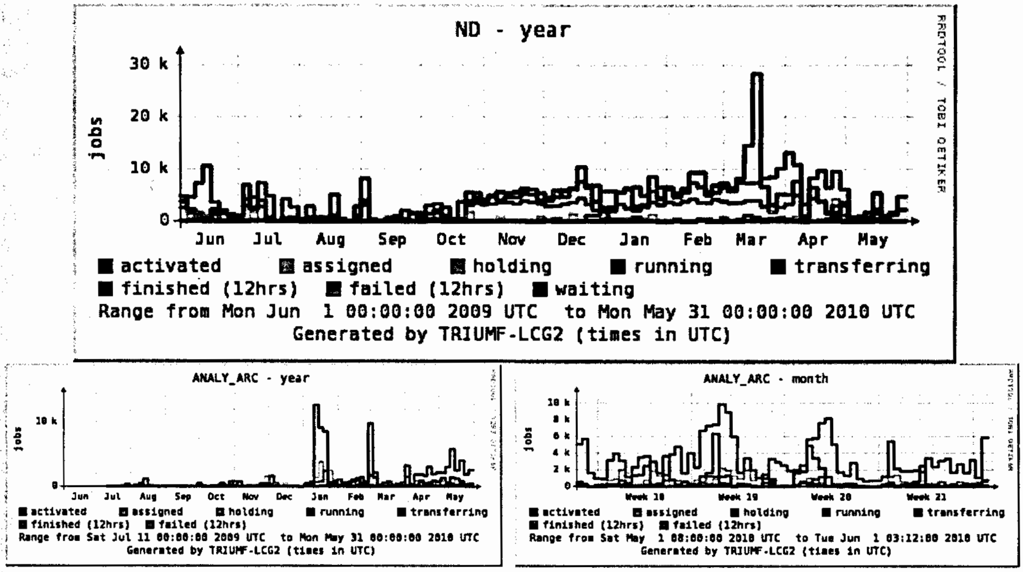


Fig. 3: ATLAS simulation jobs in the NDGF cloud over one year (upper plot) and ATLAS data analysis jobs over one year and one month (lower plots)

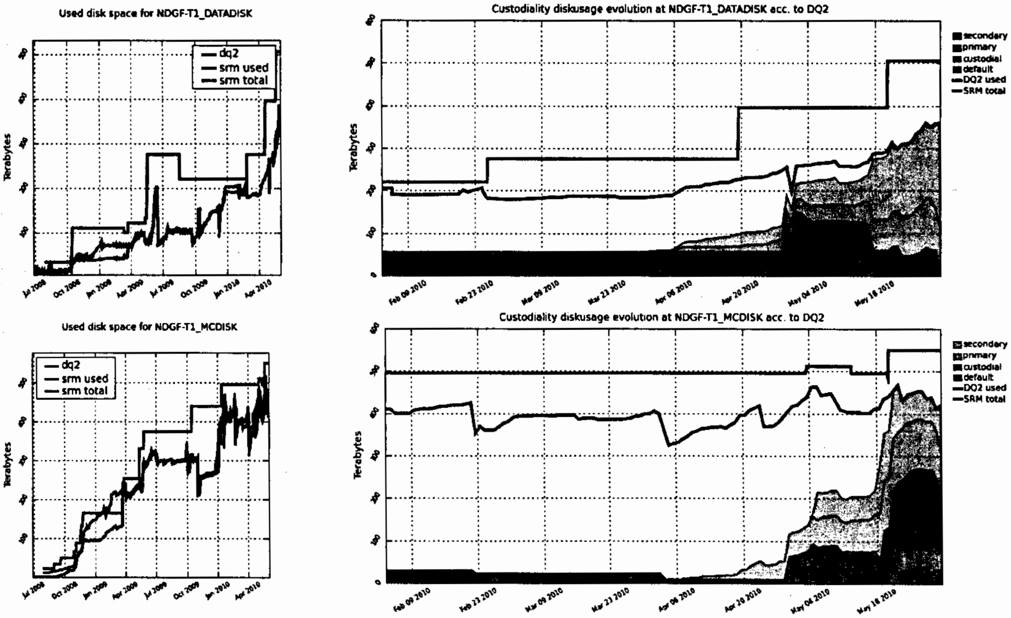


Fig. 4: Evolution of ATLAS storage capacity of NDGF Tier1: upper plots show amount of stored physics data, while the lower plots correspond to simulated data

Data management activities are as important for ATLAS as computational jobs. Figure 4 shows evolution of ATLAS storage capacity of the NDGF Tier1 since 2008. It can be seen that NDGF storage resources allocated to ATLAS are heavily used, with sharp increase of demand coinsiding with the start of LHC data taking.

It is not the purpose of this paper to compare performance of the ARC-based resources in general and NDGF in particular against other infrastructures, since there are no commonly accepted unambiguous criteria. By studying publicly available performance monitors, such as the used here ATLAS dashboard, one can conclude that NDGF performs at least as well as other Tier1 sites in all aspects, and in terms of efficiency is clearly among the top sites.

It should be also noted that in addition to the necessary middleware services, the NDGF-based infrastructure implements all the operational procedures and tools required by the relevant large Grid projects, such as already mentioned WLCG and the recently launched European Grid Initiative (EGI) [10], being thus an integral part of larger international infrastructures despite being based on non-mainstream middleware.

3 Overview of the latest ARC release

To maximize efficiency of resource usage in international computing infrastructures built by independent resource owners with different policies, distributed computing solutions must be highly portable, flexible in configuration, and standards-compliant. At the time of writing, NDGF uses ARC v0.8.2 release as the solution that satisfies these requirements best. Although ARC 0.8.2 contains both standard and pre-standard implementations of services, only pre-standard ones are currently used in production. There are several reasons for that: primarily, the pre-standard components have a solid success record and are well understood; meanwhile, standards still undergo changes, and other relevant middlewares do not implement them widely anyway, thus there is no pressing need to upgrade. This situation is likely to change in near future, when major middleware providers will agree on the common set of standards, see discussion in Section 4.

In general, ARC is designed as a de-centralized solution, such that the system has no single point of failure. Distributing such services as information indexing and metascheduling, while keeping service and task status information locally at the resource are among key design choices that ensure high fault tolerance of ARC-based infrastructures.

The version of ARC used in production is focused on providing remote computing functionality and general purpose client tools. The software can be compiled on many POSIX-like operating systems, and binary packages are available for major Linux distributions. On the service side, ARC is interfaced through custom plugins with many cluster batch systems like Torque, LSF, SGE, Condor, etc..

ARC computing service also features advanced data staging capabilities. The data caching algorithm implemented in computing service significantly reduces overhead of accessing significant amount of input data in a data-intensive computing usage pattern. With data staging embedded into the front-end it becomes possible to reduce ineffective input/output-bound usage of computing nodes. Support for various data transfer share algorithms allows effectively to avoid congestion of data staging resources and provides fair share functionality among different consumers. Possibility to have multiple front-ends allows to spread the network load. This efficient data caching and staging mechanism of ARC allows to decouple data processing and data archiving facilities, making creation of a distributed storage facility, like the one implemented by NDGF, a feasible solution.

The ARC middleware as such includes only very basic file storage service, and in order to use third party storage solutions, ARC client part features easily expandable wide range of supported data indexing, management and access protocols. In this respect, ARC client is to a large extent integrated with other middlewares regarding operations with data files.

3.1 New developments: ARC Nox

With the above principles and approaches remaining at the core of ARC design, significant effort was made to re-implement same functionalities in a standard-compliant, extensible and scalable manner. Such re-implemented components are distributed with ARC 0.8.2 alongside with the pre-standard ones.

The new components add support for computing service with standardized Web-based interface, enhanced data staging capabilities, client tools with support for multiple data and computing services and multi-language development API. Although invisible for end-users, code base of ARC undergoes quite dramatic changes, including a complete re-structuring. Perhaps the most important change is introduction of the single service hosting layer, the Hosting Environment Daemon (HED) [11]. HED is a Web Service (WS) container that provides communications between the hosted services, and handles all external interfaces. This results in much increased modularity and extensibility of services. For example, communication chain can now be composed of a number of pluggable components; this allows, among others, to formalize GSI as a separate plugin, which in turn removes dependency of core components on Globus. The restructured code significantly improves portability, such that the new ARC tools and services are being made available not just on Linux (included in some standard distributions), but also on Microsoft Windows, Mac OS and Solaris.

The packages containing new ARC services and tools are currently labeled **Nox,** in order to distinguish from the pre-standard ones. The ARC Nox packages were first made available as a separate technology preview release in November 2009 [12], and later became included in the 0.8.2 releases line. Figure 5 shows the generic service decomposition of ARC Nox, and their relations to eventual external services.

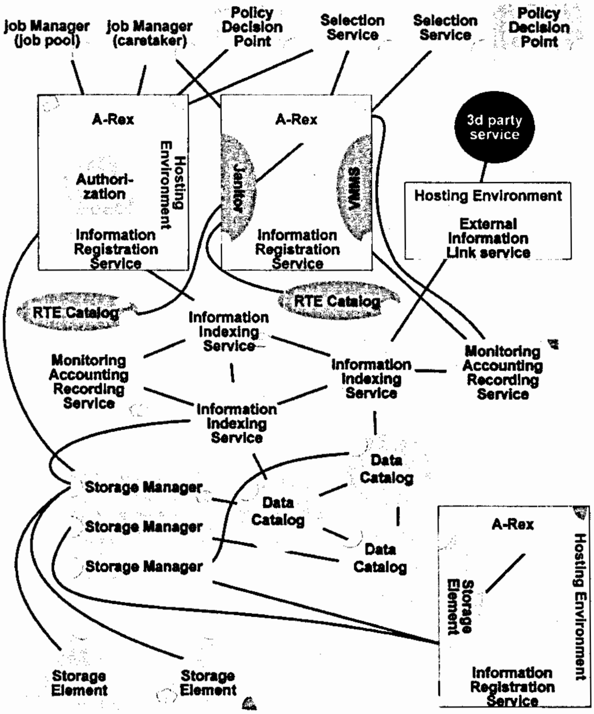


Fig. 5: Service decomposition of ARC Nox

The new ARC components can be classified in the same conventional areas: compute, data, information, security and clients. Since all new services, and even the client tools, rely on HED or its libraries, one should always include HED as a common infrastructure component for all the areas. The following sections present brief overview of each such area in ARC context. Pre-standard components, not relying on HED, will be referred to as *pre-WS* ones, while the new — as WS-components.

3.1.1 Compute area

In the compute area, two services, offering largely the same functionalities, are available. The first one is the pre-WS computing element exposing a proprietary interface via a GridFTP [13] plugin, and based on the *Grid Manager* [14] process that handles LRMS communications, data staging, cache management, application environments and such. Information about the computing resource and its activities (jobs) is accessible via an LDAP interface, and is arranged in the proprietary ARC schema. Resource usage records are collected and published to the SGAS accounting system by a dedicated *URlogger* process.

The new WS-based computing element was introduced in Nox. The Grid Manager is replaced with *A-REX* [15], which provides much the same functionality, but is more versatile. Through HED, it offers standard compliant WS interface, which is currently based on BES [16], with few extensions. It also is compliant with other key standards, such as WS-RF [17], JSDL [18] and Glue2.0 [19]. The modularity of the system allows addition of several new modules, such as the *Janitor* for application environments management, and *JURA* for job resource usage record publishing to accounting systems.

It is expected that the new WS-based computing element will replace the pre-WS one completely. Meanwhile, complex deployment scenarios are possible: for example, it is possible to deploy both services simultaneously, or to deploy A-REX with a GridFTP plugin interface.

3.1.2 Information area

In the information area, pre-WS solutions included in the ARC 0.8.2 release underwent substantial changes. Although there is no change in LDAP interface and schema, the underlying technology is completely different: Globus LDAP back-end is replaced with BDII [20], and Globus GIIS is re-implemented in pure LDAP. These changes further reduce dependency on Globus, and are expected to improve stability of information services.

The most notable change in Nox components information-wise is that they are compliant with the Glue2.0 standard. Moreover, Nox offers a preview of a new WS-based solution for the information system: the local information system *(LIDI)* offers a WS-RF interface through HED; every service thus describes itself through the LIDI interface. The information is indexed in *ISIS* — the information indexing service, implemented over a peer-to-peer information system backbone; ISIS offers a WS-interface as well.

3.1.3 Data area

As it was already mentioned, pre-WS ARC has no own storage solution, but is interoperable with third-party storages. Nox fills this gap by offering a preview of the *Chelonia,* a distributed storage solution. Chelonia is designed as a self-healing flexible storage cloud-like system. Self-healing is provided by redundant services and automatic restoration of number of replicas if storage components fall out. Chelonia implements a global namespace, and supports collections and sub-collection to any depth. The system is also intended to be user friendly, coming with a complete command-line interface and FUSE mounting possibility (though with a limited functionality). Chelonia is implemented as a set of redundant services deployed on the top of basic file storage systems. In case of no such storage, one can use the light-weight *Hopi* HTTP server of Nox. *Shepherd* services manage storages, and provide a simple interface for storing files on storage nodes. *Bartenders* provide a high-level interface for the users and for other services, while *Librarians* handle metadata and hierarchy of collections and files, location of replicas and health data of the Shepherd services. Finally, *A-Hash* services implement a replicated database to store metadata. The overall system is expected to handle small to medium size storage systems.

ARC is different from many other middlewares in one important aspect: it has powerful data handling mechanisms built into computing elements, optimizing them for data-intensive high-throughput computing. While being a part of the computing element, cache management and data staging tools and libraries can be classified as belonging to the data area as well. These functionalities are practically identical in pre-WS and WS-based ARC, and in ARC v0.8.2 come with some enhancements. Multiple Grid Managers (or A-REXes) can now be run under one GridFTP server to improve throughput. C ache-wise, Grid Managers co-deployed on one site can now use each other caches; user authentication is now cached in case the same data source is perused; caches can be now cleanly drained before taken offline, and the cleaning tool is in general optimized; finally, it is now possible to specify a lifetime for cached files. Concerning data staging, ARC now implements a fair-share system for transfers, which splits transfer slots evenly between users or groups of them. Intelligent retry strategy for failed data transfers with exponential back-off for temporary errors is introduced as well. A new possibility to specify output files through a dynamically created list is added. Improvements are made in handling SRM [21] port/protocol ambiguity, by attempting various combinations and caching successful ones. ARC v0.8.2 offers two data management libraries - for the pre-WS solutions, and for the WS-based ones.

While these changes improve significantly ARC performance, especially in such a challenging application as ATLAS data analysis, there is still a strong need for a new data staging framework. This is being in development at the time of writing, and will be only available for the WS-based ARC components.

3.1.4 Security area

Like many Grid solutions, ARC primarily relies on Globus GSI for authentication and authorisation handling. Introduction of HED, however, allows to diversify security mechanisms without re-writing services themselves. Through HED, new ARC services are capable of handling almost every protocol, such as TLS, SAML, VOMS, MyProxy, and certainly the GSI. Integration with external security and policy services is therefore quite straigtforward, as long as the interface is known. HED comes with a set of powerful security libraries that can be used to develop new modules.

On the client side, ARC now comes with the *arcproxy* command line utility, which is capable of creating all kinds of proxy certificates. It does not rely on any third-party library, and thus is available for all kinds of end-user systems, including Microsoft Windows.

Noc preview came also with a large variety of proof-of-concept services and clients handling different security aspects; at the moment, however, there are no plans to develop them further, and instead rely on existing third-party solutions.

3.1.5 Client area

In the client area, ARC v0.8.2 provides both the pre-WS CLI *(ngsub, ngls* etc) and the new WS-compliant CLI based on new ARC client libraries. Since new client tools offer significantly extended functionalities, the names can not be reused, therefore new commands are introduced: *arcsub, arcls* etc., with different command line options. The new CLI is available not just on Linuxes, but also on Microsoft Windows and Mac OS systems.

ARC keeps the tradition of providing stand-alone client tar-balls that can be quickly deployed anywhere, providing this the fastest way to access Grid resources. It is important to stress that ARC clients perform resource discovery, matchmaking and brokering prior to submitting jobs directly to computing elements. Corresponding libraries are available, allowing to develop custom-made clients. One of such newly developed client tools is the graphical user interface *Ar-cJobTool.* It is written in Python using the ARC client library; the code can be easily converted to a Web portal. ArcJobTool is now distributed as a part of ARC v0.8.2 release.

4 Future of European middleware development and ARC

As it was demonstrated in Section 2, pre-WS ARC solutions are already successfully deployed in international Grid infrastructures along with other middlewares, most notably, gLite [22]. However, the interoperability is predominantly on the application level, with the only common standards being GSI and GridFTP. In order to be accounted for and monitored by the EGI operators, NDGF successfully developed and deployed all the add-ons necessary to hook ARC-based sites into the EGI infrastructure. Most of these add-ons are custom solutions tailored for NDGF, and as such can not be distributed with

ARC. The most notable exception is the Glue1.3 translator, which translates ARC information representation into Glue1.3 one used currently by gLite. Unfortunately, neither schema is a standard one.

In order to be truly interoperable, and with many middlewares, common standards have to be accepted and implemented. The European community sees open standards as the starting point for providing a common European Grid infrastructure, where different providers use different middlewares - be it ARC, gLite, UNICORE [23] or Globus. Unfortunately, currently available standard specifications are of a very basic nature and are not suitable for production needs. The necessity to amend such standards lead to middlewares developing incompatible extensions, devaluating the notion of being " standard-conformant". Open Grid Forum (OGF) [24] mandated a dedicated Production Grid Interfaces Working Group (PGI-WG) in end-2008, bringing together key middleware providers from all over the World, and aiming at producing a set of common specifications. Being so diverse, however, the Group couldn't so far deliver a tangible output.

Aiming to address the middleware convergence issue on the European level, the European Union supported creation of the European Midleware Initiative (EMI) [25] in May 2010. In many aspects it is complementary to EGI, and is expected to be a major provider for the Unified Middleware Distribution (UMD) of EGI. EMI brings together ARC, dCache, gLite and UNICORE developers, with the high-level objective of achieving convergence between the middlewares. The first obvious step is to agree on common interfaces for key services; common solutions are expected to be developed where possible (e.g., in virtualisation), while redundant components are scheduled for removal.

Given the status of the middlewares involved in EMI, changes will be introduced in several steps. The "day Zero" release will include all the components that are used today in production infrastructures, but built and packaged using the common infrastructure, and therefore having no conflicting dependencies.

From ARC perspective, the necessity to implement EMI objectives coincides with that of replacing pre-WS components with the new ones. Clearly, interfaces will have to be adjusted to those recommended by EMI, but since no production infrastructure deploys the new components yet, there will be no disruption caused by this. In the EMI framework, ARC Grid Manager will be replaced with A-REX, while pre-WS CLI and client library will be replaced with new ones. EMI information system is likely to be developed a-new, but in any case it will rely on the Glue2.0 standard. Storage solutions, being largely independent from computing ones, and already benefiting from a common SRM standard, will not undergo major changes in EMI, except of the planned transition from GSI to industry-standard protocols.

Several components found in Nox and ARC 0.8.2 are currently not endorsed by EMI - most notably, Chelonia and ISIS. If there will be a significant user demand, ARC community will support and distribute such components outside EMI.

5 Conclusion

The latest ARC release is a result of several years of development by NorduGrid and several other projects. It represents a major step towards a truly modular Grid middleware solution, which can be easily extended with new services and functionalities, while keeping standard interfaces. As such, it is ready to become one of the key contributing middlewares to the future Unified Middleware Distribution to be deployed over EGI resources. The solutions described in this paper are constantly being improved and extended, following the general technology progress and addressing requirements from end-users, system administrators, collaborating projects and others. The new European EMI project is one of the frameworks in which future ARC development will proceed. EMI releases will provide an opportunity to phase out pre-WS components and otherwise to improve services offered by ARC.

References

[1] Worldwide LHC Computing Grid. Web site: http://www.cern.ch/lcg
[2] Ellert M. et al. Advanced Resource Connector middleware for lightweight

computational Grids. *Future Gener. Comput. Syst.*, 23(1):219-240, 2007.

[3] Nordic DataGrid Facility. Web site: http:// www.ndgf.org

[4] de Riese M., Fuhrmann P., Mkrtchyan T., Ernst M., Kulyavtsev A., Podstavkov V., Radicke M., Sharma N., Litvintsev D., Perelmutov T., and Hesselroth T. *dCache Book.*

[5] Swedish Grid Accounting System (SGAS). Web site: http://www.sgas.se

[6] Foster I. and Kesselman C. Globus: A Metacomputing Infrastructure Toolkit. *International Journal of Supercomputer Applications,* 11(2):115-128, 1997. Available at: http://www.globus.org

[7] Smirnova O. *The Grid Monitor.* The NorduGrid Collaboration. NORDUGRID - MANUAL-5.

[8] ATLAS dashboard. Web site: http://dashboard.cern.ch/atlas/

[9] Jones R. and Barberis D. *The ATLAS computing model.* J. Phys.: Conf. Ser. 119 072020 2008.

[10] European Grid Initiative. Web site: http://www.egi.eu

[11] Cameron D. et al. *The Hosting Environment of the Advanced Resource Connector middleware.* NORDUGRID-TECH-19.

[12]ARC Nox release announcement. Web site: http://www.nordugrid.org/arc/releases/nox-100

[13] Allcock W. et al. Data management and transfer in high-performance computational grid environments. *Parallel Comput.,* 28(5):749-771, 2002.

[14] Konstantinov A. *The NorduGrid Grid Manager And GridFTP Server: Description And Administrator's Manual.* The NorduGrid Collaboration. NORDUGRID-TECH-2.

[15] Konstantinov A. *The ARC Computational Job Management Module - A-REX.* NORDUGRID-TECH-14.

[16] Foster I. et al. OGSAT^{M} Basic Execution Service Version 1.0. GFD-R-P.108, August 2007.

[17] OASIS. OASIS Web Services Resource specification., April 2006.

[18] Anjomshoaa A. et al. Job Submission Description Language (JSDL) Specification, Version 1.0 (first errata update). GFD-R.136, July 2008.

[19] Andreozzi S. et al. GLUE Specification v2.0. GFD-R-P.147, March 2009.

[20] Flechl M. and Field L. *Grid interoperability: joining Grid information systems.* J. Phys.: Conf. Ser. 119 062030 2008.

[21] Sim A., Shoshani A., et al. The Storage Resource Manager Interface (SRM) Specification v2.2. GFD-R-P.129, May 2008.

[22] gLite, Lightweight Middleware for Grid Computing. Web site: http://cern.ch/glite.

[23] UNICORE (Uniform Interface to Computing Resources). Web site: http://www.unicore.eu

[24] Openl Grid Forum. Web site: http://www.ogf.org

[25] European Middleware Initiative (EMI). Web site: http://www.eu-emi.eu

ON DEVELOPMENT OF RESTFUL WEB SERVICE FOR A COMPUTER ALGEBRA SYSTEM IN MATHCLOUD ENVIRONMENT[1]

S. A. Smirnov

Moscow Institute of Physics and Technology, 141700, Dolgoprudny, Russia
sasmir@gmail.com

1. Introduction

Maxima [1] is an open source computer algebra system (CAS) for the manipulation of symbolic and numerical expressions, including differentiation, integration, Taylor series, Laplace transforms, ordinary differential equations, systems of linear equations, polynomials, and sets, lists, vectors, matrices, and tensors. Maxima yields high precision numeric results by using exact fractions, arbitrary precision integers, and variable precision floating point numbers.

The Maxima source code can be compiled on many systems, including Windows, Linux, and MacOS X.

Maxima is a descendant of Macsyma, the legendary computer algebra system developed in the late 1960s at the Massachusetts Institute of Technology. It is the only system based on that effort still publicly available and with an active user community, thanks to its open source nature.

MathCloud [2] is a distributed environment focused on the support of mathematical research and based on the technologies of Web and grid. The purpose of the environment is to provide unified access to network services of solution of various classes of mathematical problems. The REST (Representational State Transfer) architectural style [3] was used to unify the mechanism of remote access to services in the MathCloud at the level of protocols and data formats. The REST architectural style lies at the heart of the modern World Wide Web. In particular, HTTP provides a number of main REST constraints. The key concepts of REST are the resource, the identifier and the representation of a resource. Web services that satisfy the REST constraints are referred to as RESTful Web services [4].

In the unified MathCloud REST interface [5] a service means a single operation represented as a resource. To execute the operation the client has to use the POST method of HTTP. In response, the service will create a job resource which allows to query the status and the result of the client's request. Thus in the current MathCloud interface implementation all requests are processed asynchronously. GET method allows to get the description of the service. The interface also allows to transfer and store parameters and results of requests in the form of files.

The paper is concerned with the problem of transformation of the Maxima CAS into a RESTful Web service. Two ways of using Maxima from another program (sockets and unnamed pipes) are discussed. Different approaches to providing an HTTP access to the service are considered: an embedded web server and a special container.

2. MaximaAPI library

Let us consider the Maxima features which allow other applications to interact with it interactively. The complex part is that the Maxima project didn't document the protocol or an API for interactive interaction with the Maxima. Thus all the information was gathered from the project's mailing list archives and from analysis of the wxMaxima's (a GUI for Maxima) source code.

[1] Partially supported by the RFBR grant 08-07-00430-a and "SKIF-Grid" project.

Two ways to interact with the Maxima were discovered: TCP-based sockets and the use of unnamed pipes. Let's start with the sockets.

This technique is the most popular and is used in the wxMaxima and many other applications that interact with the Maxima. In this design there are two processes that must run on the same host: the Maxima process and a client process (e.g. wxMaxima). The client process creates a listening TCP socket on a well-known port such as 12345. Then it starts the Maxima process with -s option, e.g. maxima -s 12345. In this case Maxima makes a TCP connection to localhost:12345. Then Maxima redirects the standard I/O streams to this TCP connection. As a result the client application gets a socket connected to the Maxima instance and behaving like a Maxima console. We would like to emphasize that in this scheme the client application plays the role of a server while the Maxima behaves like a client.

In the second method the client application process starts a child Maxima process. Client process opens the Maxima's standard I/O streams as unnamed pipes. Thus the clients receives access to the Maxima console.

Let us compare these two methods of interaction with the Maxima. Since both methods become more or less equivalent from the moment of established connection we will focus on the specifics of the process of establishing a connection. Firstly let's list the specifics of the approach based on the sockets:

- Similar BSD sockets API can be found in most modern operating systems.
- Classes providing a portable sockets API are present only in large libraries (e.g., ACE).
- Separate means are needed to start the Maxima.

Specifics of using the unnamed pipes:

- Different operating systems have different APIs for generating child processes and for redirecting standard input/output.
- There is a small, portable, header-only library that implements the above mentioned functionality (Boost.Process [7]).

In this work we chose the second approach as it provides access to a Maxima console with less efforts although adding a dependency on a header-only third-party library.

After connecting to the Maxima console we can move to the problem of proper interaction. Console communication with the Maxima is quite simple: in response to a prompt from the Maxima user enters an expression. Then Maxima returns a result. Further, the cycle repeats. Sometimes if user enters certain expressions the Maxima behaves incorrectly. It doesn't print the result of an operation or doesn't issue a prompt. Judging from the behavior of existing applications that interact with the Maxima this is a bug in the Maxima. Thus if the user adheres to some undocumented format, the communication protocol with the Maxima can be characterized as a request-response.

Automatic processing of Maxima console output is quite complicated. It is hard to deduce the general form of the Maxima output which can be a result, a message or a prompt. According to the Maxima documentation the system has a number of variables that control the highlighting of significant fragments in the console output. These variables allow to greatly simplify the automatic processing of Maxima output. Let us outline the most useful variables and their meaning:

- **prompt-prefix** – a string printed before each prompt;
- **prompt-suffix** – a string printed after each prompt;
- **alt-display2d** – a function that replaces the standard one when Maxima prints expressions in 2-dimensional mode.

These variables can be set in a Common Lisp file specified by the -p command line option of Maxima. Maxima executes this file during its startup.

We've implemented the above-described low-level interaction with Maxima in a separate library called MaximaAPI [6]. The library uses the unnamed pipes approach to Maxima console interaction. The approach was based on the Boost.Process library. The -p command line option was used to supply Maxima with an appropriate startup script. The script sets up the prompt and the result highlighting. It also enables one dimensional output mode so that Maxima output can be used in other expressions without modification. The output parsing was based on the regular expression matching provided by the Boost.Regex [8] library.

The MaximaAPI library contains a single class named MaximaInstance. The class resides in the MaximaAPI namespace. Creation of a Maxima process and linking to its I/O streams are implemented in the constructor. Interaction with the Maxima is done via a single method that has one string argument representing an expression to be passed to the Maxima. The method returns another string containing the result. There is also a method to asynchronously interrupt the current calculation.

3. Web service based on the Mongoose web server

The first version of the service was based on the Mongoose web server which was chosen due to its simplicity, embeddability and relatively wide functionality. This web server allows you to setup handlers for URI templates. When one requests a resource, the server calls an appropriate handler supplying a structure with a partially parsed client's request. The structure contains the requested URI as a string, the HTTP method, an array of request headers, request body length, and more. Also, if a query was done through an HTML form, the server provides a helper function to retrieve the name-value pairs for the form fields.

Despite the handy functionality described above, the server is missing some necessary tools for correct handling of the HTTP. For example, there is no code for working with the content of Accept headers which is necessary to select the most suitable resource representation for a client. Also, you have to generate raw HTTP response yourself in the URI handler. As a result we had to implement the missing functionality ourselves.

Also some functionality not specific to a web server had to be developed from scratch. That was the job management in accordance with the MathCloud REST API. However the ability to use files in service parameters, that would be very useful in the Maxima service, have not been implemented.

Finally, a service prototype which worked successfully in the MathCloud environment and in particular with the WfMS [9] was implemented. However it was not suitable for calculations requiring large amount of data as it didn't support passing parameters in the form of files. The service had the following interface description:

```
      {
        "name" : "maxima",
        "description" : "CAS Maxima REST-service accepting one Maxima expresion at
a time",
        "inputs" : {
          "input expression" : {"type" : "string", "title" : "Input expression"}
        },
        "outputs" : {
          "result" : {"type" : "string", "title" : "Result returned by Maxima"}
        }
      }
```

4. Web service based on the EveREST container

After creating the first version of the service it became clear that its further support and development will result in the creation of another container for the MathCloud. It did not make much sense as existing containers already had a wide range of additional functionality, such as automatic generation of HTML-based interface, automatic termination of hanging jobs, support of file parameters, etc. Hence it was decided to make a version of the service for the EveREST container (Java). It was also decided to implement the service in Jython [10] instead of Java to simplify its debugging and support.

Since the MaximaAPI library was written in C++, a JNI wrapper had to be developed to use the main library functionality from the service running in a JVM. Hence individual Maxima instances can be created using the ru.isa.dcs.ssmir.maxima.MaximaInstance Java class with the following interface:

```
      public class MaximaInstance {
        public native void initialize(String maximaPath, String utilsDir) throws
MaximaException;
        public native void destroy() throws MaximaException;
        public native String executeCommand(String command) throws MaximaException;
        public native void interruptMaxima() throws MaximaException;
        private long _ptr;
      }
```

The class provides access to the most necessary methods of the MaximaAPI::MaximaInstance class. The _ptr field is used to store a pointer to an object of the MaximaAPI::MaximaInstance class. Maxima process startup is carried out in the initialize method and not in the constructor.

To make a service for EveREST one has to create a class that implements the everest.java.JavaServiceI interface. The interface has the following three methods:

```
      void init(Map<String, String> config);
      Map<String, Object> run(Map<String, Object> inParams, String jobDir,
JobStatus status);
      void destroy();
```

The first and the third methods are called by the container during the startup and termination, respectively. Options specified in the container's configuration file are passed to the service through the config argument of the init method.

The run method is invoked on each request to the service. The method contains the main service logic. If an exception is being raised in the method then the client receives an HTTP error. If a client decides to terminate a running job (with the DELETE method of HTTP) then the thread running this job (i.e. the run method) will get an interrupt status which can be checked from Java. To let the container complete the job termination, the run method have to raise the InterruptedException exception. The “inParams” argument contains the values of parameters passed to the service during a call to it. At the same time file parameters will have the File class. The “jobDir” argument contains the path to a directory in which the service should save the output parameters of type “file”. The “status” argument allows you to set additional data that is returned to the client querying the job status.

To make it possible to implement services in Jython a simple adapter was developed. The adapter consists of one Java class (everest.java.JavaServiceI) implementing the everest.java.JavaServiceI interface. Adapter loads a Jython service implementation and then delegated all the service interface calls to it. At the same time the Jython service has to implement the same everest.java.JavaServiceI interface. When a client interrupts a job (DELETE method of HTTP) time.sleep() and similar operations throw the KeyboardInterrupt exception. As a result the service must abort the job and raise the InterruptedException exception.

The Maxima service has been implemented in Jython. That allowed to speed up its development, reduce the amount of code and simplify its debugging in comparison to a possible implementation in Java. Internally the service has a pool of spare Maxima processes and a thread pool which are used to process several jobs concurrently. The container calls the run method concurrently from multiple threads. It yields effective use of Maxima in multiprocessor systems although every Maxima process is single-threaded.

Client may use the service as follows:

1. Let the service (its resource) be located on SERVICE_URI.
2. The client does a GET request on the SERVICE_URI and receives a response with the description of the service as shown below.
3. The client does a POST request on the SERVICE_URI, passing a JSON object with the input parameters of the service (e.g. "command" : "2+3") in the body of the request.
4. In response the server creates a job resource and returns its URI in the Location header. Let this URI be REQUEST_URI.

5. The client polls the REQUEST_URI with the GET method and receives the job status containing completeness state of the job. The client can also cancel and remove the job with the DELETE method.

   Description of the service in the EveREST configuration file is the following:

```
{
  "name" : "maxima",
  "description" : "CAS Maxima REST-service accepting one Maxima expresion at a time",
  "inputs" : {
    "command" : {"type" : "string", "title" : "Input expression"},
    "script.mac" : {"type" : "file", "optional" : true,
      "title" : "Script to load before executing the command"},
    "data.lsp" : {"type" : "file", "optional" : true,
      "title" : "Lisp script to load after the script file but before the command"}
  },
  "outputs" : {
    "result" : {"type" : "string", "title" : "Result returned by the command"},
    "output.lsp" : {"type" : "file",
      "title" : "Output file (output.lsp) which was used by the command"}
  },
  "implementation" : {
    "type" : "java",
    "class" : "ru.isa.dcs.ssmir.JythonService",
    "config" : {
      "pythonPackage" : "MaximaService",
      "pythonClass" : "MaximaService",
      "maximaPath" : "/opt/local/bin/maxima"
    }
  }
}
```

All the above service description, except for the implementation field, is returned to the client in reply to the GET request. Moreover the same description is used by the EveREST container to build HTML forms which allow the use of the service through a plain web browser.

In accordance with the description of the service, request to the service contains one mandatory parameter (command) and two optional (script.mac and data.lsp):

— command – a string with a single Maxima expression;

— script.mac – a file in the Maxima scripts file format;

— data.lsp – a file in the "raw" format (LISP expressions).

Execution of a request is done the following way (if we imagine a Maxima console):

```
load('script.mac'); /* If the script.mac is present */
load('data.lsp'); /* If the data.lsp is present */
command;
```

After the job completion, queries to its URI will begin to return the result in addition to the job status. The result, similarly to the input parameters, can be presented in various formats depending on the container implementation and on the value of the Accept header of the client's request. Let's list the output parameters contained in the response. All of them are mandatory:

— result – a string containing the console output generated by the "command" expression. It is designed to transmit small amounts of data such as diagnostic messages.

— output.lsp – a file in "raw" LISP format that could be empty. This file can be written to from Maxima during the job execution. This file is designed to transmit data (perhaps large amount) between the steps of an algorithm.

Let's describe the contents of the implementation field of the service definition. It's responsible for the container-specific details:

— type – the type of the service implementation in EveREST. In this case, a Java class.

— class – the name of the class providing the service implementation. In our case, the name of the Jython adapter class.

— config – an object containing the configuration parameters for the service. The pythonPackage and pythonClass options are Jython adapter specific and indicate the package and the class name of the implementation of the service, respectively. The maximaPath option is the only Maxima service parameter. It specifies the path to the Maxima executable.

5. Conclusion

The service enables running Maxima commands remotely using a RESTful HTTP-based API. The API is compatible with the WfMS and its Web UI which enables visual programming of complex computation scenarios with the use of the service and other facilities of the MathCloud.org environment. The API is asynchronous giving an opportunity to execute long running tasks without continuous interaction with the client software. The service manages a queue of requests which are dispatched in parallel by a pool of spare Maxima processes. It yields effective use of Maxima in multiprocessor systems (e.g. in a multi-core system).

The service was involved in computational experiments on error-free inversion of ill-conditioned matrices. During the experiments with high dimension matrices it appeared that Maxima executing under SBCL (Steel Bank Common Lisp) performs much better than under GCL (GNU Common Lisp). The service currently supports three Common Lisp implementations (SBCL, GCL, CLISP) and can be easily updated to support more if needed.

References

[1] Zhitnikov V. Computers, Mathematics and freedom. // Computerra. - 2006. - I. 16 (636). - pp. 40-43. (in Russian).

[2] Astafiev A.S., Afanasiev A.P., Lazarev I.V., Suhoroslov O.V., Tarasov A.S. Scientific service-oriented environment based on Web technology and distributed computing. // Proceedings of the All-Russian scientific conference (Novorossiysk) "Scientific service on the Internet: scalability, parallelism, efficiency." - 2009. (in Russian).

[3] Fielding R.T. Architectural styles and the design of network-based software architectures. // PhD Dissertation. Dept. of Information and Computer Science, University of California, Irvine, 2000.

[4] Richardson L., Ruby S. RESTful Web Services. // O'Reilly, 2007.

[5] Suhoroslov O.V. A unified interface to access algorithmic services on the Web. // Problems of computing in a distributed environment, Ed. S.V. Emelyanov, A.P. Afanasyev. Proceedings of ISA RAN. V. 46. (in Russian).

[6] Simple C++ CAS Maxima API and a RPC service based on it, http://code.google.com/p/remote-maxima/

[7] Boost.Process, http://www.highscore.de/boost/process/

[8] Karlsson B. Beyond the C++ Standard Library: An Introduction to Boost. // Addison-Wesley Professional, 2006.

[9] Lazarev I., Sukhoroslov O. On Development of Workflow Management Service for Distributed Computations. // Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 3rd Intern. Conf. (Dubna, June 30 - July 4, 2008). - Dubna: JINR, 2008. - 401 p. (pp. 291-294).

[10] Jython: Python for the Java Platform, http://www.jython.org/

ON DEVELOPMENT OF GRID-ENABLED APPLICATIONS AND SERVICE-ORIENTED SCIENTIFIC ENVIRONMENTS[1]

O. V. Sukhoroslov

Centre for Grid Technologies and Distributed Computing,
Institute for Systems Analysis, Russian Academy of Sciences,
Prosp. 60-let Oktyabrya 9, 117312 Moscow, Russia
os@isa.ru

The paper presents a set of tools which simplify development of grid-enabled applications and provision of these applications as services in service-oriented scientific environments.

1. Introduction

The emerging service-oriented scientific environments [1] provide a context for sharing and reuse of computing applications across scientific communities. In order to achieve desired performance and scalability these applications often leverage distributed computing resources. Grid-enabled application is an application that runs on grid resources. It is often not written from scratch but ported to grid. The development of grid-enabled applications is accompanied by several challenges such as low-level grid access mechanisms, lack of interoperability between grids, application porting, implementation of distributed coordination, load balancing and fault recovery. Providing such application to users as a service also represents a significant challenge because existing grid middleware don't provide adequate tools for building service-oriented scientific environments.

The paper presents a set of tools which overcome aforementioned problems by simplifying development of grid-enabled applications and provision of these applications as services. These tools can be divided into three layers described below.

The "Infrastructure Interfaces" layer contains high-level APIs and programming libraries for accessing heterogeneous distributed computing resources such as clusters, grid infrastructures, desktop grids and cloud computing services. These tools hide from the application developer the complexity of the underlying middleware and infrastructure. Section 2 presents jLite library which is an example of such tool providing simple API for accessing resources of EGEE/EGI, the largest production grid infrastructure. jLite supports complete grid job management lifecycle. It is cross-platform and doesn't require installation of gLite User Interface.

The "Application Frameworks" layer contains ready-to-use implementations of common patterns of distributed computing (coordination, load balancing, fault recovery, etc.). Such frameworks provide high-level programming models which simplify application development by allowing developer to concentrate on implementation of problem-specific parts of application. Section 3 presents MaWo framework which implements common parts of master-worker pattern such as worker allocation, communication with master, task scheduling, data transfer, failure recovery, etc. MaWo supports declarative application description and includes easy-to-use tools for running applications across heterogeneous computing resources including local workstations, clusters and grids.

The "Service-Oriented Toolkits" layer contains tools for development, deployment, discovery and composition of computing services. The role of this layer is to transform existing applications into services by using standard protocols and description formats. The industrial standard-de-facto, SOAP-based Web services suffer from complexity and performance issues. Section 4 presents a simpler alternative for describing and accessing scientific services based on the REST architectural style. The proposed approach is being implemented in the context of MathCloud, a service-oriented environment

[1] The work is supported by the RFBR (grant 08-07-00430-a) and RAS Presidium (Prog. П1).

for mathematical research. The implemented service container supports rapid deployment of command-line and grid applications as services. The Web-based workflow editor and runtime environment enable composition of services into new applications.

2. jLite

jLite [2, 3] is a Java library providing simple API for accessing gLite based grid infrastructure, such as EGEE/EGI. The API provides functionality similar to gLite User Interface commands and can be used for development of grid-enabled Java applications, cross-platform tools, grid portals and services.

jLite is intended for Java developers who struggle with gLite middleware and want to reduce time and effort needed to build a grid application. Existing Java APIs for gLite expose low-level grid service operations and are scattered among several packages with complex external dependencies. Available API usage examples often imply the presence of gLite User Interface (UI) environment installed on Scientific Linux. The use of APIs in the absence of gLite UI requires non-trivial configuration. This complicates the use of these APIs for development of cross-platform grid applications.

jLite is addressing these problems by providing a high-level Java API with functionality similar to gLite shell commands. Current implementation supports complete gLite job management lifecycle including VOMS proxy creation and delegation, transfer of job input files, job submission, job status monitoring and download of job output files. It supports normal, collection and parametric job types. The API hides complexity of underlying middleware and its configuration. jLite is easy to install because it includes all external dependencies and does not require installation of gLite UI. The library is pure Java and can be used on any Java-capable platform including Windows.

By providing an easy-to- use and portable API jLite simplifies development of cross-platform grid applications on top of the EGEE grid infrastructure. In contrast to Simple API for Grid Applications (SAGA) [4], which is an effort to define standard middleware-independent APIs, jLite focuses on one middleware platform and one programming language. jLite also includes a command-line interface which can be used as a simple cross-platform alternative to gLite UI on Windows and other operating systems. This complements efforts to port gLite UI on platforms different from Scientific Linux and enables grid users to submit jobs directly from their desktop machines.

jLite code [3] is available under Apache License 2.0. The library was used to build several grid-enabled applications and systems, e.g. [5] and MaWo framework presented below.

3. MaWo

Grid is an ideal platform for Bag-of-Tasks (BoT) applications composed of many (possibly thousands of) independent tasks, e.g. parameter study, Monte-Carlo simulations and image processing. The run time of such applications in a grid strongly depends on strategies used for scheduling of tasks, fault recovery and data management. A well-known "master-worker" pattern proved to be efficient for heterogeneous, dynamically available distributed resources. It can also improve a run time of application in a grid by bypassing central grid scheduler.

In order to reduce time and effort needed to port a BoT application to grid a generic master-worker framework called MaWo [6, 7] was developed. The framework implements generic parts of master-worker pattern such as worker allocation, communication with master, task scheduling, data transfer, failure recovery, etc. (Fig. 1) MaWo provides a programming interface for implementation of problem-specific parts of application. The framework also supports declarative description of BoT applications with support for arbitrary executable files which enables quick porting of applications without using MaWo API.

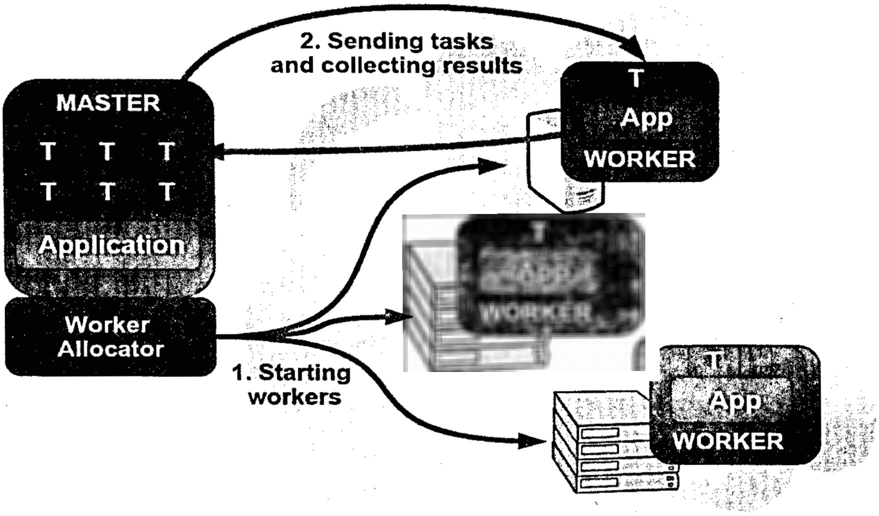


Fig. 1: MaWo architecture

MaWo master is started on a user desktop or a server outside the grid. It is responsible for task scheduling and collection of results. The master runs a built-in FTP sever which is used for application data transfer between master and workers with support for large data files. It also provides a Web interface for monitoring the status of running application.

Since many users have access to several computing resources and infrastructures MaWo supports simultaneous use of various types of resources by means of pluggable adapters. Current implementation includes adapters for allocation of workers on local machine, cluster and EGEE grid infrastructure. Each adapter provides a command for starting desired number of worker processes. The EGEE adapter utilizes jLite library (see Section 2) in order to achieve portability.

In contrast to existing master-worker frameworks for EGEE, such as DIANE [8], MaWo requires less effort to develop or port applications because it doesn't require installation of gLite User Interface and supports simple declarative description of an application in addition to programming interface. The applicability of the framework was demonstrated by successfully running several large-scale applications in EGEE including ray tracing, atomic cluster conformation problem and parameter study for geophysical models.

4. MathCloud

The concept of Service-Oriented Science [1] introduced by Ian Foster in 2005 refers to scientific research enabled by distributed networks of interoperating services. The service-oriented architecture defines standard interfaces and protocols for provision of applications as remotely accessible services. This opens up new opportunities for science by enabling wide-scale sharing, publication and reuse of scientific applications, as well as automation of scientific tasks and composition of applications into new services. We argue that existing grid middleware, though providing a mature software infrastructure for federation of computing resources, is too complex and don't provide adequate tools for building service-oriented scientific environments.

Therefore we propose a novel software infrastructure for enabling service-oriented science aimed on radical simplification of service development, deployment and use. In contrast to modern grid middleware based on Web Services specifications the proposed infrastructure embraces a more lightweight approach by using the Representational State Transfer (REST) architectural style [9], Web technologies and Web 2.0 application models.

According to the proposed approach each service represents a RESTful web service with a unified API enabling service introspection, job submission and retrieval of job results. A service has predefined sets of input and output parameters which are passed during job submission and returned as a job result respectively. A client can retrieve the description of service parameters as a JSON Schema and then submit a job by sending a JSON document with input parameters. The RESTful API supports asynchronous job processing and passing large data files as links.

The proposed approach is being implemented in the context of MathCloud [10, 11], a service-oriented environment for mathematical research.

The core component of proposed software infrastructure is a service container which implements the RESTful API and provides a hosting environment for services. The service container simplifies service development and deployment by providing ready-to-use adaptors for command-line, Java and grid applications. For example, in order to expose a command-line application as a new service a user has only to provide a declarative description mapping service parameters to command line options and files. The similar approach is used for grid applications, with the exception of the user has also to provide a job description file. In addition to RESTful API each service deployed in the service container has a browser-accessible Web interface. A secured access to services is supported by using OpenID accounts or X.509 certificates for user authentication.

The service composition is a crucial aspect of service-oriented systems enabling various application scenarios. Therefore we implemented a workflow management system with a Web-based graphical user interface (Fig. 2). The user interface is inspired by Yahoo! Pipes and provides easy-to-use tools for building workflows by connecting services with each other. The created workflow can be published as a new service thus contributing back to the environment.

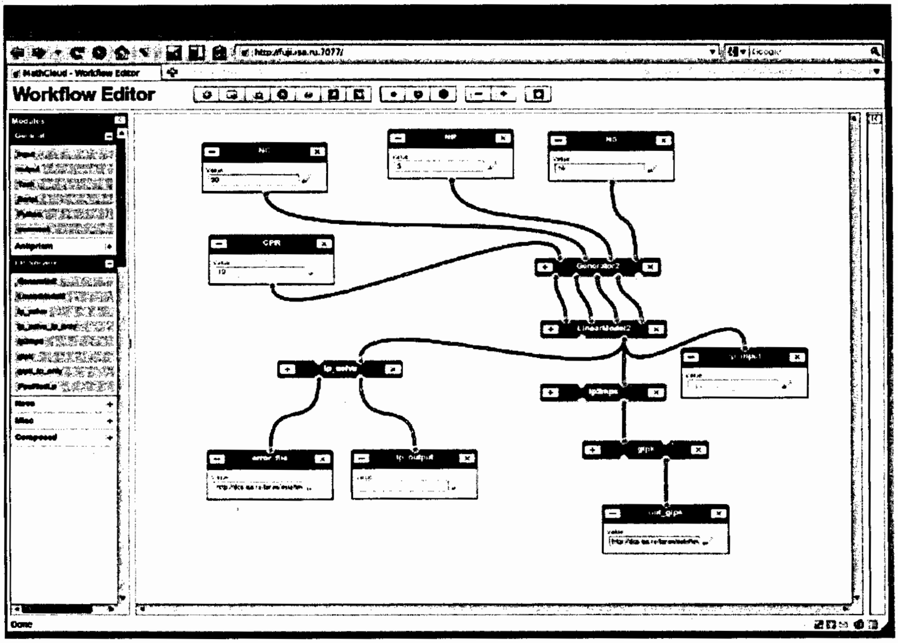


Fig. 2: Graphical workflow editor for MathCloud environment

5. Conclusion and Future Work

The paper presented a set of tools which simplify development of grid-enabled applications, starting with porting an application to a grid and ending with publishing the application as a service. The future work will focus on further development of presented tools, as well as integration of these tools with each other.

References

[1] Foster I. Service-Oriented Science // Science, 2005. V. 308. N. 5723. P. 814-817.

[2] Sukhoroslov O.V. jLite: A Lightweight Java API for gLite//Proc. of Int. Conference "Distributed Computing and Grid-Technologies in Science and Education", Dubna, 2008. Dubna: JINR, 2008. P. 201-204.

[3] jLite, http://jlite.googlecode.com/

[4] Simple API for Grid Applications, http://saga.cct.lsu.edu/

[5] Ketan Maheshwari, Paolo Missier, Carole Goble, Johan Montagnat. Medical Image Processing Workflow Support on the EGEE Grid with Taverna// Proceedings of the Intl Symposium on Computer Based Medical Systems (CBMS'09), IEEE, Albuquerque, New Mexico, USA, 2009.

[6] Sukhoroslov O. Running Bag-of-Tasks Applications in EGEE with MaWo // 5th EGEE User Forum, Uppsala, Sweden, 2010.

[7] MaWo, http://mawo.googlecode.com/

[8] DIANE, http://cern.ch/diane/

[9] Fielding R.T. Architectural styles and the design of network-based software architectures. PhD Dissertation. Dept. of Information and Computer Science, University of California, Irvine, 2000.

[10]Afanasiev A., Lazarev I., Tarasov A. MathCloud - a distributed mathematical environment // Proceedings of the 3rd International Conference "Distributed Computing and Grid-technologies in Science and Education", GRID'2008, Dubna, Russia, 30 June - 4 July, 2008.

[11]MathCloud project, http://www.mathcloud.org/

LHC DATABASES ON THE GRID: ACHIEVEMENTS AND OPEN ISSUES[1]

A. V. Vaniachine

Argonne National Laboratory, 9700 S Cass Ave, Argonne, IL, 60439, USA

To extract physics results from the recorded data, the LHC experiments are using Grid computing infrastructure. The event data processing on the Grid requires scalable access to non-event data (detector conditions, calibrations, etc.) stored in relational databases. The database-resident data are critical for the event data reconstruction processing steps and often required for physics analysis.

This paper reviews LHC experience with database technologies for the Grid computing. List of topics includes: database integration with Grid computing models of the LHC experiments; choice of database technologies; examples of database interfaces; distributed database applications (data complexity, update frequency, data volumes and access patterns); scalability of database access in the Grid computing environment of the LHC experiments. The review describes areas in which substantial progress was made and remaining open issues.

1. Introduction

In 2010 four experiments at the Large Hadron Collider (LHC) started taking valuable data in the new record energy regime. In preparations for data taking, the LHC experiments developed comprehensive distributed computing infrastructures, which include numerous databases. This paper reviews LHC experience with database technologies for the Grid computing, including areas in which substantial progress was made. I was responsible for Database Deployment and Operations (formerly called Distributed Database Services) in the ATLAS experiment since 2004. As a result, this review is biased towards my personal views and experience. Beyond ATLAS, I started compiling information on databases in LHC experiments for my earlier invited talks on the subject [1, 2].

As an example of what relational databases are used for in each LHC experiments, I briefly describe ATLAS database applications. In ATLAS, there are more than fifty database applications that reside on the central ("offline") Oracle server. By February 2010 ATLAS accumulated more than 8 TB of data, which are dominated by 4 TB of slow control data. Most of these database applications are "central" by their nature, like the ATLAS Authorship Database used to generate author lists for publications. The central databases are traditional applications developed according to standard Oracle best practices with the help of our OCP database administrators. Because these database applications are designed by traditional means, I will not cover LHC experience with these central applications in this review, since I cannot claim that LHC advanced the existing knowledge base in these traditional areas.

In contrast to the central database applications that are accessed by people or by limited number of computers and do not have to be distributed, a subset of LHC database applications must be distributed worldwide (for scalability) since they are accessed by numerous computers (Worker Nodes) on the Grid.

2. Database Applications and Computing Models of LHC Experiments

The LHC experiments are facing an unprecedented multi-petabyte data processing task. To address that challenge LHC computing models adopted Grid computing technologies. These computing models of LHC experiments determined the need for distributed database access on the Grid. The LHC Computing models are well represented at this conference in several talks and reviews

[1] Invited talk presented at the IV International Conference on "Distributed computing and Grid-technologies in science and education" (Grid2010), JINR, Dubna, Russia, 28 June - 3 July, 2010.

[3, 4]. In essence, the LHC computing models are mostly focused on the problem of managing the petabyte-scale event data that are kept in a file-based data store, with files catalogued in various databases. These event store databases are an integral part of the LHC computing models. A brief description of a growing LHC experience with the event store database as it approach petasacles is provided in the last section.

In addition to the file-based event data, LHC data processing and analysis require access to large amounts of the non-event data (detector conditions, calibrations, etc.) stored in relational databases. In contrast to the file-based LHC event store databases, the database-resident data flow is not detailed in the "big picture" of LHC computing models. However, in this particular area the LHC experiments made a substantial progress compared with other scientific disciplines that use Grids. That is why I will focus on the LHC experience with distributed database applications introduced in the next section.

3. Distributed Database Applications Overview

In ATLAS there are only few database applications that have to be distributed: Trigger DB, Geometry DB, Conditions DB and Tag DB. ATLAS developed the Trigger DB for central ("online") operations. A small subset of the whole database is distributed on the Grid in SQLite files for use in Monte Carlo simulations. To manage the detector description constants ("primary numbers") ATLAS developed the Geometry DB with contributions from LHC Computing Grid (LCG). It is the first ATLAS database application that was deployed worldwide. It is distributed on the Grid in SQLite replica files. The Conditions DB was developed by LCG with ATLAS contributions. The LCG technology for Conditions DB is called COOL. The Conditions DB is a most challenging database application. It is a hybrid application that includes data in RDBMS and in files. Conditions data are distributed worldwide via Oracle Streams and via files. The ATLAS Tag DB stores event-level metadata for physics (and detector commissioning). It was developed by LCG with ATLAS contributions. It is distributed worldwide in files and also 4 TB are hosted at select Oracle servers. The Tag DB is expected to collect 40 TB of data per nominal LHC year of operations. Given the limited data taken to date, we have not yet gathered much experience in large-scale Tag DB access.

Another LHC experiment that adopted common LCG technology for Conditions DB—COOL (Conditions database Of Objects for LHC)—is LHCb. In COOL database application architecture the Interval-of-Validity (IOV) metadata and a data payload, with an optional version tag, usually characterize the conditions data. Similar Conditions database architecture was developed by the CMS experiment (Fig. 1). The CMS conditions database stores time-varying data (calibration and

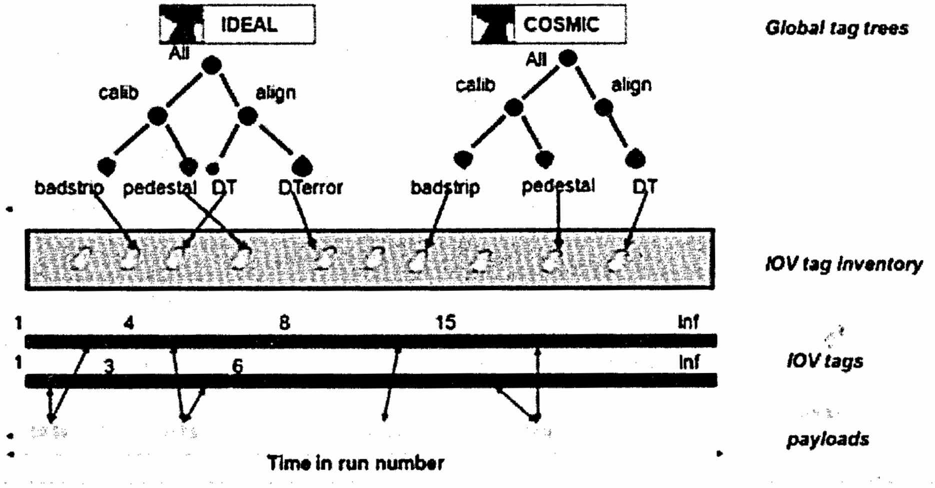


Fig. 1: The CMS conditions database architecture

alignment) together with their sequences versions IOV. IOV sequences are identified by tags, which are organized as trees. A tag tree can be traversed from any node (global tag).

As in ATLAS, the ALICE Conditions DB is also hybrid, comprised of data stored in the Offline Conditions Database (OCDB) a set of entries in the AliEn file catalogue pointing to files stored in the Grid. Together with the conditions data, the reference data are also stored in an analogous database.

4. Distributed Database Applications Details

This section describes database applications requirements, data complexity, update frequency, data volumes, usage, etc. Generally, these factors dictate the choice of database technologies: relational or hybrid (chosen in ATLAS and ALICE experiments for Conditions DB implementation).

4.1. ATLAS Geometry DB

Due to differences in requirements and implementation, ATLAS Geometry DB is separated from the Conditions DB to keep static information, such as nominal geometry. Only the time-dependent alignment corrections to Geometry are stored in the Conditions DB. Such separation of concerns resulted in a moderate data complexity of the Geometry DB. A recent statistics (February, 2010) counted 434 data structures described in 872 tables in one schema. The total number of rows was 555,162, resulting in an SQLite replica volume of 33 MB. The update frequency of the Geometry DB is "static" i.e. upon request, when the geometry corrections or updates become necessary. The database is accessed via a low-level common LCG database access interface called Common Object-Relational Access Layer (CORAL).

A typical data reconstruction job makes about 3K queries to the Geometry database. The master Geometry DB resides in the "offline" Oracle, where it is not used for production access. For example, for the Tier-0 operations an SQLite snapshot replica is made nightly. The Geometry DB is replicated on the Grid via SQLite files. During 2009 twenty-nine SQLite snapshots were distributed on the Grid, in 2008 it was eighteen.

4.2. ATLAS Conditions DB

Driven by the complexity of the subdetectors requirements, ATLAS Conditions DB technology is hybrid: it has both database-resident information and external data in separate files, which are referenced by the database-resident data. These external files in a common LHC format are called POOL. ATLAS database-resident information exists in its entirety in Oracle but can be distributed in smaller "slices" of data using SQLite. Since Oracle was chosen as a database technology for the "online" DB, ATLAS benefits of uniform Oracle technology deployment down to the Tier-1 centers. Adoption of Oracle avoids translating from one technology to another and leverages Oracle support from CERN IT and WLCG 3D Services [6].

Historically, ATLAS separated conditions database instances for Monte Carlo simulations and for the real data. The two instances still remain separate to prevent accidental overwrite of the Conditions DB for real data. Both Conditions DB instances are accessed via common LCG interface COOL/CORAL. This approach is similar to the CMS Conditions DB partitioning by usage (see below).

The complexity of the ATLAS Conditions DB data for simulations is high. According to a representative snapshot of February, 2010 the instance has 2,893 tables in four schemas. The total number of rows is 842,079 and the data volume of the SQLite replica is 376 MB. There are additionally 247 MB of data in 1049 POOL/ROOT files grouped in 25 datasets. The update frequency is "static," i.e. the database is updated upon request typically in preparation for major Monte Carlo simulations campaigns. All conditions data for Monte Carlo simulations is replicated on the Grid vial files (the full snapshot in SQLite plus the external POOL/ROOT files and their catalogs.).

The ATLAS Conditions DB for real data has a very high complexity. In February 2010, the database had 7,954 tables in 29 active schemas out of 45 schemas total. The schema count is determined by the number of ATLAS detector subsystems: 15 subsystems each having two schemas ("online" and "offline") plus one inactive combined schema (to be decommissioned). The total number of rows is 761,845,364 and the Oracle data volume is 0.5 TB. There are additionally 0.2 TB in

POOL/ROOT files grouped in 48 datasets. The Conditions DB for real data is updated continuously. Because of the large volume, use of the full database replica on the Grid is not practical. Only the required "slices" of the ATLAS Conditions DB data are distributed on the Grid. To process a 2 GB file with 1K raw events a typical reconstruction job makes about 11K queries to read more than 70 MB of database-resident data (with some jobs read tens of MB extra) plus about ten times more volume of data is read from the external POOL files.

4.3. LHCb Conditions DB

The LHCb reconstruction and analysis jobs are making direct connection via COOL/CORAL libraries from the Worker Nodes on the Grid to the Oracle replicas at the Tier-1 sites. Jobs require a limited amount of data transfer (~40 MB) in the first few minutes. SQLite replicas are used in the special cases, such as Monte Carlo simulations.

4.4. ALICE Conditions DB

Figure 2 shows conditions data flow in Shuttle—a special service providing an interface between the protected online world and the external computing resources [5]. Since 2008 the ALICE Conditions DB accumulated more that 30 GB of data for about 183,000 files plus more than 8 GB of the reference data for more than 29,000 files. All collected conditions data are exported on the Grid, thus making them accessible for the reconstruction and analysis.

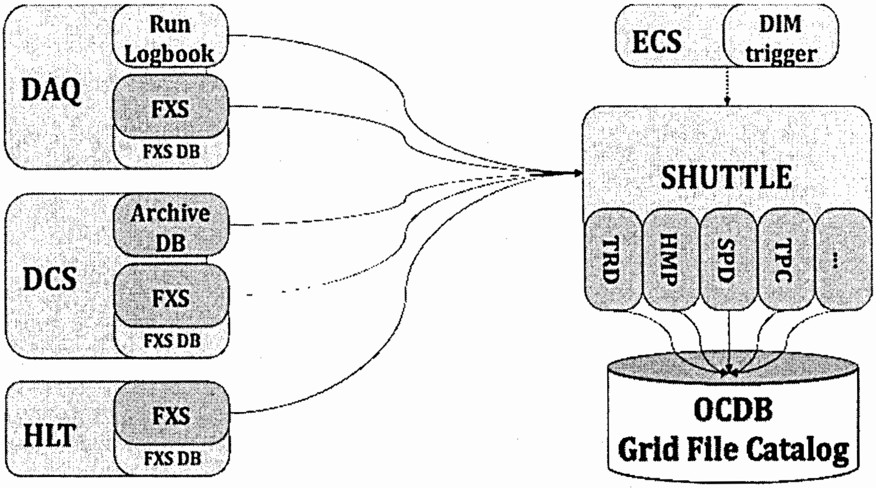


Fig. 2: ALICE Shuttle framework for Offline Conditions DB (OCDB) [5]

4.5. CMS Conditions DB

All relations in Conditions DB are purely logical and application specific. As in case of other LHC experiments, no RDBMS consistency enforced, which allows full flexibility in copying (deep and shallow) at all level of the structure. As in case of ATLAS, data consistency is enforced not by database design but through a set of policies, such as NO DELETE, NO UPDATE. In CMS, only the current IOV sequence can be extended. In contrast to ATLAS, the data payloads are implemented as POOL/ORA objects stored in the database internally.

As in ATLAS, the CMS Conditions DB has been "partitioned" into schemas following development and deployment criteria, which keep separated areas of independent development: by sub-detectors, by software release. These are "partitioned" further by use-cases to keep separated independent workflows use cases. In case of Monte Carlo simulations, all relevant data are copied into a dedicated schema including even a single SQLite file. In case of re-processing at remote Tier-1 sites, a read-only snapshot of the whole Conditions DB is made for access through Frontier. Making the replica copy read-only prevents accidental overwrites, since the master Conditions DB is continuously updated for use in prompt data processing: reconstruction, calibration, and analysis at Tier-0. The database is managed through application-specific tools described in the next section. A CMS data reconstruction job reads about 60 MB of data.

5. Database Integration

This section provides examples of custom database interfaces and tools on top of CORAL/COOL and describes integration of databases with software frameworks and into an overall data acquisition, data processing and analysis chains of the experiments.

Figure 3 presents an example of a database tool in the CMS PopCon (Populator of Condition objects). Fully integrated in the overall CMS framework, PopCon is an application package intended to transfer, store, and retrieve condition data in the "offline" databases. PopCon also assigns metadata information: tag and IOV.

Support for on-demand data access — a key feature of the common Gaudi/Athena framework — emphasizes the importance of database interfaces for LHCb and ATLAS experiments. On-demand data access architecture makes Oracle use straightforward. In contrast, the delivery of the required Conditions DB data in files is challenging, but can be implemented for a well-organized workflow, such as reprocessing. In the on-demand data access environment having a redundant infrastructure for database access turns out to be advantageous. The redundancy is achieved through common LHC interfaces for persistent data access, which assure independence on available technologies (Oracle, SQLite, Frontier...). No changes in the application code are needed to switch between various database technologies (Fig. 4). In ATLAS, each major use case is functionally covered by more than one of the available technologies, so that we can achieve a redundant and robust database access system.

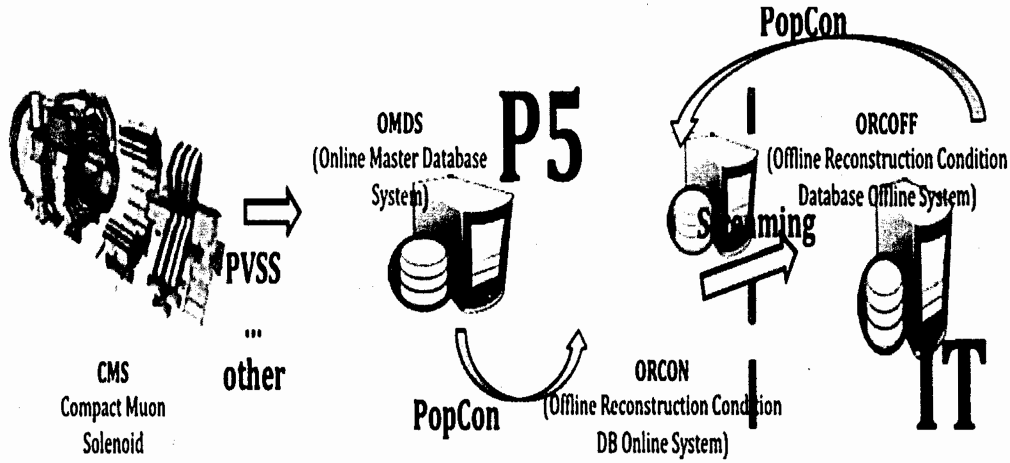


Fig. 3: CMS database tool PopCon is used in all three CMS Oracle databases for the conditions data

In addition, various tools can be built on top of the interfaces. For example, since the large volume of ATLAS Conditions DB prevents use of the full database replica on the Grid, an advanced "db-on-demand" tool was developed to produce "slices" of the required conditions data for the Grid jobs [7].

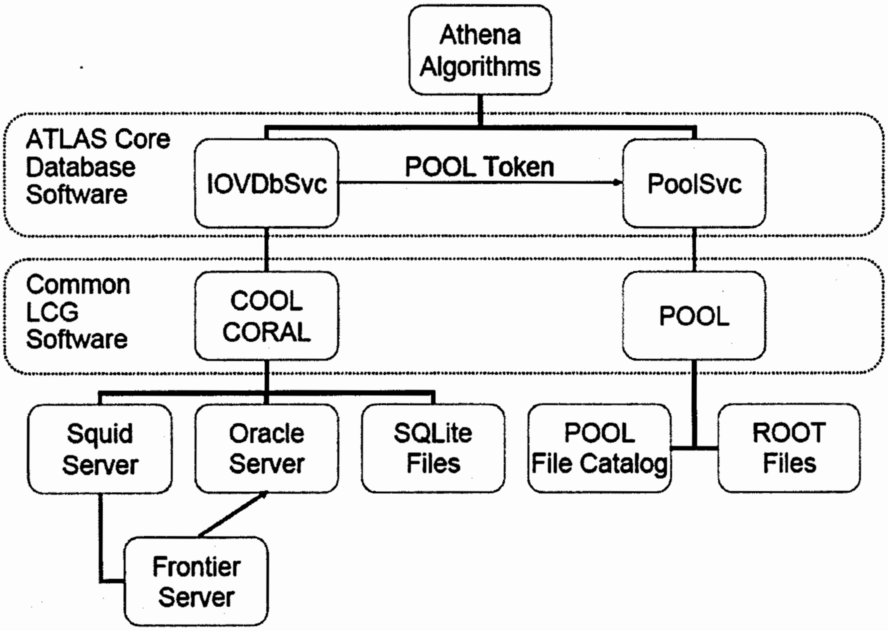


Fig. 4: Integration of common LCG interfaces for database access in case of the ATLAS software framework Athena

6. Scalability of Database Access on the Grid

Scalability of database access in the distributed computing environment is a challenging area in which a substantial progresse was made by the LHC experiments.

6.1. ATLAS Database Release Technology

In a non-Grid environment, in case of ATLAS, two solutions assure scalability of access to Conditions DB database: a highly replicated AFS volume for the Conditions POOL files and the throttling of job submission at Tier-0 batch system. None of Tier-0 solutions for scalable database access is available on the Grid. As a result, ATLAS experience with database access on the Grid provided many useful "lessons learned."

In 2004, we found that the chaotic nature of Grid computing increases fluctuations in database load: daily fluctuations in the load are fourteen times higher than purely statistical [8]. To avoid bottlenecks in production, the database servers capacities should be adequate for a peak demand [6]. In 2005, to overcome scalability limitations in database access on the Grid, ATLAS introduced the Database Release concept [9]. Conceptulally similar to the software release packaging for distribution on the Grid, the Database Release integrates all necessary data in a single tar file:

- the Geometry DB snapshot as an SQLite file,
- a full snapshot of Conditions DB data for Monte Carlo in the SQLite file,
- plus corresponding Conditions DB POOL files and their POOL File Catalogue.

Years of experience resulted in continuous improvements in the Database Release approach, which now provides solid foundation for ATLAS Monte Carlo simulation in production [10]. In 2007 the Database Release approach was proposed as a backup for database access in reprocessing at Tier-1 sites (Fig. 5).

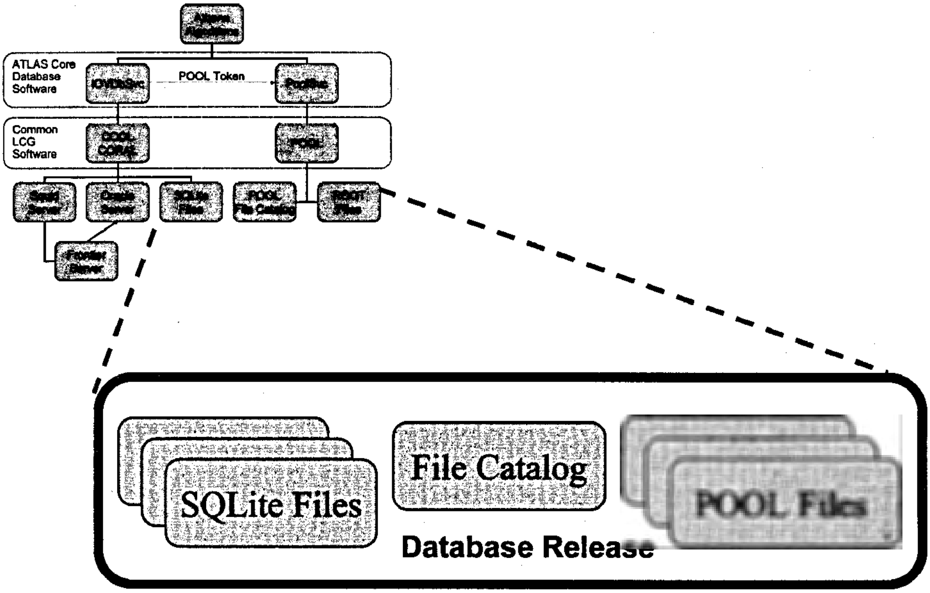


Fig. 5: ATLAS Database Release technology hides the complexity of Conditions DB access (Fig. 4)

In addition to Database Releases, ATLAS Conditions DB data are delivered to all ten Tier-1 sites via continuous updates using Oracle Streams technology [11]. To assure scalable database access during reprocessing ATLAS conducted Oracle stress-testing at the Tier-1 sites. As a result of stress-tests we realized that the original model, where reprocessing jobs would run only at Tier-1 sites and access directly their Oracle servers, would cause unnecessary restrictions to the reprocessing throughput and most likely overload all Oracle servers [12].

Thus, the DB Release approach, developed as a backup, was selected as a baseline. The following strategic decisions for database access in reprocessing were made:

- read most of database-resident data from SQLite,
- optimize SQLite access and reduce volume of SQLite replicas,
- maintain access to Oracle (to assure a working backup technology, when required).

As a result of these decisions ATLAS DB Release technology fully satisfies the Computing Model requirements of data reprocessing and Monte Carlo production: it is fast (less than 10 s per job), robust (failure rate less than 10^{-6} per job) and scalable: (served ~1B queries in one of reprocessing campaigns). The read-only Database Release dataset guarantees reproducibility and prevents access to unnecessary data (similar to CMS partitioning by usage).

6.2. CMS Frontier/Squid Technology

Frontier/Squid is a data caching system providing advantages for distributed computing. To achieve scalability, the system deploys multiple layers of hardware and software between a database server and a client: the Frontier Java servlet running within a Tomcat servlet container and the Squid—a single-threaded http proxy/caching server (Fig. 6) [13]. Depending on a fraction of the shared data required by the jobs, the time needed to retrieve conditions data at the beginning of a job is reduced by factors of 3 to 100, depending on the distance between the site running the job and the remote site providing the Oracle conditions database.

To reduce a chaotic load on the Oracle databases at the Tier-1 sites caused by the the analysis jobs, ATLAS adopted the CMS Frontier/Squid technology, which have been shown to drastically reduce this load. This greatly improves the robustness of the conditions data distribution system. Having multiple Frontier servers has provided redundancy. For that, ATLAS has implemented Frontier servers at most of the Tier-1 sites and Squid servers at all Tier-0/1/2. Work is underway to provide Squid servers at most ATLAS Tier-3 sites.

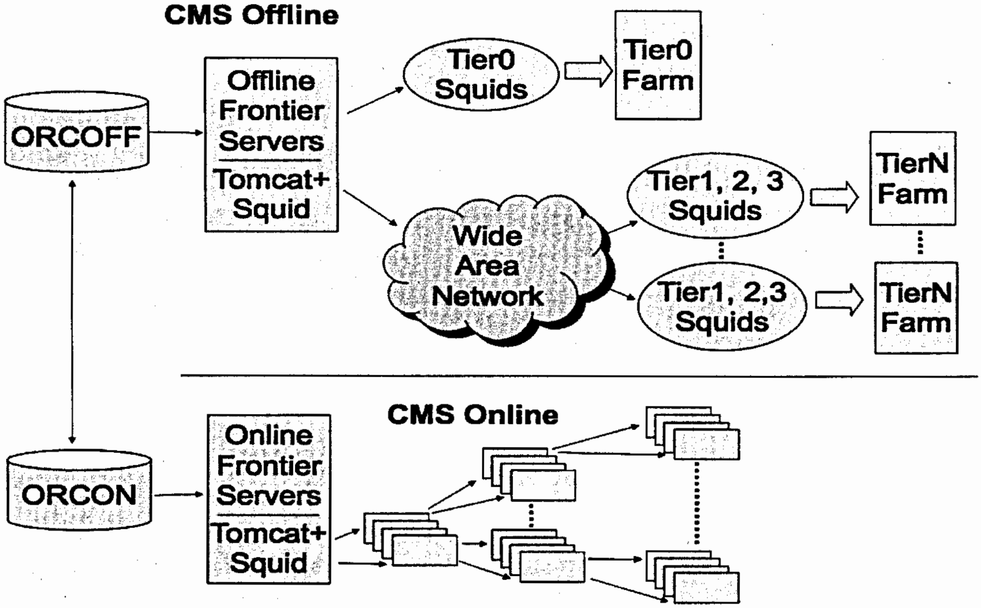


Fig. 6: CMS Frontier/Squid deployment architecture

7. Scalability of the LHC Event Store Database

Due to a hardware errors data corruption is inevitable in a large-scale data store. A recent CERN Tier-0 study found end-to-end byte error rates of as high as one error per 3×10^7 bytes [14]. The low-level TCP/IP checksum is weak and can fail to detect errors in packets. Thus, it is possible for a packet to be corrupted in transmission. In a petascale event store, this is guaranteed to happen occasionally. Also, data corruption happens before LHC data gets to TCP/IP, e.g. due to software or a memory error. In case of a compressed file, a single bit-flip often results in a corruption of the whole file[2]. Similarly, in case of the ATLAS event store, we must discard a whole dataset with thousands of files if the single file is corrupted. However, at a certain data corruption rate this approach is not scalable, since a very large dataset will waste a lot of attempts during production. To assure event store scalability, LHC experiments introduced redundant higher-levels checksums to detect these types of errors. In ATLAS, every event store file is checked immediately after it was produced. The check verifies that the POOL/ROOT zlib compressed data buffers have correct checksums. If the file is

[2] To avoid that, some special data recovery techniques should be deployed.

unreadable, the job marked as failed and re-executed. This is a critical benefit provided by file compression, since the size reduction is only ~5%.

LHC experience shows that we must introduce considerable redundancy in order to detect and recover from data corruption errors, since these errors are costly to fix. The next redundant check is done at the end of each Grid job, when the checksum is calculated for each file produced at the Worker Node. This checksum is compared with the checksum calculated for the file transferred to the Storage Element by the LHC data transfer tools. In case of the checksum mismatch, the job is marked as failed and re-executed. Sites, that did not implement this check, produce silent data corruption, where the mismatch is discovered at a later stage. This is not scalable, since, correcting silent data corruption in a distributed petascale event store is very costly. To assure scalability, the data corruption must be detected at the spot.

Learning from the initial operational experience, LHC experiments realized that the end-to-end data integrity strategies need to be developed for petascale data store. In a petascale event store, every layer of services should not assume that the underlying layer never provide corrupted or inconsistent data. As a result of these improvements, in ATLAS Grid operations only about fifty cases of corrupted files were detected in 2010, while the ATLAS event store reached 40 PB of data in 0.14 × 10^9 files. This is a critical improvement over the raw data corruption rates of one bad file in 1500 files observed in a local environment without redundant checks [14].

8. Summary

LHC experiments developed and deployed distributed database infrastructure ready for the LHC long run In ATLAS each major use case is functionally covered by more than one of the available technologies to assure a redundant and robust database access. In CMS a novel http data caching system—Frontier/Squid—assures scalability of Conditions DB access on the Grid. The Frontier/Squid technologies are adopted by ATLAS as a part of end-to-end solution for Conditions DB access in user analysis. These technologies for distributed database access represent the area where the LHC experiments made a substantial progress, compared with other scientific disciplines that use Grids. Remaining open issues provide roadmap for future R&D in the area of scalable tools for data-intensive scientific discovery.

Acknowledgements

I wish to thank the Conference organizers for their invitation and hospitality. I also thank all my collaborators and colleagues who provided materials for this review. This work is supported in part by the U.S. Department of Energy, Division of High Energy Physics, under Contract DE-AC02-06CH11357.

References

[1] Vaniachine A. “Databases in LHC Experiments: Usage and Lessons Learned,” invited talk presented at the *SuperB Computing R&D Workshop*, Ferrara, Italy, March 9-12, 2010.

[2] Vaniachine A. “Experiment Requirements: ALICE, ATLAS, CMS & LHCb,” invited talk presented at the *WLCG Distributed Database Workshop*, Barcelona, Spain, April 20-21, 2009.

[3] Shabratova G. (on behalf of ALICE) “The ALICE Grid Operation,” in these proceedings.

[4] Kreuzer P. (for the CMS Computing Project) “Experience with CMS Computing on the GRID since the Start of the LHC Data Taking,” in these proceedings.

[5] Grosse-Oetringhaus J. F. et al. “The ALICE online-offline framework for the extraction of conditions data,” *J. Phys.: Conf. Ser.* **219** 022010, 2010.

[6] Vaniachine A.V. and J.G. von der Schmitt, “Development, Deployment and Operations of ATLAS Databases,” *J. Phys.: Conf. Ser.* **119** 072031, 2008.

[7] Borodin M. (for the ATLAS Collaboration) “Optimizing On-demand Database Access for ATLAS Grid Data Processing,” in these proceedings.

[8] Vaniachine A., Malon D., Vranicar M. “Advanced Technologies for Distributed Database Services Hyperinfrastructure.” *Int. J. of Mod. Phys. A* **20** (16): 3877, 2005.

[9] Vaniachine A. “ATLAS Database Readiness for LHC Data Processing on the Grid” in *Proceedings of the III International Conference on "Distributed computing and Grid technologies in science and education" (Grid2008), Dubna, Russia, 30 June - 4 July, 2008,* JINR, 2008, pp. 54-60.

[10] Vaniachine A. “Scalable Database Access Technologies for ATLAS Distributed Computing,” ATLAS Note ATL-COM-SOFT-2009-011; ANL-HEP-CP-09-085, Geneva, CERN, 2009.

[11] Barberis D. et al. “Strategy for Remote Access to ATLAS Database Resident Information,” ATLAS Note ATL-COM-SOFT-2009-017, Geneva, CERN, 2009.

[12] Basset R. et al. “Advanced technologies for scalable ATLAS conditions database access on the grid,” *J. Phys.: Conf. Ser.* **219**: 042025, 2010.

[13] Blumenfeld B. et al. “CMS conditions data access using FroNTier,” *J. Phys.: Conf. Ser.* **119**: 072007, 2008.

[14] Panzer-Steindel B. “Data integrity,” CERN/IT Report, 8 April, 2007.

PROBLEMS OF CLOUD COMPUTING UTILIZATION FOR INTELLIGENT KNOWLEDGE TRANSFER

D. G. Velev[1], P. V. Zlateva[2]

[1] *Department of Information Technologies and Communications,*
University of National and World Economy,
Bulgaria, 1700 Sofia, Studentski grad "Hr. Botev", UNSS, dvelev@unwe.acad.bg
[2] *Institute of Control and Systems Research, Bulgarian Academy of Sciences,*
Bulgaria, 1113 Sofia, Acad.G.Bonchev Str., Bl.2, P.O.Box 79, plamzlateva@abv.bg

This work presents a brief analysis of possible problems in cloud- based knowledge transfer. The transition to Cloud computing provides for a higher level of cooperation, interaction and possibilities for knowledge and expertise transfer for the research organizations, universities and business as a whole. The advantages and disadvantages of cloud computing in knowledge transfer are described, followed by a short discussion of major identified problems. The results can be used for the development of software tools and information systems for intelligent knowledge transfer in the corresponding research areas of interest.

1. Introduction

The efficient implementation of the European strategy for economics support, based on knowledge, demands the integration of modern information and communications technologies in the knowledge transfer process.

The information and communication technologies for knowledge transfer (acquisition, processing, storage and publishing) represent important components of the information systems for the support of educational technologies and decision support systems in the area of management. The software tools for the knowledge transfer support belong to that class. These technologies integrate into a unified information system all the intelligent informational resources of the experts and all intelligent software tools for the user activity support. The users are students, lecturers, trainers, managers of educational and management systems. The combinations of such software and information resources define the term intelligent systems for knowledge transfer. These systems include knowledge models in this area too.

The rapid development of the informational tools for supporting the user activity in these processes provides for the "elimination" of the expert's knowledge from himself i.e. each user can use knowledge without the presence of an expert. Nevertheless the computer-based tools for examining the knowledge assimilation are not developed fast enough. The limited intelligent options of the information systems for the provision of the interactivity between experts and users do not overcome the restrictions of the standard interfaces of the operating systems and integrated dialogue environments.

The scientists, working on the problems concerning the development of models, software tools and information systems for intelligent knowledge transfer, which include also automated learning systems, intelligent learning systems, decision support systems, consider that models for data access by cognitive methods must be developed. This point of view is based on the need for modeling the behaviors and the intelligent activities of the experts and users in order to develop efficient tools for communication, monitoring and control the knowledge transfer process [1, 2].

Thus the research, targeted at the development of models, software tools and information systems for intelligent transfer of knowledge complies with the contemporary scientific and practical progress of the mathematical and software provision of the modern information systems.

The development of models, software tools and information systems worldwide is accompanied with a significant workload in combination with increase in their complexity. Most software tools are characterized with features, depending on the corresponding knowledge domain.

It is necessary to point that the Cloud computing describes a new delivery model for IT services based on the Internet and involves the provision of dynamically scalable and virtualized resources as a service over the Internet. Cloud computing providers deliver common business applications online which are accessed from another web service or software like a web browser, while the software and data are stored on servers [3, 8].

The transition to Cloud computing provides for a higher level of cooperation, interaction and possibilities for knowledge and expertise transfer for the research organizations, universities and business as a whole.

The aim of this work is to analyze possible problems of Cloud computing utilization in information exchange and on that basis to propose principles for intelligent knowledge transfer in areas such as education and business. The results can be used for the development of software tools and information systems for intelligent knowledge transfer in the corresponding areas of interest.

2. Knowledge transfer information systems

In recent years the Service-Oriented Architecture (SOA), Software as a Service (SaaS), Internet networks and Internet portals, decision support systems, information systems for knowledge transfer, etc. develop and interact with each other. These activities are transforming in anew modern form of existing the information technologies, known as Web 2.0.

The Web 2.0 concept defines the methods and means for the penetration of the corresponding technologies and tools in business and education for facilitating the mutual activities among employees, partners and users in the establishment of common information networks in a given sphere and for transferring information and knowledge for competitiveness improvement.

Major factors, defining the integration of the technologies into a new business model, could be summarized as follows [4, 6]:

- Growing attention to new ways for improving the operational efficiency in research, educational and business organizations;
- Global continuous accelerated growth of data access and participation in common and multifunctional technologies that brings to the accumulation of new information and knowledge;
- Developments concerning the newest technologies and choice of communication tools, utilization and operation;
- Continuous process for business optimization, leading to transformations and outsourcing. This leads to integration of computer and information systems that support business operations and their continuous expansion in the organization' limits through SOA;
- The Service-Oriented Architecture is transforming into a light-weight, pragmatic and Web-oriented technology;
- Security will remain a major concern since Web 2.0 applications expose each organization to risks;
- The unstructured information from blogs and wikis will grow rapidly and will lead to an increased need for solutions for accumulation, integration, extraction and publication of information and knowledge;
- The information and communications technologies will need a financial support for the of next-generation infrastructure for the development of SaaS, SOA, mashups, etc.
- The emergence of powerful user platforms, based on Rich Internet Applications (RIA).

The knowledge transfer problem is completely new not only towards the information technologies, but as well as towards the organizational and informational infrastructure of the modern companies and research institutes. The Web 2.0 concept and early developments mark their initial stage back in 2007. The leading software companies nowadays are USA-based, such as IBM, Oracle, Microsoft, SAP, etc. Unfortunately Europe traditionally lags behind in the development and utilization of the latest scientific and practical achievements in the information and communication technologies in comparison with the USA.

3. Cloud computing

The Cloud computing describes a new supplement, consumption and delivery model for IT services based on the Internet, and it typically involves the provision of dynamically scalable and often virtualized resources as a service over the Internet. It is a byproduct and consequence of the ease-of-access to remote computing sites provided by the Internet.

The term "cloud" is used as a metaphor for the Internet to depict the Internet in computer network diagrams as an abstraction of the underlying infrastructure it represents. Typical cloud computing providers deliver common business applications online which are accessed from another web service or software like a web browser, while the software and data are stored on servers [5].

Gartner, the most influential ITC consultancy agency in the world, gives the Cloud computing a second place according to the possibilities it provides the business with regarding information support, especially in the conditions of global economic and financial crisis. The technology is absolutely new and it is in the process of standardization. It is expected the financial support for its implementation in business to exceed tens of billions dollars till 2012, and its ROI to overpass them multifold [6].

The Cloud computing services and possibilities will gradually transfer into a model, which many companies and organizations in the sphere of education and training, healthcare, manufacturing, scientific research, etc. will use as means for efficiency and competitiveness provision.

The direct advantages of Cloud computing introduction in knowledge transfer can be summarized as follows [4, 7]:

- Lower implementation costs;
- Lower capital cost;
- Enhanced market offerings;
- The technology offers the addition of new modules without the need to invest in new hardware or software;
- Additional services – secured environment, data replication elimination, faster transfer of data, information and knowledge, elimination of experts.

The transition to Cloud computing services provides for a higher level for mutual activities and many possibilities for knowledge transfer and expertise for the universities, scientific research units and business as a whole. All parties in the knowledge transfer process can work and communicate from any place by using applications in the cloud. Although the users do not need to use any special devices to access cloud services, units such as notebooks, netbooks, PDAs, smartphones, etc. will represent an adequate technical possibility for applying the corresponding cloud services.

The Cloud model provides for enhanced possibilities for conducting scientific research. Nowadays using Cloud computing, lecturers and researchers will be able to scale their applications and tasks to comply with their corresponding needs.

Despite the fact there is a limited set of Cloud services, the research organizations, educational and training institutions, universities and business will turn to the utilization of such Cloud services such as e-mail, mutual operation, operational efficiency increase, implemented in a secured environment. The users will be offered flexibility in the management of their own information (documents, graphics, audio, video, etc.), access control to other traditional and Web based information resources, governed by the idea for intelligent knowledge transfer [8].

4. Possible problems in cloud-based knowledge transfer

It is necessary to point that the Cloud computing model have certain disadvantages nevertheless the fact that the Cloud computing offers a cost-effective solution to provide services, data storage and computing power to an increasing number of Internet users without financial investments in physical machines that need to be maintained and upgraded on-site. Unlike traditional software packages that can be installed on a local computer, backed up, and are available as long as the operating system supports them, cloud-based applications are services offered by companies and service providers in real time [9].

The main problems of Cloud computing utilization for intelligent knowledge transfer can be identified as lack of reliability and availability, privacy and security, loss of control over operations and data [10, 12].

The increasing new technology developments in cloud and virtualisation environment create definite security threats. Cloud computing and virtualization, while offering significant benefits and cost-savings, move servers outside the traditional security perimeter and expand the zone for cybercriminals. They can either manipulate the connection to the cloud or attack the data centre and the cloud itself. As a consequence the user's data can be revealed.

The introductions of e-mail, and the explosion of social media, their growth and adoption rates have been slowed by initial fears over security concerns and the loss of control over data and operations. Certainly, privacy and security questions will need to be addressed as institutional data and applications move into a cloud environment [11].

If given organizations try to cut their ICT expenditure, they turn to external providers to host applications on their behalf. These cloud computing services are entrusted to third parties. At the same time the organizations that are increasing their dependence on other organizations for the provision of their IT services, have become a constant target of new cyber attacks [10, 17].

Privacy - Privacy settings for user accounts are established by the individual service. It is user's responsibility to ensure that his account content is locked down or made available the way he prefers.

There are no guarantees that a certain service will continue. Applications in the cloud are provided by companies in order to profit: sometimes they are still in beta, sometimes they are from start-ups funded by venture capital which may run out, sometimes it's decided they are not viable. That means there are no guarantees used services will continue to exist.

Lack of interoperability and transferability - Cloud computing does not promote aggregation of content as a number of separate services are likely to be used to host and store information and content. Additionally it does not facilitate the establishment of interoperable school administrative and learning systems. Individual cloud providers may have little or no interest in interoperability or transferability as it's in their interests to keep you tied to their service.

Terms and conditions - Some cloud services reserve intellectual property (IP) rights over everything that is posted so may lead to lose IP in critical materials or to collections of materials. This may be in conflict with employment contracts, organizational policy, and have implications for intelligent knowledge transfer [8, 10].

Content issues - Many cloud services are supported by advertising, and/or they may have unsuitable content. It is important that the uncontrolled content is considered as part of the risk mitigation strategy.

Intellectual property Restrictions - Some apps let you restrict access from certain IP addresses. This might work if you're prepared to forego the benefits of device independence, but to my mind this is one of the great advantages of working in the cloud [12].

Backups – Cloud-based knowledge transfer should not rely on the cloud for backups. Instead it must keep backup copies so that in the event of a sudden outage or service closure to be still able to access its content and give it to the researchers.

Data leakage threat - The increasingly inter-connected educational environment and prevalence of externally provided services is reflected by a growing data leakage threat. That threat is driving an increased demand for assurance over third parties [13]. The standard ISO 27001 is becoming a common standard for compliance. Many cloud providers are being asked to demonstrate compliance with the standard. User postings to social networking sites pose a new data leakage risk. At the same time social networking is increasingly important to intelligent knowledge transfer. Research organizations must allow effective use of the Internet, but reduce inappropriate use. Use of software to block access to inappropriate websites is slightly up on two years ago. Web access logging and monitoring is relatively static. However, more sophisticated use is being made of these tools than in the past.

Most researchers have a personal online identity already [14]. They have a personal email addresses, blogs, Facebook and LinkedIn accounts, etc. If their social media accounts are allowed to connect to the research organization cloud-based apps and perform the authentication process, it could mean that a user would be able to access a certain application simply by logging into a Facebook or LinkedIn account. A breach in the application's security would then only come at the expense of a breach in the security of a user's personal account. This way the responsibility for maintaining security would be would be shared. As soon as one researcher in an research organization has revealed his account details the entire system is compromised and all the organization's information is open to whoever gets hold of the password and there is no real way of know if or when this has happened.

Perpetuity - Services may allow you to delete the account but may retain all your content forever, and continue to use it in whatever way they wish. This will often be specified in the terms and conditions of service. This means control of your digital identity may be in the hands of others [15,17].

Denial of service – It may happen sometimes that services decide that a certain user have infringed their terms and conditions of service. Usually the services will suspend his account immediately and it will be the user to convince them to allow him to access his account. With international cloud services with hundreds of millions of users, it can be challenging to get them to respond in a timely manner [13, 17].

The easiest way to conduct fraud online is through stealing a valid user name and password. Key logging, phishing, social engineering and network sniffing present a risk or possible impact to Cloud computing applications. According to Jason Hart [15], a former ethical hacker, provides the following six steps for improving cloud security, citing: "Cloud computing security risks can be easily mitigated by implementing already existing solutions and so it is vital that businesses review their security policies immediately if they are to continue to protect their data and assets."

Teach all users safe internet skills. It is essential all users are aware of what the dangers are [16]:

- Perform a detailed vulnerability assessment - and review security policies immediately to ensure that they are adequately protected;
- Ensure anti-virus protection is current and kept up to date on all devices;
- Use a firewall to protect every point in the organization;
- Use VPN or SSL/VPN technology for secure connections and encryption for all information on portable devices;
- Deploy strong authentication for remote users, requiring a strong password, PIN and separate token.

Lower device and computing power costs means research organizations will get more for their infrastructure, but this is likely to be offset by increased costs in wireless capability, connectivity, support and infrastructure. If the software applications and data are all online then an 'always connected' capability will be essential for researchers, as will ubiquitous wireless connections. While cloud computing can solve some organizational IT needs, some services will be better left on local machines and/or running from local servers for performance and privacy reasons.

Integration with existing in-house applications can be difficult as many cloud services are standalone and are provided default. For seamless administrative services and the ability to move and mashup your own data, cloud computing may not be the best solution. Mission critical services with high privacy needs – such as finance and human resources may also be best kept in-house. In summary, a shift to Cloud computing may mean a greater variety of internet-ready devices being used, applications accessed from the web rather than running from local machines or servers, data stored in the cloud, and enterprise applications managed and hosted by third party service suppliers.

5. Conclusions

The cloud-based knowledge transfer model is a relatively new idea in terms of adoption. Many applications such as word processing, spreadsheets, presentations, databases and more can all be accessed from a web browser, while the software and files are housed in the cloud. Research organizations can take advantage of cloud applications to provide researchers with free or low-cost alternatives to expensive, proprietary productivity tools. Before full adoption, research organizations must consider key issues, which may include lack of reliability and availability, privacy and security, loss of control over operations and data and other problems. It is expected the cloud-based knowledge transfer to undergo many changes regarding miscellaneous issues, risks, best practices and standards.

References

[1] Al-Zoube M. E-Learning on the Cloud. International Arab Journal of e-Technology, 2009. V. 1, N. 2. P. 58-64.

[2] Katz R.N., Goldstein P.J., Yanovsky R. Demystifying Cloud Computing for Higher Education. Educase Center for Applied Research. Research Bulletin, 2009. V.l, issue 29.

[3] Wyld D. Cloud Computing 101: Universities are Migrating to The Cloud for Functionality and Savings. Cloud Computing in Universities Today, 2009.

[4] Chorafas D.N. Cloud Computing Strategies, CRC Press, 2010. P. 337.

[5] Miller M. Cloud Computing - Web-Based Applications That Change the Way You Work and Collaborate Online, QUE, 2010. P. 292.

[6] Linthicum David S. Cloud Computing and SOA Convergence in Your Enterprise, Addison-Wesley, 2010. P. 265.

[7] Katz R. The Tower and the Cloud: Higher Education in the Age of Cloud Computing, A New EDUCAUSE e-Book, 2008. http://www.educause.edu/ thetowerandthecloud

[8] Reese G. Cloud Application Architectures, Oreilly, 2009. P. 206.

[9] Harrison D. Is Cloud Computing a Credible Solution for Education? IT Trends, 2009. http://campustechnology.com/articles/2009/11/12/is-cloud-computing-a-credible-solution-for-education.aspx

[10] McLear J. Heading into the cloud: cloud computing and education, Blogs: educationau, 2009. http://blogs.educationau.edu.au/jmillea/2009/06/23/heading-into-the-cloud-cloud-computing -and-education/

[11] Leesanguansuk S. Social sites, cloud computing pose new threats. Security Report, 2010. http://www.bangkokpost.com/tech/technews/31830/social-sites-cloud-computing-pose-new-threats

[12] Read Write Cloud channel: Cloud Security Using ...Social Networks? 2010. http://www.readwriteweb.com/ cloud/2010/02/cloud-security-using-social-ne.php

[13] Net Security News: Cloud computing and social networking expose businesses to attacks, 2010. http://www.net-security.org/secworld.php?id=9172

[14] O'Reilly E. Social Networking Behind a Firewall, Scalable Collaboration Solutions for Business Application, 2010. http://social-networking-tagging.suite101.com/article.cfm/social-networking-behind-a-firewall

[15] Securitypark: Steps to improve security in the Cloud, 2010. http://www.securitypark.co.uk/security_article264596.html

[16] Tout S., Sverik W., Lawver G. Cloud Computing and its Security in Higher Education, 2009 // Proc. ISECON, vol. 26

[17] Marther T., Kumaraswamy S., Latif S. Cloud Security and Privacy, Oreilly, 2009. P. 335.

ERROR-FREE INVERSION OF ILL-CONDITIONED MATRICES IN DISTRIBUTED COMPUTING SYSTEM OF RESTFUL-SERVICES OF COMPUTER ALGEBRA[1]

V. V. Voloshinov[1], S. A. Smirnov[2]

[1] *Institute for System Analysis RAS, Moscow, Russia, vladimir.voloshinov@gmail.com*
[2] *Moscow Institute of Physics and Technology, 141700, Dolgoprudny, Russia, sasmir@gmail.com*

Error-free inversion of an ill-conditioned matrix is a well known challenging task in computer science. However it can be easily accomplished using symbolic computation techniques available in computer algebra systems. Unfortunately, symbolic computations are time and memory consuming. To address this issue matrix inversion distributed algorithm have been implemented on the base of MathCloud RESTful framework and Maxima (GNU Computer Algebra System) RESTful service. The scenario is based on block decomposition of input matrix and Schur complement. The approach is demonstrated by inversion of Hilbert matrices with exponential growing of condition number w.r.t. matrix's size, up to 500x500. Besides, it was shown that MathCloud framework including workflow editor can be used for rapid prototyping and implementation of distributed algorithms.

1. Introduction

Symbolic computations in distributed computing environment have attracted more and more attention for the last decade [1]. In particular, it enables to perform error-free computation for complex computing science problem e.g. solving ill-conditioned systems of linear equations. This problem is also crucial for parallel computing systems based on "traditional" arbitrary precision float arithmetic [2]. In the previous report [3] on the same subject we presented results of experiments with CAS Maxima (a well-known GNU Computer Algebra System, maxima.sourceforge.net) services implemented by Smirnov S.A. via object-oriented Ice middleware, www.zeroc.com. (The service is available at http://code.google.com/p/remote-maxima). That approach implies that researcher is skilled enough to implement "scenario-specific" workflow manager application himself, e.g. in Java. This feature reduces possible usage of that implementation of CAS Maxima service.

Now we present a workflow scenario of distributed, and partially parallel, error-free inversion of ill-conditioned matrices in MathCloud framework, [4], [5], www.mathcloud.org, based on Web 2.0 paradigm (HTTP, JavaScript Object Notation for data representation, Reach Web Application as user interface). The scenario is based on block decomposition of input matrix and Schur complement. The approach has been announced in [3] and also includes parallel multiplication of intermediate rectangular matrices by their block decomposition. Another CAS Maxima RESTful service, [6], has been developed and deployed on multi-core desktops. One more goal of the work was to verify usability of MathCloud workflow design and management system for asynchronous and unpredictable in advance computing flow.

2. Block decomposition and Schur complement as reasons for parallelism

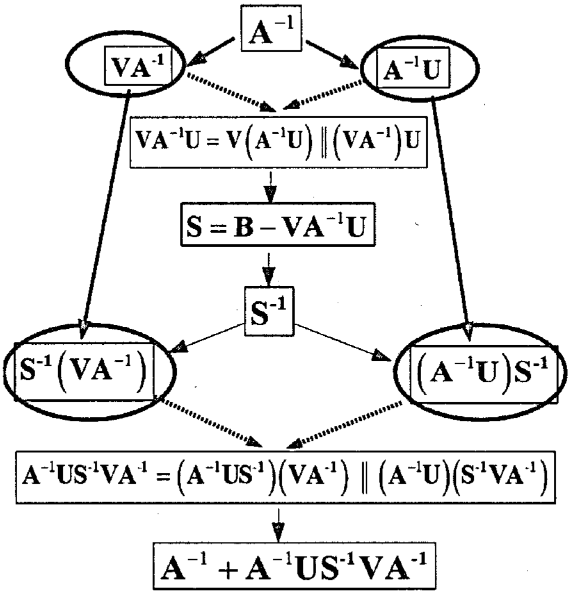
For the distributed scenario we use well-known in the theory of matrices [7] and computer science [8] inversion approach based on «block representation» and the so-called Schur formula (or

[1] Partially supported by the RFBR grant 08-07-00430-a and "SKIF-Grid" project.

Schur complement). Let M be a square $N\times N$ matrix represented as four sub-matrices: A - an upper left, square block, $N_A\times N_A$; B - a lower right, square block, $N_B\times N_B$, where $N_B=N-N_A$; U - an upper right, rectangular block, $N_A\times N_B$; V - a lower left, rectangular block, $N_B\times(N-N_A)$. Then, see expressions (1), we can define a square, $(N-N_A)\times(N-N_A)$, matrix $\mathbf{S}\overset{\text{def}}{=}\mathbf{B}-\mathbf{V}\cdot\mathbf{A}^{-1}\cdot\mathbf{U}$, called *Schur complement of the block A of matrix* ***M***. Well-known statement (see [7]): if M^{1} exists, than (possibly after some cells permutations) A^{-1} and S^{1} exist, and M^{1} may be subdivided into four sub-matrices of the same sizes (as A, U, V, B) as it is shown below

$$\mathbf{M}=\left[\begin{array}{c|c}\mathbf{A}&\mathbf{U}\\\hline \mathbf{V}&\mathbf{B}\end{array}\right],\exists\mathbf{M}^{-1}\ \Rightarrow\ \exists\mathbf{A}^{-1},\ \exists\mathbf{S}^{-1},\ \ \mathbf{M}^{-1}=\left[\begin{array}{c|c}\mathbf{A}^{-1}+\mathbf{A}^{-1}\cdot\mathbf{U}\cdot\mathbf{S}^{-1}\cdot\mathbf{V}\cdot\mathbf{A}^{-1} & -\mathbf{A}^{-1}\cdot\mathbf{U}\cdot\mathbf{S}^{-1}\\\hline -\mathbf{S}^{-1}\cdot\mathbf{V}\cdot\mathbf{A}^{-1} & \mathbf{S}^{-1}\end{array}\right].\quad (1)$$

This representation of M^{1} enables speed up of its computing on the base of concurrent (parallel) inversions and multiplications of their sub-matrices (of less size than original M). Brief (not all data flows are shown) block-scheme of appropriate scenario is presented in Fig. 1. All arrows correspond to data flow between blocks. There two kinds of arrows. Solid one means that appropriate input data must be delivered to begin the next step, e.g. multiplication $(A^{-1}U)S^{-1}$ may be performed only after product $(A^{-1}U)$ and inverse S^{-1} become ready. Dashed arrows mean that any of inputs enables to begin next step, e.g. any of products VA^{-1} or $A^{-1}U$ enables to calculate $VA^{-1}U=(VA^{-1})U=V(A^{-1}U)$. Those steps that may be performed in parallel are marked out by ovals.



A^{-1}
VA^{-1}
$A^{-1}U$
$VA^{-1}U=V(A^{-1}U)\,\|\,(VA^{-1})U$
$S=B-VA^{-1}U$
S^{-1}
$S^{-1}(VA^{-1})$
$(A^{-1}U)S^{-1}$
$A^{-1}US^{-1}VA^{-1}=(A^{-1}US^{-1})(VA^{-1})\,\|\,(A^{-1}U)(S^{-1}VA^{-1})$
$A^{-1}+A^{-1}US^{-1}VA^{-1}$

A^{-1}	=> $\sim O((^{N}/_{2})^3)$
$VA^{-1}\,\|\,A^{-1}U$	=> $\sim O((^{N}/_{2})^3)$
$VA^{-1}U$	=> $\sim O((^{N}/_{2})^3)$
$B-VA^{-1}U$	=> $\sim O((^{N}/_{2})^2)$
S^{-1}	=> $\sim O((^{N}/_{2})^3)$
$S^{-1}(VA^{-1})\,\|\,(A^{-1}U)S^{-1}$	=> $\sim O((^{N}/_{2})^3)$
$A^{-1}US^{-1}VA^{-1}$	=> $\sim O((^{N}/_{2})^3)$
$A^{-1}+A^{-1}US^{-1}VA^{-1}$	=> $\sim O((^{N}/_{2})^2)$

Fig. 1: Possible parallelism and workload evaluation (for fixed bit length float arithmetic)

Let us present a rough evaluation of potential acceleration if the above scenario is running in parallel computing environment. Suppose that we can use in parallel two computing unit implementing fixed length float arithmetic operations, and these units are supplied with standard numerical linear algebra algorithms (including square matrix inversion via Gaussian elimination algorithm and rectangular matrices multiplication). Then we can evaluate possible speed up of matrix inversion in distributed scenario in comparison with "standalone" one. Remember well-known facts in computer science: nxn matrix inversion via Gaussian elimination requires $O(n^3)$ operations; multiplication of two nxn matrices costs also $O(n^3)$ operation (disregarding asymptotically faster Strassen [8] or Coppersmith-Winograd [9] algorithms). Let N be an even number and $N_A=N/2$, then brief evaluation gives (costs of each steps are at the left side of the Fig.1) that distributed inversion "costs" $\frac{3}{4}\cdot N^3+\frac{1}{2}\cdot N^2$ arithmetic operations against N^3 of "standalone" $N\times N$ matrix inversion. Thus, we have about 3/4 less duration. Unfortunately, all these evaluations are hardly applicable for symbolic computing, because the cost of *symbolic arithmetic* operation depends on number of digits in operands' rational representation. Further only experimental timing will be presented.

The scheme at Fig.1 is recursive, because it may be applied for inversion of intermediate matrices ***A*** and ***S*** etc. Moreover, the scenario includes rectangular matrices multiplications which is well suited for paralleling. By now we use only the last trick. Namely, let's consider two rectangular matrices *X*, $M_X\times C$, and Y, $C\times N_Y$, subdivided respectively into couples of horizontal (X_1, $M_{X_1}\times C$, X_2, $(M_X-M_{X_1})\times C$) and vertical (Y_1, $C\times N_{Y_1}$, Y_2, $C\times(N_Y-N_{Y_1})$) blocks. Then, see (2), the product $\mathbf{X}\bullet\mathbf{Y}, M_X\times N_Y$ consists four blocks of appropriate sizes which may be calculated independently in parallel.

$$\mathbf{X}=\left[\frac{\mathbf{X}_1}{\mathbf{X}_2}\right],\mathbf{Y}=\left[\mathbf{Y}_1 \,|\; \mathbf{Y}_2\right] \;\Rightarrow \mathbf{X}\bullet\mathbf{Y}=\left[\begin{array}{c|c} \mathbf{X}_1\bullet\mathbf{Y}_1 & \mathbf{X}_1\bullet\mathbf{Y}_2 \\ \hline \mathbf{X}_2\bullet\mathbf{Y}_1 & \mathbf{X}_2\bullet\mathbf{Y}_2 \end{array}\right]. \qquad (2)$$

3. Hilbert matrices inversion as test case

Hilbert $N\times N$ matrix is defined as $\mathbf{H}_N=\left\{h_{m,n}\right\}_{m=1,n=1}^{N,N}$ where $h_{m,n}=(m+n-1)^{-1}$. The matrix may be considered as upper-left squire segment of an infinite set of Gram coefficients of the monomial basis in Hilbert space $L_2[0,1]$, because $h_{mn}=\int_0^1 t^{m-1}\cdot t^{n-1}dt$. These matrices are well-known class of ill-conditioned ones with exponential growth of their condition number w.r.t. their size, $cond(\mathbf{H}_N)=\left\|\mathbf{H}_N\right\|\cdot\left\|(\mathbf{H}_N)^{-1}\right\|\sim O\left((2.5)^{4n}/\sqrt{n}\right)$ (exact values are in the left table). Thus, inversion of Hilbert matrices by standard linear algebra algorithms (e.g. Gaussian elimination for LU-decomposition, ***H=L.U***, with subsequent solving of matrix equation ***L.U.X=E***, where ***E*** is identity matrix) becomes impossible for even rather small *N* at computing unit operating with fixed bits length float numbers. It is not so for symbolic computation operating with exact rational numbers $x=p/q$ represented by pairs of natural numbers {numerator, denominator} *{p,q}*.

N	*cond(H_N)*
10	$1.6\cdot10^{13}$
50	$1.5\cdot10^{74}$
70	$5.5\cdot10^{104}$
100	$4.1\cdot10^{150}$
150	$1.2\cdot10^{227}$

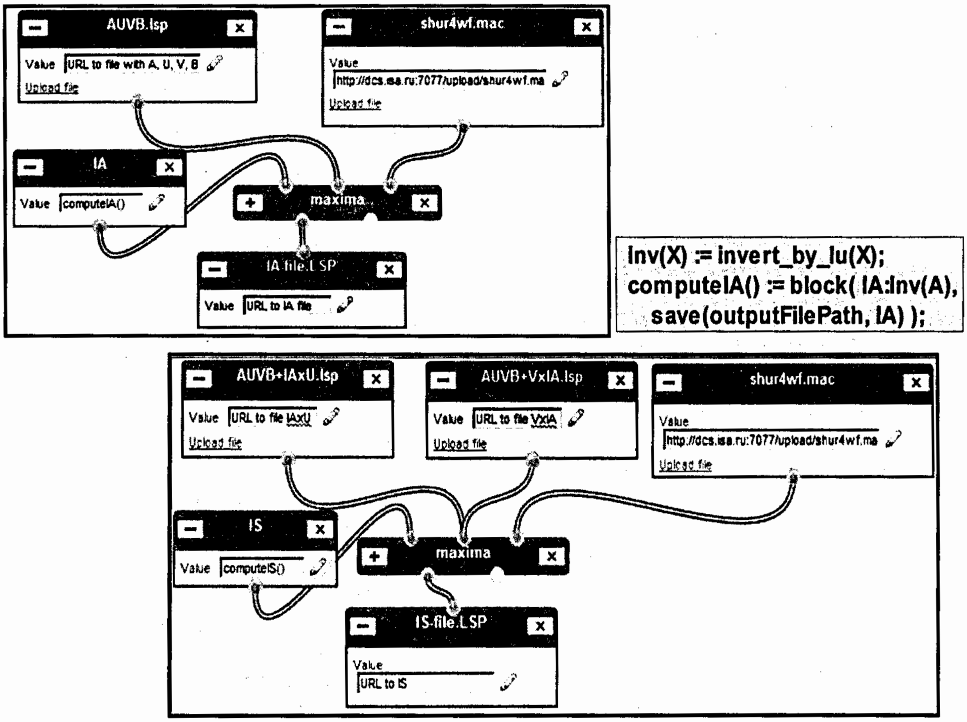
On the other hand, ill-conditioned property means that the length of symbolic representation of inverse matrices' rational coefficients becomes very large, and elapsed times of arithmetic operations become very long. At the left table you can see timing of Hilbert matrices inversion by standard Maxima subroutine *"invert_by_lu"* (solving of ***L.U.X=E*** matrix equation after LU-decomposition). At the second column we present durations of computing to verify that we really get true inverse matrix. Both operations are rather time consuming for rather large *N*. It is significant that Maxima system may be built "upon" various Lisp interpreter (we used Steel Bank Common Lisp, www.sbcl.org), but all of available Lisp-systems are single threaded even on multicore processors. It is the reason to use distributed computing techniques to speed-up. It will be shown below that "parallel" implementation of algorithm presented above in section 2 enables to reduce computing time more than twice. As to sizes of symbolic representation of exact (!) inverse *$(H_N)^{-1}$* it is about 34 Mb for *N=300* (in textual Lisp format) and about 140 Mb for *N=500*.

CPU Intel Xeon E5410 2.33GHz (x8 cores)		
N	**invert_by_lu(H), 1 core, elapsed time, min**	**Checking, H.invH == E?, min**
100	**0.3**	**0.1**
200	**3.4**	**1.0**
300	**15**	**4**
400	**45**	**12**
500	**109**	**27**

4. Computing scenario in MathCloud environment

MathCloud framework [5] has three main features: 1) unified remote access to existing applications or to shell scripts via light weight RESTful (HTTP, JSON) services; 2) Web-based editor for fast development of rather complex computing scenario, which in their turn may be included in other scenarios as a "composed" service; 3) workflow management system enabling execution multiple long-running scenarios. Each RESTful service involved in scenario is represented below (see Fig. 2-5) as rectangular block where top circled "ports" correspond to input parameters and bottom - output ones. A number of data types are supported, but *string* and *file* are enough for "inversion scenario". Data flow are represented by "wires" connecting output and input ports. Note that data-files are sent to recipient-services as URLs and recipient loads these files directly from sender-service independently of workflow management system running on the "third" host.

Two kinds of services have been used for inversion scenario: simple "cat" service concatenating two input files into one output; "maxima" service providing access to an instance of Maxima application to perform computations. Maxima service [6] (Fig. 2) has three input parameters: 1) a "command" (as string) in Maxima script language that should be invoked by Maxima instance; 2) a file with data (in Lisp format) to be used during the command execution; 3) a file with auxiliary (to simplify the "command") Maxima script (e.g. script-functions definitions) required for a given computing scenario. There are two output parameters: 1) a data-file with results of command execution written in Lisp format by standard Maxima script "save" function; 2) a string result of Maxima command.



Value URL to file with A, U, V, B

Value http://dcs.isa.ru:7077/upload/shur4wf.ma

Value computeIA()

Value URL to IA file

Value URL to file IAxU

Value URL to file VxIA

Value http://dcs.isa.ru:7077/upload/shur4wf.ma

Value computeIS()

Value URL to IS

```
Inv(X) := invert_by_lu(X);
computeIA() := block( IA:Inv(A),
    save(outputFilePath, IA) );
```

```
computeIS() := block(
  VxIAxU : If (member('VxIA, values)) then VxIA.U else V.IAxU,
  S : B-VxIAxU, IS : Inv(S), save(outputFilePath, IS) );
```

Fig. 2: Examples of "elementary" blocks of inversion scenario in MathCloud workflow and corresponding Maxima script functions

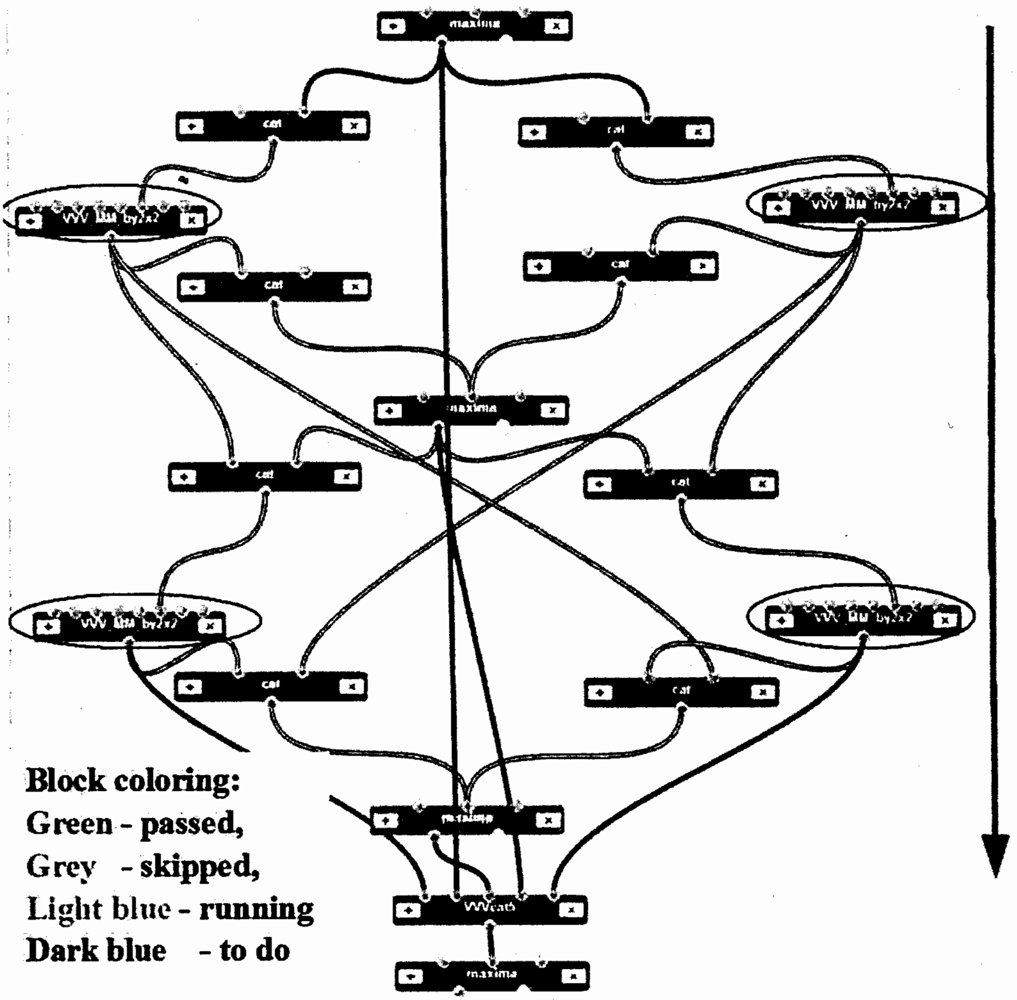
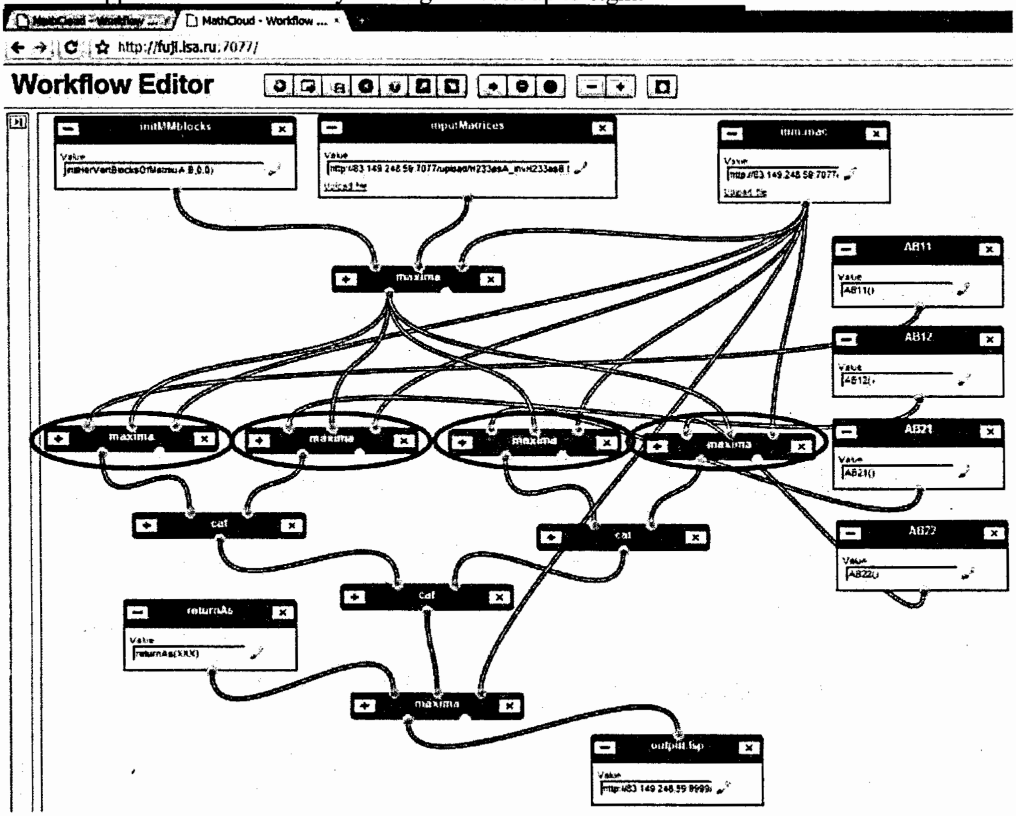


Fig. 3: Main inversion scenario in running mode (all input/output blocks are hidden)

In the Fig.3 you can see (running) implementation of "theoretic" scenario (Fig. 1) in MathCloud workflow editor (all input parameters i.e. commands and files are hidden of). Detailed view of top elementary block (A^{-1} computing) with corresponding Maxima-scripts is presented at the left of Fig. 2 (result is uploaded by "save" command as Lisp "IA" variable). At the right you can see the same concerning "central" block (S^{1} computing). Note that here we demonstrate non-trivial behavior of scenario (two input wires to one "file" port), when presence of any of products VA^{-1} and $A^{-1}U$ in input file enables to compute Schur complement S. During execution the first coming data is used and the another possible workflow remains inactive (see gray skipped "cat" blocks in the Fig. 3).

The "main" scenario presented in the Fig. 3 has four "matrices multiplication" blocks (marked out by ovals) which may be run in parallel. Moreover, see explanations before formula (2). Each of those multiplications may be parallelized. Possible parallel scenario of matrices multiplication is presented in the Fig. 4. Here four Maxima applications run in parallel. This composed scenario has been used in main scenario just as another service ("VVV_MM_by2x2"). Thus, the number of

Maxima applications concurrently running has been up to eight.



MathCloud - Workflow ...

http://fuji.isa.ru:7077/

Workflow Editor

initMMblocks Value: initHerVertBlocksOfMatrixA 8,0,0)

inputMatrices Value: http://83.149.248.59:7077/upload/H233asA_invH233asB

mm.mac Value: http://83.149.248.59:7077/

AB11 Value: AB11()

AB12 Value: AB12()

AB21 Value: AB21()

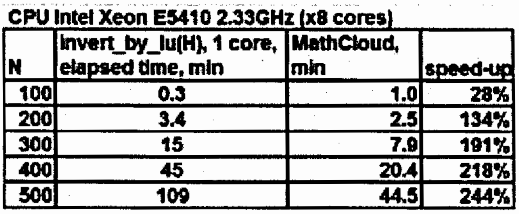
AB22 Value: AB22()

returnAs Value: returnAs(XXX)

output.lisp Value: http://83.149.248.59:8999/

Fig. 4: Workflow scenario of two matrix multiplications by two-block decomposition (2)

In the table in the Fig.5 we compare elapsed time of H_N inversion by "standalone" Maxima process (2nd column) with duration of MathCloud workflow (Fig. 3) execution (3d column). Elapsed time in the 3d column includes all MathCloud "overheads" (start/stop REST-services, data file exchange). In these experiments all Maxima services run at the same 8 core host, and MathCloud workflow management system run at another host.



CPU Intel Xeon E5410 2.33GHz (x8 cores)

N	invert_by_lu(H), 1 core, elapsed time, min	MathCloud, min	speed-up
100	0.3	1.0	28%
200	3.4	2.5	134%
300	15	7.9	191%
400	45	20.4	218%
500	109	44.5	244%

Fig.5: Performance of H_N inversion in MathCloud framework

Conclusions

1. Distributed computing scenario for error-free symbolic inversion of ill-conditioned matrices has been implemented in MathCloud framework and tested on Hilbert matrices.
2. Performance of the proposed approach becomes good enough for really hard computing tasks (e.g. for inversion of Hilbert, *N×N*, matrices, with *N* more than 300) when computations exceeds overheads of MathCloud workflow management system and data exchange.
3. Usability of MathCloud workflow editor and management system including creation of complex (composed) scenarios have been verified.

References

[1] Hammond K., Al Zain A., Cooperman G., Petcu D., and Trinder P. SymGrid: a Framework for Symbolic Computation on the Grid. In Proc. EuroPar'07 — European Conference on Parallel Processing, LNCS, Rennes, France, 2007. Springer, p. 457-466.

[2] Germanenko M.I. Error-free Rational Calculations Software and Application for Solution Of Linear Systems // Vestnik of Lobachevsky State University of Nizhni Novgorod, 2009, N 4, c. 172-180 (in Russian).

[3] Voloshinov V.V. Symbolic Inversion of Ill-conditioned Matrices in Grid-environment of services of access to the Computer Algebra System Maxima. In: Proc. of Int. Conference "Mathematical Modeling and Computational Physics" (MMCP'2009), July 7 – July 11, 2009, Dubna, Russia, pp.175-176 (in Russian), http://mmcp2009.jinr.ru/pdf/Voloshinov.pdf (slides in English).

[4] Astafiev A.S., Afanas'ev A.P., Lazarev I.V., Sukhoroslov O.V., Tarasov A.S. Scientific service-oriented environment based on Web-technologies and distributed computing // Scientific Internet Services: Scalability, Parallelism, Efficiency: Proceedings of All-Russian Supercomputing Conference (September, 21-26, 2009, Novorossisk, Russia). – Moscow: MSU publisher, 2009. – p. 524 (in Russian).

[5] Lazarev I.V., Sukhoroslov O.V. Implementation of Distributed Computing Scenarios in MathCloud framework. // Problems of distributed computing/ Ed. S.V. Emel'yanov , A.P. Afanas'ev. Proceedings of ISA RAS, vol. 46. - Moscow.: KRASAND, 2009. - pp. 6-23 (in Russian).

[6] Smirnov S.A. On Development of RESTful web Service for a Computer Algebra System in MathCloud Environment // The 4th International Conference "Distributed Computing and Grid-technologies in Science and Education"(GRID'2010), June 28 - July 3, 2010 Dubna, Russia, http://grid2010.jinr.ru/files/pdf/maxima.pdf.

[7] Gantmakher F.R. The Theory of Matrices, 5th Ed., Moscow, FIZMATLIT, 2004 (in Rusian)

[8] Introduction to Algorithms, 2nd Ed. T. H. Cormen, C. E. Leiserson, R. L. Rivest, Clifford Stein. MIT Press, 2001.

[9] Coppersmith D., Winograd S. Matrix Multiplication via Arithmetic Progressions // J. Symbolic Computation, 1990, No. 9, pp. 251-280.

THE DISTRIBUTED INTELLIGENT LEARNING SYSTEM BASED ON COGNITIVE AND REACTIVE SOFTWARE AGENTS

E. I. Zaytsev

Moscow State University of Instrument Engineering and Computer Science
107846, 20, Strominka st., Moscow, Russia
zei@tsinet.ru

New methods of teaching with the use of advanced technologies, in particular, intelligent Knowledge-Based Systems (KBS), take root into educational process of the higher school. Knowledge-Based Systems of educational assigning, such as Intelligent Learning Environments (ILE) and Intelligent Tutoring Systems (ITS), increase efficiency of the labour of the teachers and quality of preparation of the students. Today Distributed Teaching Systems (DTS) are actively creating and use too. Further development of DTS is the integration of distributed information processing and methods of an artificial intelligence with the purpose of development of distributed intelligent systems of educational assigning, such as Multi-agent Knowledge Banks (MKB) [1].

Example of Intelligent Learning Environments is the Static Knowledge Banks (SKB) [2], which form the answers to the inquiries of the users by means of execution of specialized procedures of searching and logic processing of knowledge. In the SKB of knowledge of static data domains, modeled in bases, are represented with the help of special frames-prototypes. Frames-prototypes describe objects and their states, operations and events, and also the processes (understood ordered combinations of events and/or of other processes) realizable with the purpose of the solution of those or other problems. The SKB build the answers to the inquiries of the users about values of the various characteristics of objects and events, about the comparison and analysis of events, detection of communications between events, and also inquiries about synthesis of the plans of operations for the solution of those or other problems, that is about formation of ordered combinations of events ensuring these solutions.

Development of the concept of Static Knowledge Banks is concept of Multi-agent Knowledge Banks, which not only execute functions of Intelligent Learning Environments, but also represent itself as Intelligent Tutoring Systems. MKB actuates general and special knowledge of data domain about learning process and model of trainee, associating knowledge with the reactive and cognitive program agents [3], which realize procedures of processing of these knowledge. Multi-agent Learning System also gives the answers to the inquiries of the users and create the rational strategy of training, improving pursuant to accumulation of the data. In opposition to the Static Knowledge Banks, the success of which bodily depends on motivation of the pupils and their self-discipline, MKB check the students' operations with the use of a dynamical feed-back for an adaptive response on operations of the pupils, and also with postponed feed-back for a periodic evaluation of their knowledge.

For formation of the answers on the inquiries of the users about values of the various characteristics of objects and events, about a comparison and analysis of events, detection of communications between events, and also inquiries about formation of ordered combinations of events ensuring solution of those or other problems, in MKB, instead of one problem solver the multilevel grid of the program agents is used. The program agents realize the reasoning using knowledge of data domains, modeled in their bases, which, as well as in Static Knowledge Banks, can be representation as the frames-prototypes.

The cognitive agents of Multi-agent Knowledge Bank build the answers to the inquiries of the users in an outcome of the specification of properties of essences (events and their subjects), calculus causal, temporary and other relations on the set of essences, and also in an outcome of planning of problem solving. Thus, a calculus of the relations and synthesis of the plan of operations for the solution of some problems execute not only due to fulfilment of production, reduction or transformation rules, but also in an outcome of interaction of agents of Multi-agent Knowledge Bank. The multilevel architecture of MKB proposes to use both horizontal and vertical connections between the agents. At it there are levels responsible for a cooperative behavior, local planning, formation of intentions, perception and fulfilment of operations, reactive behavior and training of the agent. Each agent functions pursuant to the cooperative obligations, which are assigned to the agent by other agents of MKB.

One of the basic advantages of MKB is the weak connectivity dictated by the approach because of the contracts. The contracts represent the specifications of the requirements to interfaces submitted and required by the agent, which realize the protocol of interaction. Partition of the program on the agents, whose interactions are controlled, is corrected by the specified contracts, facilitates identification of natural parallelism, which exists in a context of data domain, and facilitates understanding, how it is necessary to conduct decomposition of activities, which can be executed simultaneously. The development of knowledge banks on the basis of the agents gives the developer a capability to concentrate on correct modelling of problems solvable by the agents, instead of rushing to control parallelism in the program explicitly.

Using for formation of the answers to the inquiries of the users several program agents simultaneously, not only increases performance of a system at the expense of parallelism, but also expands capabilities of knowledge bank on granting to the users of the generalized information. The agents, distributed in units of the local or global computer network, are capable to submit or to recommend learning materials appropriate to outcomes of generalization of preferences, behavior and representations of certain groups of the users of the system. The protocols used in Multi-agent Knowledge Bank, should actuate a capability of a realization of mobile processes, the communications between which can be interrupted, and they are restored.

The realization of the mobile program agents allows a system to redistribute dynamically a computing load, depending on a condition of a network. If the computing on one of units has become not effective, the program agent can suspend the activity, to move on the less loaded computer and to continue activity on it. The mobile agent can sequentially visit computing units, interesting for it, or clone a set of the derived agents, which will be executed in parallel. Thus, the mobile agents should support "strong" mobility model, at which together with a segment of a code, also move the descriptor of process. This model allows the working process to execute from that place, on which this process was suspended before transferring on another computer.

At presence in computing units of the special software, called as an agent's platform, the mobile agents can work on different hardware under the control of various operation systems. Agent's platform answers for the life support of the agents and represents a middleware, which is between the agents and operation system. The main functions of an agent's platform consist in control of the agents, maintenance of message passing between the agents, in searching of the agents and data about them inside a system, support the ontology. Agent's platform allows to transmit, to receive, to register the agents, provides safety of a unit and stability, that is ability of the agents to restore the condition after abort.

Important difference of Multi-agent Knowledge Bank is neural-logical model which underlies MKB and integrates logical (based on knowledge) and connective (neural networks) approaches in an artificial intelligence. In the neural-logical model of MKB a gears of logic reasoning, models of artificial neural networks and methods of information processing based on fuzzy logic are integrated. Fuzzy logic is used both in the gear of nonmonotonic reasoning of the cognitive agents, and in a structure of artificial neural networks of the reactive agents. Due to this, the agents of MKB are capable to decide poorly formalizable problems in open, dynamic problem areas, in which data and knowledge circumscribing essences and the communications, as a rule, are incomplete, contradictory, inexact and indefinite.

The reactive agents are capable to extract knowledge from acting samples, interpreting them as learning samples, and also to build and to realize the fuzzy queries to a knowledge base. The cognitive agents have not full knowledge of the environment and have only partial representation about a problem. They execute inexact, presumable reasoning, which is subject to change at obtaining by the agent of the additional information incompatible to obtained earlier inferences (beliefs revision), and also at change of model of the world in an outcome of updating of beliefs of the agents (beliefs update). The cognitive agents use special gears permitting them to operate by indistinct concepts and to realize fuzzy forward-chaining reasoning or fuzzy backward-chaining reasoning. At a forward-chaining fuzzy reasoning the facts of a knowledge base of the cognitive agents will be transformed to particular values of membership functions of condition of fuzzy productions and find values of membership functions of conclusion on everyone from fuzzy rules. The process of backward-chaining fuzzy reasoning consists of a substitution of separate values of membership functions of inferences and finding of membership functions of conditions, which are received as next subgoal, and further can be used as a membership function of new inferences.

The Multi-agent Knowledge Banks' creation is a challenge, which requires experience of designing, conceptual realization and engineering solutions in such areas as representation and processing of knowledge, network communications, artificial neural networks and fuzzy logic. Designing and realization of the Multi-agent Knowledge Banks considerably become simpler at the use of special tools for support of process of creation of copies and assembly of the learning agents, for automatic generation of partial or full realizations on the basis of the specifications. Specialized libraries and the tools of development of multi-agent systems of educational assigning, similar problem-oriented AgentITS (MSUIECS) environment, support processes of designing and realization of the learning agents with a specific behavior.

For the description of the program systems the theory of the agents offers such a high level concepts as a role of the agents, knowledge, beliefs, desires of the agents, plans, goals, protocols of dialogue and negotiating. The increase of a level of abstraction facilitates the software engineering, thus, it limits a scope of abstraction and volume of the control of details of a realization, which entrust to the developers. The agents, as the specific abstraction, contain more knowledge of data domain, in comparison with general and, therefore, less knowledge is required from the developer for the solution of a problem. However, more knowledge concentrated in abstraction narrows down area of application. The high level abstraction, to be used multiply, should suppose adaptation through some internal gears of variability or through external adapters. In creation of Multi-agent Knowledge Banks with the use of environment AgentITS at a realization of abstraction of the agent the user has a capability to configure visual representation of the solution, and then to generate a source code, create direct instances of classes and adjust them in appropriate way.

The system AgentITS includes instrumental environment of development of Multi-agent Knowledge Banks and the agent's platform which ensure creation of network connections between the agents, searching of the necessary agents. This toolkit, consisting of the interactive masters and panels of properties is optimized for the creation of intelligent systems of educational assigning, which should execute adaptive training with the use of the personal learning agents. Personalization in teaching is reached at the expense of representation in MKB of the metaknowledge for realization of an individual selection and formation of educational materials. The direct access to educational materials is executed by the agents of resources of teaching. The process engineering of Multi-agent Knowledge Bank allows to move these agents to remote resources and to make the analysis of the received information in parallel on several computing units. The agents of monitoring of individual trajectory of teaching, agents of testing and control of a regularity of operations of learning, agents of a feed-back of MKB with learners and teachers are projected and realized on the MKB development.

References

[1] Zaytsev E.I. Multi-agent Knowledge Banks: architecture and methods of a realization// Proc. of Int. Conference "Fundamental and applied problems of instrument engineering, computer science and economy", 2010, MSUIECS, Moscow, Russia (in Russian).

[2] Mironov A.S. Representation and processing of knowledge in Static Knowledge Banks// Proc. of Int. Conference “Fundamental and applied problems of instrument engineering, computer science and economy”, 2005. MSUIECS, Moscow, Russia (in Russian) P. 137-141.

[3] Zaytsev E.I. Methodology of representation and processing of knowledge in distributed intelligent information systems. Automation and Modern Process Engineering, N. 1, 2008. Moscow, Russia. P. 29-34.

A GENERIC RESOURCE FRAMEWORK FOR CLOUD SYSTEMS[1]

R. Zhelev[1], V. Georgiev[2]

[1]Institute on Parallel Processing, Bulgarian Academy of Sciences
[2]Faculty on Mathematics and Informatics, University of Sofia "St. Cl. Ochridsky"

In this paper we propose design architecture of a generic framework for management and monitoring of diverse types of resources in large-scale distributed systems. The framework is mainly focused to meet the requirements of organizing vast datacenters with the purpose of Cloud servicing. Our motivation comes from improving disadvantages in existing Grid system approaches and meeting new requirements of Cloud systems. We define a framework that represents datacenter resources in a uniform way, allowing generic administration without knowledge of the underlying resource access protocol. The framework design allows the variety of managed resources to be conveniently extended by dynamic deployment of resource provider modules. For the purpose of extensibility, our system is "service-oriented" [1], but we passionately keep away from Web Services and WSRF [2] approaches vastly popular for building distributing systems. We consider them too heavy weight and overly complex, especially adapted to asynchronous communication and event support, which we consider of vital importance for high performance systems, ambitious to cover business competitive SLA for end user servicing. Instead, we build our services upon the so called OSGi platform [3], which provides high dynamics in modularity - loading and lifecycle of modules, lightweight service registry, and in the same time is a specification standard enabling interoperability. The resource framework is flexible enough to employ agent-based [19] [20] and remote (centric) resource provider modules, so that appropriate approaches could be integrated independently for the specific types of resources. We chose a Cluster-to-Cluster management distribution, that is - one management cluster per datacenter, aiming to achieve a best balance between scalability and performance efficiency in administration of intentionally built datacenters as Cloud systems are.

Introduction – Related Work Overview

A base functionality of distributed systems middleware is the organization of existing distributed resources into a consistent system, making them available for monitoring and management by applications and system operators.

Grid Monitoring Architecture

Global Grid Forum (GGF)[7], GMA-Working Group (GMA-WG)[9] defines the Grid Monitoring Architecture[8] as a general recommendation for grid monitoring system. It consists of three types of components (Fig. 1a):

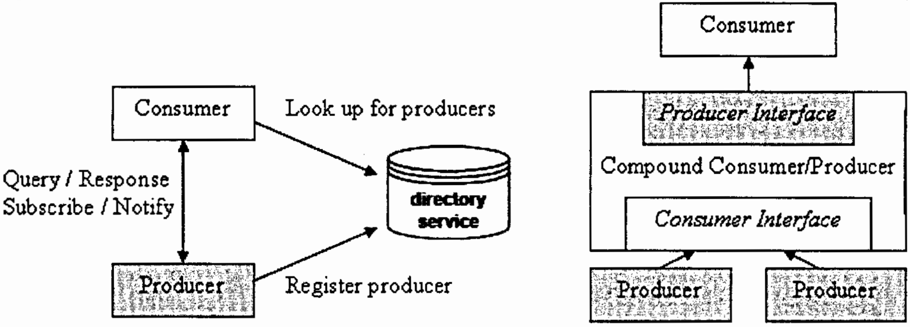


Fig. 1: a) Grid Monitoring Architecture; b) Compound Producer/Consumer

[1] This paper is supported by the Proj. 156/2010 of the Sofia University Research Fund

- Directory Service – lookup registry for available producers of monitoring data;
- Producer – interacts with underlying resource and makes monitoring data available;
- Consumer – receives resource monitoring data.

GMA also defines Compound Producer/Consumers (Fig. 1b) known as well as Republishers, who act as intermediary for building more complex grid-wide infrastructures.

A grid monitoring taxonomy [13], based on the presence and complexity of involved GMA components classifies the monitoring systems into four levels:

Level 0: self-contained systems with no Producer component involved. The Consumer gets monitoring data directly from the underlying resource via the resource specific protocol.

Level 1: Underlying resources are abstracted by Producer components, and Consumers obtain monitoring data via generic API provided by the Producers.

Level 2: There are certain Republishers acting as intermediary between the 'first line' Consumers and Producers. A second level system is differentiated by a semantically equivalent first level system by the distribution of the functionality (that would otherwise be provided by a single Producer) among different hosts.

Level 3: Hierarchy of Republishers take place here to enable highly flexible monitoring. Republishers are configurable and access other Republishers or Producers from the lower level, allowing their organization into a scalable arbitrarily structured hierarchy.

Resource Abstractions

Flexible Grid systems maintain the available resources behind an abstraction that unifies the representation of diverse types of resources having different characteristics. There are two basic approaches for modeling a resource – schema based and object based [10]. In a schema based approach, a description language describes the data that a resource comprises. In an object model, the operations on the resources are defined as part of the resource model. Both approaches are further characterized as fixed or extensible, regarding the ability for resource descriptions to be extended.

Web Services

Web services are typically APIs that are accessed via Hypertext Transfer Protocol and executed on a remote system hosting the requested services. There is often a machine-readable description of the operations offered by the service written in the Web Services Description Language (WSDL) [18]. Web Services are the most popular way for implementing a Service Oriented Architecture (SOA) [1]. SOA-based architectures provide loosely coupled units of functionality (services) that can be used within multiple domains.

The Global Grid Form defines Open Grid Service Architecture (OGSA) [4][5][6] for a service-oriented grid computing environment, based on the Web Services technology. The concept is further adopted by the Globus Alliance [15] [16], and Web Services model is extended to be suitable for representing resources in a distributed environment like Grid. This is not an easy task, since Web Services are stateless in principle, while resources in Grid definitely have states, and hence – they need stateful representation. This leads to specifying the so called Web Services Resource Framework (WSRF), which enables the modeling of stateful resources with Web Services [17]. The WSRF comprises a whole suit of specifications – WS-Resource, WS-ResoureProperties, WS-ResourceLifetime, WS-ResourceGroup, WS-BaseFaults. Along with all newly introduced specifications, WSRF strongly relies on WS-Addressing and WS-Notification for asynchronous communication and event handling.

Open Cloud Computing Infrastructure (OCCI)

Recently GGF has formed an Open Cloud Computing Infrastructure Working Group (OCCI-WG) [21] with the purpose to specify common open architecture for Cloud systems, similarly to the

OGSI and OGSA for Grids. The specifications are still in progress and up to now there is no artifact really ready. But a notable fact is that OCCI-WG focuses only on the IAAS (Infrastructure as a Service) related Clouds, and specifies only a fixed schema for representing hardware/virtual machine resources. Concretely, the specification comprises exact modeling of compute, storage and network resources trying to achieve standardization and interoperability, but is quite limited regarding dynamics and meeting new requirements.

Analysis and Motivation

Disadvantages in existing approaches

Although the usage of Web Services and WSRF for modeling of resources is considered to be a state of the art (up to now-days) in terms of using the latest modern technology, we consider usage of Web Services too heavy weight and overly complex for employment in resource modeling. Adapting a stateless in nature Web Services (based on request-response model) to support stateful representations, asynchronous communication and event-driven notifications is done through overweighting the model with additional complex specifications. Adding to this the heavy weight protocol of SOAP and REST (basically used to implement remote access to Web Services) also overburdens the architecture and hits down the performance.

Cloud system requirements

Unlike the Grid systems, Clouds are characterized by some principle differences that imply new requirements to the resource management:

- Single Ownership and Central Administration – while Grids encompass different VOs with respective resource domains sharing only part of their resources into the Grid, Cloud systems have single Cloud owner - usually one business company owning the datacenter(s) that requires total control over the existing resources;
- Intentionally built datacenters – the volume of resources in Clouds is precisely known or at least estimatable. Resources are organized in an optimized environment, demanding adequate topology for management distribution, that provides best efficiency with regards to the Cloud physic specifics;
- Tasks in Grid and Cloud – Clouds are serving users making them share common resources or services in isolation. In computational grids, tasks are large distributed calculations – DNA analysis, processing of large volume text databases [14], etc. where a user can occupy lots of (potentially all) existing resources, causing other users to wait for their availability. Task queuing functionality is not important for Cloud systems. Instead, Cloud tasks could be limited to management operations upon the resources – adding virtualizations, re-provisioning of new applications, reconfiguring applications to serve new tenant users and so on;
- Predefined applications - although submitting a task in Grid is also related to provisioning of software (i.e. the client execution program), this software is not preliminary known to the Grid systems, while in Clouds the provisioned software is always well defined. Clouds are working with certain set of VM images, applications, predefined services, well known configurations, etc. Working with well known applications gives us the advantage to describe them with metadata and administer software components more effectively.

Motivation

Disadvantages we have identified in existing approaches as well as different requirements imposed by Cloud servicing systems give us the motivation to design a new framework for administering of resources in Clouds.

Intentionally built datacenters providing internally optimized environment motivate the building a Cluster-to-Cluster management topology, placing one management cluster in each datacenter. Each cluster's volume (in terms of number participating hosts) could be of arbitrary size calculated against the volume of handled resources in the datacenter.

The service oriented approach defined by OGSA and adopted by Globus, influences with its principles - distinct entities providing service to one another, but we will abandon the heavy weight usage of Web Services and will build our service components on the top the OSGi (Open Service Gateway Initiative) platform [3], chasing after higher dynamics and efficiency. We are also defining our own resource abstraction that is more suitable for meeting the Cloud systems use cases. Since tasks in Clouds are mainly operations upon the resources, we will define the resource abstraction in the object-based manner, describing not only schema for the resource states, but also the operations available on resources.

Generic Resource Framework – Distributed Topology

Our distributed topology, as well as the load balance scheme is described in details in our previous paper [22]. In short summary, we assume that that Clouds are typically built of one or more datacenters physically spread in strategic geographical locations over the world. We build a Cluster to Cluster topology - one Management Server (MS) cluster in each datacenter, handling directly the management of target resources. The cluster could be configured of arbitrary size (it terms of number of cluster hosts) with respect to the volume of datacenter resources. Our load balance algorithm distributes the load in the cluster so that every cluster host is responsible for even subset of resources. The algorithm ensures efficient failover when some hosts are not operable and provides abilities to use high performance non-replica caches, ensuring that calls for certain resources will go to the exact place where data is cached. We also define Remote Access Servers (RAS) that provide interface for remote accessing of resources in the Cloud by administrator UI consoles or web browsers. RAS Servers are capable to perform inter-datacenter connections and consolidate the information from all Management Server clusters. We argue that this distribution topology provides best balance between scalability and performance efficiency for the Cloud, benefiting from efficient dynamic load balance of the cluster, and two-level hierarchy branching defined between RAS and MS management roles.

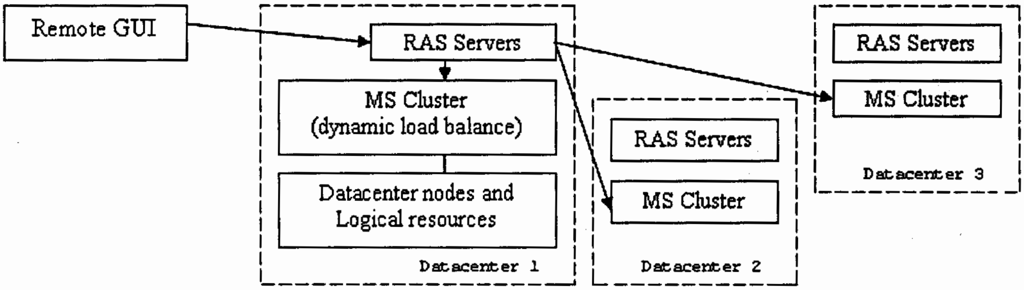


Fig. 2: Generic Resource Framework Distributed Topology

Generic Resource Framework – Resource Abstraction

To unify the control and representation of diverse types of resources we define the so called **Control Unit** abstraction providing uniform interface to arbitrary resources regardless of the underlying access protocol. The principle of this abstraction is defining the state and the operations available on the resource. Resource state is defined as a set of **State Variables** – each State Variable has name and value. The operations available over the resources we refer as **Control Unit Actions.** Actions consist of action name, input and output arguments. Resources that support dynamic creation/deletion upon request, may define specially treated Constructor and Destructor actions in their interface. The set of State Variable names and Actions form the **Control Unit Interface**, while the set of State Variable values form the **Control Unit State**.

Each instance of Control Unit has a type and an ID associated with it. The **type** of the Control Unit identifies its interface. There may be many Control Unit instances with the same type, which means that they have the same set of state variable and action names, but may have different states. In order for management applications to dynamically obtain information about the state variables and actions supported by the Control Units of a given type, we define a **metadata** structure that describes the Control Unit Interface.

Control Units may be arranged **hierarchically** - every Control Unit instance may have one or more child Control Units and one or more parent Control Units. We consider such organization essential for representing more complex resources - devices, hardware and software systems, which may be decomposed to a hierarchy of sub-components, achieving arbitrary level of granularity. Both parent and child Control Units of given Control Unit may be of different types. As an example - we may have a running application instance modeled as Control Unit and dynamic set of tenant users that share this application in isolation - provided as sub Control Unit instances. User preference configurations could be sub Control Units of the tenant users and so forth.

For the purpose of extensibility, it is a responsibility of child unit to "attach" itself under the appropriate parents in the tree. This means that the parent Control Units do not need to know its sub-control units, but the child control unit has to know its parent(s). In this way existing control unit type implementations do not need changes when new type appears in the hierarchy.

In system wide scope, we can say that resource hierarchies are physically or logically hosted on some autonomous datacenter entity that we can call a **Resource Host**. A Resource Host could be for instance the datacenter computational node hosting different resource entities – native processes, user applications, application configurations, log files, etc. Resource Hosts could also be the network router devices if they are subject of remote monitoring and configuration. Hosts are not necessarily physical devices, but could also be completely logical units like user accounts or predefined maintenance procedures currently maintained in the datacenter.

To follow the natural organization of resources and provide efficient management with accordance to their real-life relations, we define in a system wide scope two basically different kinds of Control Units - **Host Control Units** and **Component Control Units**. Host Control Units are the roots of a hierarchy of resources, while Component Control Units are said to belong to a certain Resource Host and potentially have other parent or sub Control Units in the hierarchy. This division is essential for the efficiency of our System since we imply specific treatment semantics to both kinds. Host Control Units are considered to be the connection end-points for management, for instance if the resource host is a remote device in the datacenter keeping physical connection to some Management Server node transmitting events and request-response queries, it is most likely the same connection to be used for most of the Components in the hierarchy underneath. In this sense our load balance algorithm (described in details in [22]) will be based on distributing only the Hosts Control Unit IDs, which means the whole hierarchy starting from the Host Control Unit is distributed at once and handled on the same Management Server cluster node.

Generic Resource Framework – The OSGi Platform

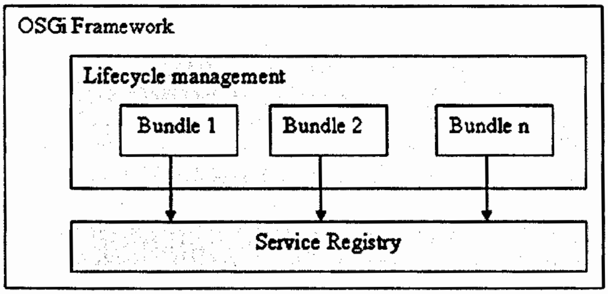


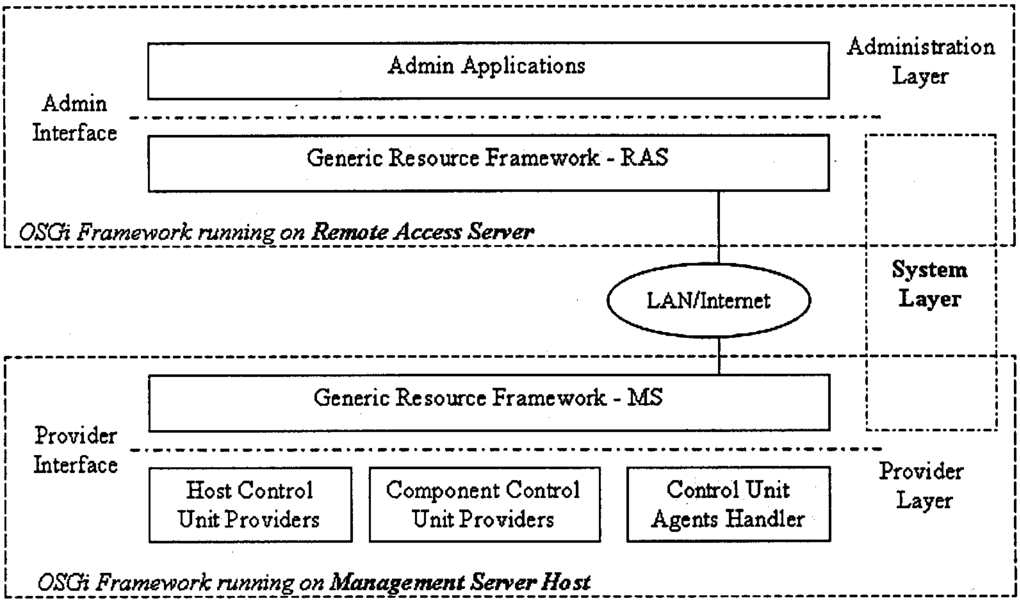
Fig. 3: OSGi Platform

We build our Generic Resource Framework (fig.3) components upon the so called OSGi (Open Service Gateway initiative) framework [3], benefiting from its dynamic lifecycle nature of application modules (called bundles) and the flexible cooperative model for resource sharing – the Service Registry. Key characteristics of the OSGi environment are:

- Class Loading - the OSGi Framework has a powerful and rigidly specified class-loading model. It is based on top of Java but adds modularization. In Java, there is normally a single classpath that contains all the classes and resources. OSGi adds private classes for a module as well as controlled linking between modules;

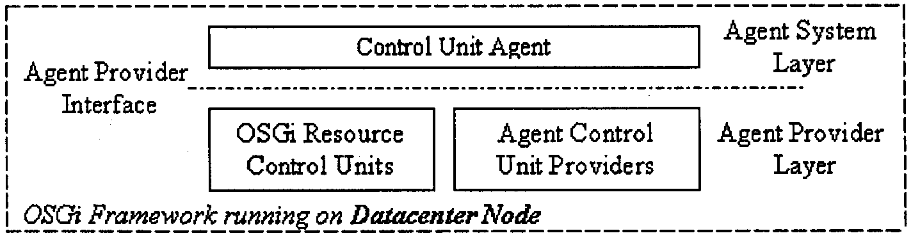
- Application Life Cycle - OSGi defines bundles that can be dynamically installed, started, stopped, updated and uninstalled. This introduces dynamics that are normally not part of an application. The framework provides API for managing the modules (bundles) in runtime, which makes it a good fit for remote management and automations;
- Service Registry - the Service Registry provides a cooperation model for bundles that takes the dynamics into account. Bundles can cooperate via traditional class sharing but class sharing is not very compatible with dynamically installing and uninstalling code. The service registry provides a comprehensive model to share objects between bundles, making the OSGi environment efficient lightweight basement for SOA.

Generic Resource Framework – System Architecture



Admin Applications
Administration Layer
Admin Interface
Generic Resource Framework - RAS
OSGi Framework running on Remote Access Server
LAN/Internet
System Layer
Generic Resource Framework - MS
Provider Interface
Host Control Unit Providers
Component Control Unit Providers
Control Unit Agents Handler
Provider Layer
OSGi Framework running on Management Server Host

Fig. 4a: System Architecture – Remote Access Server and Management Server Host



Control Unit Agent
Agent System Layer
Agent Provider Interface
OSGi Resource Control Units
Agent Control Unit Providers
Agent Provider Layer
OSGi Framework running on Datacenter Node

Fig. 4b: System Architecture – Optional components on the Managed Datacenter Node

In Fig. 4a, we define one OSGi Framework running on every Remote Access Servers (RAS) and Management Server (MS) cluster node. We define the System Layer components situated on the RAS and MS and providing remote communication internally. The System interacts with Administration Applications on one side and Control Unit Providers on the other. All modules –

Admin Applications, System modules and Control Unit Providers are deployed as OSGi bundles on the respective OSGi Framework. Cooperation is done via the OSGi Service registry, where every layer implements the respective interface and provides it to the upper layer. Extending the variety of managed resources is done by implementing a dedicated provider for new arbitrary type of manageable resource. The new provider can be packed and installed as additional bundle. Providers installed on the Management Servers, we will refer as Backend Providers.

Optionally, an OSGi Framework could be installed on some of the managed Datacenter Nodes (where applicable) to enable Agent-based Control Unit implementation of resources – Fig. 4b. Usage of agent-based Control Unit implementations introduces the benefits of less functional and performance limitations due to the federation of sensors placed closer to the underlying resources. Since agent-based model requires Java and OSGi running on the remote node, the agent-based OSGi approach will not always be applicable for employment in all situations. For instance a Cloud system providing IAAS (Infrastructure as a Service) usually works with resources on a lower level, for instance provisioning virtual Operating Systems on the managed nodes, hence there would be no place for Java on that node to handle the providing of resources (HW/OS level VMs). Devices like network routers and load balancers, if also administered globally would have no place for java and will be monitored on a lower level protocol with Backend Providers instead of Agent-based ones. All logical resources like user accounts and maintenance procedures should also be represented via Backend Providers. However, Cloud systems that provide PAAS (Platform as a Service) or SAAS (Software as a Service) would find it quite appropriate to install the OSGi Framework on their datacenter nodes and use an Agent-based approach for administering the applications serving users.

Here is a description of the system architecture layers and participating components.

Provider Layer

This layer provides Control Unit compliant implementation of the resources. A Control Unit compliant resource can be any resource, which state is represented as a set of State Variables and its functionality is represented as a set of Actions. This representation is handled by the Control Unit Providers, which are the only ones that directly deal with the corresponding underlying resources and abstract them in uniform way, suitable for the Generic Resource Framework. There should be one Control Unit Provider corresponding to a Control Unit type. Responsibilities of the Control Unit providers are: to provide resource states upon request, to fire events to the System for all changes that have to be delivered to the administration listeners, to provide metadata descriptions for their Control Unit type, to create and destroy Control Units upon request if supported, and to interpret invocation of Control Unit Actions via the actual underlying resource specific mechanism. The System divides the Control Unit Providers into separate provider definitions - for Component and for Host Control Units, regarding the differences in the treatment of both kinds. In lots of the cases, Providers will need to keep persistent Control Unit State information. Resource information persistency is a major topic in the resource management, and will be subject of our future materials.

Provider Interface

The provider interface represents a respective Java interface that Providers should implement and register as service in the OSGi service registry on the MS cluster nodes. For the purpose of load handling and task parallelization, the interface is consisted of asynchronous methods to prevent bottlenecks and blocked threads during I/O operations potentially performed by providers. Once a Provider is registered, the System grants to it a Callback Interface where the Provider may fire resource events to the System.

System Layer

The System Layer is responsible to organize the resource management in a multi-host distributed environment. Its modules are spanned in the RAS and MS server roles, organizing the load distribution and inter-node communication of the resource management infrastructure. System Layer provides to the Administration Layer a common interface for accessing and controlling of all resources. The administration applications never access providers directly. Instead, the System acts as intermediary, abstracting away common task implementations – consolidating distributed information, resource filtering, event and subscriptions dispatching, load balance, etc., allowing developing of providers and admin applications with minimal effort.

Administration Layer

The Administration Layer covers the Cloud applications and services, interested in management and monitoring of resources. They can list and search resources, retrieve their states, receive notifications for state changes they are interested in, and perform management of resources by invoking resource supported actions.

Administration Interface

Administration interface is Java interface implemented by the System and registered in the OSGi Service Registry of RAS role servers, where it can be accessed by modules of the Administration Layer.

Control Unit Agents Handler

System-implemented generic Backend Provider of Control Units exported on datacenter nodes running an OSGi framework. The module accepts connections from a system-implemented agent bundle – the Control Unit Agent, and both sides transmit control units' data "bringing" the Control Units from managed Datacenter Nodes to the Management Server hosts. The Control Unit Agents Handler module represents the Agent-based Control Units into Backend Control Units making them available to the Generic Resource Framework via the Control Unit Provider Interface on the MS.

Control Unit Agent

System-provided agent bundle, that enables the management of Datacenter Nodes running an OSGi Framework. Responsibility of the Control Unit Agent is to handle the agent-based Control Unit Providers registered in the local OSGi Framework, and to forward the Control Units data to the MSs.

Agent Provider Interface

Similarly to the Provider Interface on the MSs, the Agent Provider Interface is Java interface that should be implemented by Agent-based Control Unit Providers representing local resources as Control Units and should be registered as Service in the OSGi Framework on the Datacenter Node.

Agent Provider Layer

This layer provides Control Unit compliant implementation locally for the Datacenter Node resources. The implementation of local Control Units is naturally lighter, since implementations are not aware of multi-managed-nodes environment; method calls are not asynchronous (as on MS), etc.

OSGi Resource Control Units

A gracefully employed example of the dynamics introduced by OSGi and our Control Unit abstraction: as part of the System, we provide a Control Unit implementation of the OSGi Bundles. The Control Unit representation of OSGi Bundles automatically gets handled by the Control Unit Agent and all local bundles become available on the MS for global administration. That means we can dynamically install start stop and update bundles on the managed Datacenter Nodes, as these supported operations can be modeled via the Control Unit Abstraction. Since modules from the Agent Provider Layer are also deployed as bundles, once a new Control Unit implementation of resources is developed to extend the monitoring functionality, it could be massively deployed on millions of nodes via the bundle Control Units Actions.

Generic Resource Framework – Events Handling

If we keep to the GMA architecture [8] that defines consumers and producers of state change events, we can say that Admin Applications are the consumers in our System, and Control Unit Providers are the producers. The System also takes part here as an intermediary, and plays the role of republisher [13] or a compound consumer/producer [8], i.e. the System acts as consumer to the Control Unit Providers, and as producer to the Admin Applications.

Generally we know two patterns for event notification triggering – Publish/Subscribe and Subscribe/Publish [4]. In the Publish/Subscribe pattern, publishers disseminate information to consumers without any prior knowledge about them. Subscribers specify which published messages are of interest to them. In the Subscribe/Publish pattern, messages are created in response to a subscription. The first approach is convenient for the publishers since they do not need to implement event filtering, while the latter influences efficiency and system performance since event data is not transmitted in case no one is interested in it. Again our System tries to find the balance employing the benefits of both approaches. We define that Admin Applications subscribe for events in the System, providing filters about the changes they want to receive notifications about. The System spans the subscriptions to all physical locations where System components are running – MS Hosts and Control Unit Agents, but does not forward the subscriptions to the Control Unit Providers. Instead, the Control Unit Providers publish all state changes into the System, and the System in turn decides whether notifications should be delivered processing the registered subscription filters. Providers that connect remote physical resources and want to handle subscriptions in order to enable/disable the publishing of events in the 'first instance', may implement additional interface, where the System will supply the subscriptions for resources of the related type.

Our event structures follow the general Control Unit Abstraction and do not need additional metadata. We simply have state change events containing changed State Variable values, or events for added and removed Control Units bearing the respective Control Unit Identification.

Conclusions

Achieved Goals

We have defined a generic resource framework that is suitable for employment in resource management of Distributed Computing Environments. We define the distributed management infrastructure as set of Remote Access Servers (RAS) and Management Server (MS) clusters, making it suitable for Cloud systems leveraged on the top of intentionally built datacenters. The service oriented approach is based on OSGi, allowing lightweight and performant service architecture, while still providing higher dynamics and flexibility. The framework employs own resource abstraction optimized for usage in Cloud datacenters – object based modeling, component hierarchies, host and component Control Unit entities.

GMA Classification Analysis

As mentioned in the "Events Handling" chapter, we may say that our System implements the GMA architecture. If we have to classify our System to the taxonomy described in the related work overview, we can say our System can be determined as a Level 2 implementation of GMA. We have certain intermediaries, but we don't employ dynamic hierarchy of republishes to reach the Level 3 of arbitrary structured hierarchy. For our purpose of administering datacenters offering Cloud servicing, we have found this level of complexity most suitable and optimal in balancing between scalability and performance efficiency. Regarding the Directory Service defined by GMA as lookup registry for available Producers, we can say that unlike the majority of the existing systems where a database is used for implementing such Directory Service [11] [12] [16], listing of our producers (Control Unit Providers) is done via the OSGi Service Registry, where Providers are registered as services.

Future Work

Our framework provides conceptual basements for implementing a wide-featured middleware for Clouds. The framework can be functionally extended with important features having certain dedicated place in Cloud servicing systems.

User Management and Access Control

Since Cloud systems are characterized by single ownership and central administration, and hence – they have a common access-control policy, we can define a system-wide access-control mechanism that is integrated in the base middleware. To make a distinction, Grids encompass different VOs with respective resource domains sharing only part of their resources into the Grid, so different resource providers should consider different private access control policies related to the VO they belong. In our Cloud resource framework, the access-control mechanism could be easily integrated within the Control Unit abstraction and could be handled as common task by the resource framework, freeing resource providers from considering any access control restrictions. For instance, end users (served by the Cloud), could be restricted in a generic way, making them able to access only given Control Unit types (for instance applications providing services that user has paid for). Users cloud be further restricted to access only limited set of State Variables for certain Control Unit types, or to invoke only limited set of Control Unit Actions.

Rule-based Automations

The System has entire potential basis to employ a good extent of self-management features. For instance - restarting of physical nodes on failure, migrating of user applications if nodes do not recover after restart, firing of alarms to mark broken hardware to be checked by technicians. In nowdays the most frequently mentioned property of Cloud System is - 'elasticity'. Clouds must automatically scale up and down in terms of changing the number of application and data instances as demand require. An achievable use case for our System would be an Automation Service part of the Cloud middleware, to be able to process rules of the kind "if you get 10 Control Unit Events that CPU occupation is above 90%, install a new application instance (by invoking respective Control Unit Action)". In result, the user load would be 'elastically' handled by more application instances. The opposite shrinking can also be done by destroying application instances when CPU metrics fall under 10%.

References

[1] Bell, M. Introduction to Service-Oriented Modeling, Service-Oriented Modeling: Service Analysis, Design, and Architecture/ Wiley & Sons, 2008. P. 3

[2] Foster I. (ed), Frey J. (ed), Graham S. (ed), Tuecke S. (ed), Czajkowski K., Ferguson D., Leymann F., Nally M., Sedukhin I., Snelling D., Storey T., Vambenepe W., Weerawarana S. Modeling Stateful Resources with Web Services, 2004. V. 1.1.

[3] Open Services Gateway Initiative. "OSGI Service Platform," Specification Release 4.2, September, 2009.

[4] Treadwell J., and Von Reich J. The Open Grid Services Architecture, Version 1.5, Global Grid Forum, Lemont, Illinois, USA, GWD-I.080, July 2006.

[5] Foster I., Gannon D., Kishimoto H., Von Reich J. Open Grid Services Architecture Use Cases. Information Document, Global Grid Forum (GGF), October 28, 2004. P. 5-11.

[6] Foster I., et al. The Physiology of the Grid: An Open Grid Services Architecture for Distributed Systems Integration, Globus Research, Work-in-Progress 2002.

[7] Global Grid Forum, http://www.gridforum.org

[8] Tierney B., Aydt R., Gunter D., Smith W., Swany M., Taylor V. and Wolski R. A Grid Monitoring Architecture, Global Grid Forum, Lemont, Illinois, U.S.A., GFD.I.7, 2002.

[9] Global Grid Forum, Grid Monitoring Architecture Working Group, http://www.didc.lbl.gov/GGF-PERF/GMA-WG/

[10]Krauter K., Buyya R., Maheswaran M. A taxonomy and survey of grid resource management systems for distributed computing. Software: Practice and Experience, 2002. V. 32. 2. P. 135-164.

[11]R-GMA details are available, http://www.r-gma.org

[12]Volckaert B., Thysebaert P., De Leenheer M., De Turck F., Dhoedt B., Demeester P. A scalable and performant grid monitoring and information framework// International Conference on Parallel and Distributed Processing Techniques and Applications, ISBN 1-932415-48-3.

[13]Zanikolas S.and Sakellariou R. A taxonomy of grid monitoring systems. Future Generation Computer Systems (FGCS) Journal. Elsevier Science, The Netherlands, January, 2005. V. 21, I.1. P. 163-188.

[14]Foster I., Kesselman C., Tuecke S. The anatomy of the grid: enabling scalable virtual organizations// International Journal of High Performance Computing Applications, 2001.

[15]A Globus Primer 4 available, http://www.globus.org/toolkit/docs/4.0/key/

[16]MDS4: The GT4 Monitoring and Discovery System, http://www.globus.org/toolkit/docs/4.0/info

[17]Atkinson M., DeRoure D., Dunlop A., Fox G., Henderson P., Hey T., Paton N., Newhouse S., Parastatidis S., Trefethen A., Watson P., and Webber J. (2004-07-31) (PDF). Web Service Grids: An Evolutionary Approach. UK e-Science Technical Report Series.

http://www.nesc.ac.uk/technical_papers/UKeS-2004-05.pdf.

[18] World Wide Web Consortium. Web Services Description Language. Specification 2.0, June, 2007.

[19]Rochford Keith, Coghlan Brian and Walsh John. An Agent-based Approach to Grid Service Monitoring. Scalable Computing: Practice and Experience V. 8, N. 3, P. 281-290. http://www.scpe.org ISSN 1895-1767

[20]Goel Kunal, Chana Inderveer. Agent-Based Resource Monitoring For Grid Environment, M.E. Thesis, July 2008. Thapar University.

[21]Open Grid Forum, Open Cloud Computing Interface Working Group, http://www.occi-wg.org

[22]Zhelev R., Georgiev V. Resource Administration Load Balance Scheme for Cloud Systems. Information Systems and Grid Technologies, Fourth International Conference, May 2010, (to be published).

MD-GRID NGI: СОВРЕМЕННОЕ СОСТОЯНИЕ И ПЕРСПЕКТИВЫ РАЗВИТИЯ GRID-ТЕХНОЛОГИЙ В МОЛДОВЕ

А. А. Алтухов[3], П. П. Богатенков[1,3], Е. В Васюкова[2], Г. В. Секриеру[1,3]

[1]Институт математики и информатики АНМ, Кишинев, Молдова
[2]Университет «Дубна», Дубна, Россия
[3]Ассоциация RENAM, Кишинев, Молдова

The questions of Grid-technologies using in Moldova and MD-GRID NGI current sate and prospects of development are described. The concept and fundamentals of the grid technologies development directions in Moldova are outlined, as well as hierarchical structure of the organizational model of national of the Grid-infrastructure construction, operation and its the basic elements are described.

Развитие современных распределенных Grid-систем обусловлено широкими перспективами их применения в науке, образовании и других областях человеческой деятельности. Главным образом Grid-технологии предназначены для решения сложных научных, производственных и инженерных задач, которые невозможно решить в разумные сроки на отдельных вычислительных установках. Существующие вычисленные Grid-инфраструктуры ориентированы на реализацию распределенных вычислений для решения сложных научно-технических задач. Информационные Grid-инфраструктуры призваны обеспечить доступ к распределенным неоднородным данным большого объема.

Grid-инфраструктура претендует на роль универсальной вычислительной инфраструктуры для обработки данных, в которой функционирует множество служб (Grid Services), которые позволяют решать не только конкретные прикладные задачи, но и предлагают различные сервисы: поиск необходимых ресурсов, сбор информации о состоянии ресурсов, хранение и доставку данных.

В докладе представлены основные этапы и результаты реализации концепции построения Grid-инфраструктуры в Молдове. Описана иерархическая структурно-организационная модель инфраструктуры и приведены некоторые направления внедрения Grid-технологий.

По своей топологии Grid-система представляется как географически распределенная инфраструктура, объединяющая множество ресурсов разных типов (процессоры, долговременная и оперативная память, компьютерные сети, базы данных и т. п.), доступ к которым пользователь может получить из разных точек, независимо от места их расположения. Концепция Grid предполагает коллективный разделяемый режим доступа к ресурсам и к связанным с ними услугам в рамках глобально распределенных виртуальных организаций, состоящих из предприятий, групп пользователей и отдельных специалистов, совместно использующих общие ресурсы [1, 2].

В каждой виртуальной организации имеется своя собственная политика поведения ее участников, которые должны соблюдать установленные правила. Виртуальная организация может образовываться динамически и иметь ограниченное время существования.

Потенциал использования Grid-технологии уже сейчас оценивается высоко: он имеет стратегический характер, и по своему функциональному назначению в перспективе должен стать вычислительным инструментарием для развития высоких технологий в науке, образовании и других сферах человеческой деятельности [3]. Такие оценки можно объяснить способностью Grid на основе безопасного и надежного удаленного доступа к ресурсам глобально распределенной инфраструктуры решить следующие принципиальные проблемы:

- создание Grid-инфраструктуры высокой пропускной способности из серийно выпускаемого оборудования при одновременном повышении эффективности (до 100%) имеющегося парка вычислительной техники путем предоставления в Grid временно не использующихся ресурсов;
- создание распределенных вычислительных систем способных поддерживать решение сложных научных, инженерных и производственных задач;
- создание широкомасштабных систем мониторинга, управления, комплексного анализа и обслуживания с глобально распределенными источниками данных, повышающих жизнедеятельность научных учреждений, организаций и других структур общества.

Национальная Grid инициатива в Молдове (MD-Grid NGI) была основана в 2006-2007 годах в рамках организационной структуры Ассоциации RENAM (Research and Educational Networking Association of Moldova), которая построила и администрирует общую сетевую инфраструктуру с доступом в Интернет для научно-образовательного сообщества с одноименным названием RENAM. Эта сеть объединяет оптоволоконными каналами связи институты Академии наук Молдовы (АНМ), ведущие университеты и другие научно – образовательные учреждения. Министерство Информационного Развития Молдовы и руководство АНМ, учреждения сферы науки, образования и медицины поддержали инициативу создания Grid-инфраструктуры и Национальной Grid инициативы в Молдове.

Состав и организация процесса функционирования Grid-инфраструктуры основываются на трех базовых элементах: вычислительные ресурсы, высокоскоростное и надежное подключение этих ресурсов к сети, промежуточное программном обеспечении (middleware), которые объединяет эти ресурсы в единый вычислительный комплекс. Участие Молдовы в ряде международных проектов способствовало разработке принципов функционирования MD-Grid NGI и ее базовых элементов [2,4]

Одним из важнейших условий построения Grid-инфраструктуры является наличие скоростного и надежного канала доступа к сети Internet. Для этой цели используется сетевая инфраструктура RENAM, которая объединяет ресурсы АНМ, ведущих университетов и других учреждений сферы науки и образования в единую сеть с доступом в Internet. По своей топологии сеть RENAM представляет собой трех уровневую архитектуру. Первый уровень это локальные сети кампусов, организаций и учреждений. Второй уровень это сетевые узлы оптоволоконные каналы (пропускная способность 1Gbps) для подключения локальных сетей исследовательских институтов АНМ и университетов к сети RENAM. Третий уровень это центральный узел, обеспечивающий работу внешнего магистрального оптоволоконного канала Кишинев - Яссы (Румыния) (пропускная способность оптического оборудования 10 Gbps) с последующим выходом на Транс - европейскую академическую сеть GEANT. Создание и ввод в эксплуатацию в 2010 году оптоволоконного канала для доступа к сети GEANT реализовано в рамках международных проектов SEE-GRID-SCI (финансированный Европейской Комиссии) и NIG 982702 - New RENAM-RoEduNet gateway based on CWDM technologies implementation (финансированный NATO). Это дало возможность увеличить пропускную способность канала с 300 Mbps до 1Gbps и обеспечить условия для последующего увеличения скорости обмена до 10 Gbps.

Главная задача внедрения и развитие распределенных вычислений в Grid-средах является повышение эффективности фундаментальных и прикладных исследований проводимых научно-исследовательскими институтами и университетами и требующих значительных вычислительных ресурсов. Необходимыми условиями для реализации этой задачи являются:

- надежное функционирование и развитие скоростной, защищенной информационно-вычислительной сетевой инфраструктуры (в нашем случае – RENAM);
- развитие и надежное функционирование распределенной высокопроизводительной вычислительной инфраструктуры;

- информационная, алгоритмическая и программная поддержка научно-технических учреждений, создающих прикладные системы, работающие в распределенной вычислительной инфраструктуре;
- надежное функционирование и развитие Grid-сегмента Молдовы как элемента глобальной Grid-инфраструктуры;
- внедрение технологий распределенной обработки информации и доступа к распределенной информации;
- разработка и адаптация существующих методов "гридификации" прикладного программного обеспечения и обеспечение взаимодействия различных Grid-систем.

Создание национального сегмента Grid–инфраструктуры в Молдове преследует цель расширения внедрения информационных технологий в сферах науки, образования и медицины. Приоритетными направлениями этого сегмента являются:

- Интеграция необходимых элементов единого национального Grid-сегмента: коммуникационных, компьютерных и программных ресурсов;
- Внедрение современных технологий в научных исследованиях и образовании, интегрирование научных учреждений в европейское и мировое научное пространство и привлечение молдавских ученых к участию в международных проектах и виртуальных научных сообществах;
- Обеспечение компьютерной обработки больших объемов результатов метеорологических, геофизических, экологических и других измерений;
- Создание условий для внедрения новых современных методов медицинского обслуживания с использованием распределенных баз диагностических данных и распределенной обработки медицинских данных;
- Разработка и реализация системы подготовки и повышения квалификации специалистов для работы в Grid-системах и разработчиков Grid-приложений.

В настоящее время Grid-сегмент Молдовы объединяет вычислительные ресурсы различных учреждений. В Табл. 1 представлены основные параметры Grid-сайтов MD-GRID NGI.

Таблица 1: Основные параметры Grid-сайтов MD-GRID NGI

MD-GRID NGI site	**Available CPUs**	**Available storage**	**Network** (External)
Certified sites			
MD-01-TUM	5 Intel P-IV 3,0 GHz CPUs	320 GB on Storage Element	100 Mbit Ethernet
MD-03-SUMP	5 x CPU AMD Athlon 64 X2 6000+ (3.0GHz)	650 GB on Storage Element	100 Mbps Ethernet
MD-04-RENAM	6 Quad Core Xeon 5130 CPUs	2 TB on Storage Element	100 Mbit Ethernet
MD-02-IMI	12 Quad Core Xeon 5130 CPUs	3,5 TB on Storage Element	100 Mbit Ethernet
Planned to be integrated into MD-GRID NGI			
MD-05-SUM	4x2xAMD 275 Dual-Core 2.2GHz and 3x2xAMD 280 Dual-Core 2.4GHz CPUs	2x500GB 7.2k SATA and 4x80 GB 7.2k SATA	100 Mbit Ethernet

Ассоциация RENAM координирует функционирование и развитие MD-GRID NGI, активными участниками которой в настоящее время являются:

- ИГС АНМ — Институт Геологии и Сейсмологии Академии Наук Молдовы;
- ИМИ АНМ — Институт Математики и Информатики Академии Наук Молдовы;
- ИПФ АНМ — Институт Прикладной Физики Академии Наук Молдовы;
- ФРТ ТУМ — Факультет Радиоэлектроники и Телекоммуникаций Технического Университета Молдовы;
- ГГСМ — Государственная Гидрометеорологическая Служба Молдовы;
- НЦМСП – Национальный научно-практический центр медицины скорой помощи.

Приложения, активно использующие Grid-технологии, развиваются по трем направлениям - сейсмология, экология и медицина [5]. Последнее направление связано с созданием системы для обмена и хранения информации в формате DICOM (стандарт изображений, получаемых с медицинского оборудования), используя технологии Grid. Создание такой системы призвано обеспечить автоматизацию хранения, обработки и доступа к результатам медицинских исследований, накапливаемым в формате DICOM. Общая структура такой системы показана на Рис. 1. При решении задач информационного характера, когда потребитель должен получать интересующую его информацию в момент порождения или когда в ней возникает потребность, Grid-технологии применяются для интеграции всех накапливаемых данных.

Основной чертой развития информационных технологий в Молдове является стремление эффективно использовать распределенные разнородные вычислительные ресурсы и системы хранения информации для решения как чисто научных, так и практических задач.

Исходя из этого, интерес представляет поддержка и расширение научно-исследовательской и прикладной деятельности в сфере Grid-технологий и High-Performance Computing в следующих основных направлениях:

1. Развитие тестовой Grid-структуры:
   - обмен опытом по установке и эксплуатации систем виртуализации для развертывания Grid-сайтов;
   - обмен опытом по настройке и эксплуатации middleware, систем мониторинга и учета ресурсов;
2. Апробация и адаптация приложений в Grid (гридификация приложений):
   - обмен опытом по созданию приложений в областях распределенных и параллельных вычислений, систем визуализации и распределенных баз данных;
   - разработка и создание приложений в области параллельных компьютерных технологий и высокопроизводительных вычислений;
   - консультации и помощь пользователям в создании Grid-ориентированных приложений;
   - Разработка, адаптация и внедрение удобного для пользователей сервиса доступа к ресурсам Grid.
3. Привлечение новых пользователей ресурсов Grid-инфраструктуры:
   - подготовка и издание учебно-методических материалов для популяризации Grid и HPC технологий в научно-образовательных и других структурах общества;
   - обучение пользователей работе в Grid-инфраструктурах;
   - включение пользователей в существующие и создаваемые Виртуальные Организации учебно-тестовой и реальной Grid-инфраструктур;
4. Осуществление совместных международных научно-технических программ и проектов в сфере Grid-технологий и High-Performance Computing.

В условиях бурного развития информационных и коммуникационных технологий владение навыками работы и умение создания приложений в Grid-средах является важным фактором развития для многих областей, в которых необходимо применение высокопроизводительных вычислений и обработка больших объемов данных. Заключенный в начале 2010 года договор о сотрудничестве между ЛИТ ОИЯИ (Лаборатория Информационных Технологий Объединенного Института Ядерных Исследований) и MD-GRID NGI призван вносить существенный вклад в решение этой задачи. Успешная реализация процесса обучения пользователей обусловлена наличием в ЛИТ ОИЯИ современного учебного полигона, позволяющего продемонстрировать работу в различных Grid-средах.

Участники MD-GRID NGI с самого начала стремились установить сотрудничество с университетами и научно - исследовательскими институтами АНМ и другими заинтересованными организациями Молдовы. В настоящее время в Молдове созданы необходимые предпосылки для внедрения Grid-технологий в научных и производственных сферах. Это объективный процесс движения от традиционных способов работы к использованию передовых технологий обработки данных.

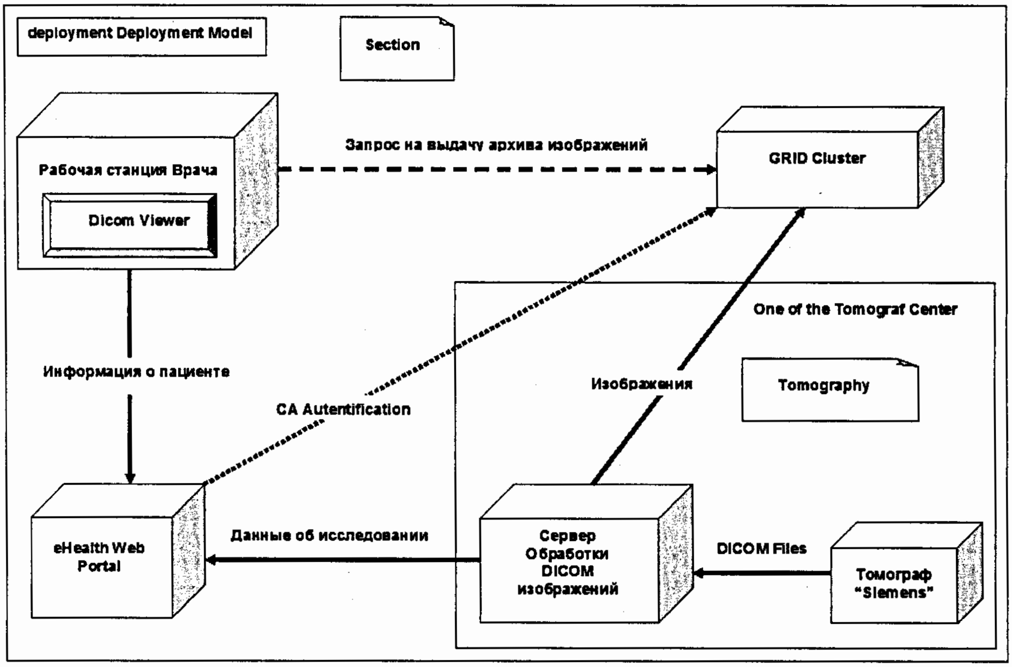


Рис. 1: Общая структура системы для обмена и хранения информации в формате DICOM (стандарт изображений, получаемых с медицинского оборудования), используя технологии Grid

Литература

[1] South-East European GRid eInfrastructure Development, http://www.see-grid.eu/

[2] SEE-GRID eInfrastructure for regional eScience, http://www.see-grid-sci.eu/

[3] Altuhov A., Bogatencov P., Secrieru G., Sidorenco V., Pocotilenco V. Participation of scientific - educational community of Moldova in the European project of Grid development in the Easernt Europe countries // Proc. of All-Russia scientific conference "Scientific service in Internet: Technologies of parallel programming". Novorossisk, September 18-23, 2006. Moscow University, M. 2006. P. 127-128 (in Russian).

[4] EGI.eu- pan-European Grid Initiative organization, http://www.egi.eu

[5] SidorencoV., Bogatencov P., Altuhov A. MD-GRID JRU Consortium and its Role in SEE-GRID-SCI Project // Proc. of the Third International Conference, Dubna, June 30 – Jule 4, 2008.

ПОДГОТОВКА КАДРОВ В СФЕРЕ ГРИД-ТЕХНОЛОГИЙ И РАСПРЕДЕЛЕННОГО КОМПЬЮТИНГА

А. П. Афанасьев[2], Л. А. Калиниченко[3], М. А. Посыпкин[2], С. А. Ступников[3], В. А. Сухомлин[1], О. В. Сухорослов[2]

[1] МГУ им. М.В. Ломоносова, Москва, Россия
[2] Институт системного анализа РАН, Москва, Россия
[3] Институт проблем информатики РАН, Москва, Россия

В последнее время грид-технологии получили широкое развитие как за рубежом (проекты EGEE, DEISA), так и в нашей стране (проекты RDIG, СКИФ-ГРИД, РИСП). Можно с уверенностью утверждать, что владение навыками работы и создания приложений в грид-инфраструктуре является важнейшим фактором, определяющим прогресс во многих областях, в которых требуются применение высокопроизводительных вычислений и обработка больших объемов данных. Поэтому, подготовка кадров в этом направлении является важной и актуальной задачей российского образования. Современные грид-инфраструктуры разнообразны по своему функциональному назначению. Различаются вычислительные гриды, ориентированные на распределенные вычисления с целью образования «виртуального суперкомпьютера» многими связанными посредством сети компьютерами. В e-science требуются информационные гриды, обеспечивающие доступ к неоднородным, распределенным репозиториям данных большого объема наряду с разделяемым доступом к другим видам ресурсов. В настоящее время в российских ВУЗах изучаются различные аспекты вычислительных и информационных гридов. При этом отсутствует целостная программа, объединяющая различные стороны грид-технологий. В статье рассматривается проект магистерской программы, структуры и объема знаний, необходимых для подготовки специалистов в области грид-технологий. Предлагаемая программа включает в себя теоретические вопросы распределенных и параллельных вычислений, технологии сервисных гридов (gLite, Globus, Unicore) и гридов рабочих станций (BOINC, XTremWeb, Condor), методы интеграции информационных ресурсов (SOA, OGSA DAI), распределенные хранилища данных (SRB). На основании опыта преподавания различных аспектов грид-технологий, имеющегося у коллектива авторов, обосновывается последовательность, в которой должны преподаваться курсы программы, а также распределение часовой нагрузки по курсам. Также рассматривается распределенная инфраструктура для проведения практических занятий по программе.

Введение

В последние годы быстрое развитие получили технологии организации распределенной обработки информации и высокопроизводительных вычислений. Одним из классов таких технологий являются грид-технологии - инфраструктурные технологии промежуточного слоя, предоставляющие возможность интеграции вычислительных и информационных ресурсов глобальных сетей для решения сверхсложных и ресурсоемких задач вычислительного характера и/или обработки информации. Грид-инфраструктуры являются разновидностью распределенных параллельных систем, определяемых наборами открытых стандартов и протоколов, и служащих для обеспечения доступа к данным, вычислительным мощностям, средам хранения и широкому набору других ресурсов, доступных при помощи Интернета.

Грид-инфраструктуры разнообразны по своему функциональному назначению. Большой класс грид-технологий составляют вычислительные гриды, ориентированные на распределенные вычисления с целью образования «виртуального суперкомпьютера» из большого числа компьютеров, связанных друг с другом посредством сети и работающих совместно при решении сложных задач, требующих значительных вычислительных и информационных ресурсов. В e-science все более широкое применение находят информационные гриды, обеспечивающие доступ к неоднородным, распределенным репозиториям данных большого объема наряду с разделяемым доступом к другим видам ресурсов (включая вычислительные).

Целью настоящей работы является создание системы подготовки профессиональных кадров в сфере грид-технологий и распределенного компьютинга. В качестве первоочередной задачи коллективом авторов ведется разработка соответствующей магистерской программы в рамках направления 010300 «**Фундаментальная информатика и информационные технологии – ФИИТ**». Ниже рассмотрены первые результаты работы в данном направлении.

1. Актуальность и основные шаги создания системы грид-образования

В настоящее время происходит стремительное развитие технологий грид с целью создания согласованной, открытой и стандартизованной технологической среды, обеспечивающей гибкое, безопасное, скоординированное совместное использование вычислительных ресурсов глобальной сети для решения сложных и ресурсоемких задач в важнейших областях современной науки и техники.

Различают следующие классы грид [1-18]: Computational Grid - грид ориентированный на распределенные вычисления, Data Grid - грид ориентированный на обработку больших потоков данных, Informational Grid - грид ориентированный на интеграцию крупных распределенных хранилищ (OGSA-DAI), в подобных архитектурах используется централизованный реестр, хранящий метаданные всех сервисов и распределенных хранилищ, Hybrid Grid - грид сочетающий в себе как Computatuinal/Data Grid так и Informational Grid, Semantic Grid - это любой из описанных типов грид-архитектур, в котором описывается семантика ресурсов (интерфейсы, характеристики производительности, особенности безопасности).

Сейчас уже можно утверждать, что освоение грид-инфраструктуры является важным фактором развития ряда наукоемких приложений, для которых требуются высокопроизводительные вычисления и массовая обработка информации. Однако создание распределенных грид-приложений относится к наукоемким задачам и является существенно более сложным процессом по сравнению с созданием обычных последовательных программных систем. В связи с этим весьма актуальным становится задача развертывания системы подготовки высокопрофессиональных кадров в области распределенных вычислений и грид-технологий.

Анализ федеральных государственных образовательных стандартов (ФГОС) нового поколения показал, что наиболее адекватной учебно-методической платформой для построения системы грид-образования являются направления, разработанные факультетом вычислительной математики и кибернетики МГУ им. М.В. Ломоносова:

- 010400 - Прикладная математика и информатика,
- 010300 Фундаментальная информатика и информационные технологии – ФИИТ (*до 2010 года «Информационные технологии»).*

Указанные направления позволяют построить полную уровневую систему подготовки, включающую как уровень бакалавра (в виде профиля), так и уровень магистра (магистерская программа). При этом магистерская подготовка в данном случае имеет более весомое значение, как с точки зрений конечных целей обучения (подготовка высокопрофессиональных креативных кадров), так и в плане полноты объема знаний для подготовки профессионалов данной направленности. Поэтому в качестве первоочередной задачи авторами ставится цель разработки и внедрения в учебную практику магистерской программы, получившей название «Высокопроизводительные распределенные технологии и ГРИД».

К основным шагам решения этой задачи следует отнести следующее:

- создание учебно-методического обеспечения (спецификаций объема знаний, учебных программ, учебных курсов и практических занятий, УМК и пр.);

- создание учебной грид-инфраструктуры, полигона GRID-EDU;
- развертывание программного обеспечения промежуточного слоя (middleware) и разработка средств унифицированного доступа к грид-сервисам разного класса грид-инфраструктур;
- применение технологий электронного обучения, организация учебных процессов для классической и смешанных форм обучения.

Кратко рассмотрим решения, разработанные при реализации указанных выше шагов.

2. Создание учебно-методического обеспечения

В данном проекте первостепенное значение уделяется разработке объема знаний системы грид-образования (Knowledge of Body of Grid-Education – KoB GE). При этом используется подход, аналогичный подходу, применяемому при разработке типовых программ учебных курсов организациями IEEE и ACM [19]. В данном подходе объем знаний определяется в виде иерархической конструкции с тремя уровнями иерархии, включая уровни предметных областей знаний, разделов областей и тем (топиков) разделов. Процесс проектирования объема знаний системы грид-образования носит коллективный характер. Для интенсификации этого процесса планируется использовать процесс консорциумной стандартизации, организованный на основе ресурса it-edu.ru .

Ниже представлен состав предметных областей KoB GE:

- DC1: Архитектура параллельных и распределенных вычислительных систем,
- DC2: Парадигмы и методы распределенных вычислений и процессов обработки информации,
- DC3: Грид-системы и ПО промежуточного слоя,
- DC4: Распределенные объектные технологии,
- DC5: Технологии облачных вычислений,
- DC6: Онтологическое моделирование в грид-среде,
- DC7: Методы обеспечения безопасности грид-систем,
- DC8: Администрирование грид-систем,
- DC9: Методы и средства разработки грид-приложений,
- DC10: Организационно-методическое обеспечение грид-систем.

3. Создание полигона GRID-EDU

Эффективный учебный процесс в области высокопроизводительных вычислений невозможно представить без проведения практических занятий и исследований на современной технологической базе с использованием современных средств разработки программного обеспечения. Полигон GRID-EDU создается на основе следующих требований:

1. включение в состав полигона высокопроизводительных вычислительных ресурсов и инфраструктурных решений различных типов, в частности, многопроцессорных систем с общей памятью, систем с распределенной памятью, сервисных гридов, грид-систем на базе объединенных в сети персональных компьютеров;
2. функциональная полнота инструментальных средств - на компьютерах полигона GRID-EDU должен присутствовать широкий набор стандартных программных средств построения грид-приложений;
3. предоставление доступа для различных организаций к сервисам и ресурсам грид-инфраструктур при обеспечении надежных средств аутентификации и авторизации, сводящих к минимуму риск несанкционированного проникновения в компьютерные сети организаций.

Создание полигона GRID-EDU, удовлетворяющего определенным выше условиям, ведется посредством интеграции ресурсов организаций, участвующих в рассматриваемом проекте.

4. Базовые грид-технологии для организации учебного процесса

Важным вопросом создания образовательной системы в области грид и распределенного компьютинга является выбор грид-технологий для учебного процесса и оснащение ими грид-полигона. Как следует из вышесказаного, одно из требований к оснащению гридами учебной базы - это обеспечение возможности для учащихся работать с гридами разных классов, в частности, как с вычислительными, так и информационными.

При выборе вычислительных гридов для учебного процесса был проведен анализ наиболее распространенных решений в этой области, который показал, что в настоящее время успешно развиваются два основных подхода к построению вычислительных гридов.

Первый подход, получивший называние сервисного грида (Service Grid), предполагает развертывание распределенной сервис-ориентированной инфраструктуры, обеспечивающей унифицированный удаленный доступ к выделенным ресурсам уровня кластеров или суперкомпьютеров. Поставщиками ресурсов в подобных системах являются достаточно крупные организации, обладающие ресурсами указанного уровня. Как правило, включаемые в сервисный грид ресурсы являются гомогенными, то есть функционируют под управлением одной версии ОС и предоставляют одинаковое окружение для запускаемых заданий. Число пользователей сервисных гридов гораздо больше числа поставщиков ресурсов. При этом каждый пользователь может использовать ресурсы грида для запуска своих приложений. Примерами сервисных гридов являются EGEE, NorduGrid, TeraGrid. Базовым промежуточным ПО подобных систем служат технологии Globus Toolkit, gLite, ARC, UNICORE. Недостатком сервисных гридов является высокая сложность установки и администрирования указанного ПО, что ограничивает круг потенциальных поставщиков ресурсов.

Второй подход, так называемый грид рабочих станций (Desktop Grid), предполагает использование ресурсов большого количества простаивающих персональных компьютеров, подключенных к сети. Поставщиками ресурсов в подобных системах являются рядовые пользователи. Подключаемые в грид ресурсы рабочих станций являются гетерогенными по своей архитектуре и программному обеспечению. При этом данные ресурсы, как правило, доступны не постоянно, а только в моменты их простоя. Поэтому, в отличие от сервисных гридов, состав ресурсов грида рабочих станций является гораздо более динамичным. В подобных системах число поставщиков ресурсов обычно гораздо больше числа пользователей, использующих ресурсы грида для запуска приложений. Примерами технологий для организации грида рабочих станций являются BOINC, Condor, XtremWeb. В отличие от технологий сервисных гридов, данные технологии позволяют легко и быстро подключать к системе новые ресурсы.

На данном этапе разработки магистерской программы для ее реализации выбраны следующие инфраструктурные технологии:

- промежуточное программное обеспечение gLite, которое является наиболее распространенным способом организации грид-вычислений в Европе;
- грид рабочих станций (desktop grids) BOINC (Berkley Open Infrastructure for Network Computing), в качестве вычислительных ресурсов которого планируется использовать мощности компьютерных классов и выделенные ресурсы организации EGEE, подключенные к гриду рабочих станций при помощи технологии EDGeS.

Выбор информационных грид-инфрастрктур для их использования в образовательном процессе еще не завершен, в виду их большого многообразия, включающего, как уже отмечалось гриды знаний, онтологические гриды, семантические гриды, и др.

5. Применение технологии электронного обучения

Технологии распределенных и параллельных вычислений являются быстроразвивающейся динамично изменяющейся областью знаний. Поэтому чрезвычайно остро стоит вопрос создания образовательного контента и поддержание его в соответствии с современными технологиями параллельных и распределенных вычислений. Специалистами ИСА РАН, ВМК МГУ, МФТИ и других организаций накоплен значительный опыт преподавания методов и технологий параллельных и распределенных вычислений. Разработаны учебные пособия и слайды презентаций. Размещение этого материала в едином хранилище позволит создать информационный ресурс, полезный как для преподавателей, так и для студентов, обучающихся по тематике распределенных вычислений и грид. С этой целью авторами ведется работа по созданию соответствующего учебно-методического обеспечения на базе ресурса виртуальной кафедры http://vitu.oit.cmc.msu.ru/ .

6. Краткое содержание базового годового курса

Рассмотрим краткое содержание программы базового семестрового курса магистерской программы, разработанного авторским коллективом, который апробируется в настоящее время в рамках учебного процесса на факультете ВМК МГУ имени М.В. Ломоносова. Данный курс называется «Высокопроизводительные распределенные технологии и ГРИД», его программа включает следующие темы:

- Введение в параллельные и распределенные вычисления,
- Основные технологии разработки программ для систем с общей и распределенной памятью,
- Практикум по параллельным вычислением,
- Введение в Грид-технологии,
- Основы технологии Десктоп-грид,
- Установка и настройка BOINC,
- Запуск распределенных приложений в платформе BOINC,
- Разработка и сборка приложений для платформы BOINC –I,
- Разработка и сборка приложений для платформы BOINC –II,
- Введение в технологию MapReduce, реализация на платформе Hadoop,
- Разработка и запуск приложений на платформе Hadoop,
- Высокоуровневые технологии на базе MapReduce, язык PIG, библиотека Cascading.

В заключение следует отметить, что рассмотренный выше проект является открытым. Его реализация в части учебно-методического обеспечения проводится с помощью консорциумного подхода на базе сайта поддержки учебно-методического совета для направлений ПМИ и ФИИТ www.it-edu.ru .

Литература

[1] Foster Ian, Kesselman Carl. The Grid: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers. ISBN 1-55860-475-8. 1999.

[2] Anderson D. BOINC: A System for Public-Resource Computing and Storage// In proceedings of the 5th IEEE/ACM International GRID Workshop, Pittsburgh, USA, 2004.

[3] Anderson David P. Public Computing: Reconnecting People to Science. Conference on Shared Knowledge and the Web. Residencia de Estudiantes, Madrid, Spain, Nov. 17-19, 2003.

[4] Urbach E., Kacsuk P., Farkas Z., Fedak G., Kecskeméti G., Lodygensky O., Marosi A.Cs., Balaton Z. Zoltán; G. Caillat, G. Gombás, A. Kornafeld, J. Kovács, He H., Lovas R.: EDGeS: Bridging EGEE to BOINC and Xtrem Web, Journal of Grid Computing, 2009, Vol 7, No. 3, P. 335 -354.

[5] Kacsuk P., Marosi A., Kovacs J., Balaton Z., Gombas G., Vida G., Kornafeld A. SZTAKI Desktop Grid - a Hierarchical Desktop Grid System, Cracow Grid Workshop, Krakow, 2006.

[6] BOINC: User manual [HTML], http://boinc.berkeley.edu/wiki/User_manual

[7] SZTAKI: DC-API manual [HTML], http://www.desktopgrid.hu/storage/dcdoc/general.html

[8] The Open Grid Services Architecture, http://www.ggf.org/documents/GFD.30.pdf

[9] The Information Grid: A Practical Approach to the Semantic Web, Oracle, http://www.oracle.com/technology/tech/semantic_technologies/pdf/informationgrid_oracle.pdf

[10] AstroGrid, http://www.astrogrid.org/

[11] Information grid, Japan project, http://informationgrid.org/wiki/index.php/Main_Page

[12] Global Information Grid (GIG) , DoD, http://www.globalsecurity.org/space/systems/gig.htm

[13] Department of Defense Global Information Grid Architectural Vision, June 2007, http://cio-nii.defense.gov/docs/gigarchvision.pdf

[14] Nicholas R. Jennings David De Roure and Nigel R. Shadbolt. The Semantic Grid: Past, Present, and Future, http://eprints.ecs.soton.ac.uk/9976/1/procieee.pdf

[15] Kalinichenko L.A., Stupnikov S.A., Martynov D.O. SYNTHESIS: a Language for Canonical Information Modeling and Mediator Definition for Problem Solving in Heterogeneous Information Resource Environments. Moscow: IPI RAN, 2007. P. 171.

[16] Kalinichenko L.A. Compositional Specification Calculus for Information Systems Development Proceedings of the East-West Conference on Advances in Databases and Information Systems (ADBIS'99), Maribor, Slovenia, September 1999, Springer Verlag, LNCS.

[17] Computing Curricula 2005. Association for Computing Machinery and Computer Society of IEEE.

ОБЪЕДИНЕНИЕ ВЫСОКОУРОВНЕВЫХ ВЫЧИСЛИТЕЛЬНЫХ РЕСУРСОВ В РАСПРЕДЕЛЁННОЙ СРЕДЕ

А. П. Афанасьев, В. В. Волошинов, М. А. Посыпкин, О. В. Сухорослов

Институт системного анализа РАН, Москва, Россия,
Московский физико-технический институт (государственный университет)

Современные грид-системы ориентированы на интеграцию высокопроизводительных вычислительных ресурсов для решения задач с предельно высокими требованиями к подобным ресурсам, а также задач переборного и поискового типа, допускающих декомпозицию на множество небольших независимых подзадач. Представляется, что концепция грид-вычислений может быть использована для решения более широкого класса задач, отличительной особенностью которых является возможность их декомпозиции на относительно «крупные» типовые подзадачи. Данный класс фактически охватывает широкий спектр вычислительных задач математики, физики, химии, биологии и т.д. Для решения таких задач требуется набор сервисов решения типовых вычислительных математических задач, связанных друг с другом в соответствии со схемой решения исходной задачи. Появление возможности решения сложных задач путем композиции проблемно-ориентированных сервисов вывести распределенные вычислительные среды на качественно новый уровень.

Предлагаемый подход, охватывающий фактически все этапы научных исследований, состоит в построении распределенных вычислительных сред нового поколения, предоставляющих доступ к проблемно-ориентированным сервисам и образующих универсальную инфраструктуру для научной кооперации. Данная инфраструктура базируется на сервис-ориентированном подходе: пользователи преобразуют свои приложения в удаленно доступные сервисы, которые могут быть обнаружены и использованы другими пользователями для решения интересующих их задач. Данный подход обобщает идеи совместного использования вычислительных ресурсов в грид, расширяя при этом возможности системы и ликвидируя разрыв между прикладными задачами и вычислительной инфраструктурой.

Следует подчеркнуть, что сервис-ориентированные научные среды не являются отрицанием грид-систем, а, напротив, - должны базироваться на вычислительной грид-инфраструктуре, используя ее для проведения сложных вычислений и хранения больших массивов данных. Таким образом, речь идет о естественной эволюции грид-систем и реализации новых системных уровней над уже созданной инфраструктурой. Новизна же предлагаемого подхода состоит в смещении акцента от агрегации вычислительных ресурсов на решаемые с помощью агрегированных ресурсов задачи. Если грид-системы развивались снизу вверх, начиная с “сырых” ресурсов, то среды нового поколения нацелены на отображение прикладных задач на доступные в грид ресурсы путем создания проблемно-ориентированных сервисов.

Актуальность данного направления исследований подтверждается сформулированной в 2005 году концепцией “сервис-ориентированной науки” (Service-Oriented Science) [1], автором которой выступил Иэн Фостер, один из основположников грид-вычислений. В соответствии с данной концепцией, сервис-ориентированный подход позволяет организовать повсеместный доступ к разнородным научным ресурсам и автоматизировать процесс научных исследований, тем самым, открывая новые возможности для науки в целом. Концепция “сервис-ориентированной науки” перекликается с более широкой концепцией “электронной науки” e-Science, сформулированной Джоном Тэйлором в 1999 году. Предлагаемый подход также согласуется с распространенной в последнее время моделью “приложение-как-сервис”

(Software-as-a-Service, SaaS), заключающейся в оформлении приложения в виде удаленно доступного сервиса.

MathCloud

Среда MathCloud [2] является сервис-ориентированной математической средой, базирующейся на технологиях Web и грид. Целями данной среды являются предоставление унифицированного доступа к проблемно-ориентированным вычислительным сервисам и поддержка интеграции данных сервисов при решении прикладных задач. Во главу предлагаемого подхода к реализации среды MathCloud ставятся удобство разработки сервисов, простота доступа к сервисам пользователей и использование открытых технологий.

Проблемно-ориентированный вычислительный сервис среды MathCloud (далее - просто сервис) представляет собой доступный по сети программный компонент, поддерживающий решение определенного класса задач с помощью соответствующих вычислительных алгоритмов. В соответствии с моделью клиент-сервер, сервис обслуживает приходящие к нему запросы клиентов на решение конкретных задач. Запрос клиента содержит параметризованное описание задачи, формулируемое в виде конечного набора входных параметров. После успешной обработки запроса сервис возвращает клиенту результат, оформленный в виде конечного набора выходных параметров.

Для унификации удаленного доступа к сервисам MathCloud на уровне протоколов и форматов данных используется архитектурный стиль REST (Representational State Transfer) [3]. REST обладает рядом преимуществ по сравнению с технологиями Web-сервисов на основе протокола SOAP: простой унифицированный интерфейс на основе открытых стандартов HTTP и URI; максимальная свобода выбора языка программирования и средств разработки; высокая масштабируемость за счет грамотного использования ключевых элементов архитектуры Web. Для среды MathCloud был разработан и подробно описан стандартный REST-интерфейс [4], который должны реализовывать все сервисы среды. Данный интерфейс поддерживает обмен данными в формате JSON, асинхронную обработку запросов и получение описания сервиса.

Разработан контейнер сервисов, реализующий указанный интерфейс и поддерживающий быстрое преобразование в сервисы приложений с интерфейсом командной строки. Каждый сервис, развернутый в контейнере, доступен пользователям среды через веб-браузер. Произведена интеграция контейнера сервисов с грид-инфраструктурой EGEE, позволяющая преобразовывать в сервисы MathCloud существующие грид-приложения.

Поддержка объединения или композиции сервисов является ключевой функцией сервис-ориентированной среды, позволяющей пользователям данной среды "собирать" из существующих сервисов новые приложения и, что особенно важно, новые сервисы, развивая, тем самым, среду. Для этих целей в рамках MathCloud предусмотрены готовые средства, упрощающие композицию сервисов и доступные пользователям с минимальной квалификацией. В частности, разработана workflow-система [5], позволяющая описывать сценарий совместного использования нескольких сервисов MathCloud при помощи визуального редактора. Описанный сценарий может быть затем преобразован в новый сервис MathCloud и запущен на выполнение путем вызова данного сервиса.

Предлагаемый подход позволяет скрыть от пользователя детали реализации вызовов сервисов и передачи данных между ними, оставив пользователю только необходимость правильного соединения сервисов друг с другом. Таким образом, многие задачи, решение которых сводится к компоновке стандартных сервисов, становятся доступными для пользователей не обладающих навыками распределенного программирования. Графическое представление сценария позволяет сделать наглядными связи между сервисами. Кроме того, такое представление позволяет быстро вносить изменения в уже работающие сценарии.

Визуальный редактор сценариев реализован в виде самостоятельного Web-приложения, запускаемого в браузере. В свою очередь, среда выполнения сценариев реализована в виде

REST-сервиса, позволяющего хранить сценарии и экспортировать их в виде сервисов MathCloud.

Грид-системы из персональных компьютеров

Современные грид-инфраструктуры можно условно разделить на два класса: сервис-ориентированные гриды (СГ) и грид системы из персональных компьютеров (ГПК). В англоязычной литературе принята терминология Service Grids и Desktop Grids для обозначения систем первого и второго класса соответственно. Основной задачей сервис-ориентированных гридов является предоставление вычислительных ресурсов и ресурсов хранения данных для пользователей грид-инфраструктуры. При этом аппаратные ресурсы для этих целей специально выделяются и поддерживаются организациями-участниками грид-ассоциаций. Безопасность доступа к ресурсам контролируется с помощью механизма грид-сертификатов, выдаваемых пользователям. В качестве примеров успешной реализации концепции сервис-ориентрованных гридов можно привести европейские грид-ассоциации EGEE (EGI) и DEISA.

Недостатком сервис-ориентированных гридов является достаточно высокая стоимость их создания и эксплуатации, обусловленная необходимостью приобретения, установки и поддержания в работоспособном состоянии выделенных ресурсов. Существенно более дешевой альтернативой, получившей широкое распространение в настоящее время, является использование простаивающих ресурсов персональных компьютеров. Замечено, что при выполнении большинства задач, для которых предназначен персональный компьютер, в среднем загрузка центрального процессора не превышает 5-10% от полной. Оставшийся ресурс можно использовать для полезной работы. На этой идее основаны грид-системы из персональных компьютеров. Начав свое развитие с решения конкретных задач (SETI@HOME) в распределенной среде, сейчас ГПК развиваются в рамках больших международных проектах, таких как, как например DEGISCO[6], направленных на радикальное увеличение вычислительной мощности подобных систем. В ГПК пользователи предоставляют ресурсы персональных компьютеров для решения вычислительных задач, регистрируя их на сервере. Затем сервер распределяет по подключенным компьютерам задания, которые, как правило, являются частями одной большой задачи. В функции промежуточного программного обеспечения ГПК входит распределение работы по вычислительным узлам (подключенным персональным компьютерам), восстановление вычислительного процесса в случае сбоя или отключения одного или более узлов, верификация результатов расчетов с помощью дублирования заданий и сопоставления результатов, полученных на разных узлах. Безопасность обеспечивается на уровне приложений: выполняемый файл содержит цифровой сертификат, проверяемый при загрузке на вычислительный узел. ГПК эффективны при решении задач, допускающих декомпозицию на достаточно большое число независимых частей. К этому классу относятся различные переборные задачи из области биоинформационных технологий, обработки сигналов, моделирования структуры химических соединений. В настоящее время наиболее распространенной платформой для организации ГПК является BOINC[7]. Также известны, хотя используются в значительно меньшей степени такие системы, как XTremeWeb[8] и OurGrid[9].

Тенденцией последнего времени является создание механизмов обеспечения интероперабельности между сервис-ориентированными гридами и грид-системами из персональных компьютеров. Одной из основных международных инициатив, направленных на решение задачи интеграции СГ и ГПК, является проект EDGI[10]. Разработанный в рамках этих проектов инструментарий 3G-Bridge[11] позволяет использовать ресурсы ГПК для обработки задач СГ и наоборот.

BNB-Grid

Задача поиска глобального минимума (максимума) функции $f(x)$ на допустимом множестве $X \subseteq R^n$ состоит в отыскании такой точки $x_* \in X$, что $f(x_*) \leq f(x) (f(x_*) \geq f(x))$

для всех $x \in X$. Решение задач глобальной оптимизации требует больших вычислительных ресурсов. Поэтому для ускорения методов их решения применяют параллельные и распределенные вычисления. Разработан программный комплекс BNB-Grid[12], позволяющий решать такие задачи в распределенной вычислительной среде. В состав среды могут входить рабочие станции, параллельные системы с общей памятью, суперкомпьютеры.

Первый этап решения задачи оптимизации в BNB-Grid состоит в запуске приложений, выполняющих вычисления (солверов) на узлах распределенной среды. В результате формируется вычислительное пространство. Формирование вычислительного пространства происходит автоматически с использованием средств удаленного доступа, предусмотренных конкретной системой: SSH, Грид-сервисы и др. На данный момент поддерживается сервисный грид ППО СКИФ-Грид (на базе Unicore) и грид-системы из персональных компьютеров BOINC. После создания вычислительное пространство используется для решения задачи. Обмен данными между солверами производится центральным процессом – супервизором и осуществляется средствами, предусмотренными для взаимодействия с конкретным узлом. Если имеется возможность установления прямого сетевого соединения, то используются способы обмена, основанные на протоколах TCP/IP. В некоторых случаях обмен данными с приложениями возможен только через передачу файлов средствами промежуточного программного обеспечения Грид.

Балансировка нагрузки производится на двух уровнях: на верхнем уровне супервизор распределяет вычислительную нагрузку между солверами. На нижнем уровне (в пределах одного вычислительного узла) распределение работы производится солвером методами, предназначенными для конкретного типа вычислительного узла. Для разработки солверов применяется библиотека BNB-Solver [13], предназначенная для решения задач на многопроцессорных системах с общей и распределенной памятью.

На данный момент программный комплекс BNB-Grid был успешно применен для решения задачи о ранце, задачи нахождения энергетически-оптимальной конформации атомного кластера и задачи криптоанализа шифра A5/1. Причем в последнем случае в вычислениях было задействовано до 6000 вычислительных ядер различных суперкомпьютеров одновременно.

Сервисы оптимизационного моделирования

Методы оптимизационного моделирования являются одним из основных инструментов системного анализа и обработки данных в инженерных и физических задачах. За долгую историю наработан обширный парк солверов и библиотек численных методов оптимизации для решения различных классов задач. Рассмотрим один из самых распространенных - задачи нелинейного математического программирования

$$f_o(x) \to \min_{x \in Q}, Q \doteq \left\{ x \in \mathbb{R}^n, f_i(x) \le 0 \ (i \in I), g_j(x) = 0 \ (j \in J) \right\}.$$

Практическое применение существующих солверов подразумевает программную реализацию процедур вычисления (при заданном векторе переменных x): значений целевой функции и ограничений, $f_o(x), f_i(x) \ (i \in I), \ g_j(x) \ (i \in J)$; векторов градиента целевой функции и якобианов вектор-функций ограничений $\nabla f_o(x) \in \mathbb{R}^n$, $\nabla f_i(x) \in \mathbb{R}^{|I| \times n}$, и $\nabla g_j(x) \in \mathbb{R}^{|J| \times n}$; гессиана (матрицы вторых производных) “расширенной” функции Лагранжа (с множителем и при целевой функции) при заданных множителях Лагранжа $\alpha_i, \beta_j, \gamma$)
$\nabla^2_{xx} L(x, \alpha, \beta, \gamma) \doteq \gamma \nabla^2 f_o(x) + \sum_{i \in I} \alpha_i \nabla^2 f_i(x) + \sum_{j \in J} \beta_j \nabla^2 g_j(x)$.

Для автоматического создания указанных процедур непосредственно из символьного описания исходных данных оптимизационной модели с подстановкой числовых значений ее параметров, указанных отдельно от символьного представления, принято использовать специальные языки оптимизационного моделирования [14]. Наиболее распространенными являются системы на основе языков AMPL (A Modeling Language for Mathematical Programming) [15] и GAMS (General Algebraic Modeling System) [16]. Основой обоих является транслятор, преобразующий формулы символьного описания задачи в специальную структуру, т.н. стаб (например, на основе префиксной польской записи), пригодную для последующего применения средств автоматического дифференцирования [17]. Подобные системы развиваются с 80-х годов прошлого века. Активные исследования в этом направлении велись тогда и в СССР. В статье [17], в частности, сказано «...В ВЦ АН СССР в 1979-1982 гг. В.П. Мазуриком было разработано программное обеспечение для дифференцирования элементарных функций на ЭВМ БЭСМ-6 и СМ-4. В 1983-1984 гг. Е.Н. Веселовым был создан язык ДИФАЛГ с дифференциальной семантикой. Язык используется в системе ДИСО в качестве входного языка постановки оптимизационных задач на персональных компьютерах...».

Большинство современных программных реализаций решателей оптимизационных задач рассчитаны на использование одного (или нескольких) языков оптимизационного моделирования. Например, в своих исследованиях мы применяем солверы LP_SOLVE, lpsolve.sourceforge.net, GLPK, www.gnu.org/software/glpk (оба - для решения задач линейного программирования), и, для нелинейных задач, - Ipopt, projects.coin-or.org/Ipopt. Для всех трех существуют модификации, предназначенные для ввода данных в форматах AMPL и GAMS.

С учетом сказанного выше, решение оптимизационных задач в распределенной среде специализированных сервисов удобно декомпозировать на две подзадачи: 1) преобразование символьного описания модели и числовых значений параметров в стаб; 2) собственно вызов солвера для обработки стаба, т.е. запуска численной процедуры поиска оптимального решения. Соответственно, целесообразно создание сервисов-“генераторов” стабов на основе программных компонент системы AMPL или GAMS и сервисов-оберток для солверов. Именно такой подход уже применяется нами в исследованиях структурного состава углеродных наноструктур [18], где применяется метод оптимизационной идентификации, средствами программного инструментария MathCloud [19].

Проблема каталогизации сервисов

При развертывании масштабной инфраструктуры сервисов обработки данных чрезвычайно важной является проблема каталогизации ресурсов системы, т.е. регистрации подключаемых и обнаружения уже подключенных сервисов. Обычно, эта проблема решается путем создания специализированного сервиса - информационной службы. Необходимость разработки такой службы была заявлена в самом начале «эпохи» Grid [20].

В разработанном нами инструментарии создания сервисов IARnet информационная служба (ИС) [21] реализована на основе технологий Semantic Web [22] и разработок проекта Semantic Grid [23]. ИС позволяет публиковать, как сведения о типах программных ресурсов, так и более подробную информацию о конкретных экземплярах сервисов на языке описания онтологий Web Ontology Language (OWL), адаптированного для описания математических сервисов. Соответствующие информационные сообщения пересылаются в формате RDF/OWL. Запросы к ИС от программных компонентов представляются на языке SPARQL. Система предоставляет простой интерфейс для типовых RDF-запросов, например, поиска зарегистрированных ресурсов по их типам. Программная реализация ИС основана на комплекте разработки RDF-баз данных Jena [24].

Литература

[1] Foster I. Service-Oriented Science. Science. 2005. V. 308, N. 5723. P. 814-817.

[2] Астафьев А.С., Афанасьев А.П., Лазарев И.В., Сухорослов О.В., Тарасов А.С. Научная сервис-ориентированная среда на основе технологий Web и распределенных вычислений. // Научный сервис в сети Интернет: масштабируемость, параллельность, эффективность: Труды Всероссийской суперкомпьютерной конференции (21-26 сентября 2009 г., г. Новороссийск). М.: Изд-во МГУ, 2009. 524 с. (С. 463-467).

[3] Fielding R.T. Architectural styles and the design of network-based software architectures. PhD Dissertation. Dept. of Information and Computer Science, University of California, Irvine, 2000.

[4] Сухорослов О.В. Унифицированный интерфейс доступа к алгоритмическим сервисам в Web. // Проблемы вычислений в распределенной среде / Под ред. С.В. Емельянова, А.П. Афанасьева. Труды ИСА РАН, Т. 46. М.: КРАСАНД, 2009. 304 с. (с. 60-82).

[5] Лазарев И.В., Сухорослов О.В. Реализация распределенных вычислительных сценариев в среде MathCloud // Проблемы вычислений в распределенной среде/ Под ред. С.В. Емельянова, А.П. Афанасьева. Труды ИСА РАН, Т. 46. М.: КРАСАНД, 2009. 304 с. (С. 6-23).

[6] DEGISCO (Desktop Grids for International Scientific Collaboration), http://degisco.eu/ Public Computing: Reconnecting People to Science. David P. Anderson. Conference on Shared Knowledge and the Web. Residencia de Estudiantes, Madrid, Spain, Nov. 17-19, 2003.

[7] Public Computing: Reconnecting People to Science. David P. Anderson. Conference on Shared Knowledge and the Web. Residencia de Estudiantes, Madrid, Spain, Nov. 17-19, 2003.

[8] Fedak Gilles, Germain Cecile, Neri Vincent, Cappello Franck. XtremWeb: A Generic Global Computing System // Proceedings of the 1st International Symposium on Cluster Computing and the Grid, 2001. P. 582.

[9] Cirne Walfredo, Brasileiro Francisco, Andrade Nazareno, Costa Lauro, Andrade Alisson, Novaes Reynaldo, and Mowbray Miranda. Labs of the world, unite!!! Journal of Grid Computing, 4(3):225–246, 2006.

[10] EDGI (European Desktop Grid Initiative), http://edgi-project.eu/

[11] Urbach E., Kacsuk P., Farkas Z., Fedak G., Kecskeméti G., Lodygensky O., Marosi A. Cs., Balaton Z., Caillat G., Gombás G., Kornafeld A., Kovács J., He H., Lovas R. EDGeS: Bridging EGEE to BOINC and XtremWeb, Journal of Grid Computing, 2009. Vol. 7. No. 3. P. 335-354.

[12] Посыпкин М.А. Решение задач глобальной оптимизации в среде распределенных вычислений // Программные продукты и системы. № 1. 2010.

[13] Evtushenko Y., Posypkin M., Sigal I. A framework for parallel large-scale global optimization // Computer Science - Research and Development 23(3), 2009. P. 211-215.

[14] Hürlimann T. Modeling Languages in Optimization: A New Paradigm // Encylopedia of Optimization, 2nd ed. / Editors C.A. Floudas, P.M. Pardalos, Springer, 2009. P. 2323-2330.

[15] Fourer Robert, Gay David M., and Kernighan Brian W. AMPL: A Modeling Language for Mathematical Programming. 2nd Ed. Brooks/Cole Publishing Company / Cengage Learning, 2002, (http://www.ampl.com).

[16] GAMS, The General Algebraic Modeling System, http://www.gams.com

[17] Айда-заде К.Р., Евтушенко Ю.Г. Быстрое автоматическое дифференцирование на ЭВМ. // Математическое моделирование. Январь, 1989. Т. 1. № 1. С. 120-131.

[18] Неверов В.С., Афанасьев А.П., Велигжанин А.А., Волошинов В.В., Зубавичус Я.В., Кукушкин А.Б., Марусов Н.Л., Свечников Н.Ю., Семенов И.Б., Станкевич В.Г., Тарасов А.С. Моделирование рентгеновской дифракции на углеродных

наноструктурах и определение их и определение их возможного топологического состава в осажденных пленках из токамака Т-10 // Журнал "Вопросы атомной науки и техники", серия "Термоядерный синтез", Вып. 1, 2010. С. 7-21.

[19] Лазарев И.В., Сухорослов О.В. Реализация распределенных вычислительных сценариев в среде MathCloud. // Проблемы вычислений в распределенной среде / Под ред. С.В. Емельянова, А.П. Афанасьева. Труды ИСА РАН, Т. 46. М.: КРАСАНД, 2009. 304 с. (с. 6-23).

[20] Czajkowski K., Fitzgerald S., Foster I., Kesselman C. Grid Information Services for Distributed Resource Sharing // Proceedings of the 10th IEEE International Symposium on High-Performance Distributed Computing (HPDC-10), IEEE Press, 2001.

[21] Afanasiev Alexander, Sukhoroslov Oleg, Posypkin Mikhail. A High-Level Toolkit for Development of Distributed Scientific Applications. // Victor E. Malyshkin (Ed.): Parallel Computing Technologies, 9th International Conference, PaCT 2007, Pereslavl-Zalessky, Russia, September 3-7, 2007, Proceedings. Lecture Notes in Computer Science 4671 Springer 2007, ISBN 978-3-540-73939-5, P. 103-110.

[22] Berners-Lee T., Hendler J., Lassila O. The Semantic Web. Scientific Am., May 2001, P. 34-43.

[23] De Roure D., Jennings N.R., Shadbolt N.R. The Semantic Grid: Past, Present, and Future //Proceedings of the IEEE, Volume 93, Issue 3, 2005. P. 669-681.

[24] Jena – A Semantic Web Framework for Java, 2007, http://jena.sourceforge.net/

ПРИМЕНЕНИЕ ГРИДА С НЕКЛАСТЕРИЗОВАННЫМИ РЕСУРСАМИ ДЛЯ ЗАДАЧ ПРОЕКТИРОВАНИЯ ФИЗИЧЕСКИХ УСТРОЙСТВ

П. С. Березовский, А. С. Родин

Институт прикладной математики им. М.В.Келдыша РАН
125047, Москва, Россия
bps@keldysh.ru, rals@bk.ru

Введение

На сегодняшний день применение технологий одноуровневого грида, т.е. грида, состоящего из некластеризованных (отдельных) компьютеров, выходит за рамки исследовательских проектов [1], [2] и публичного компьютинга [3]. Программные системы, объединяющие такие ресурсы, применяются для решения реальных практически важных прикладных и научных задач. Несмотря на это, даже наиболее развитые в своём классе системы обладают рядом ограничений, которые существенно снижают потенциал их применения. Такой вывод делается в работе [4] на основе анализа ряда программных разработок и предлагаются требования к программному обеспечению грида с некластеризованными компьютерами, которые учитывают специфику таких ресурсов. В работе [5] рассматривается программная реализация системы SARD (StandAlone Resource Dispatcher) для одноуровневого грида с неотчуждаемыми, т.е. используемыми их владельцами, компьютерами, которая выполнена в соответствии с предложениями работы [4] и теоретическими результатами, изложенными в [6] и [7].

Настоящая работа посвящена применению системы SARD для решения практически важных прикладных задач. Наиболее эффективно некластеризованные ресурсы используются при расчёте сериализуемых приложений. Класс таких приложений довольно широк и включает в себя множество задач, решаемых на основе общепринятых вычислительных методов, таких как моделирование методом Монте-Карло, или поиск оптимального значения параметров в вычислительных экспериментах. В работе рассматриваются две задачи подобного типа, по которым проведены расчёты в управляемой системой SARD вычислительной инфраструктуре Института прикладной математики им. М.В. Келдыша РАН.

1. Задача пространственного распределения энергии ионизирующего излучения

Во многих практических задачах, связанных с функционированием аппаратуры и оборудования в полях ионизирующих излучений, важной является проблема защиты этих объектов от нагрева и других поражающих факторов проникающего излучения.

В большинстве случаев оценка влияния излучения сводится к вычислению энергии, поглощенной в материалах объектов («энерговыделение»). Соответствующие вопросы актуальны и для термостойкости компонентов ядерных реакторов, и для космических аппаратов. Особое место в данной проблематике занимает задача оценки воздействия проникающих излучений на человека.

Модель взаимодействия излучения с материалами объектов строится на основе общепринятых представлений о механизмах поглощения и рассеяния гамма квантов в веществе. Основными типами взаимодействия в исследуемом энергетическом диапазоне (кванты до 1 МэВ) являются: комптоновское и когерентное рассеяние и фотопоглощение излучения.

Для решения указанных задач построены эффективные статистические методы моделирования переноса гамма излучения в сложных многокомпонентных объектах на основе весовых модификаций метода Монте-Карло [8]. Использование алгоритмов статистического моделирования на основе экономичных модификаций метода Монте-Карло [9] является одним из наиболее эффективных для решения сложных граничных задач моделирования взаимодействия и трансформации ионизирующего излучения в многокомпонентных объектах сложной внутренней структуры.

1.1. Актуальность применения технологий грида

В рассматриваемом классе задач, для обеспечения заданной точности обычно необходимо моделирование более миллиарда фотонных историй, что требует больших временных затрат порядка 5–10 часов счёта на современных компьютерах. В то же время, так как фотонные истории моделируются независимо, моделирование всех историй в рамках одного запущенного приложения и моделирование всех историй частями в рамках многих запущенных приложений с последующим суммированием результатов приведут к идентичным результатам (в рамках статистической погрешности). В этой ситуации разделение «тяжёлой» задачи на несколько «лёгких» подзадач и их одновременное выполнение позволяет кратно уменьшить время расчёта. Подобное ускорение может быть достигнуто и при использовании суперкомпьютеров, однако для этого необходимо модифицировать программный код.

1.2. Проведение расчётов на некластеризованных компьютерах

1.2.1. Описание расчётной задачи

С помощью системы SARD решена задача по расчёту распределения плотности потока инжектируемых электронов с внешней и внутренней поверхностей трубы.

На рис. 1 изображена схема модельного эксперимента, в котором алюминиевая труба радиуса R со стенками толщиной $R/10$ облучается источником гамма излучения S на основе изотопа Se^{75}, который широко применяется в дефектоскопии. Расположение совокупности точек Tk на внутренней и внешней поверхностях трубы, в которых рассчитываются характеристики электронных потоков, ясно из рис. 2.

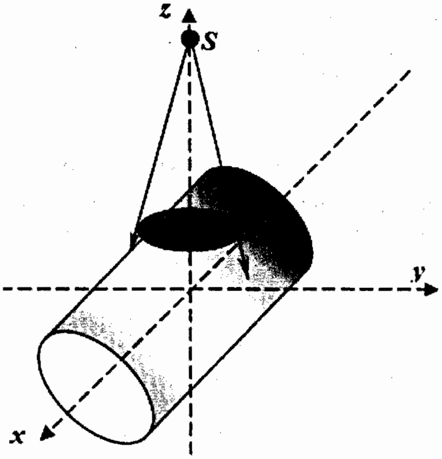


Рис. 1: Схема эксперимента

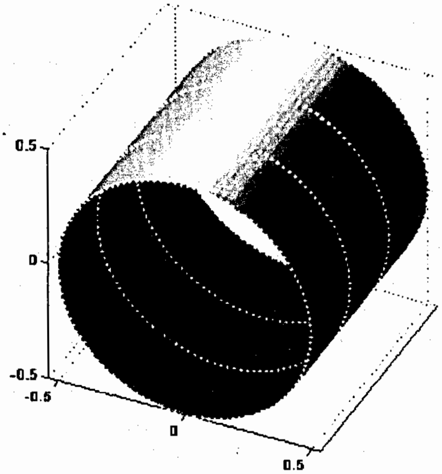


Рис. 2: Расположение детекторов (Tk - белые точки)

1.2.2. Описание программы расчёта

Построенные статистические методы моделирования переноса гамма излучения реализованы в виде отдельного приложения, которое может быть запущено практически на любом современном персональном компьютере под управлением операционной системы семейства Unix или MS Windows. Размер исполняемого файла не превышает 60 КБ вместе с файлами конфигурации. Входные данные зависят от каждой конкретной задачи, в нашем случае суммарный объём таких данных составил около 30 МБ.

Конфигурационный файл приложения состоит из набора блоков, в каждом из которых указываются имена файлов, содержащих описание материала, а также параметры источника или объекта.

Описание материала содержит таблицы некогерентного и когерентного углов рассеяния, а также коэффициентов поглощения. В отдельном входном файле содержится информация о спектре излучения источника.

Для каждого объекта, являющегося детектором, создаётся выходной файл, в котором хранится линеаризованный массив. Каждый элемент этого массива задаётся пятью индексами *m, n, i, j, k*. Индексы *i, j, k* — «пространственные», они определяют пространственную ячейку, в которой поглотилась энергия (пространственная дискретизация для каждого объекта-детектора задаётся во входных данных "space res=%resH% %resR% %resA%"). Индексы *m, n* — «энергетические»: первый указывает на то, что энергия поглощалась порциями, относящимися к *m*-той ячейке (detector energies=%resDet% %detMinNrj% %detMaxNrj%), а второй — то, что энергия поглощалась от фотона с начальной энергией, относящейся к *n*-той ячейке (source energies=%resSrc% %srcMinNrj% %srcMaxNrj%). Этот файл может быть использован для визуализации поглощённой объектом энергии.

1.3. Результаты расчётов

Проведение расчётов на инфраструктуре из десяти некластеризованных компьютеров с помощью разработанной системы диспетчеризации позволило сократить время выполнения расчётной задачи в шесть раз. Благодаря этому на этапе проведения экспериментов удалось осуществить большее количество запусков, а расчёт реальных данных выполнить с большей точностью.

2. Расчёт модели движения лайнера в магнитном компрессоре

Построение многокомпонентных физических установок сопряжено с множеством трудностей, одной из которых является определение характеристик компонентов установки, таких как геометрические размеры, свойства материала деталей, электрические характеристики, компоновка узлов и т.д. Часто компоненты системы разрабатываются одновременно и независимо друг от друга, что не даёт возможности на ранних стадиях проверять в действии работу всей системы в целом и оценивать её показатели.

Примером такого рода систем является установка «МОЛ» [10], предназначенная для генерации электрического импульса мегаджоульного уровня. Для этой установки в ГНЦ РФ ТРИНИТИ разработан макет усилительного каскада мощности — магнитный компрессор (МК), работа которого основана на сжатии магнитного потока лайнером, ускоренным электродинамическими силами до скорости 1 км/с.

Режим компрессии магнитного поля предъявляет особые требования к геометрической форме зазора между пластинами в момент сжатия магнитного поля. Для короткого генерируемого импульса отдача кинетической энергии тонкого лайнера должна проводиться одновременно по всей его плоскости, поэтому важно знать динамику деформирования пластины в процессе её ускорения, особенно на стадии электромагнитного торможения лайнера.

Одним из ключевых вопросов, стоящих перед создателями установки является определение такого набора входных параметров устройства, при котором генерируемый на выходе импульс имеет оптимальные характеристики. Для решения этой задачи в ИПМ им. М.В. Келдыша РАН были построены двумерные математические и численные модели, соответствующие поперечному и продольному сечениям магнитного компрессора, а также разработано программное обеспечение для моделирования движения лайнера.

На макете МК была проведена серия экспериментальных запусков [11], однако имеющийся объём полученных в них данных достаточно ограничен. Таким образом, математическое моделирование и вычислительный эксперимент являются практически единственными инструментами для получения более подробной информации о движении лайнера в магнитном компрессоре.

2.1. Актуальность применения технологии грида

Для получения численных характеристик поведения лайнера с помощью математического моделирования имеется специальный программный комплекс. Входящие в состав комплекса программы позволяют получить характеристики движения ленты лайнера (поле перемещений и скоростей, распределение напряжений и деформаций и т.д.) в зависимости от её формы, материала, момента замыкания цепи и т.д. Чтобы решить поставленную задачу оптимизации входных параметров устройства для получения более мощного выходного импульса, необходимо провести ряд экспериментов, варьируя значения исследуемых параметров для каждой из математических моделей лайнера. Это сопряжено с большими временными затратами, т.к. для получения информации о динамике изменения свойств системы необходимо проводить серию из 20–30 запусков. При этом однократный расчёт занимает 4–8 часов на современном персональном компьютере средней производительности.

Следует отметить, что задача расчёта движения ленты лайнера для одного набора значений параметров является сильно связной в частности по причине перестроения сетки на каждом шаге обработки. В численных алгоритмах используются неявные (по времени) схемы, что затрудняет распараллеливание данных алгоритмов. В связи с этим, ускорение может быть достигнуто путём параллельного запуска программы моделирования с различными значениями исследуемого параметра на нескольких компьютерах. Кроме того, в зависимости от получаемых результатов в ходе вычислительного эксперимента разработчиками численных моделей и программного комплекса могут вноситься соответствующие изменения с целью учёта дополнительных свойств магнитного компрессора и поведения программы моделирования.

2.2. Проведение расчётов на некластеризованных компьютерах

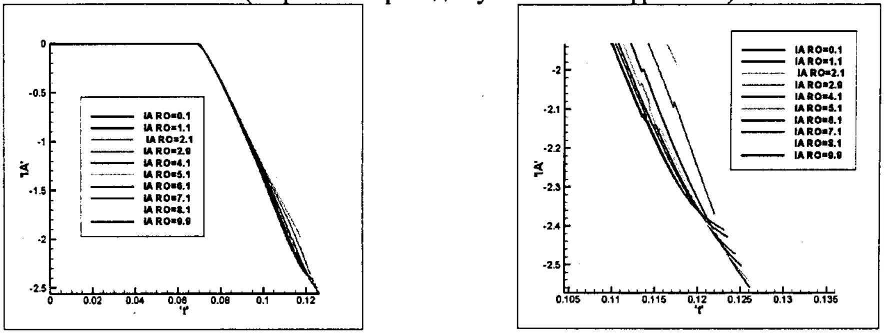
2.2.1. Описание расчётной задачи

Для дискретизации уравнений созданной математической модели применён метод конечных элементов с элементами первого порядка, для чего предварительно проводится триангуляция расчётной области. Под воздействием магнитного поля пластины лайнера двигаются навстречу друг другу, поэтому сетка в диэлектрической подобласти перестраивается на каждом шаге по времени.

С помощью разработанной системы диспетчеризации проведено две серии оптимизационных расчётов.

В первой серии варьировался момент замыкания цепи лайнера *tA* в диапазоне от 50 до 100 микросекунд (процесс ускорения лайнера занимает 120–130 микросекунд). Проведённые расчёты подтвердили, что оптимальным является значение *tA*=70, которое и используется в экспериментальных запусках (это значение было изначально рассчитано исходя из энергетических характеристик устройства).

Во второй серии варьировалось распределение плотности вещества в пластине лайнера: в начальной постановке задачи пластина имела однородную плотность, в дальнейших расчётах принималось, что плотность меняется по линейному закону (в центре пластины коэффициент пропорциональности равен 1, на краях пластины он равен RO). Значение *RO* в расчётах менялось от 0.1 до 10 (при этом общая масса лайнера во всех расчётах одинакова). На рис. 3.А показаны графики зависимости от времени выходного импульса (полного тока в цепи лайнера) для различных значений *RO* (на рис. 3.Б приведён увеличенный фрагмент).



А Б

Рис. 3: Графики выходного импульса для различного распределения плотности в лайнере

Как видно из представленных графиков, выбор распределения плотности вещества в пластине оказывает существенное влияние на выходной импульс (при этом динамика деформирования пластины для различных расчётов сильно отличается). В выбранном диапазоне оптимальным оказалось значение *RO*=4, т.е. плотность вещества на краях лайнера в четыре раза больше, чем плотность в центре пластины.

2.2.2. Описание программы расчёта

Построенные математические модели лайнера реализованы в виде двух Windows-приложений и набора файлов с данными. Первое приложение — основная программа расчёта, второе — вспомогательная программа Gridder2D [12] для перестроения сетки, вызываемая на каждом шаге работы основной программы. Размер исполняемого файла составляет около 500 КБ, а общий объём файлов задания вместе с файлами конфигурации и входными данными не превышает 3 МБ.

В качестве входных данных основная программа получает набор файлов, в которых заданы геометрические размеры области, значения физических характеристик используемых материалов, исходные распределения электромагнитных полей в расчётной области, распределения скорости и температуры в пластине лайнера, а также значения других используемых величин.

В процессе работы программы на каждом шаге по времени происходит решение систем уравнений, соответствующих электромагнитной и кинематической частям задачи. В результате работы программы создаются файлы, содержащие значения электромагнитных, скоростных и прочих характеристик лайнера, а также файлы для анимации движения и построения графиков зависимости от времени интегральных параметров системы, в том числе график выходного импульса.

2.3. Полученные результаты

В ходе проведения серий экспериментов с помощью системы SARD было подтверждено оптимальное значение момента замыкания цепи, рассчитанное исходя из энергетических характеристик устройства. По результатам расчётов была выявлена зависимость выходного импульса от распределения плотности вещества в пластине лайнера и определено оптимальное значение распределения плотности вещества в пластине лайнера в заданном диапазоне. Также была получена информация о деформировании ленты лайнера, которая в дальнейшем будет использована для определения оптимального начального профиля пластины.

Заключение

В настоящей работе рассмотрено применение этой системы и показана её практическая важность при решении ряда научных и производственных задач.

Решённые с помощью системы SARD задачи на инфраструктуре из некластеризованных компьютеров в ИПМ им. М.В. Келдыша РАН и полученные в ходе расчётов результаты свидетельствуют об актуальности применения технологий одноуровневого грида и перспективности ведущихся в этой области разработок.

Способность системы диспетчеризации объединять в грид-инфраструктуры персональные компьютеры для их использования во время слабой загрузки, а также простота установки и администрирования системы позволяют в короткие сроки получить мощный инструмент для решения широкого круга математических задач и оказать существенную поддержку исследователям при проведения вычислительных экспериментов.

Литература

[1] Zhou D., Lo V. Cluster Computing on the Fly: resource discovery in a cycle sharing peer-to-peer system// Fourth IEEE International Symposium CCGrid. 2004. P. 66-73.

[2] Cirne W., Brasileiro F., Andrade N., Santos R., Andrade A. Labs of the World, Unite!!! Journal of Grid Computing. V. 4, No. 3. September 2006. P. 225-246.

[3] Проект BOINC — http://boinc.berkeley.edu

[4] Березовский П.С., Коваленко В.Н. Состав и функции системы диспетчеризации заданий в гриде с некластеризованными ресурсами. Препринт № 67. Москва: ИПМ им. М.В. Келдыша РАН, 2007. С. 29.

[5] Березовский П.С. Реализация системы диспетчеризации заданий SARD в одноуровневом гриде. Препринт № 49. М.: ИПМ им. М.В. Келдыша РАН, 2010. С. 32.

[6] Березовский П.С., Емельянов В.Н., Коваленко В.Н., Луховицкая Э.С. Механизмы управления разделяемыми компьютерами в гриде // Распределённые вычисления и Грид-технологии в науке и образовании// Труды 3-й международной конференции. Дубна: ОИЯИ, 2008. С. 303-306.

[7] Березовский П.С., Коваленко В.Н. Планирование в гриде с разделяемыми ресурсами на основе статистических данных // Программные продукты и системы, 2009. № 85. С. 3-6.

[8] Михайлов Г.А. Весовые методы Монте-Карло. Новосибирск: Изд-во СО РАН, 2000.

[9] Жуковский М.Е., Подоляко С.В., Скачков М.В., Йениш Г.-Р. О моделировании экспериментов с проникающим излучением// Математическое моделирование, 2007. Т. 19. №5. С. 72-80.

[10]Азизов Э.А., Алиханов С. Г., Велихов Е.П., Галанин М.П. и др. Проект «Байкал». Отработка схемы генерации электрического импульса // Вопросы атомной науки и техники, серия Термоядерный синтез. 2001. № 3. С. 3-17.

[11]Галанин М.П., Лотоцкий А.П. Моделирование разгона и торможения лайнера в устройствах обострения мощности // Радиотехника и электроника, 2005. Т. 50. №2. С. 256-264.

[12]Щеглов И.А. Программа для триангуляции сложных двумерных областей Gridder2D // Препринт № 60. Москва: ИПМ им. М.В. Келдыша РАН, 2008. С. 32.

МОНИТОРИНГ РАБОТЫ СТРУКТУРНЫХ ЭЛЕМЕНТОВ СПЕЦИАЛИЗИРОВАННОГО ВЫЧИСЛИТЕЛЬНОГО КОМПЛЕКСА ННЦ ХФТИ ДЛЯ ОБРАБОТКИ ДАННЫХ ЭКСПЕРИМЕНТА CMS (ЦЕРН)[1]

О. О. Бунецкий, С. С. Зуб, С. Т. Лукьяненко, А. С. Приставка, Д. В. Сорока

Национальный научный центр Харьковский физико-технический институт (ННЦ ХФТИ)
Украина, 61108, г. Харьков, ул. Академическая, 1
Телефон: +38 (057) 335-35-30, Факс: +38 (057) 335-16-88
www.kipt.kharkov.ua, kipt@kipt.kharkov.ua

Специализированный вычислительный комплекс (ВК) ННЦ ХФТИ является элементом всемирного LHC Grid, а с 2009 г. сертифицирован как Т2 центр в грид-инфраструктуре эксперимента CMS. На ВК хранятся и обрабатываются реальные экспериментальные данные с Большого адронного коллайдера, ресурсы ВК круглосуточно доступны пользователям всемирного Grid для вычислений. Это накладывает особые требования к надежности работы всех элементов и служб ВК. При этом особое значение имеет постоянный мониторинг качества электропитания и температуры окружающей среды в помещении ВК ННЦ ХФТИ. Такой мониторинг осуществляется с использованием комплексного решения, на основе средств мобильной связи и возможностей современных источников бесперебойного электропитания. Обсуждается система контроля целостности RAID массивов данных в дисковых серверах распределенного хранилища данных. Рассматривается контроль работоспособности сетевых интерфейсов структурных элементов ВК и системы обработки пакетных задач на вычислительных узлах комплекса со своевременным оповещением о сбоях.

Введение

В эксперименте CMS, как и в других крупных экспериментах на Большом адронном коллайдере (БАК), требуется с высокой скоростью обрабатывать огромный поток информации. Это накладывает чрезвычайно жесткие условия на вычислительные комплексы (ВК), предназначенные для решения этой задачи. Для обработки и анализа данных, аккумулируемых в экспериментах БАК, создана разветвленная грид-инфраструктура, называемая “Всемирный БАК-грид” (WLCG), одним из элементов которой является ВК ННЦ ХФТИ.

1. Вычислительный комплекс ННЦ ХФТИ

Первая очередь прототипа специализированного вычислительного центра для обеспечения работ в рамках эксперимента CMS в ННЦ ХФТИ была создана в 2001 году [1] и представляла собой Linux-кластер, построенный на нескольких процессорах типа Pentium III. Прототип постоянно модернизировался с обновлением его элементной базы и наращиванием ресурсов. К моменту запуска БАК он представляет собой уже полноценный, сертифицированный ВК, который готов к участию в распределенной обработке данных CMS и приближается по своим аппаратным характеристикам к параметрам “номинального” Т2-центра этого эксперимента [2] .

[1] Работа поддержана грантом молодых ученых НАН Украины 2009-2010 г.

Комплекс объединяет более 30 высокопроизводительных двухпроцессорных узлов с CPU архитектуры x86-64. Кроме того, два сервера (4 CPU) архитектуры AMD64 обеспечивают интерактивную работу пользователей (в частности, запуск задач анализа выборок данных CMS в среде WLCG, используя сконфигурированный на них интерфейс пользователя грид (UI)). Еще один двухпроцессорный (AMD64) сервер обслуживает обмен данными между комплексом и другими субъектами грид-инфраструктуры CMS. Согласно принятой в эксперименте процедуре обмена данными, на этом узле установлены CMS VObox и комплекс PhEDEx. Помимо этого, данный узел выполняет также функции Интернет-шлюза для рабочих узлов системы, и на нем сконфигурирован прокси-сервер (SQUID) для задач, использующих специализированное программное обеспечение эксперимента CMS.

Все узлы работают под ОС ‘Scientific Linux CERN’ (SLC) (версии 4 и 5). В соответствии с принятыми в WLCG стандартами для обработки пакетных задач используется система OpenPBS/Torque с планировщиком Maui.

Внутренняя сетевая инфраструктура построена на основе двух высокопроизводительных маршрутизаторов Cisco Catalyst 2960G и Hewlett-Packard ProCurve 2848. Внутренняя пропускная способность локальной сети составляет 1 Гбит/с. Ширина интернет канала составляет примерно 200 Мбит/с.

Комплекс также включает хранилище информации (SE) — массовую дисковую память типа DPM [3], которое является распределенным. Его образуют 13 дисковых серверов с аппаратными RAID5- и RAID6-дисковыми массивами, снабженными дисками быстрой («hot-spare») аварийной замены. Общая емкость этого SE составляла примерно 80 Тбайт.

С 2005 года специализированный ВК ННЦ ХФТИ, для участия в программе эксперимента CMS, зарегистрирован в структурах WLCG/EGEE под именем Kharkov-KIPT-LCG2. [4] После конфигурации и отладки необходимых версий программного обеспечения эксперимента CMS (CMSSW, PhEDEx и проч.), он был также зарегистрирован в базе данных CMS с именем T2_UA_KIPT. [5] Внешний вид ВК ННЦ ХФТИ приведен на рис 1.

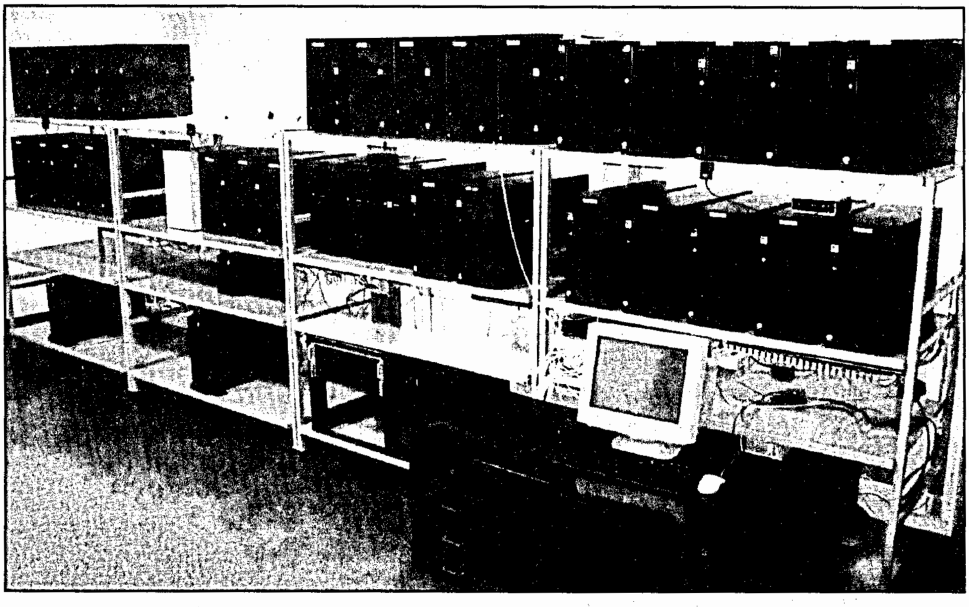


Рис. 1: Внешний вид вычислительного комплекса ННЦ ХФТИ. Справа на стойках расположены сервера распределенного дискового хранилища данных типа DPM, слева – счетные сервера. Внизу на стойках установлены источники бесперебойного питания APC Smart UPS RT 5000. Вверху на стойках закреплены датчики температуры

В июне 2009 г. комплекс T2_UA_KIPT успешно выполнил все сертификационные тесты CMS и с тех пор находится в составе группы региональных центров, готовых к участию в распределенной обработке данных CMS на уровне центра 2-го яруса WLCG.

2. Цели и задачи мониторинговых систем структурных элементов ВК ННЦ ХФТИ

Такое положение ВК не допускает сбоев и аварий в работе, так как это может привести к потере экспериментальных данных, которые накапливаются и хранятся на ВК ННЦ ХФТИ. Но работа ВК, состоящего из большого числа элементов (сетевые маршрутизаторы, счетные сервера, дисковые сервера, источники бесперебойного питания, кондиционеры и т.п.), без сбоев невозможна. Поэтому нами была создана комплексная мониторинговая система, которая круглосуточно «следит» за параметрами работы всех элементов T2_UA_KIPT и, в случае «аварий», своевременно информирует системных администраторов ВК по средствам SMS и E-mail. А в некоторых «простых» ситуациях мониторинговые службы способны самостоятельно устранить неисправности в работе ВК (например, очистка дисков от временных файлов или перезагрузка «зависшего» сервера). Так же все «аварии» и «сбои», обнаруженные мониторинговыми службами, записываются автоматически в лог-файл, который при необходимости можно просмотреть. Это помогает нам выявлять причины возникновения «аварий». Постоянно анализируя записи мониторинговых служб, мы усовершенствуем работу всего ВК в целом и повышаем его надежность и отказоустойчивость.

Мониторинг работы узлов ВК постоянно совершенствуется, и сегодня мы имеем комплексное решение на основе возможностей источников бесперебойного питания (ИБП) APC Smart UPS RT 5000, датчиков температуры, GSM-модема и программ – написанных администраторами ВК. Основные технические характеристики аппаратной части мониторингового комплекса приведены в Табл. 1.

Таблица 1. Технические характеристики аппаратной части мониторингового комплекса

НАИМЕНОВАНИЕ	ПАРАМЕТР
Источник бесперебойного питания APC Smart UPS RT 5000	
Диапазон входного напряжения при работе от сети	160 – 280 В
Максимальная выходная мощность	3,5 кВт
Время работы от аккумуляторной батареи при отключении электроэнергии	15,4 Минуты (1750 Вт)
Датчик температуры APC AP9619	
Диапазон рабочих температур	0° - 40° C
Длина шнура	3,6 м
Размеры (ширина х высота х глубина)	38 x 64 x 22 мм
GSM/GPRS/EDGE модем Teltonika T-modem PCI	
Рабочие диапазоны	GSM 900/1800 (EDGE, GPRS, HSCSD и CSD
Наличие внешней антенны	Да
Интерфейс	PCI
Совместимость с ОС	Windows, Linux

А начиналось все в 2003 г. с нескольких утилит, поставляемых производителями ИБП. С их помощью можно было следить за параметрами электропитания и при его отключении «корректно» выключать сервера, что уменьшало вероятность потери накопленной информации.

Об отправке, каких либо сообщений тогда не могло быть и речи. Информацию можно было получить, только просмотрев лог-файл или же визуально, когда сервер уже отключен.

Количество мониторинговых служб и программ постоянно растет. Основными на сегодня являются:

1. Мониторинг качества электропитания узлов ВК ННЦ ХФТИ;
2. Мониторинг температуры воздуха в помещении ВК;
3. Система мониторинга целостности жестких дисков в RAID массивах распределенного дискового хранилища данных ВК ННЦ ХФТИ;
4. Мониторинг доступности сетевых интерфейсов рабочих узлов ВК, как внутри сети, так и извне;
5. Система оповещения о «зависших» или устаревших задачах, полученных из Grid;
6. Система очистки распределенного дискового хранилища данных от временных файлов.

Далее рассмотрим принципы и параметры работы каждой системы, их отличия и особенности.

3. Мониторинговые службы электропитания ВК и температуры воздуха в помещении

Мониторинг качества электропитания узлов ВК ННЦ ХФТИ и мониторинг температуры воздуха в помещении серверной построены на одной аппаратно-программной базе. За основу взят программный комплекс APC PowerChute, который поставляется на диске, вместе с ИБП. Так же в комплект поставки ИБП входят датчик температуры и специальная плата (AP9619), которая имеет сетевой интерфейс RJ45 и разъем для подключения датчика температуры. Работа программы PowerChute основана на принципе клиент-сервер, где ИБП – сервер, а рабочие узлы ВК – клиенты. Система позволяет контролировать более 30 параметров бесперебойного электропитания ВК, включая температуру в помещении. Наиболее важные параметры это: входное и выходное напряжение электропитания, заряд аккумуляторной батареи ИБП, нагрузка на ИБП, время работы ИБП от аккумулятора при отключении электроэнергии. Все «события», происходящие в рамках программы PowerChute, записываются в лог-файл, хранящийся на ИБП. Система позволяет очень гибко настраивать параметры работы: следить или не следить за «нужным» параметром, отправлять сообщение при «событии» или просто записать информацию в лог-файл, а после критических «событий» (например «аккумулятор ИБП разряжен на 95%») позволяет корректно завершить работу узлов.

Дополнив комплекс APC PowerChute, GSM-модемом и несколькими своими программами, мы получили достаточно гибкую систему контроля электропитания ВК и температуры воздуха в помещении комплекса. Она позволяет не только своевременно оповещать администраторов ВК о «происшествиях», отправляя SMS на мобильные телефоны, но и в случае возникновения аварий с электропитанием или кондиционером (повышение температуры воздух в помещении ВК) «корректно» завершать работу рабочих узлов, предотвращая потерю накопленных данных и поломку дорогостоящего оборудования.

4. Система мониторинга целостности жестких дисков в RAID массивах распределенного дискового хранилища данных ВК ННЦ ХФТИ

Казалось бы, решена проблема с системой электропитания и поломок аппаратуры быть не должно, однако это не так. Это всего лишь уменьшило количество поломок, а не исключило их полностью. И для этого приходиться контролировать работу отдельных частей структурных элементов ВК ННЦ ХФТИ. Это работа и жестких дисков (ЖД), и сетевых карт и оперативной памяти и многого другого оборудования.

Контроль целостности ЖД налажен для RAID-5 и RAID-6 массивов в распределенном дисковом хранилище данных. Хоть RAID-5 и предполагает наличие «hot-spare» диска. Но мы должны знать, когда произошла подмена вышедшего из строя ЖД, «hot-spare» диском. Что бы своевременно заменить поврежденный ЖД и не допустить разрушения RAID массива и потери важных данных. Этот мониторинг базируется на работе TW_CLI утилиты из комплекта поставки системных программ 3ware RAID контроллера. Эта утилита имеет большое количество команд, при помощи которых можно получать информацию о состоянии RAID массива и каждого отдельного ЖД в нем. Далее нами написана программа, позволяющая выполнять нужные нам команды, а результат выполнения отправлять на мобильные телефоны и E-mail. Поместив эту программу в CronD ОС Linux, и настроив запуск на выполнения два раза в сутки, мы получили довольно своевременное оповещение о выходе из строя ЖД. Подробный алгоритм работы мониторинга целостности ЖД в RAID массивах распределенного дискового хранилища данных ВК ННЦ ХФТИ приведен на рис. 2.

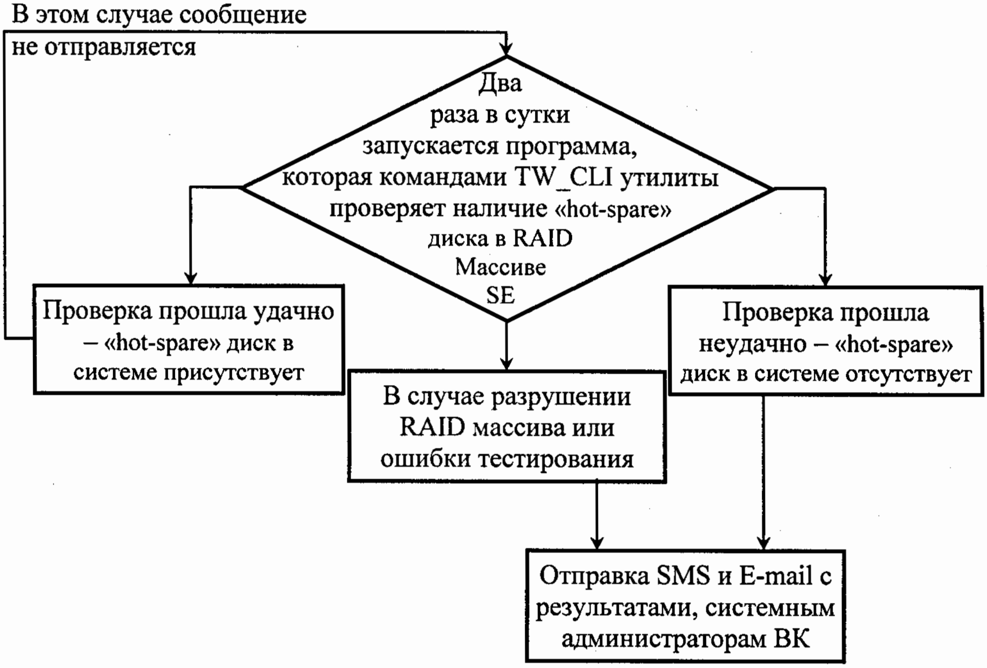

В этом случае сообщение
не отправляется
Два
раза в сутки
запускается программа,
которая командами TW_CLI утилиты
проверяет наличие «hot-spare»
диска в RAID
Массиве
SE
Проверка прошла удачно – «hot-spare» диск в системе присутствует
Проверка прошла неудачно – «hot-spare» диск в системе отсутствует
В случае разрушении RAID массива или ошибки тестирования
Отправка SMS и E-mail с результатами, системным администраторам ВК

Рис. 2: Алгоритм работы мониторинга целостности жестких дисков в RAID массивах распределенного дискового хранилища данных ВК ННЦ ХФТИ

5. Мониторинг доступности сетевых интерфейсов рабочих узлов ВК и подобные ему системы

Одним из важнейших требований, предъявляемых к T2_UA_KIPT, является его доступность 24 часа в сутки, 365 дней в году. Этот параметр (доступность ВК через интернет) необходимо постоянно контролировать. Для этого написана сложная, многоуровневая программа мониторинга, состоящая из нескольких модулей, которые самостоятельно работают в системе. Задача программы заключается в периодическом контроле (24 раза в сутки) доступности рабочих узлов ВК, как внутри сети, так и извне. Так же, программа может самостоятельно перезагрузить «зависший» рабочий узел, что избавляет людей от рутинной работы. Как и все выше упомянутые мониторинговые системы, эта система оповещает администраторов ВК о «происшествиях» (по средствам SMS и E-mail) и все результаты своей работы вносит в Log-файл, доступный для просмотра и анализа системными администраторами ВК. В отправленном сообщении находится информация о том, какой сервер не доступен и по какому IP адресу. Алгоритм работы программы напоминает алгоритм, приведенный на Рис. 2.

Аналогично построена работа программы проверяющей прохождение физических задач, поступивших из Grid. Критерием является время выполнения задачи. Если она считается больше положенного времени, значит процесс завис и бесполезно потребляет ресурсы *вычислительной системы. Если это произошло, отправляется сообщение с номерами выполняемого процесса и задачи.* Этой информации, полученной своевременно, достаточно для того, чтобы удалить задачу из очереди и дать возможность считаться другим задачам, тем самым не загружая счетные сервера зависшими процессами. Результаты работы этой программы автоматически записываются в Log-файл.

Проблема бесполезного потребления вычислительных мощностей на ВК всегда актуальна. Поэтому постоянно приходиться оптимизировать работу ВК, следить за прохождением задач, наличием свободного места на дисках распределенного дискового хранилища данных и многим другим.

В процессе работы, на дисковых серверах накапливаются временные файлы, они должны удалятся автоматически, но иногда этого не происходит, из-за «глюков» системы. Удаление временных файлов выполняет еще одна программа, написанная нами. Она запускается автоматически один раз в три дня. Ее задачей является выявление и удаление автоматически не удаленных временных файлов. По завершению выполнения, программа отправляет результаты по электронной почте. Регулярно просматривая отчет, можно постоянно контролировать работу дисковых серверов.

Заключение

Имея такой мониторинговый комплекс на ВК, который постоянно совершенствуется и дополняется новыми возможностями, мы всегда своевременно информированы о «происшествиях». Это позволяет уменьшить вероятность выхода из строя дорогостоящего оборудования и потерю накопленных данных. Имея историю сбоев, мы можем проводить их анализ и оптимизировать работу ВК. Так же некоторые мониторинговые программы постоянно «освобождают» бесполезно используемые ресурсы ВК, таким образом повышая производительность системы.

Литература

[1] Levchuk L.G., Sorokin P.V., Soroka D.V., Trubnikov V.S. Вопросы атомной науки и техники. Сер. “Ядерно-физические исследования”, 2(40) (2002) 49.

[2] Bayatian G. et al. The CMS collaboration. CMS: The Computing Project; Technical Design Report, CERN-LHCC-2005-023, CMS TDR 7, 2005.

[3] LCG Disk Pool Manager (DPM) Administrators Guide, https://twiki.cern.ch/twiki/bin/view/LCG/DpmAdminGuide

[4] Zub S., Levchuk L., Sorokin P., Soroka D. Nucl. Instr. and Meth. A 559 (2006) 35.

[5] CMS SiteDB: Site Directory, https://cmsweb.cern.ch/sitedb

СЕМАНТИЧЕСКИЙ ГРИД, ОСНОВАННЫЙ НА КОНЦЕПЦИИ ПРЕДМЕТНЫХ ПОСРЕДНИКОВ

А. Е. Вовченко, Л. А. Калиниченко, С. А. Ступников

Институт проблем информатики РАН
Россия, 119333, Москва, Вавилова, д. 44, кор. 2
itsnein@gmail.com, leonidk@synth.ipi.ac.ru, ssa@synth.ipi.ac.ru

In frame of the SYNTHESIS project being developed at IPI RAS a semantic grid infrastructure has been designed and implemented. The semantic grid is built on top of the AstroGrid that is positioned as an information grid aimed at support of virtual observatories. The infrastructure proposed significantly extends the conventional vision of semantic grids emphasizing a provision of well-defined meaning to data and service resources. We consider that such resources obtain adequate meaning in the context of specific application problems. Therefore we focus on ontological and conceptual specifications of application domains and problems. Such specifications are reflected in the definitions of *mediators* independently of information resources accessible through the information grid. The mediation middleware is positioned between the applications/users and resources. The mediation based semantic grid is used currently for problem solving in the area of astronomy. Note that mediation based semantic grid middleware is independent of particular information grid and can be integrated with any such infrastructure.

1. Введение

В настоящее время наблюдается существенный рост объема получаемых экспериментальных данных в различных областях науки. Разнородность информации вызвана большим числом организаций, накапливающих данные (в результате наблюдения, измерения), разнообразием объектов наблюдения, совершенствованием техники наблюдений. Это приводит к необходимости использования неоднородной, распределенной информации, накопленной в течение значительного периода наблюдений технологически различными инструментами.

Разрыв между исследователями и источниками данных и сервисов приводит к необходимости поиска новых путей создания информационных систем, в которых особое внимание было бы сосредоточено на специальных средствах организации решения задач над множеством распределенных информационных ресурсов, накапливаемых в разнообразных научных центрах. Разработан ряд инфраструктур, которые технически способствуют организации решения задач в такой среде. Среди них веб-сервисы, гриды, Семантический Веб, интероперабельные инфраструктуры промежуточного слоя и др.

Настоящая статья ограничивается рассмотрением информационных грид-инфраструктур для науки, которые в последнее время становятся все более востребованными. Принято различать вычислительные, информационные, облачные грид-инфраструктуры. До сих пор основное внимание при реализации таких инфраструктур было сосредоточено на организации множества компьютеров для достижения высокой производительности вычислений или на создании среды для обеспечения доступа к большому числу распределенных информационных ресурсов и их интероперабельности при решении задач. Вопросы семантики предметных областей задач, ее связи с семантикой информационных ресурсов грида, декларативного программирования приложений на основе таких семантических определений в грид-инфраструктурах практически не рассматривались. В настоящей работе кратко излагается подход к созданию прототипа *семантического грида*, компонентами которого могут быть как отдельные информационные ресурсы (базы данных и сервисы), так и вычислительные и информационные гриды со своими информационными ресурсами. Основная идея семантического грида заключается в формулировании задач на основе спецификации предметной области задачи независимо от релевантных задаче ресурсов и реализации такой формулировки в грид-инфраструктуре. Предполагается, что сами ресурсы, зарегистрированные в грид-среде, снабжены семантическими определениями, достаточными для принятия решения о целесообразности их использования при решении конкретной задачи.

Статья структурирована следующим образом. Во втором разделе рассматривается подход к решению научных задач над неоднородными информационными ресурсами на основе концепции предметных посредников в грид-инфраструктурах. В третьем разделе представлено описание различных типов грид-инфраструктур и семантического грида, основанного на концепции предметных посредников. В четвертом разделе представлена инфраструктура семантического грида, основанного на предметных посредниках, а также разработанный прототип. В пятом разделе описан пример применения разработанного прототипа для решения конкретных научных задач. Заключение статьи подводит итог обсуждению и намечает планы дальнейшего развития работы.

2. Концепция предметных посредников для организации решения научных задач над неоднородными информационными ресурсами

Основной идеей в инфраструктуре доступа к множественным неоднородным информационным источникам является введение промежуточного слоя между информационными ресурсами и потребителями информации. Основными компонентами промежуточного слоя являются предметные посредники [1], существующие независимо от информационных ресурсов. Уровень предметных посредников вводится как часть информационных систем, создаваемых для решения научных задач. Каждый предметный посредник задает спецификацию предметной области для решения некоторого класса задач, используя каноническую информационную модель для представления предметной области [2] и унифицированного отображения разнообразных видов моделей информационных ресурсов.

Различаются два принципиально различных подхода к проблеме интегрированного представления описания предметной области задачи по отношению к множеству релевантных задаче информационных ресурсов:

- *двигаясь от ресурсов к задачам*, при этом схема посредника образуется как интегрированная схема множества ресурсов независимо от приложения;
- *двигаясь от приложения к ресурсам*, при этом описание предметной области приложения образуется независимо от ресурсов в терминах понятий, структур данных, функций, процессов, а затем релевантные приложению ресурсы отображаются в это описание.

Первый подход, *движимый информационными ресурсами*, является немасштабируемым по отношению к числу ресурсов, не дает возможности достижения семантической интеграции ресурсов в контексте конкретного приложения, не ведет к доказательной идентификации релевантных приложению ресурсов, не способствует повышению стабильности спецификации посредника в процессе эволюции ресурсов, релевантных приложению.

Движимый приложениями подход предполагает создание предметного посредника, который поддерживает взаимодействие между приложением и ресурсом на основе определения прикладной области (определения посредника). Второй подход имеет очевидные преимущества по отношению к подходу, движимому информационными ресурсами. Процесс регистрации неоднородных информационных ресурсов в предметном посреднике в подходе, движимом приложениями, основан на технике GLAV[3], комбинирующей два подхода: LAV (Local As View), при котором схемы регистрируемых ресурсов рассматриваются как материализованные взгляды над виртуальными классами посредника, и GAV (Global As View), при котором глобальная схема посредника является взглядом над схемами ресурсов. В этом случае GAV взгляды служат для разрешения различных конфликтов между спецификациями ресурсов и посредника. Подобная техника регистрации обеспечивает стабильность спецификации приложения при изменении конкретных информационных ресурсов и их фактического присутствия (удаление ресурса, добавление новых ресурсов), а также масштабируемость посредников по отношению к числу регистрируемых ресурсов. Настоящая статья основана, главным образом, на подходе, движимом приложениями.

3. Семантический грид, основанный на концепции предметных посредников

Целью любой грид-инфраструктуры является совместное использование распределенных ресурсов для решения сложных задач. Предполагается, что информационный грид предоставляет для решения задач следующие средства:

- совокупность ресурсов - готовых программ решения задач, библиотек методов, информационных ресурсов – онтологий, баз данных, файлов, и т.д.;
- реестры, содержащие метаописания всех представленных ресурсов;;
- средства программирования в конкретной грид-среде;
- стандартные интерфейсы для доступа к ресурсам, например, интерфейсы сервисов в SOA.

Семантические гриды предоставляют семантические описания ресурсов и возможности простого поиска ресурсов и сервисов, обеспечивая их интероперабельное и интегрированное совместное использование в контексте задачи, формулируемой в терминах предметной области.

Предлагаемая в данной работе инфраструктура семантического грида состоит из четырех слоев:

- слой предметной области задачи (Application Problem Domain);
- слой предметных посредников (Semantic Mediation Middleware);
- слой грид-инфраструктур (Computational and Information Resource Environments)
- слой информационных ресурсов.

Для решения задач в инфраструктуре используется метод, движимый приложениями. Отправляясь от предметной области задачи, определяется онтология предметной области (понятия и связи между ними), строится концептуальная схема предметной области, содержащая информационные структуры и методы, необходимые для решения задачи [4]. Таким образом, получается семантическая спецификация решения задачи, полностью независимая от конкретных ресурсов. После этого определяются грид-инфраструктуры, необходимые для решения задачи. Если задача особенно сложна, может понадобиться совокупность баз данных информационного грида и ряд методов (сервисов) обработки информации, поддерживаемых вычислительным гридом. Далее, на уровне ресурсов идентифицируются ресурсы, релевантные задаче, используя реестры доступных грид-инфраструктур.

Для того, чтобы реализовать подобную схему решения задач, опираясь на семантику предметной области, а также обеспечить отображение в эту схему конкретных ресурсов, релевантных задаче, и, возможно, принадлежащих различным средам, необходим промежуточный слой, обеспечивающий взаимодействие программ приложений (или пользователей) с множеством ресурсов в процессе решения задачи - слой предметных посредников. Принципиально важным компонентом этого слоя является каноническая информационная модель (язык СИНТЕЗ [5]), используемая как для спецификации предметных областей и посредников, так и для унифицированного представления разнообразных информационных ресурсов, необходимых для решения задачи. Также каноническая модель используется для декларативной спецификации отображений спецификаций ресурсов в абстрактные спецификации посредника.

Каждый предметный посредник определяется концептуальной схемой задачи (или ее частью). Построение отображений спецификаций ресурсов в спецификации посредника реализуется в процессе регистрации ресурсов в посреднике [6].

Промежуточный слой предметных посредников вместе со связанными с ним гридами образует *семантический* грид. Благодаря спецификации задач (посредников) в абстрактных терминах предметной области и отображениям спецификаций конкретных ресурсов в такие семантические спецификации обеспечивается интероперабельность совокупности неоднородных ресурсов, поддерживаемых одной или несколькими грид-средами.

4. Прототип семантического грида, основанного на концепции предметных посредников

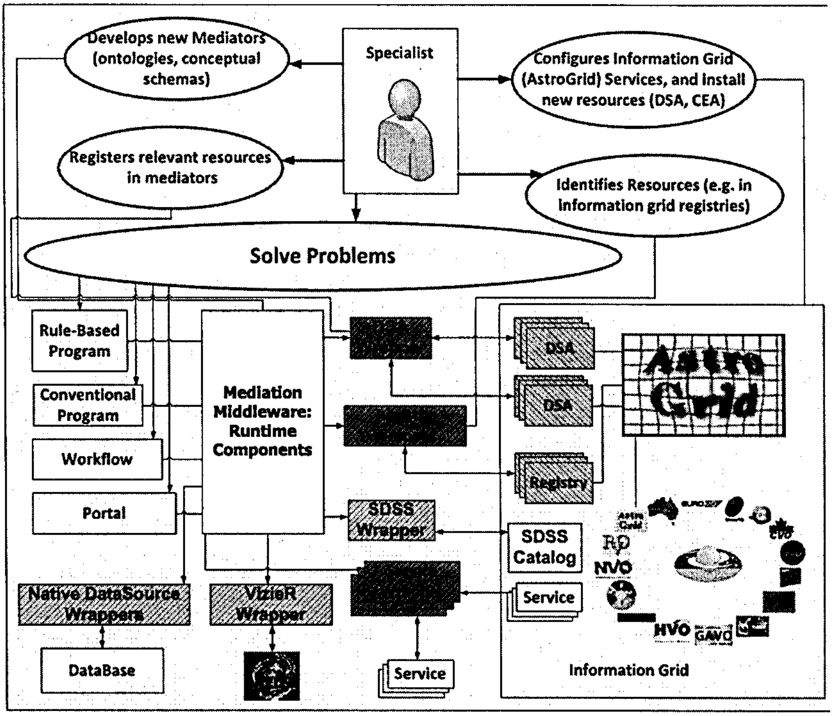

Develops new Mediators (ontologies, conceptual schemas)
Specialist
Configures Information Grid (AstroGrid) Services, and install new resources (DSA, CEA)
Registers relevant resources in mediators
Identifies Resources (e.g. in Information grid registries)
Solve Problems
Rule-Based Program
Conventional Program
Workflow
Portal
Mediation Middleware: Runtime Components
SDSS Wrapper
Native DataSource Wrappers
VizieR Wrapper
DataBase
Service
DSA
DSA
Registry
Astro Grid
SDSS Catalog
Service
RQ
NVO
HVO
GAVO
Information Grid

Рис. 1: Структура программных средств поддержки семантического грида

В соответствии с разработанной инфраструктурой для решения научных задач был реализован прототип семантического грида (Рис. 1). В качестве грид-инфраструктур использовались системы АстроГрид [7] и VizieR [8].

Система АстроГрид нацелена на поддержку инфраструктуры для решения научных задач в виртуальных обсерваториях (ВО), предоставляющей средства доступа к астрономическим каталогам и реестрам метаданных, в которых регистрируются ресурсы ВО. Система АстроГрид включает следующие основные компоненты:

- *Registry* (реестр) представляющий собой коллекцию метаданных — XML-документов, описывающих ресурсы, которые могут использоваться при решении задач с помощью ВО. Реестр реализован на основе стандарта OAI PMH [9], специализированного IVOA (Альянс Международной Виртуальной Обсерватории [10]) для нужд ВО;
- *Community*, обеспечивающий регистрацию и персональную аутентификацию пользователей;
- *MySpace* — виртуальное хранилище данных, к которым могут иметь доступ все сервисы системы АстроГрид;
- *Common Execution Architecture* (CEA — Общая исполнительная архитектура), определяющая способ оформления приложения в виде сервиса АстроГрид;
- *DataSet Access* (DSA), реализующая подключение баз данных к системе АстроГрид.

VizieR может рассматриваться как информационный грид для доступа к большинству астрономических каталогов (около восьми тысяч каталогов). Пользователю предоставляется веб-интерфейс для поиска каталогов по ключевым словам и задания запросов к каталогам.

Прототип семантического грида, основанного на предметных посредниках, предоставляет пользователям следующие возможности:

- разработка онтологий и концептуальных схем новых предметных посредников;
- обнаружение релевантных задаче ресурсов;
- регистрация релевантных ресурсов в посреднике, включающая:
  - определение онтологий и концептуальных схем ресурсов;
  - определение отображений спецификаций ресурсов в спецификации посредников [11];
- регистрация в системе АстроГрид новых ресурсов с использованием DSA и CEA и последующая регистрация их в предметных посредниках (в том случае, если ресурсы, необходимые для решения задачи, не найдены, или их недостаточно);
- непосредственное формулирование задач в виде:
  - программ на языке правил канонической информационной модели;
  - программ на традиционных языках программирования (в текущей версии прототипа – на языке Java);
  - управление потоками работ (в текущей версии прототипа – средства управления потоками работ системы АстроГрид AG Python).

Для поддержки названных возможностей в рамках реализации прототипа были разработаны следующие компоненты:

- портал, обеспечивающий возможности конфигурирования посредников, задания программ и отображение результатов;
- адаптеры (wrappers) - специальные программы, обеспечивающие унифицированный доступ к разнородным ресурсам из посредников: преобразование запросов на языке программ посредника в запросы на языке ресурса, получение результата запроса от другого ресурса, а также преобразование результатов запросов в объекты схемы посредника [12], в том числе:
  - адаптер реляционных баз данных (*Native DataSource Wrapper*);
  - адаптер к астрономическому каталогу данных SDSS [13] (*SDSS Wrapper*) с поддержкой возможности выполнения процедуры кросс-идентификации астрономических объектов XMatch на сервере SDSS;
  - адаптер DSA-ресурсов, зарегистрированных в системе АстроГрид (*DSA Wrapper*),
  - адаптер реестров АстроГрида (*RegistryWrapper*), для осуществления поиска по метаданным ресурсов или приложений в реестрах системы АстроГрид;
  - адаптер ресурсов системы VizieR (*VizieR Wrapper*);
  - адаптер системы поиска VizieR, для осуществления поиска по метаданным ресурсов или приложений в системы Vizier;
  - адаптер сервисов (*Service Wrapper*).
- средства переписывания и планирования программ над схемой посредника в частичные программы над ресурсами (Runtime Components) [14].

5. Пример решения научной задачи

5.1. Концептуальная схема задачи

Задача *определения вторичных стандартов для фотометрической калибровки оптических компонентов космических гамма-всплесков* (далее просто *задача*) поставлена Институтом космических исследований РАН [15]. Поиск стандартов необходим для проведения фотометрии оптического компонента гамма-всплесков. В качестве стандартов необходимо использовать звезды с определенным блеском, находящиеся в небольшой окрестности конкретного наблюдения гамма-всплеска. Стандарты должны быть звездами (а не галактиками или артефактами), не переменными, изолированными, близкими по цвету к ожидаемому цвету гамма-всплеска, с малым собственным движением. Фотометрические оценки блеска звезд должны иметь минимальную ошибку (статистическую и систематическую), и таких звезд должно быть не менее пятнадцати для того, чтобы оставить специалисту возможность выбора стандартов по критериям конкретного наблюдения. Таким образом, задача определения стандартов включает выбор кандидатов в стандарты, вычисление количественного критерия качества стандарта и определение фотометрических значений стандарта в различных фотометрических системах.

Общий процесс решения задачи в инфраструктуре семантического грида выглядит следующим образом:

- построение глоссария предметной области;
- построение онтологии предметной области и онтологий ресурсов;
- создание концептуальной схемы посредника;
- регистрация в предметном посреднике ресурсов, релевантных задаче;
- формулирование задачи в виде программы или потока работ над концептуальной схемой и запуск ее на необходимых входных данных.

В процессе анализа описания задачи были выявлены фрагменты текста, определяющие термины или задающие их ограничения. По таким фрагментам, с использованием существующих астрономических онтологий [16], термины были определены вербально, т.е. были составлены их текстовые описания. Например, для термина *Magnitude* (звездная величина) было составлено определение «logarithmic measure of the brightness of an object, measured in a specific passband in particular epoch». В вербальных определениях терминов была произведена идентификация связанных с ними существенных терминов, и глоссарий был дополнен этими терминами. Так, например, термин *Epoch* (эпоха) является частью определения термина *Magnitude.* Далее была проведена аннотация новых терминов вербальными определениями. Этот процесс повторялся до некоторого насыщения глоссария.

На основе анализа вербальных определений терминов контекста решаемой задачи были выявлены онтологические понятия и связи между ними. Так, понятие *PhotometricSystem* связано отношением один-ко-многим с понятием *Passband.* Также были оценены и специфицированы ограничения отношений, специфические для предметной области. Так, например, свойство *epoch* понятия *Magnitude* имеет кардинальность 1.

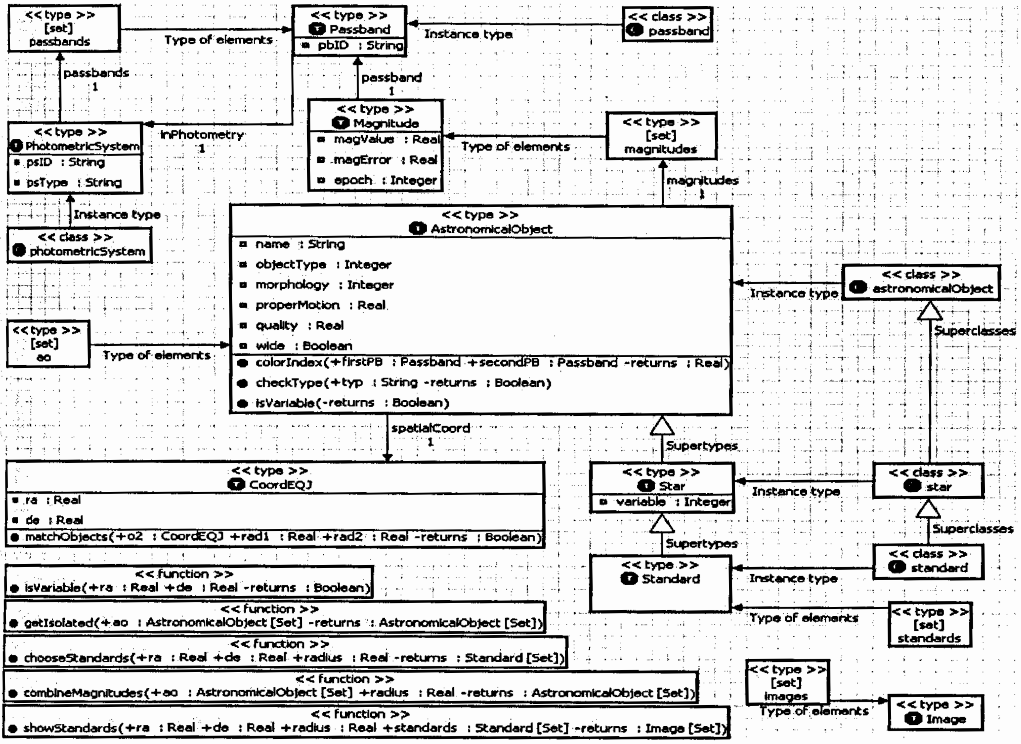
На основании анализа описания задачи и построенной онтологии была создана концептуальная схема предметной области для решения задачи (Рис. 2). Были выявлены следующие основные типы данных, необходимые для решения задачи:

- экваториальные координаты (CoordEQJ);
- фотометрическая система (PhotometricSystem);
- фотометрическая полоса (Passband);
- звездная величина в некоторой фотометрической системе (Magnitude);
- абстрактный астрономический объект (Astronomical Object);
- звезда (Star);
- стандарт (Standard);
- изображение (Image);

а также методы и функции:

- кросс-идентификация объектов (matchObjects);
- вычисление цветового индекса (colorIndex);
- проверка, является ли данный объект объектом некоторого определенного типа – звездой, галактикой и т.д. (checkType);

- проверка переменности звезды на основе фотометрических параметров (isVariable).



<< type >> [set] passbands
Type of elements
<< type >> Passband
pbID : String
Instance type
<< class >> passband
passbands 1
passband 1
<< type >> PhotometricSystem
psID : String
psType : String
inPhotometry 1
<< type >> Magnitude
magValue : Real
magError : Real
epoch : Integer
Type of elements
<< type >> [set] magnitudes
magnitudes 1
Instance type
<< class >> photometricSystem
<< type >> AstronomicalObject
name : String
objectType : Integer
morphology : Integer
properMotion : Real
quality : Real
wide : Boolean
colorIndex(+firstPB : Passband +secondPB : Passband -returns : Real)
checkType(+typ : String -returns : Boolean)
isVariable(-returns : Boolean)
Instance type
<< class >> astronomicalObject
Superclasses
<<type >> [set] ao
Type of elements
spatialCoord 1
Supertypes
<< type >> CoordEQJ
ra : Real
de : Real
matchObjects(+o2 : CoordEQJ +rad1 : Real +rad2 : Real -returns : Boolean)
<< type >> Star
variable : Integer
Instance type
<< class >> star
Superclasses
Supertypes
<< type >> Standard
Instance type
<< class >> standard
Type of elements
<< type >> [set] standards
<< function >> isVariable(+ra : Real +de : Real -returns : Boolean)
<< function >> getIsolated(+ao : AstronomicalObject [Set] -returns : AstronomicalObject [Set])
<< function >> chooseStandards(+ra : Real +de : Real +radius : Real -returns : Standard [Set])
<< function >> combineMagnitudes(+ao : AstronomicalObject [Set] +radius : Real -returns : AstronomicalObject [Set])
<< function >> showStandards(+ra : Real +de : Real +radius : Real +standards : Standard [Set] -returns : Image [Set])
<< type >> [set] images
Type of elements
<< type >> Image

Рис. 2: Концептуальная схема задачи определения стандартов

5.2. Процесс решения задачи

Задача определения стандартов была сформулирована в виде программы над концептуальной схемой, рассмотренной в предыдущем разделе. Параметром программы является площадка на небесной сфере, в которой произошел гамма-всплеск. Площадка характеризуется центром с координатами *queryRA, queryDE* и радиусом *radius*. Программу можно разбить на пять последовательных шагов.

Шаг 1.

На первом шаге среди всех астрономических объектов выбираются те, что попадают в указанную площадку. При этом нас интересуют только координаты (*ra, de*), звездные величины в различных полосах (*magnitudes*), тип объекта (*objectType*), собственное движение (*properMotion*) и качество данных (quality). Запрос к посреднику, представляющий собой правило языка СИНТЕЗ [5] (подобное правилам языка Datalog) выглядит следующим образом:

```
r(x/[ra, de, magnitudes, objectType, properMotion, quality])
:- astronomicalObject(x1/[ra: spatialCoord.ra, de: spatialCoord.de, objectType,
properMotion, quality, magnitudes])
& ra < queryRA + radius & ra > queryRA - radius
& de < queryDE + radius & de > queryDE - radius
```

Правило продуцирует коллекцию *r* объектов, содержащих только необходимые атрибуты и удовлетворяющих ограничениям на координаты, указанным в теле правила, из коллекции всех доступных астрономических объектов (*astronomicalObject*). В виртуальной коллекции *astronomicalObject* интегрированы следующие информационные ресурсы:

- из информационного грида АстроГрид:

   - каталог USNO-B1 (US Naval Observatory). Покрывает все небо, включает наблюдения 10^9 объектов за последние 50 лет, сканированные с пластин Шмидта;
   - каталог 2MASS (Two Micron All-Sky Survey). Обзор всего неба на длине волны 2 микрона, включает наблюдения $3*10^8$ объектов за 1997-2001 годы;
- из информационного грида VisieR:
   - каталог GSC (Guide Star Catalog). Включает наблюдения $945*10^6$ объектов, полученных с телескопа «Хаббл»;
   - каталог UCAC3 (The Third USNO CCD Astrograph Catalog). Покрывает все небо, включает наблюдения 10^8 объектов;
- каталог SDSS (Sloan Digital Sky Survey). Включает наблюдения $357*10^6$ объектов северной части неба с 2.5-метрового телескопа обсерватории Апач Пойнт, Нью-Мексико.

Шаг 2.

На втором шаге конструируются объекты, содержащие звездные величины из всех возможных ресурсов. Для этого производится кросс-идентификация объектов из разных ресурсов, после чего все одинаковые параметры отбрасываются за исключением звездных величин, которые объединяются в одно множество. Данная часть программы представляет собой вызов соответствующей функции:

```
combineMagnitudes (r/AstronomicalObject, r1);
```

Тем самым производится объединение фотометрических данных из различных астрономических каталогов.

Шаг 3.

На третьем шаге отсеиваются неизолированные объекты:

```
getIsolated(r1, r2);
```

На вход функции *getIsolated* поступает коллекция *r1*, полученная на предыдущем шаге, в результирующую коллекцию *r2* попадают только изолированные объекты (т.е. такие, в некоторой окрестности которых на небесной сфере не наблюдается других объектов).

Шаг 4.

На четвертом шаге отсеиваются переменные объекты и галактики, и выбираются звезды с очень малым собственным движением и качественными фотометрическими данными:

```
r3(x/[ra, de, magnitudes])
:- r2(x1/[ra, de, objectType, properMotion, quality, magnitudes])
& checkType(x1, 'G', nType) & nType = false
& isVariable(x1, isVar) & isVar = false
& objectType = Star
& properMotion < 0.01
& quality  < 0.01
```

Все подходящие объекты (из структуры которых остаются только координаты и звездные величины) попадают в коллекцию *r3*, определенную в голове правила. Выбираются объекты из коллекции *r2*, полученной на предыдущем шаге. При помощи функции *checkType* выбираются те объекты, тип которых не есть *G* (галактика). При помощи функции *isVariable* выбираются только объекты, не являющиеся переменными. Также проверяются условия на тип объекта (*objectType = Star*), собственное движение (*properMotion < 0.01*) и качество фотометрии (*quality < 0.01*).

Функция *isVariable* проверяет переменность звезд, основываясь на интегрированной информации следующих ресурсов:

- из информационного грида VisieR – каталог VSX (The International Variable Star Index). Составлен American Association of Variable Star Observers, включает $18*10^4$ переменных звезд;
- NSVS (Northern Sky Variability Survey). Покрывает северную часть неба, включает наблюдения $14*10^6$ объектов, полученной с роботизированной системы телескопов в Лос-Аламосе;

- ASAS (All Sky Automated Survey variable stars). Составлен Обсерваторией университета Варшавы;
- GSVC (General Catalogue of Variable Stars). Составлен Институтом астрономии РАН и ГАиШ МГУ, включает $78*10^3$ переменных звезд;

Функция *checkType* определяет тип объекта, используя веб-сервис к NASA/IPAC Extragalactic Database, включающей данные о $8*10^5$ объектов.

Шаг 5.

На четвертом шаге в коллекцию *r3* попадают звезды - кандидаты в стандарты. На последнем же шаге кандидаты маркируются на изображениях площадки гамма-всплеска и предоставляются пользователю для утверждения:

```
r4(im/Image):- showStadards(queryRA, queryDE, radius, r3)
```

В результирующую коллекцию изображений *r4* попадают изображения площадки с заданными координатами *queryRA, queryDE* и радиусом *radius*, на которых промаркированы кандидаты из коллекции *r3*, полученной на предыдущем шаге.

6. Заключение

В настоящей статье рассматриваются первые результаты создания семантического грида, основанного на концепции предметных посредников для решения научных задач над множеством неоднородных распределенных информационных ресурсов. Введение семантического грида призвано решить ряд семантических проблем взаимодействия ученого, решающего задачу в некоторой предметной области, и разнообразных релевантных задаче результатов наблюдений и средств их обработки. В исследованной архитектуре предметных посредников реализован подход, движимый приложениями, при котором для класса приложений формируется спецификация предметной области независимо от существующих информационных ресурсов. Далее происходит идентификация ресурсов, релевантных задаче, и их регистрация в посреднике на основе техники GLAV.

Стоит отметить два важных момента. Первый заключается в возможности использования произвольных гридов (вычислительных, информационных) для решения задач. Каждый грид представлен абстрактно - своими информационными ресурсами: базами данных, файлами, сервисами (под которыми скрываются программы из библиотек программ соответствующего грида), и пр. Для пользования этими ресурсами нужны реестры ресурсов и согласованные интерфейсы для подключения к семантической среде поддержки решения задач - среде посредников. В случае наличия интерфейсов, реестров и ресурсов конкретный грид может быть включен в семантический грид, основанный на предметных посредниках и использоваться для решения задач.

Второй момент заключается в том, что над совокупностью грид-базированных ресурсов создается семантический промежуточный слой (слой предметных посредников) для придания всей совокупности грид-ресурсов унифицированных представлений. Унифицируются спецификации концептуальных моделей решения задач, формулируемых в терминах предметных областей и воплощаемых в посредниках. Унифицируются спецификации нужных задаче (посреднику) ресурсов и задания отображений отобранных ресурсов в спецификации посредников. Унифицируется рассредоточенное планирование реализации концептуальных моделей и рассредоточенное исполнение программ поддержки решения задач над множеством грид-ресурсов. Таким образом, пользователь решает задачу в общих терминах, не заботясь о том, какие конкретно грид-инфраструктуры используются для решения задачи.

Полученные результаты свидетельствуют о перспективности исследованного подхода, существенное развитие которого планируется в ряде направлений. Планируется использование семантического грида, основанного на предметных посредниках при решении разнообразных научных задач. Возможности разработанной инфраструктуры были продемонстрированы при решении задачи определения вторичных стандартов для фотометрической калибровки оптических компонентов космических гамма-всплесков.

Литература

[1] Брюхов Д.О., Вовченко А.Е., Железнова О.П., Захаров В.Н., Калиниченко Л.А., Мартынов Д.О., Скворцов Н.А., Ступников С.А. Архитектура Промежуточного слоя Предметных Посредников для Решения Задач над Множеством Интегрируемых Неоднородных Распределенных Информационных Ресурсов в Гибридной Грид-Инфраструктуре Виртуальных Обсерваторий. Информатика и ее Применения, 2008. Т. 2. Вып. 1. С. 2-34.

[2] Захаров В.Н., Калиниченко Л.А., Соколов И.А., Ступников С.А. Конструирование Канонических Информационных Моделей для Интегрированных Информационных Систем. Информатика и ее Применения, 2007. Т.1, Вып. 2. С.15-39.

[3] Рябухин О.В., Брюхов Д.О., Калиниченко Л.А. Формирование выражений взглядов в задаче регистрации ресурсов в предметных посредниках. RCDL'2009, Петрозаводск, Россия, 2009.

[4] Вовченко А.Е., Захаров В.Н., Калиниченко Л.А., Ковалёв Д.Ю., Рябухин О.В., Скворцов Н.А., Ступников С.А. От спецификаций требований к концептуальной схеме. RCDL'2010, Казань, Россия, 2010.

[5] Kalinichenko L.A., Stupnikov S.A., Martynov D.O. SYNTHESIS: A language for canonical information modeling and mediator definition for problem solving in heterogeneous information resource environments. — M.: IPI RAS, 2007. P. 171.

[6] Briukhov D.O., Kalinichenko L.A., Martynov D.O. Source Registration and Query Rewriting Applying LAV/GLAV Techniques in a Typed Subject Mediator// Proc. of the Ninth Russian Conference on Digital Libraries RCDL'2007. Pereslavl-Zalesskij: Pereslavl University, 2007. P. 253-262.

[7] AstroGrid, http://www.astrogrid.org

[8] VizieR, http://vizier.u-strasbg.fr/cgi-bin/VizieR

[9] The open archives initiative protocol for metadata harvesting. Protocol Version 2.0 of 2002-06-14, Document Version 2004/10/12T15:31:00Z, http://www.openarchives.org/OAI/2.0/ openarchivesprotocol.htm

[10] International Virtual Observatory Alliance, http://ivoa.net/

[11] Kalinichenko L.A., Stupnikov S.A. Constructing of Mappings of Heterogeneous Information Models into the Canonical Models of Integrated Information Systems. Advances in Databases and Information Systems // Proc. of the 12th East-European Conference. Pori: Tampere University of Technology, 2008. P. 106-122.

[12] Stupnikov S.A., Kalinichenko L.A. Methods for Semi-automatic Construction of Information Models Transformations. Proc. of the 13th East-European Conference Advances in Databases and Information Systems, workshop Model – Driven Architecture: Foundations, Practices and Implications (MDA). Riga: Riga Technical University, 2009. P. 432-440.

[13] The Sloan Digital Sky Survey, http://www.sdss.org/

[14] Вовченко А.Е., Крупа А.В. Планирование запросов над множеством неоднородных распределенных информационных ресурсов в архитектуре средств поддержки предметных посредников. RCDL'2009, Петрозаводск, Россия, 2009.

[15] Вовченко А.Е., Вольнова А.А., Денисенко Д.В., Калиниченко Л.А., Куприянов В.В., Позаненко А.С., Скворцов Н.А., Ступников С.А. Применение средств виртуальной обсерватории для выбора вторичных стандартов поля при фотометрии оптического послесвечения гамма-всплесков. Труды Всероссийской Астрономической Конференции, ВАК-2010.

[16] Ontology of Astronomical Object Types. Version 1.3., http://www.ivoa.net/Documents/Notes/ AstrObjectOntology/

РАСШИРЯЕМАЯ ОБЪЕКТНО-ОРИЕНТИРОВАННАЯ СИСТЕМА РАСПРЕДЕЛЕННЫХ ВЫЧИСЛЕНИЙ[1]

К. Ю. Войтиков[1], А. Н. Моисеев[2], П. Н. Тумаев[3]

[1, 3]Филиал Кемеровского государственного университета в г. Анжеро-Судженске,
652471, Российская Федерация, Кемеровская обл., Анжеро-Судженск г., Ленина ул., 8
[2]Томский государственный университет
[1]kost_v@ngs.ru, [2]amoiseev@ngs.ru, [3]pavel.tumaev@gmail.com

В Анжеро-Судженском филиале Кемеровского государственного университета совместно с Томским государственным университетом ведутся работы по исследованию характеристик систем массового обслуживания различных конфигураций. Так как результатами таких исследований являются аналитические выражения, то требуется какое-то фактическое подтверждение того, что они соответствуют реальным событиям, описываемым системой. Но в большинстве случаев рассматриваемые модели являются довольно абстрактными, аналогов которых в реальном мире либо не существует, либо их очень трудно пронаблюдать и измерить.

В связи с этим было решено использовать в качестве такого инструмента проверки имитационное моделирование.

Но практически сразу стало очевидно, что объем вычислений, необходимых для моделирования СМО определенных конфигураций (например, с конечным, но очень большим количеством входящих заявок) займет довольно продолжительное время. Вдобавок для обеспечения адекватности результатов моделирования необходимо многократно повторять такие сеансы моделирования, чтобы накопить объем данных, требуемый для статистической обработки. В связи с этим было принято решение производить необходимые вычисления распределенно. В качестве ресурсов предполагалось использовать свободное время компьютерных классов университета, офисных компьютеров, а также Volunteer Computing студентов университета.

Система получила название ODIS – Object-oriented Distributed Imitation System [1].

Далее стало понятно, что система не будет загружена задачами основного проекта постоянно, а однажды развернутую сеть распределенных вычислений можно было бы использовать как для других проектов АСФ КемГУ, так и предоставлять для работы сторонним пользователям. Поэтому было принято решение реализовать подсистему распределенных вычислений как независимую инфраструктуру, предоставляющую возможности расширения своего функционала для решения любых требовательных к вычислительным ресурсам задач, которые отвечают определенным требованиям разделяемости и описания. Таким образом, работы по проекту ODIS становятся частным случаем использования системы распределенных вычислений, получившей название «ODIS Drops».

Общая архитектура системы и использование ее в рамках проекта ODIS представлена на рис. 1.

На рисунке показано распределение компонентов системы по трем узлам: Client – компьютер пользователя системы, составляющего задания и получающего конечные результаты их выполнения; Remote Calculator – компьютер, ресурсы которого используются для осуществления расчетов; Server – компьютер, посредством которого осуществляется координация остальных узлов.

[1] Работа выполнена в рамках аналитической ведомственной целевой программы «Развитие научного потенциала высшей школы (2009-2010 годы)», проект № 4761

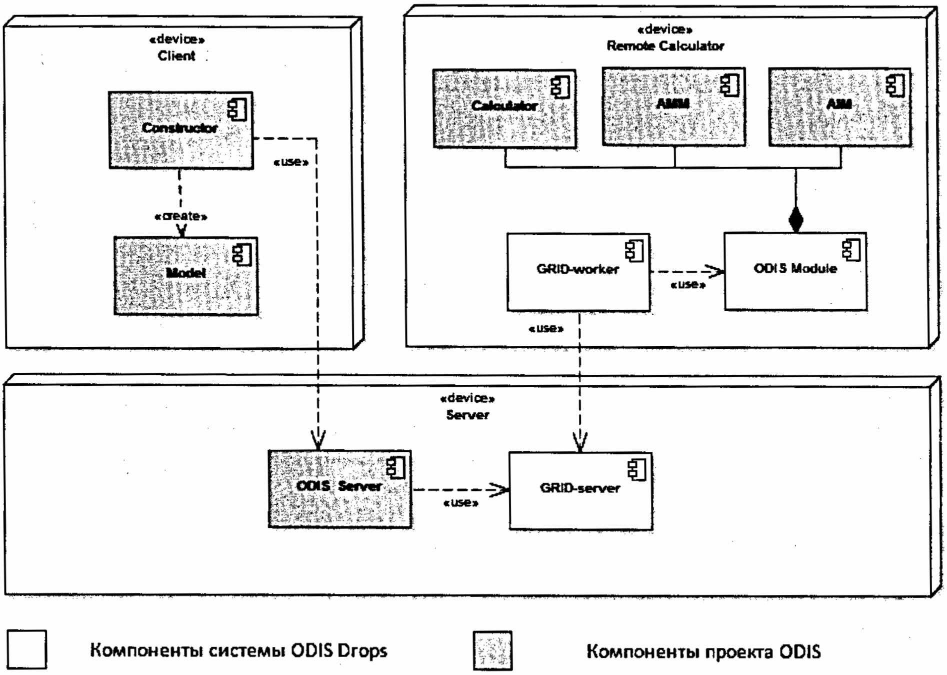


Рис. 1: Компоненты системы

Ключевыми компонентами системы являются GRID-server и GRID-worker. Компонент GRID-worker устанавливается на компьютеры, используемые в качестве ресурса, и отвечает непосредственно за сам процесс расчетов, получение новых заданий и отправку результатов на сервер. Компонент GRID-server устанавливается на сервер сети распределенных вычислений и отвечает за координацию работы объектов GRID-worker, получение заданий от пользователей системы, разделение заданий на подзадания для конкретных объектов GRID-worker и итоговую обработку результатов всех подзаданий для получения конечного результата, необходимого пользователю.

Важно отметить, что Grid-server в данном случае является пассивным. Он предоставляет два интерфейса – для клиентов системы и для объектов GRID-worker, при этом сам не производит никаких обращений к другим компонентам системы. Вместе с технологией Microsoft WCF, выбранной для обеспечения связи между компонентами это позволило сделать сервер максимально доступным для большого количества компьютеров, находящихся в разных условиях сетевого доступа к физическому серверу.

Для решения различных типов задач на однократно развернутой сети объектов GRID-worker используется механизм модульного расширения системы [2]. Так вычислительные компоненты проекта ODIS агрегируются в один из модулей системы – ODIS module.

Более подробно структура компонентов GRID-server и GRID-worker представлена на рис. 2 и рис. 3.

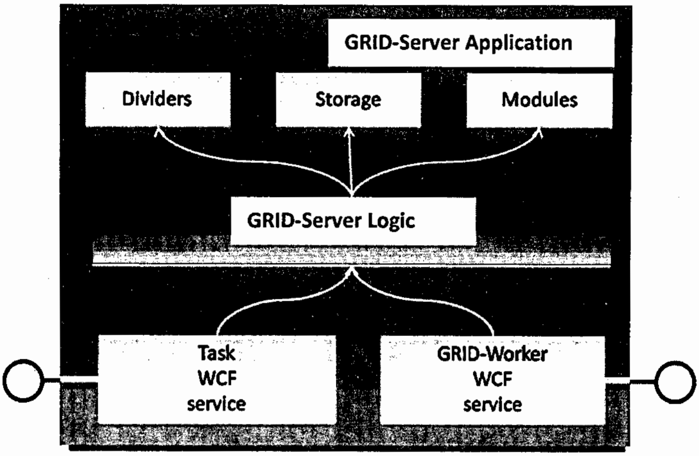


Рис. 2: Структура компонента GRID-Server

В состав компонента GRID-server входят следующие компоненты:

- Task WCF service – Windows Communication Foundation (WCF) web-служба сервера для получения заданий и выдачи результата пользовательским приложениям;
- GRID-worker WCF service – WCF web-служба для подключения объектов GRID-worker;
- GRID-server Logic – «ядро» компонента, в котором описаны классы предметной области системы и сценарии манипуляции ими;
- Dividers – набор алгоритмов разделения заданий на подзадания. Расширяемый компонент: возможно добавление новых алгоритмов;
- Storage – поставщик хранилища для хранения заданий, подзаданий, результатов и учетной информации о существующих GRID-worker'ах. Расширяемый компонент: возможно создание поставщиков хранилища для любых систем хранения информации – баз данных, бинарных или XML-файлов, облачных хранилищ и т.д.;
- Modules – каркас модульного расширения системы и набор самих модулей.

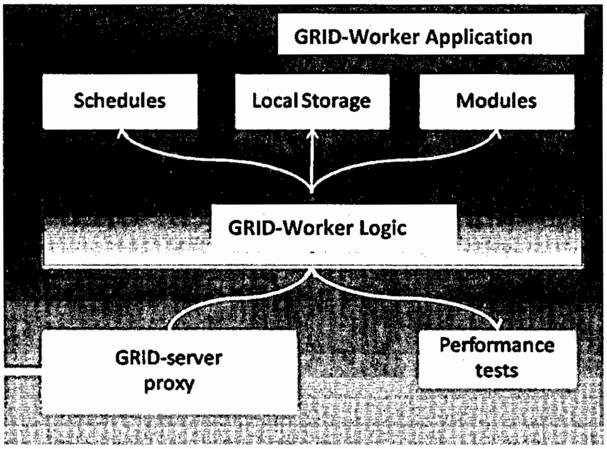


Рис. 3: Структура компонента GRID-worker

В состав компонента GRID-worker входят следующие компоненты:

- GRID-server proxy – компонент-посредник службы объекта GRID-server;
- Performance tests – набор тестов производительности компьютера, на котором установлен GRID-worker. Информация о производительности передается на сервер вместе с другими данными об объекте GRID-worker и используется в алгоритмах разделения заданий для принятия решений по поручению подзаданий данному конкретному объекту GRID-worker;
- GRID-worker Logic – аналог компонента GRID-server Logic, входящего в состав GRID-server, но описывающий предметную область на уровне абстракций объекта GRID-worker;
- Schedules – расписания работы объекта GRID-worker, настроенные пользователем компьютера, на котором он установлен. Расписания указывают, когда и при каких условиях GRID-worker может использовать ресурсы компьютера, например, начинать вычисления только после определенного периода простоя компьютера или получать новые задания только при наличии определенного объема свободного дискового пространства и т.д.;
- Local storage – поставщик локального хранилища данных подзаданий и их результатов. В данном случае – XML или бинарный файл, но компонент также расширяем, и может, например, использовать базу данных клиентского уровня;
- Modules – аналогичен компоненту Modules в составе GRID-server.

Вынесение описания расчетов в модуль дает возможность не только решать все новые задачи на существующей сети объектов GRID-worker, но и использовать для расчетов в распределенной среде ранее созданные инструменты.

Как это происходит. Допустим, существует инструмент, выполняющий некоторые полезные расчеты, вокруг этого инструмента необходимо четко определить и описать в виде классов модуля следующие понятия:

- что является заданием для инструмента, и какие начальные значения оно должно содержать;
- что является промежуточным грид-результатом, который будет получен на каждом объекте GRID-worker при помощи данного инструмента;
- как использовать инструмент для решения заданий, т.е. как с его помощью из содержимого задания получить грид-результат;
- что является итоговым результатом выполнения всего задания, и как рассчитать его, имея массив грид-результатов.

При этом в качестве такого инструмента могут выступать:

- C# классы и алгоритмы – могут быть включены в текст модуля непосредственно;
- .NET сборки – могут быть подключены и использованы в модуле;
- COM-библиотеки и OLE-приложения – могут быть использованы вызовами соответствующих методов;
- наконец, любой интероперабельный инструмент, чем бы он ни был, может быть использован из модуля, если позволяет в каком бы то ни было виде передавать в него задания и получать результаты.

Как было сказано ранее, WCF вместе с концепцией пассивного сервера позволяет привлечь к работе в системе широкий круг компьютеров, имеющих разный доступ к физическому серверу.

WCF позволяет декларативно создавать для одной службы неограниченное количество так называемых «конечных точек», состоящих из трех компонентов: адрес – уникальный адрес службы; привязка – здесь, помимо прочего, также декларативно, производится настройка способа сериализации передаваемых данных и выбор используемого протокола; и контракт – глоссарий сообщений принимаемых службой и структуры передаваемых данных.

Таким образом, имея один грид-сервер с одной действующей веб-службой можно добиться крайне эффективной гибкости подключения в зависимости от ситуации. Например, GRID-worker может быть расположен на том же компьютере, что и GRID-server, на компьютере в локальной сети или компьютере, связанном с сервером через Интернет, при этом он может находиться в корпоративной сети другой организации, быть скрытым за прокси-сервером, сетевым экраном и т.д. При этом в каждом случае возможен выбор оптимального протокола и формата передачи данных.

Использование в рамках WCF формата XML как основного формата передачи данных позволяет также достичь большой гибкости со стороны использования компонента GRID-server пользовательскими приложениями. Так веб-служба Task WCF service принимает задания для системы в виде XML. Таким образом, инструментом пользователя системы для составления заданий и получения результатов может являться что угодно, что позволяет передавать эти задания и получать результаты в XML формате. Так для работы по проекту ODIS к службе подключается ODIS-server, который в свою очередь предоставляет специальный интерфейс для специфических клиентских приложений проекта (Рис. 1).

В дальнейшем планируется создать веб-интерфейс для универсального создания заданий для всех существующих модулей. Если же разработчику модуля требуется предоставить пользователям какой-либо специальный инструмент для работы с системой, он может также создать веб- или клиентское приложение на любой платформе, обращающееся к веб-службе компонента GRID-server.

Таким образом, на данный момент разработана система распределенных вычислений, обладающая следующими возможностями расширения:

- расширяемость с точки зрения задач – объектный каркас модулей системы позволяет расширять систему для решения новых задач, в том числе адаптировать существующие инструменты решения,
- расширяемость с точки зрения алгоритмов разделения заданий – объектный интерфейс позволяет дополнять систему новыми алгоритмами разделения,
- расширяемость с точки зрения клиента – XML-интерфейс заданий позволяет создавать различные инструменты клиента на любых платформах, в том числе интегрировать систему в качестве ресурса в более крупные GRID-системы,
- гибкость и расширяемость с точки зрения ресурсов – WCF позволяет подключаться к GRID-серверу с любого компьютера, имеющего доступ в WWW, а также предоставляет возможность создания дополнительных пользовательских транспортов в рамках WCF.

Публикации по проекту доступны по адресу http://odisdrops.blogspot.com.

Литература

[1] Войтиков К. Ю., Моисеев А .Н., Тумаев П. Н. Компонентная модель распределенной объектно-ориентированной системы имитационного моделирования // Вестник Томского государственного университета. Управление, вычислительная техника и информатика - 2010, № 1, С. 78-83.

[2] Войтиков К. Ю., Тумаев П. Н. Построение архитектуры сервера распределенных вычислений // Научное творчество молодёжи: материалы XIV Всероссийской научно-практической конференции (15-16 апреля 2010 г.). Томск: Изд-во Том. ун-та, 2010. Ч. 1. С. 115-118.

ВЫЧИСЛИТЕЛЬНАЯ ХИМИЯ В ГРИД-СРЕДАХ

В. М. Волохов[1], Д. А. Варламов[1,2], А. В. Волохов[1], А. В. Пивушков[1], Г. А. Покатович[1], Н. Ф. Сурков[1]

[1]Институт проблем химической физики РАН,142432, Черноголовка, Россия
[2]Институт экспериментальной минералогии РАН, 142432, Черноголовка, Россия

The computational chemistry is one of the most interested in grid calculations of branches of a science and in the state-of-the-art is impossible without usage of the super-power parallel and distributed computing resources. In the article on a number of examples existence of different computing classes of tasks is illustrated: both tasks breaking up to a collection of independent jobs, and being the uniform big task. Applicability of grid technologies for solution of the first class of tasks on the distributed polygons is shown. Examples of adapting of standard applied quantum-chemical packages and authoring programs are shown for starts in the various distributed environments. The new computational techniques developed by authors are shortly described: a method of creation of "bunches" of formally independent jobs and a creation method of “virtual containers” (for operation in the heterogeneous computing environments).

The description of IPCP resource center uniting in the structure resource sites of next grid polygons: EGEE-RDIG, SKIF-Polygon and the National Nanotechnology Network is resulted. The resource center also includes client interfaces of different levels to resources of the specified polygons. The constituent of resource center – a grid portal (association of grid and web services) allowing users to get facilitated access to various grid services and resources by means of a web-browser through WWW interface. In the article examples of carrying out distributed calculations in the various environments and on various computing polygons are shown.

Вычислительная и квантовая химия являются одними из наиболее заинтересованных в грид-вычислениях (в том числе на входящих в грид-среды суперкомпьютерах) отраслями науки и неэффективны без использования сверхмощных параллельных и распределенных вычислительных ресурсов для решения задач самых разных классов.

Наиболее востребованы грид-вычисления (и суперкомпьютинг) в следующих областях химии, химической физики и близких к ним науках:

- изучение строения вещества;
- строение молекул и структура твердых тел;
- создание материалов с заранее заданными свойствами;
- кинетика и механизм сложных химических реакций;
- химическая физика процессов горения и взрыва;
- газодинамика экстремальных состояний;
- химическая физика процессов образования и модификации полимеров;
- предсказательное моделирование наноструктур и различные нанотехнологии;
- общие проблемы химической физики - и др.

Для проведения крупномасштабных вычислений в области вычислительной и квантовой химии и сопряженных областей науки требуется проведение высокоинтенсивных параллельных и распределенных расчетов. Например, некоторые задачи оптимизации молекулярных структур требуют выполнения до 10^9 отдельных расчетов. Подобные расчеты требуют вычислительных ресурсов, которые не может предоставить ни один из вычислительных центров.

Крупномасштабные квантово-химические расчеты – одно из направлений работы вычислительного центра Института Проблем Химической Физики в Черноголовке (ИПХФ РАН, http://www.icp.ac.ru) [1]. Институт располагает богатейшей в России библиотекой параллельных квантово-химических и молекулярно-динамических программ (авторских, "open source" и лицензионных). В течение года в институте проводится расчет от 3 до 4 тысяч вычислительных задач высокой сложности с публикацией более чем 400 печатных работ с использованием результатов проведенных расчетов.

Работы с системами распределенных вычислений в ИПХФ РАН были начаты в 2004 году по программам Президиума РАН и Федеральным целевым научно-техническим программам и продолжаются в настоящее время в рамках Программы фундаментальных исследований Президиума РАН № 1 на 2009-2011 годы «Проблемы создания национальной научной распределенной информационно-вычислительной среды на основе развития грид-технологий и современных телекоммуникационных сетей», программы Союзного Государства «СКИФ-ГРИД», программы развития Национальной Нанотехнологической Сети (ГридННС).

Были сформулированы три направления исследований: 1) адаптация прикладного ПО в области вычислительной химии к работе в различных грид-инфраструктурах и обеспечение возможностей запуска задач на распределенных ресурсах; 2) развитие ресурсного грид сайта (для нескольких распределенных сред), выступающего как в роли полигона для проведения вычислительных экспериментов в данной области, так и в роли средства для решения реальных задач; 3) создание новых методик вычислений в распределенных средах, связанных со спецификой используемого ПО.

Наш опыт проведения расчетов в области химии [2-4] позволил разделить большинство квантово-химических задач на два основных вычислительных типа :

1. задачи, распадающиеся на совокупность независимых заданий, число которых зависит от количества параметров задачи или от «сетки» разбиения искомой области данных;
2. задачи, представляющие собой единый вычислительный процесс, как правило, требующий единовременного выделения большого количества ресурсов (количество CPU, оперативная память на ядро, дисковые массивы) – например, моделирование молекулярных структур.

Задачи первого типа наиболее востребованы для работы в грид-средах, поскольку, распределяя независимые задания на множество небольших кластеров (каждый кластер – 10-20 процессоров, задание может исполняться на нем как параллельное), можно добиться высокой эффективности использования вычислительных ресурсов. При этом возможно использование весьма больших вычислительных полигонов (до 10^3–10^4 процессоров) как в локальном варианте (в гетерогенных вычислительных средах типа Condor), так и в условиях распределенных сред (совокупность кластеров – ресурсных сайтов). Хорошим примером является траекторные расчеты химических реакций. Характерной системой для подобных расчетов является реакция $H_2 + O_2$. Расчет представляет собой моделирование с помощью классических траекторий элементарного акта столкновения. Для расчета полного сечения подобной реакции для набора необходимой статистики следует разыграть: по два угла взаимной ориентации для каждой молекулы, начальные колебательные и вращательные квантовые числа, параметр столкновения, относительную энергию столкновения. Как правило, расчет одной траектории занимает от нескольких минут до первых часов. Последовательный перебор указанных параметров приводит к расчетам десятков миллионов траекторий, что ведет к нереальности решения подобных задач на локальных ресурсах и необходимостью перехода к их решению на распределенных полигонах. Аналогичным способом могут решаться многочисленные многопараметрические задачи химии, для которых свойственен относительно независимый перебор многомерных «сеток» входных параметров, что ведет к увеличению до 10^8 – 10^9 числа независимых расчетов.

Задачи второго типа обычно предназначены для решения на суперкомпьютерах, т.к. эффективность их решения непосредственно связана с высокими требованиями как к ресурсам суперкомпьютера (вовлекается значительное число процессоров), так и к ресурсам расчетных узлов (значительные объемы оперативной памяти и дискового пространства), а также эффективностью распараллеливания вычислительного процесса. При этом для характерных задач исследования наноструктур и молекулярных кристаллов возможно использование до нескольких тысяч процессоров с потреблением процессорного времени порядка месяца, т.е. использование кластеров терафлопного уровня. При этом использование грид-сред возможно для запуска подобных задач на суперкомпьютерных ресурсах, входящих в вычислительный полигон. При этом с точки зрения пользователя работа в грид-среде отличается в лучшую сторону от обычного удаленного запуска заданий (через SSH и т.п.) в сторону доступности нескольких ресурсов одновременно и упрощения выбора необходимого (и доступного!) вычислительного ресурса. Также пользователь становится независимым от какого-то конкретного ресурса.

Основой для проведения работ с распределенными средами стал созданный ранее в ИПХФ РАН ресурсный грид центр, включающий сайты распределенных сред gLite, Unicore и Globus. Целью создания ресурсных сайтов стало формирование опытного полигона по проведению вычислительных экспериментов в российском грид сегменте по вычислительной и квантовой химии. Основными задачами в рамках этого направления стали:

- разработка и использование системы запуска исходящих задач (т.е. запускаемых пользователями на удаленных ресурсных узлах) различного типа в распределенных средах;
- обеспечение проведения вычислительных экспериментов и расчета реальных входящих задач на собственном ресурсном узле ИПХФ (выступающем в роли удаленного распределенного ресурса и тестового полигона).

Существующий ресурсный грид центр ИПХФ был достаточно детально описан авторами ранее [1,4, http://cc-ipcp.icp.ac.ru]. Сейчас он объединяет в своем составе полнофункциональные ресурсные сайты следующих распределенных полигонов:

- узел консорциума EGEE-RDIG (Enable GRID for E-sciencE и Russian Data Intensive GRID, http://www.egee-rdig.ru, с мая 2010 года EGI – European Grid Infrastructure) на основе среды gLite (http://glite.web.cern.ch), виртуальная организация (ВО) RGSTEST;
- сайт категории «А» СКИФ-Полигона (http://skif-grid.botik.ru) на базе промежуточного ПО Unicore (http://www.unicore.eu);
- сайт Национальной Нанотехнологической Сети (ГридННС, http://www.ngrid.ru, виртуальная организация NanoChem) – среда Globus Toolkit 4, http://www.globus.org).

Данные ресурсные сайты позволяют решать входящие задачи как с использованием прикладных квантово-химических пакетов, так и общего характера.

В состав ресурсных сайтов входит также комплекс клиентских интерфейсов различных уровней для взаимодействия адаптированного квантово-химического прикладного ПО с грид-средами. Они позволяют запускать исходящие задачи вычислительной химии на распределенных ресурсах, обеспечивая возможность формирования заданий, запуск на удаленных сайтах через брокеры ресурсов, мониторинга прохождения заданий, сбора результатов и статистики.

Работа в рамках ВО RGSTEST обеспечивает доступ к вычислительным мощностям до 500-700 процессоров и дисковым массивам порядка 8-12 терабайт в нескольких географических зонах (Москва, Дубна, Харьков, Черноголовка и др.). Разнородность узлов данной ВО позволяет легко варьировать параметры запускаемых задач, ориентируясь на различные типы ресурсов. Использование подобного полигона обеспечивает проведение достаточно масштабных вычислительных экспериментов как научного, так и прикладного характеров.

Ресурсный сайт для среды Unicore позволяет выполнять входящие задачи сертифицированных пользователей СКИФ-Полигона, производить мониторинг задач и передавать полученные результаты пользователям. Обеспечена возможность мониторинга состояния сайта извне. Клиентский интерфейс обеспечивает запуски исходящих задач через среду Unicore на собственном ресурсном сайте ИПХФ (в роли удаленного ресурса - https://unicorgw.icp.ac.ru:8080) и на доступных (через брокер ресурсов ИПС (https://testbed.botik.ru:9999) ресурсных узлах СКИФ-Полигона – в основном ИПС РАН, СевКавГУ, СКИФ-МГУ и др.

В рамках создаваемой с 2010 года Национальной Нанотехнологической Сети (ГридННС) ресурсному сайту ИПХФ предоставлен доступ к вычислительному полигону с общим числом CPU более 8000 (http://mon.ngrid.ru/stats?page=usage) и большим количеством виртуальных организаций, в том числе поддерживающих квантово-химические расчеты (на базе сайта ИПХФ создана и функционирует ВО nanochem).

Составная часть ресурсного центра – грид-портал, объединение грид и Web сервисов. Создан высокоуровневый интерфейс, позволяющий более эффективно использовать все преимущества грид-расчетов. Эта среда позволяет пользователям получить доступ к грид-ресурсам и сервисам, вызывать и настраивать их с помощью web-браузера. Архитектура грид портала основана на идее, что портальная система является контейнером для пользовательских интерфейсов, обеспечивающих работу с грид-службами. Преимущество данной архитектуры в том, что она достаточно легко позволяет встраивать в портал интерфейсы новых грид-служб и изменять существующие. Портальные сервисы контролируют и визуализируют пользовательский интерфейс. Сформирован WWW портал (http://grid.icp.ac.ru, Grid Enabled Chemical Physics – GECP), включающий WWW интерфейсы к следующим прикладным пакетам:

1. Квантово-химический комплекс GAMESS-US, методы ab initio которого могут использовать параллельные вычисления;
2. Вычисление многопараметрических функций, под которыми следует понимать целый класс задач химической физики, обладающих свойством параллелизма по данным (Data Parallel).

Данные интерфейсы позволяют определять входные параметры и условия (включая загрузку данных, конфигурационных файлов, сертификатов пользователя), формировать сложные первичные файлы запуска, производить (при условии авторизации пользователя) запуск данного ПО в распределенных средах, осуществлять мониторинг выполнения заданий и сбор результатов. Интегрирована также технология работы через web-интерфейс с «пучками» независимых заданий на «нарезаемых» областях данных. Заметим, что основная часть программного кода web-интерфейсов не связана напрямую с выбранной распределенной средой, поэтому они могут быть подключены к нескольким вариантам таковых сред. Данные интерфейсы значительно снижают трудоемкость работы пользователя в части формирования задач и работы с первичными данными и значительно облегчают работу с пакетами в распределенных средах.

Нами проводилась экспериментальная проверка и апробация возможности использования грид-ресурсов для расчетов с использованием стандартных прикладных пакетов программ (в том числе параллельных версий), применяемых в вычислительной химии, а также различных авторских программ, разработанных в ИПХФ и НЦЧ РАН. Для адаптации в распределенных средах ранее [2] был выбран ряд прикладных пакетов (GAMESS-US, VASP, Gaussian-98,-03, Dalton-2, CPMD, NAMD, авторские программы по решению многопараметрических задач из области квантовой химии и молекулярной динамики). Для них был проведен детальный анализ модульной структуры квантово-химического кода и изучены особенности работы различных реализаций однопроцессорных и параллельных версий, определены стратегии реализации выбранных типов квантово-химических вычислений применительно к распределенным средам.

Для выбранных прикладных пакетов были созданы и протестированы на реальных задачах низкоуровневые интерфейсы для запуска их в распределенных средах (в основном – для среды gLite, в меньшей степени для сред Unicore и Globus). Данные интерфейсы включают набор скриптов по формированию исходящих заданий, запуску через брокер ресурсов на удаленных узлах, мониторингу выполнения задач, возвращению полученных результатов с удаленных ресурсов и «сборку» окончательных результатов на интерфейсе пользователя. Реализованы интерфейсы для однопроцессорных и параллельных (SMP, сокетные, MPI-1,2) вариантов указанного ПО. На ресурсном грид-узле ИПХФ, использованном в качестве удаленного распределенного ресурса, проведены запуски указанного прикладного ПО через инфраструктуры ВО RGSTEST (EGEE-RDIG), СКИФ-Полигона, ГриННС. Запуски адаптированного ПО проводились в разных режимах и конфигурациях (с разным количеством процессоров и использованием разных вариантов параллельных расчетов).

Для решения части задач «первого» вычислительного типа (например, широкого класса многопараметрических задач вычислительной химии) с использованием грид-технологий был создан метод запуска “пучков” независимых заданий для использования всех доступных ресурсов распределенной среды. При этом полная задача разбивается на огромное количество независимых подзадач (каждая определяется группой значений совокупности параметров). Для решения многопараметрических задач квантовой химии были разработаны методы формирования «пучков» независимых заданий с варьирующими параметрами – до 10^4, в перспективе до 10^7 «атомарных» заданий на задачу.

Следует отметить, что большинство прикладных пакетов вычислительной химии отличаются сложностью конфигураций и повышенными требованиями к среде выполнения, особенно при параллельных расчетах. Обычно эта проблема решается созданием виртуальных организаций, т.е. объединением через распределенные среды во многом однотипных (по установленному ПО и настройкам) вычислительных ресурсов. Для них прикладные пакеты (вместе со средствами конфигурирования и настройки) распространяются из единого репозитория (например, для ПО ЦЕРНа – Atlas, CMC, Alice). В большинстве же случаев неподготовленный ресурсный сайт не имеет нужного заранее установленного прикладного ПО или не сконфигурирован должным образом, поэтому запуск непредустановленных сложных прикладных пакетов для таких ресурсов обычно оканчивается неудачей. В общем случае необходима трудоемкая ручная или полуавтоматическая перенастройка ресурсных узлов распределенных сред, включающая установку собственно пакетов, конфигурирование центрального узла и расчетных узлов (настройка переменных окружения, общих NFS ресурсов, PBS очередей), установка дополнительных системных библиотек и исполняемых файлов (включая параллельные среды типа Mpich-2). При условии этого возможны запуски пакетов на распределенных узлах.

Для частичного решения данной проблемы авторами был разработан метод создания виртуальных перемещаемых программных «контейнеров», включающих собственно ПО, набор необходимых системных файлов, скрипты по развертыванию и настройке среды исполнения, файлы данных и конфигурационные файлы. Он доставляется на ресурсный узел грид-среды стандартными средствами распределенного middleware. Далее происходит развертывание пакета, настройка среды исполнения, запуск задания и отправка результатов на пользовательский интерфейс, затем «очистка» среды исполнения. Так могут быть решены многие проблемы запуска прикладного ПО в распределенных средах. Данный метод грид-вычислений описан в другой статье авторов в этом сборнике (Волохов и др., «Технология запуска ...»).

В итоге наши работы позволили создать в рамках грид-технологий вычислительную среду для проведения крупномасштабных расчетов в области вычислительной химии:

1. создан комплекс адаптированных к различным грид-средам различных вычислительных полигонов прикладных программных пакетов вычислительной химии с интерфейсами различного уровня (от низкоуровневых интерфейсов до Web-портала),
2. разработаны новые методы вычислений (методы формирования «пучков» независимых заданий, метод «виртуальных контейнеров» и т.д.) в распределенных и параллельных средах;
3. создан ресурсный центр (включающий ресурсные сайты полигонов EGEE-RDIG, СКИФ-Полигона, ГридННС) для проведения вычислительных экспериментов в области химии, объединяющий как ресурсы для решения входящих заданий в средах gLite, Unicore, Globus, так и пользовательские интерфейсы распределенных сред для решения исходящих задач.

Литература

[1] Варламов Д.А., Волохов В.М., Пивушков А.В., Сурков Н.Ф., Покатович Г.А. Распределенные и параллельные вычисления в области химии на ресурсном узле ГРИД ИПХФ РАН // “Distributed Computing and Grid-Technologies in Science and Education: Extended Proceedings of the 3rd Intern.Conf.” (Dubna, 2008). Dubna: JINR, 2008. С.127-130.

[2] Волохов В.М., Варламов Д.А., Пивушков А.В., Сурков Н.Ф., Покатович Г.А. ГРИД и вычислительная химия//Вычислительные методы и программирование. М.: МГУ, 2009. Т. 10. № 2. С. 78-88.

[3] Волохов В.М., Варламов Д.А., Пивушков А.В. Крупномасштабные задачи химии на параллельных и распределенных вычислительных полигонах: современное состояние и перспективы// Научный сервис в сети Интернет: решение больших задач, Всероссийская научная конференция, (Новороссийск, 22-27 сентября 2008). М.: Изд-во МГУ. 468 с. С. 210-212.

[4] Волохов В.М., Варламов Д.А., Пивушков А.В., Покатович Г.А., Сурков Н.Ф. Технологии ГРИД в вычислительной химии// Вычислительные методы и программирование, М.: МГУ, 2010. Т. 11. № 1. С. 42-49.

ТЕХНОЛОГИЯ ЗАПУСКА ПАРАЛЛЕЛЬНЫХ ЗАДАЧ В РАЗЛИЧНЫХ РАСПРЕДЕЛЕННЫХ СРЕДАХ

В. М. Волохов[1], Д. А. Варламов[1,2], Н. Ф. Сурков[1], А. В. Пивушков[1], А. В. Волохов[1]

[1]Институт проблем химической физики РАН,142432, Черноголовка, Россия
[2]Институт экспериментальной минералогии РАН, 142432, Черноголовка, Россия

The article is devoted the description of technology of dynamic creation of the virtual environment of execution of parallel tasks on any grid resources of various polygons. The created technology allows to start complex configured applied packages (including tasks of computing chemistry) in the conditions of unprepared in advance parallel environments. Application of the given technology of "virtualization" of applications allows to expand essentially a range accessible grid resources for performance on them of complex configured applied packages without their preinstallation.

The described technology includes the method (developed by authors) of creation moved «virtual containers» which contain: "personal" copies of necessary system files and libraries (including parallel Mpich-2 environment), scripts for "customization" of the operating system of the work nodes, necessary file "trees", applied package itself, data files etc. After dynamic creation of "container" on the client interface it as usual grid job is supplied by resources of the distributed environment to a remote resource site, developed there, customizes accessible work nodes and the site environment «under itself» and is started as the usual parallel application under MPI-2. Upon successful termination of application operation there is "cleaning" of the environment of execution and return of results on user interface. As an example quantum-chemical package GAMESS-US for which "virtual containers" for gLite, Unicore and Globus Toolkit environments are created efficient is used.

На основе опыта работы в различных распределенных средах [1] авторами был сделан вывод, что значительными препятствиями на пути применения грид технологий в вычислительной химии (а в целом – для запуска в грид средах любых сложно сконфигурированных прикладных пакетов) являются следующие проблемы:

- гетерогенность распределенных вычислительных узлов (на уровне архитектур процессоров, операционных систем, сетевых настроек, параллельных сред и т.п.);
- необходимость создания вычислительной среды для многих ресурсоемких параллельных приложений, состоящей из конфигурационных настроек, дополнительных служб, специфичных параллельных сред, хранилищ данных и прочих компонентов;
- невозможность (или избыточная трудоемкость) перенастройки работающих вычислительных ресурсов (особенно класса "production farms") для целей распределенных вычислений или под нужды конкретных прикладных пакетов.

Одним из способов решения большинства этих проблем может стать применение технологий виртуализации, включающих: (а) создание распределенных ресурсов и сервисов на базе виртуальных машин, (б) формирование виртуальных «контейнеров-приложений» как единых распределенных задач; (в) создание полнофункциональных виртуальных машин, выступающих в роли исходящих/входящих распределенных заданий.

Рассмотрим один из разработанных авторами методов виртуализации вычислительного объекта (параллельного приложения), перемещаемого и выполняемого в грид среде. Метод основан на реализации виртуальной динамически формируемой параллельной MPI среды в форме «виртуального контейнера». Он включает адаптацию программы (прикладного пакета) для работы в роли приложения в составе «контейнера», создание образа виртуальной среды исполнения, формирование собственно «контейнера» и процесс выполнения его как исходящего грид задания на неподготовленном удаленном грид узле (т.е без его предустановки). Применение метода показано на примере пакета GAMESS-US, одного из наиболее востребованных пользователями квантово-химических пакетов.

Для работы многих приложений на уровне локального кластера требуется создание целой системы из приложений, служб, сетей, хранилищ данных и прочих компонентов вычислительной инфраструктуры, которые часто плохо совместимы с режимами работы ресурсного узла в целом. Для пакетов прикладного ПО и сервисов нужны комплексные среды с необходимым набором приложений и политиками безопасности. Ключевыми требованиями являются скорость и простота предоставления таких сред, их тщательная изоляция друг от друга, квотирование вычислительных ресурсов для каждой среды, независимость от базовых настроек узла. Часто это необходимо делать без прерывания работы узлов и остановки вычислительной среды, особенно для «production farms», не допускающих долгих остановок и реконфигурирования системы.

Другой проблемой решения задач (особенно параллельных) в условиях распределенных вычислений является необходимость виртуализации программных сред для исходящих задач. Например, для проведения параллельных вычислений требуется наличие установленной на ресурсных узлах системы параллельного программирования (MPI, OpenMP и др.) или предустановленных специфичных математических библиотек. Используемые пакеты прикладных программ вычислительной химии (GAMESS, Gaussian, NAMD, VASP и др.), как и большинство инженерных пакетов (ANSYS, Abacus, FlowVision и т.п.), отличаются сложностью конфигураций и повышенными требованиями к среде выполнения, особенно для параллельных расчетов. Они требуют обязательной настройки большого количества переменных окружения операционной системы до запуска параллельного приложения на каждом из расчетных узлов. Такая настройка обычно осуществляется в два этапа:

1. при ручной/полуавтоматической установке ПО системным администратором на каждом узле ресурсного сайта на уровне операционной системы (например, при формировании сайтов виртуальной организации);
2. при настройке соответствующих скриптов запуска задания для каждого пользователя, согласно требованиям приложения и системы параллельного программирования.

При этом даже традиционный подход со статическим линкованием библиотек (не говоря уже о динамическом) к исполняемому модулю часто не способен создать полностью работоспособное параллельное задание на произвольном ресурсе среды грид из-за отсутствия на ресурсе необходимых системных файлов.

Для решения данной проблемы авторами был проведен анализ процедуры исполнения типичного параллельного задания на ресурсном узле грид (для сред gLite и Unicore), позволивший определить требования к создаваемому виртуальному образу среды исполнения, а также принципиальную возможность формирования динамической среды исполнения для тестируемых ресурсов. Также был сделан анализ систем параллельного программирования для выбора оптимального виртуального образа среды исполнения параллельного приложения на грид ресурсах. В качестве базового пакета для разработки виртуального образа среды исполнения параллельного приложения была выбрана среда Mpich-2. Детальнее этот анализ и последующая работа с библиотеками MPI-2 описаны в [2, 3]. Отметим, что на данном этапе работ были введены некоторые ограничения для используемых программных сред:

- использованы аппаратные архитектуры x86 и em64t;
- на расчетных узлах грид ресурсов предполагается использование операционной системы Linux (клоны на базе RedHat – собственно RedHat, ScientificLinux, Fedora и т.п.), что связано с особенностями размещения системного ПО, хотя принципиальных ограничений на использование других ветвей Linux дистрибутивов нет;
- по стандарту настройки ресурсных узлов ГРИД для коммуникации между расчетными узлами используется интерфейс TCP/IP и беспарольный доступ по ssh (включая копирование файлов), возможна поддержка NFS ресурсов;
- некоторые версии прикладных пакетов с целью повышения производительности вычислений имеют привязку к сетевым продуктам конкретных производителей и используют по-

ставляемые этими производителями низкоуровневые драйверы. Нами пока такие версии, несмотря на их высокую эффективность, использоваться не будут.

После анализа процедуры выполнения грид задания в разных распределенных средах и выбора среды параллельных вычислений была разработана технология создания динамически формируемых образов исполняемых сред, или виртуальных «контейнеров». Был сформирован перемещаемый программный пакет MPI-2. Полученный пакет далее использовался в качестве базового прототипа для разработки виртуального образа среды исполнения конкретных параллельных приложений.

В качестве тестового приложения была использована программа вычисления числа π (‘cpi.c’) из пакета Mpich-2, правильность работы которой легко проверяется в параллельной среде с различным количеством узлов. Исходная тестовая программа была доработана с учетом особенностей запуска прикладных приложений на грид узлах, был получен ее исполняемый модуль и скрипты запуска с использованием библиотек MPI-2. Тестовый модуль и перемещаемый пакет MPI-2 были собраны и упакованы в единый «контейнер», для запуска которого в средах грид (gLite и Unicore) была разработана серия низкоуровневых скриптов пользовательского интерфейса.

Была принята следующая схема запуска: на удаленный ресурсный узел сети грид через брокер ресурсов (или непосредственно – как в Globus) передается главный скрипт и упакованный «контейнер», содержащий исполняемые файлы, необходимые системные библиотеки, файлы конфигурации и данных. Далее главный скрипт выполняет (упрощенно) следующую последовательность шагов: сбор информации о текущем узле грид, распаковка «контейнера» в директории псевдопользователя грид и перемещение файлов в общедоступную область, настройка среды, запуск сервера mpd (с правами mapped-user) на стартовом узле и проведение его тестирования, распределение необходимых файлов по списку свободных узлов, запуск «кольца» серверов mpd, запуск параллельного приложения и его работа как обычного распределенного задания с последующей передачей результатов на брокер ресурсов и пользовательский интерфейс, удаление всех библиотек и временных файлов со всех узлов. Более детально последовательность работы «контейнера» описана здесь [2,3].

Тестовый вариант «контейнера» (на примере нескольких простых параллельных задач) был отлажен на ресурсном грид центре ИПХФ (его сайты использовались в качестве удаленного ресурса) для сред gLite, Unicore, Globus. Дальнейшее тестирование было проведено на ресурсных узлах RDIG в рамках ВО RGSTEST (узлы НИИЯФ МГУ), а также на узлах СКИФ-Полигона и ГридННС, что показало применимость данного метода для большинства ресурсных сайтов данных сетей.

В качестве реального прикладного пакета с повышенными требованиями к конфигурации системы был выбран пакет GAMESS-US (http://www.msg.ameslab.gov/GAMESS) – одна из популярных программ для теоретического исследования свойств химических систем, основное направление – развитие методов расчета сверхбольших молекулярных систем. Основные программные модули GAMESS-US поддерживают параллельный режим вычислений как на многопроцессорных компьютерах, так и на кластерах рабочих станций UNIX. Пакет отличается сложностью установки и конфигурации, а также требует нестандартных настроек параллельной среды вычислений.

В пакете GAMESS-US ранее была реализована модель интерфейса с распределенным размещением данных (DDI – Data Distributed Interface), которая была оптимизирована для многопроцессорных SMP-архитектур общего вида, особенно работающих с памятью в стиле System V. В настоящее время практически все ab initio методы, включенные в пакет GAMESS, могут использовать параллельные вычисления. Интерфейс DDI использует в качестве базовой сокетную TCP/IP модель межпроцессорных коммуникаций. Использование такого метода распараллеливания для работы на локальном кластере достаточно эффективно и довольно просто в конфигурации. Но при работе в грид средах возникает ряд принципиальных проблем: а) необходимо заранее явно указывать используемые расчетные узлы (обычно нереально); б) неправильно оценивается загруженность расчетных узлов (учитывается только первый расчетный узел); в) отсутствует возможность контроля выполнения удаленной задачи средствами распределенного middleware; (г) на ряде кластеров сокетная модель неработоспособна из-за политик безопасности кластера.

Конфигурации же GAMESS-US с использованием библиотеки MPI авторами пакета разработаны только для ряда мейнфреймов (Cray, IBM, SGI). В общем случае конфигурации с MPI не рекомендуются.

Для работы с пакетом GAMESS-US в среде грид на ресурсных узлах ИПХФ РАН первоначально была установлена последняя наиболее широко распространенная версия Mpich-1.2.7 (реализация стандарта MPI-1). Достоинством данной версии является то, что она явно включает интерфейс Globus-2, основанный на Globus Runtime System, что было бы эффективно для запуска Globus заданий. Однако, получить работоспособную конфигурацию GAMESS-US для MPI-1 не удалось по двум причинам:

- запуск исполняемого задания GAMESS-US осуществляется скриптом, активизирующим более 150 переменных окружения. На главном узле среда создается правильно, но механизм передачи переменных окружения на подчиненные узлы в библиотеке Mpich-1 стандартно отсутствует. В пакет Mpich был включен “secure server”, одной из задач которого являлась ликвидация этого недостатка. Но из-за неполной совместимости с клонами RedHat эту функцию безопасного сервера использовать не удается. При запуске задания на локальном узле пакет Mpich-1 использует команды оболочки такие, как ‘.’ , eval, exec, которые не наследуют среду окружения запускающего процесса, что ведет к краху дочерних процессов;
- особенности реализации команды запуска параллельных заданий mpirun пакета Mpich-1. Запуск заданий на главном и подчиненных узлах существенно различаются – строки команды удаленного запуска (rsh или ssh) на подчиненных узлах дополняются служебными переменными. Стандартное расположение строчных аргументов задания GAMESS-US нарушается, что ведет к краху запуска.

После установки библиотек MPI стандарта 2.0 (версия 1.0.3 пакета Mpich2) была проведена модификация конфигурационных скриптов пакета GAMESS-US (compddi, comp, compall, lked), а также программных модулей ddi_init.c и ddi_base.h. Был полностью переписан соответствующий раздел в запускающем скрипте rungms, который сначала запускает кольцо серверов mpd, а затем уже и само задание. Запуск параллельных заданий осуществляется командой mpiexec, которая не имеет указанных выше недостатков команды mpirun (библиотеки MPI-1).

Использование модифицированной авторами библиотеки MPI позволило впервые получить исполняемое задание для работы GAMESS-US в параллельной среде под управлением MPI. Была проведена компиляция модифицированных исходных кодов GAMESS-US с использованием библиотек MPI-2 и получен бинарный пакет. Затем была создана система компоновки необходимых системных файлов, модифицированного GAMESS-US, конфигурационных файлов и скриптов настройки, файлов данных в единый «контейнер», выступающий в роли исходящего задания распределенной среды. Запуск контейнера аналогичен описанному выше для прототипа.

Серия первичных запусков была проведена на ресурсном сайте грид ИПХФ РАН для сред gLite, Unicore, Globus (запуск задач через грид инфраструктуру с использованием данного сайта как удаленного). Дальнейшее тестирование было проведено на ресурсных узлах RDIG в рамках ВО RGSTEST (узлы НИИЯФ МГУ, среда gLite), в настоящее время проводится тестирование созданного «виртуального контейнера» на ресурсных сайтах СКИФ-Полигона и ГридННС. Были проведены успешные запуски пакета GAMESS-US с применением данной технологии (рассчитаны тестовые примеры молекулярных структур из дистрибутива GAMESS, например, серия ab initio расчетов по оптимизации геометрии в 15-атомной системе ($P_3O_9H_3$) на уровне HF/6-31G) [2-4].

Ввиду того, что на ряде кластеров запрещено или ограничено использование скриптовых языков, были проведены работы по переводу всех действий по развертыванию и настройке подобных «контейнеров» в полностью бинарные исполняемые программы, которые действуют схожим образом, но не требуют доступа к shell языкам. Таким образом, впервые разработана технология запуска GAMESS-US в грид среде в виде единого бинарного файла. При этом входящая задача порождает единичный процесс, который распаковывает библиотеки и системные файлы, прикладной пакет, файлы данных, настраивает среду исполнения, запускает параллельные процессы GAMESS-US, собирает полученные результаты, удаляет «мусор» и отправляет выходные данные на пользовательский интерфейс грид среды.

Заключение

Разработан метод создания виртуальных перемещаемых «контейнеров», которые содержат: «персональные» копии системных файлов и библиотек, скрипты по настройке операционной системы, необходимые файловые «деревья», собственно приложение, файлы данных и т.п. Использование таких «контейнеров» позволяет проводить запуски сложных прикладных пакетов без их предустановки на узлы грид сетей.

В качестве примера использован классический квантово-химический пакет GAMESS-US, для которого создан работоспособный «виртуальный контейнер» для сред gLite, Unicore (вариант для Globus Toolkit 4 разрабатывается).

Литература

[1] Волохов В.М., Варламов Д.А., Пивушков А.В., Покатович Г.А., Сурков Н.Ф. Технологии ГРИД в вычислительной химии // Вычислительные методы и программирование, М.: МГУ, 2010, Т. 11. № 1. С. 175-182.

[2] Волохов В.М., Варламов Д.А., Сурков Н.Ф., Пивушков А.В. Виртуальные вычислительные среды: использование на GRID полигонах // Вестник ЮУрГУ, серия «Математическое моделирование и программирование», 2009. № 17 (150). Вып. 3. С. 24-35.

[3] Варламов Д.А., Волохов В.М., Пивушков А.В., Сурков Н.Ф. Виртуализация параллельных приложений квантовой химии для запуска на ресурсных узлах распределенных сред // 3-я Межд. науч. конф. «Суперкомпьютерные системы и их применение» SSA'2010, Минск, май 2010. Минск: ОИПИ НАН Беларуси. Т. 2. С.12-16.

[4] Варламов Д.А., Волохов В.М., Сурков Н.Ф., Пивушков А.В. Виртуализация вычислительной среды в ГРИД // Параллельные вычислительные технологии 2010, ПаВТ-2010, (Уфа, март 2010). Челябинск: изд-во ЮУрГУ. С. 63-70.

ОБ ОДНОМ МЕТОДЕ РАСПОЗНАВАНИЯ ФОРМЫ ОБЪЕКТОВ, ОСНОВАННОМ НА СИГНАТУРНОМ АНАЛИЗЕ

И. М. Гостев[1], Д. Д. Кириллов[2]

[1] Лаборатория информационных технологий,
Объединенный институт ядерных исследований, 141980, Дубна, Россия
[2] Московский государственный институт электроники и математики,
109028, Россия, Москва, Б. Трехсвятительский пер., д. 3
[1]igostev@gmail.com, [2]kirillov@corp.mail.ru

Введение

Процесс распознавания образов это отнесение некоторых данных к определенному классу на основе выделения характерных признаков из общего потока входной информации. Для решения задачи распознавания обычно определяют некоторую меру близости между объектами. Конкретная формулировка здесь сильно зависит от последующих методов и этапов распознавания в соответствии с выбранной методологией.

Под распознаваемыми объектами понимают различные предметы, явления, процессы, сигналы. Каждый объект описывается совокупностью основных характеристик, записываемых как вектор свойств $\vec{x} = \left(x_1, \ldots, x_n\right)$, где i-я координата вектора $\vec{x}$ определяет значения i-го свойства объекта, и характеристикой S, которая указывает на принадлежность объекта к некоторому классу (образу). Набор заранее расклассифицированных объектов, у которых известны характеристики $\vec{x}$ и S, используется для обнаружения закономерных связей между значениями этих характеристик, и называются обучающей выборкой [1]. Те объекты, у которых характеристика S неизвестна, образуют контрольную выборку. Отдельные объекты обучающей и контрольной выборок называются реализациями.

Основная задача распознавания образов – выбор правила (решающей функции) D, в соответствии с которым, по $\vec{x}$, устанавливается его принадлежность к одному из образов. Выбор решающей функции D производится так, чтобы свести расходы распознающего устройства к минимуму.

Примером задачи распознавания образов такого типа может служить задача классификации нефтеносных и водоносных пластов по косвенным геофизическим данным. По этим характеристикам сравнительно легко обнаружить пласты, насыщенные жидкостью. Значительно сложнее определить наполнены они нефтью или водой. Требуется найти правило использования информации, содержащейся в геофизических характеристиках, для отнесения каждого насыщенного жидкостью пласта к одному из двух классов — водоносному или нефтеносному. Для решения этой задачи в обучающую выборку необходимо включить геофизические данные уже вскрытых пластов.

Успех в решении задачи распознавания образов зависит в значительной мере от того, насколько удачно выбраны признаки $\vec{x}$. Поскольку исходный набор характеристик может существенно превышать число информативных параметров, то большое значение уделяется методам отбора параметров используемых в процессе классификации. Используемая в работе идеология распознавания формы графических объектов включает в себя несколько этапов:

Первый этап – предварительная обработка изображения. Он состоит из загрузки изображения в систему. Далее изображение обрабатывается фильтром низких частот для устранения шумов. Завершающим этапом обработки исходного изображения является получение его контуров как показано на Рис. 1. Для чего можно воспользоваться такими известными методами как CANNY, SUSAN или Дельта-сегментация [2]. Каждый из них решает поставленную задачу, но качество получаемых контуров зависит от самого входного изображения и параметров методов. Так же необходимо помнить, что выбор метода определяется типом поставленной задачи, когда один алгоритм дает лучший результат, нежели чем другой.

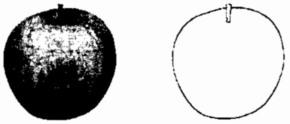


Рис. 1: Исходное изображение (слева). Контуры изображения (справа)

Второй этап включает построение из двумерного контура – одномерной функции. Этот процесс наиболее часто выполняется на основе методов сигнатурного анализа. То есть построение контурной функции основан на круговом движении некоторого луча между центром тяжести фигуры и некоторой характеристикой контура. В её качестве могут выступать функция касательного угла, функция изгиба, длина радиуса дуги и т.п. [3]. Сигнатурная функция фигуры описывает ее форму, а точность определяется как дискретными размерами изображения, так и особенностями конкретного сигнатурного метода.

Несмотря на ряд положительных свойств, открывающих возможность построения идентификационных метрик инвариантных к группам аффинных плоскостных преобразований [4], они имеют недостаток, выраженный в зависимости формы контурной функции от положения центра тяжести объекта. Его смещение из-за шумов и помех, воздействующих на контур, может приводить к нарушению точности при идентификации.

Постановка задачи и решение

Рассмотрим понятие информационной составляющей некоторого вектора свойств $\vec{x}$, приписываемого некоторому объекту [5]. Пусть она определена как множество признаков I, состоящее из подмножеств $I^{(i)} \subset I,\ i=\overline{0,n}$. Индекс i означает уровень подмножества. Так, например, $I^{(0)}$ будет подмножеством первичных признаков, $I^{(1)}$ - вторичных признаков, и т.д. Таким образом, вся информационная часть вектора свойств будет состоять из

$$I=\bigcup_{i=0}^{n} I^{(i)};\quad I^{(i)}\bigcap I^{(j)}=\varnothing;\quad i\neq j.$$

Кроме того, возможна ситуация, когда $I^{(i)} = F(I^{(i-1)}, I^{(i-2)}...)$.

Причем в реальных случаях некоторые подмножества $I^{(i)}$ могут быть пусты или не использоваться, а процесс распознавания может быть построен с использованием, как признаков отдельных уровней, так и их комбинаций.

Введем формальное описание сигнатурной функции следующим образом. Пусть определены элементы множеств $I^{(0)}$ и $I^{(1)}$ для некоторого объекта. Назовем функционалом преобразования сигнатуры контура $r=\nu_3(\nu_2(\nu_1(I^{(0)}, I^{(1)})))$ следующую последовательность функций ν, применяемых к $I^{(0)}$ и $I^{(1)}$ на множестве $G^{(0)}=\left[0,360^{\circ}\right]$:

1. $I^{(0')}=\nu_1(I^{(0)}, I^{(1)})$, где ν_1 - функция преобразования декартовых координат в полярные в виде $(R,\varphi)=\{((x-x_c)^2+(y-y_c)^2)^{1/2}, arctg\frac{x}{y}\}$, то есть получения набора $I^{(0)}$ в полярных координатах - $I^{(0')}=(R_l,\varphi_l)$, $l=\overline{1,k}$.

2. $I^{(0'')} = \nu_2(I^{(0')})$, где ν_2 - функция сортировки множества $I^{(0')}$ по углу φ в порядке его возрастания. Далее предполагается, что если встречаются две и более точек с одинаковым углом, то выбирается точка с максимальным значением вектора R.

3. $r = \nu_3(I^{(0'')})$, где ν_3 - функция интерполяции точек объекта по всей окружности с заданным фиксированным шагом, определенным как $\Delta\varphi = 0.5, 1, 2...$ угловых градуса, что дает 720, 360 и 180 ... точек развертки исследуемого контура, в зависимости от требуемой точности представления объекта.

<u>*Замечание 1.*</u> Очевидно, что преобразования ν_1 и ν_2 будут однозначными только для выпуклых контуров графических объектов. Однако необходимо заметить, что эти преобразования осуществляются относительно центра тяжести объекта. То есть понятие выпуклости здесь определяется через число линий контура объекта, которые будет пересекать луч, идущий от его центра тяжести. Для выпуклого в этом смысле объекта это число всегда должно быть равно 1. В противном случае возникают неоднозначности в выполнении преобразований ν_1 и ν_2, разрешаемые по 3 пункту определения.

Примеры разверток контурных функций приведены на Рис. 2.

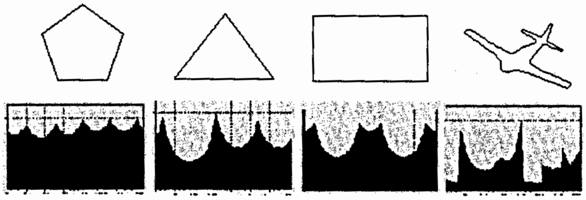


Рис. 2: Примеры фигур и их контурных функций

Несмотря на тот факт, что получаемая сигнатурная функция однозначно описывает контур, тем не менее, существуют формы, при реализации которых сигнатурная функция имеет неоднозначность описания, и, что еще более интересно, когда центр тяжести выходит за пределы фигуры. Примеры таки фигур показаны на Рис. 3.

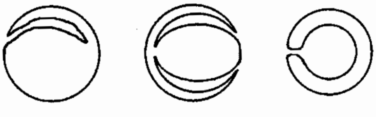


Рис. 3: Примеры невыпуклых фигур

На этом изображении для каждой фигуры в некоторой области одному положению радиус–вектора соответствует несколько значений контурной функции в полярных координатах. Такие фигуры возникают, например, в дефектографии при диагностике деталей на наличие внутренних трещин.

Для изображенных фигур на Рис. 3 применение функционала преобразования сигнатуры контура приводит к потере значительной части информации (внутренней) о форме объекта. Отметим, что эти фигуры практически неразличимы при использовании методов геометрической корреляции для их идентификации.

Для решения задачи об распознавании формы объекта необходимо модифицировать область определения контурных функций r – множество $G^{(0)}$ [5] следующим образом.

Пусть имеется контур некоторой фигуры, изображенный например, на Рис. 4a. Контурная функция от радиус–вектора, проведенного из центра тяжести для него приведена на Рис. 4b.

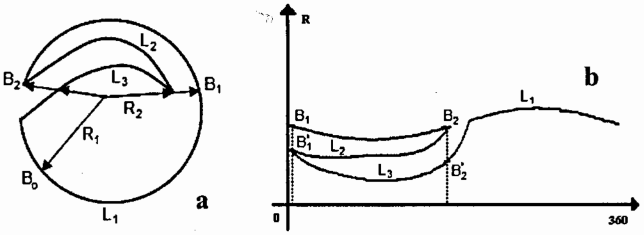


Рис. 4: Контур фигуры и его контурная функция

Двигаясь из точки B_0 вдоль контура по дуге L_1 против часовой стрелки попадаем в точку B_1, в которой появляется второй радиус–вектор, обозначенный как B_1^1. При дальнейшем продвижении радиус–вектор имеет уже три значения, по одному на каждую ветвь контура (рис. 4a и 4b). Однако уже в точке B_2 все ветви сходятся в одну, которая идет до точки B_0. Таким образом, в такой контурной функции существует один интервал, где она задается однозначно, две точки, где она определена дважды и отрезок, где она определена трижды.

Для формализации представления полученной контурной функции введем некоторые новые понятия и переопределим область определения контурной функции r – множество точек $G^{(0)}$ на отрезке $\left[0,360^\circ\right]$ в следующем виде.

<u>*Определение 1*</u>. Назовём *расширенным множеством точек* $G_{Ext} \subset G^{(0)}$ в полярной системе координат и образованное *контурными интервалами* g_i при условиях:

1. $g_i = \left[\tau'_i, \tau''_i\right], \quad \tau'_i \le \tau''_i, \quad \tau'_i, \tau''_i \in \left[0,360^\circ\right]$, так что $G = \bigcup\limits_{i=1}^{N} \{g_i, L_i\}$,

2. $\tau'_1 = \tau'_2 = \ldots = \tau'_N$ и $\tau''_1 = \tau''_2 = \ldots = \tau''_N$,

где $i = \overline{1,N}$, а N - число ветвей контурной функции, в которых существует неоднозначность её определения. Вторая часть условия 1 означает, что интервалы объективно связаны с соответствующими им ветвями. В условии 2 цифры нижних индексов показывают номер ветвей L_i.

Введенное множества G_{Ext} точно отображает существующее положение, однако неоднозначность контурной функции на некоторых интервалах не позволяет применять метрики, основанные на геометрической корреляции, используя которые можно сравнивать форму таких объектов [4]. Для применения этих метрик необходимо модифицировать G_{Ext} следующим образом.

<u>*Определение 2*</u>. Назовём *расширенным множеством точек* $G_{Ext} \subset G^{(0)}$ в полярной системе координат и образованное множеством *контурных интервалов*

$$g_i = \left[\tau'_i, \tau''_i\right], \quad \tau'_i \le \tau''_i, \quad \tau'_i, \tau''_i \in \left[360 \cdot i, 360(i+1)\right], \text{ так что } G = \bigcup_{i=1}^{N} \{g_i, L_i\},\ i = \overline{0,(N-1)}.$$

Теперь неоднозначность в определении контурной функции устранена, но множество G_{Ext} задано на интервале $\left[0,360^\circ \cdot N\right]$, как показано на Рис. 5.

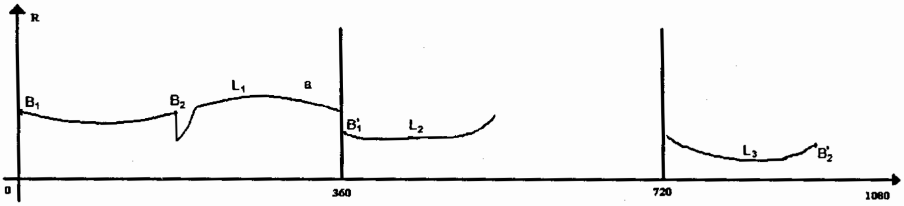


Рис. 5: Примеры расширенного множества точек G_{Ext}

Рассмотрим теперь, как можно модифицировать одну из метрик геометрической корреляции, определённых в [6]. В качестве основы возьмем две ее разновидности называемые метриками *распознавания на основе идентификации по части контура №1* (ИЧК1) и *№2* (ИЧК1).

Определение 3. Пусть эталонная $x(\varphi_i)$ и идентифицируемая – $y(\varphi)$ контурные функции определены и непрерывны на отрезках g_i в полярной системе координат. Запишем $\eta_{ixy}(\varphi,\tau)$ как *частичную функцию разности* значений x и y с учетом номера g_i

$$\eta_{ixy}(\varphi,\tau)=x(\varphi_i)-y(\varphi_i-\tau_i),\quad \varphi_i,\tau_i\in g_i\quad i=\overline{0,(N-1)}. \tag{1}$$

В этой формуле разность значений двух функций вычисляется только на множестве контурных интервалов $g_i\subset G_{Ext}$, независимо от их величины.

Определение 4. Частичную функцию отклонения $\delta_{xy}(\tau)$ *для* x *от* y вычислим в дискретных точках на интервалах $g_i\subset G_{Ext}$ как:

$$\delta_{xy}(\tau)=\frac{1}{N}\sum_{i=0}^{N-1}\left(\frac{1}{m_i}\sum_{\varphi_i\in g_i}\left|\eta_{ixy}(\varphi,\tau)\right|\right),\ \tau_i\in g_i\quad i=\overline{0,(N-1)}, \tag{2}$$

где m_i – число точек интервала g_i.

Определение 5. Частичную функцию среднего отклонения $\sigma_{xy}(\tau)$ для x от y запишем как:

$$\sigma_{xy}(\tau)=\frac{1}{N}\sum_{i=0}^{N-1}\left|\left(\frac{1}{m_i}\sum_{\varphi_i\in g_i}\left|\eta_{ixy}(\varphi,\tau)\right|\right)-\eta_{ixy}(\varphi,\tau)\right|,\quad \tau_i\in g_i\quad i=\overline{0,(N-1)}. \tag{3}$$

Теоретически количество частей $g_i\subset G$ и их размер может быть произвольным. Однако необходимо заметить, что внешние части контура объекта и эталона могут полностью совпадать. В этом случае если исключать из формул (1-3) идентичные фрагменты, то процесс вычисления функций существенно упрощается.

Определение 6. Для метрик типа $\rho_{E1}=\min\limits_{\tau}\delta_{xy}(\tau)$ и $\rho_{E2}=\min\limits_{\tau}\sigma_{xy}(\tau)$ функции *распознавания на основе расширенных контурных функций №1* (РК1) и *№2* (РК2) на базе методов геометрической корреляции запишем как:

$$\lambda_{E1}=\begin{cases}1, & (\rho_{E1}<\varepsilon_{E1})\\ 0, & (\rho_{E1}\ge\varepsilon_{E1})\end{cases}, \tag{4}$$

$$\lambda_{E2}=\begin{cases}1,(\rho_{E2}<\varepsilon_{E2})\\ 0,(\rho_{E2}\ge\varepsilon_{E2})\end{cases}, \tag{5}$$

где, ε_{E1} и ε_{E2} есть классификационные допуска, а $\rho_{E1}(\tau)$ и $\rho_{E2}(\tau)$ вычисляются на всех интервалах g_i, $i=\overline{1,N}$. Равенства $\lambda_{E1}=1$ и $\lambda_{E2}=1$ будет означать успешную идентификацию объекта.

Замечание 2. При необходимости распознавать хиральные объекты, то есть объекты зеркально развернутые на плоскости, необходимо дополнить формулы (4) и (5) частями с зеркальной функцией как это сделано в [6]. А процесс распознавания строить в зависимости от необходимости относить эти объекты в один или разные классы.

Результаты и выводы

При обработке большого потока изображений вопросы быстродействия системы начинают выходить на первое место. Здесь решающую роль начинает играть архитектура построения вычислительного комплекса. Так, например, для обработки видео информации при управлении крылатой ракетой производительность комплекса определяется разрешающей способностью камеры наблюдения. Для небольших разрешений, например 640x380 процесс обработки и распознавания может проходить на одном процессоре [7]. Но большие изображения при организации процесса в реальном времени требуют уже построения мультипроцессорной системы. Теперь скорость обработки определяется уже двумя параметрами. Во-первых, архитектурой программного обеспечения, и во-вторых, архитектурой вычислительной системы.

Методология процесса идентификации опирается на принцип последовательного взвешивания [5], когда объекты последовательно проверяются на принадлежность к эталону различными методами. Причем наиболее трудоемкие методы используются на последних этапах. Обработка графики или видеопотока организована по принципу конвейера [8]. Поскольку все используемые методы требуют только один проход по изображению, то небольшой фрагмент изображения, обработанный на предыдущей операции, поступает на следующую и т д.

Несмотря на кажущуюся трудоемкость идентификации предложенным методом, она обладает двумя несомненными преимуществами. Во-первых, позволяет проводить идентификацию контуров с постоянной скоростью, независимой от сложности объекта. Во-вторых, обеспечивает заданную точность идентификации, которая может регулироваться за счет количества точек множества G_{Ext} и величины классификационного допуска.

Таким образом, возможность организации работы процесса в параллельном режиме, невысокая вычислительная сложность и обеспечение заданной точности идентификации делает предложенный метод актуальным и перспективным в ряде областей, требующих распознавание графических объектов в реальном масштабе времени.

Литература

[1] Ту Дж., Гонсалес Р. Принципы распознавания образов. М. : Мир, 1978. С. 414.

[2] Гостев И. М. Об одном методе получения контуров изображений// Изв. РАН ТиСУ. № 3, 2004.

[3] Loncaric S. A survey of shape analysis techniques //Pat. Rec., 31(8):983–1001, 1998.

[4] Gostev I. M. Recognition of Graphic Patterns: Part 1 // Izv. Ross. Akad. Nauk, Teor. Sist. Upr., N. 1, 2004. [Comp. Syst. Sci. 43 (1), 129 (2004)].

[5] Гостев И.М. О принципах построения эталона в системах распознавания графических образов // Изв. РАН ТиСУ. № 5, 2004. С. 135-142.

[6] Гостев И.М. Методы идентификации графических объектов на основе геометрической корреляции// Физика элементарных частиц и атомного ядра. Т.41. Вып.1. 2010.

[7] Гостев И.М., Яковлева В.С. О построении высокоскоростной системы по обработке изображений и распознаванию образов. //Изв. вузов. Приборостроение. №2, 2005. С.59-62.

[8] Гостев И.М., Подгорбунский А.Г. О построении высокоскоростной системы по обработке изображений и распознаванию образов. Изв. вузов. Приборостроение. №2, 2009.

ОБ ОДНОМ АЛГОРИТМЕ ВЫЧИСЛЕНИЯ ПРОИЗВОДНЫХ ВЫСШИХ ПОРЯДКОВ, ОСНОВАННОМ НА МЕТОДЕ НУМЕРОВА

И. М. Гостев[1], Т. Д. Радченко[2]

[1] Лаборатория информационных технологий,
Объединенный институт ядерных исследований, 141980, Дубна, Россия
[2] Московский государственный институт электроники и математики,
109028, Россия, Москва, Б. Трехсвятительский пер., д. 3

Введение

Несмотря на успехи в области прикладной математики и вычислительной техники, создание систем распознавания графических образов до сих пор остается сложной теоретической и технической задачей. Существует множество приемов решения этой задачи, одним из которых является идентификация незамкнутых кривых [1]. Эту методологию удобно использовать, когда необходимо идентифицировать, например, потоки графических данных в виде треков элементарных частиц, снимаемых с ускорителя в реальном масштабе времени, или определить объект по контуру, заданному фрагментами некоторых кривых, а также для решения множества других прикладных задач вычислительной математики.

Под идентификацией некоторой плоской незамкнутой кривой будем понимать процесс сравнения двух групп признаков, выделенных из функций $f(t)$ и $h(t)$ на основе разработанной метрики [2]. Сначала установим систему параметров, по которой метрика должна вычисляться.

Определим набор необходимых информативных признаков для процесса идентификации, то есть таких свойств образа, по которым его можно выделить из окружающей группы объектов. Здесь необходимо остановиться на таком вопросе, а что же представляют собой такие информативные признаки для плоской незамкнутой кривой? Для их определения будем отталкиваться от контрольных точек, инвариантных к сдвигу, масштабированию, повороту и зеркальному отображению, составляющих основу математического описания кривой.

Рассмотрим случай идентификации функций заданных таблично. Возьмем отрезок $[a,b]$, на котором определены значения функций $f(t)$ с постоянным шагом *h*. В качестве информативного признака выберем «особые точки», в данном случае это нули производных. Для построения математического описания множества этих точек на отрезке использована методология $k-jet$ [1].

Напомним, что $k-jet$ от *k*-раз непрерывно дифференцируемой на $[a,b]$ функции $f(x)$ представляет собой ряд Тейлора, в котором проведена замена переменной с $(x-x_0)$ на $z\in[a,b]$

$$\left(J_{x_0}^k f\right)(z)=f(x_0)+f'(x_0)z+\ldots+\frac{f^{(k)}(x_0)}{k!}z^k .$$

Введем следующие понятия.

Определение 1. Пусть $f(t)$ достаточно гладкая функция на отрезке $[a,b]$. Назовем $k-jet$ нулями $j-$того порядка $(1\le j\le k)$ функции f точки $t_1^{(j)},t_2^{(j)},\ldots,t_{n_j}^{(j)}$, в которых ее $j-$тая производная обращается в нуль: $f^{(j)}(t_r^{(j)})=0,\ r=1,\ldots,n_j$.

Определение 2. Пусть $f(t)$ достаточно гладкая функция на отрезке $[a,b]$. Множество точек $G_f=\{a=t_0<t_1<\ldots<t_n=b\}$, в которых какая-либо из производных обращается в ноль: $f'(t)=0$,

$f''(t)=0,\ldots, f^{(k)}(t)=0$, будем называть множеством нулей $k-jet$ функции $f(t)$.

Например, для функции четвертого порядка $y(x)=5x^4-100x^2+2x-1$ на отрезке $[-5,5]$ это множество будет состоять из пяти точек (трех экстремальных и двух точек перегиба), как показано на Рис.1.

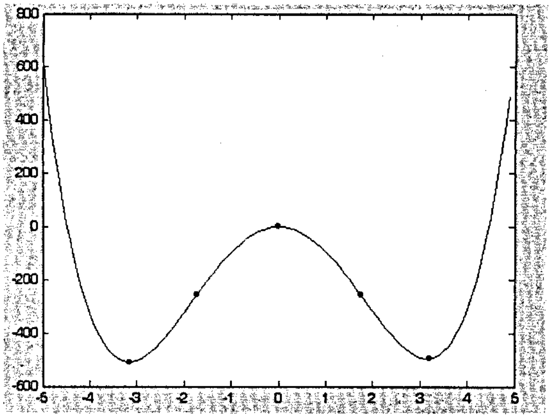


Рис.1: Нули *k-jet* функции *y(x)*

Определение 3. Если у k – раз непрерывно дифференцируемых на отрезке $[a,b]$ вещественных функций f и h совпадают все $k-jet$ нули всех порядков до k – того порядка включительно и, кроме того, значения функций f и h совпадают во всех $k-jet$ нулях, тогда мы будем говорить, что функции f и h являются слабо $k-jet$ идентичными на отрезке $[a,b]$.

Теорема. Пусть функция *f(t)* *k*-раз непрерывно дифференцируема на $[a,b]$, G - множество нулей $k-jet$ $f(t)$. Тогда множество $\{\langle t_i, f(t_i)\rangle, t_i \in G\}$ представляет собой все информативные признаки, необходимые и достаточные для определения класса слабой $k-jet$ идентичности кривой.

Постановка задачи и метод решения

Основная цель настоящей работы заключается в построении множества $k-jet$ нулей для любой таблично заданной функции с высокой точностью, а также в эффективной реализации соответствующей автоматической процедуры. Для того, чтобы получить высокую точность вычислений, были исследованы различные методы численного дифференцирования. Для повышения эффективности автоматизированных вычислений использовались распределенные вычисления.

Построение множества нулей $k-jet$ для плоской незамкнутой кривой требует вычисления с высокой точностью производных первого и второго порядков на всем интервале, на котором рассматривается функция. Метод компактных аппроксимаций (метод Нумерова) позволяет сделать это наиболее оптимально [3]. Здесь имеется ввиду достижение высокого порядка аппроксимации производных на трехточечном (компактном) шаблоне.

Пусть на отрезке $[a,b]$ длиной L определена периодическая достаточно гладкая функция $u(x)$ такая, что для любого x выполняется $u(x+L)=u(x)$. Выберем следующую сетку $\omega=\{x_j=jh, j=0,1,\ldots,N\}$, где N – число интервалов сетки и $h=L/N$ – постоянный шаг сетки. В каждой точке сетки значение функции $u(x)$ определено и равно $u(x_i)=u_i$ где *i=0,1,...,n*. Найдем связь между значениями в трех соседних узлах сетки функции $u(x)$ и аппроксимацией ее производных высшего порядка. Обозначим: $u''(x_j)=f_j$, тогда формулами компактного численного дифференцирования или формулами Нумерова для приближенного вычисления производных второго порядка называется следующее соотношение:

$$\frac{1}{12}f_{j-1}+\frac{5}{6}f_j+\frac{1}{12}f_{j+1}=\frac{u_{j-1}-2u_j+u_{j+1}}{h^2},\ j=1,...,N-1 \qquad (1)$$

Правая часть представляет собой формулу для приближенного вычисления второй производной. Левая часть получена разложением правой части в ряд Тейлора.

Аналогичная формула имеется и для первой производной, для чего введем обозначение: $u'(x_j)=g_j$. Тогда формулами компактного численного дифференцирования или формулами Нумерова для приближенного вычисления производных первого порядка называется следующее соотношение:

$$\frac{1}{6}g_{j-1}+\frac{2}{3}g_j+\frac{1}{6}g_{j+1}=\frac{u_{j+1}-u_{j-1}}{2h},j=1,...,N-1\,. \qquad (2)$$

Для вычисления производных по формулам Нумерова необходимо решать систему линейных алгебраических уравнений относительно искомых производных в каждой точке сетки. При этом необходимо добавить еще 2 уравнения в качестве краевых условий: $f_0=f_1$ и $f_{N-1}=f_N$.

Следующая формула представляет собой разложенную в ряд правую часть формулы Нумерова:

$$f_j=u''(x_j)+\frac{u^{(4)}(x_j)}{12}h^2+O(h^4)$$

Преимущественной особенностью этого метода является малое значение остаточного члена. Метод Нумерова обеспечивает более высокий порядок точности $O(h^4)$ и $O(h^3)$ для производной второго и первого порядка соответственно, по сравнению с обычными сеточными методами дифференцирования – $O(h^3)$ и $O(h^2)$.

При решении подобной системы на интервале с большим количеством точек, с необходимостью вычисления производных высоких порядков, обычными последовательными алгоритмами время вычислений существенно возрастает. При этом процесс распознавания объектов в реальном времени становится проводить невозможно. Для сокращения времени необходимо использовать высокопроизводительные мультипроцессорные вычислительные системы. Это означает, что необходимо разрабатывать новые эффективные параллельные алгоритмы для работы в распределенной среде.

В настоящее время решение систем линейных уравнений возможно при применении метода параллельной прогонки [4]. Реализация этого метода была встроена в автоматическую процедуру вычисления множества нулей $k-jet$.

Рассмотрим алгоритм решения полученной алгебраической задачи с использованием многопроцессорной вычислительной системы с p процессорами [5]. Введем равномерное линейное разбиение множества номеров узлов сетки $\Omega = \{0,1,...,\mathrm{N}\}$ на связные подмножества $\Omega_m=\{i_1^{(m)},...,i_2^{(m)}\}$, $m=\overline{0,(p-1)}$ соответствующие разбиению вектора неизвестных по процессорам.

В результате такого разбиения процессор с номером m будет обрабатывать $i_2^{(m)}-i_1^{(m)}+1$ точек (Рис. 2).

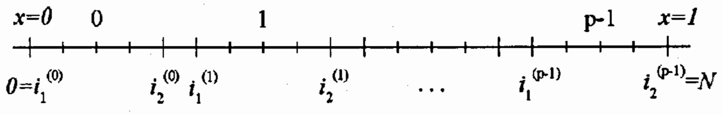


Рис. 2: Разбиение расчетной области по процессорам

Результат, получаемый на каждом процессоре, представляет собой некоторую линейную комбинацию из значений искомой функции на границе множества и во внутренних узлах, что дает часть общего решения. Общий вектор-решение получается на управляющем процессоре.

Для реализации алгоритма была использована технология MPI, где основным способом взаимодействия параллельных процессов является передача сообщений друг другу. Интерфейс MPI поддерживает создание параллельных программ в SIMD-модели, в которой для всех параллельных процессов используется один и тот же код.

В качестве иллюстрации была выбрана гладкая непрерывно-дифференцируемая функция $f(x) = 2 - 1.7\sin x$, которая носит имя улитки Паскаля. Рассчитаем по формулам (1) и (2) шесть ее производных и представим их графики в полярных координатах (Рис. 3).

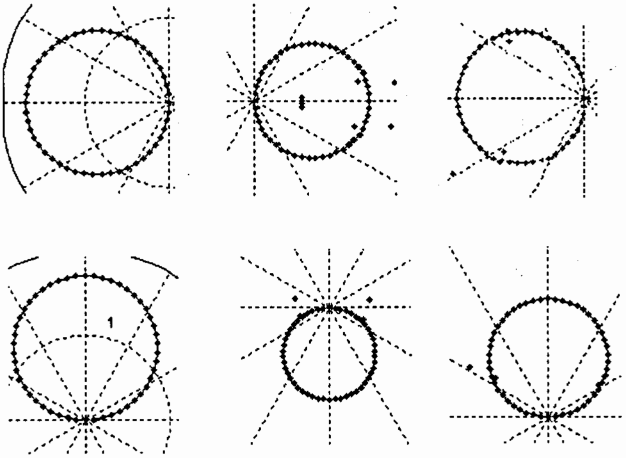


Рис. 3: Первые шесть производных для улитки Паскаля

Выводы

На основании анализа полученных результатов можно сделать следующие выводы об особенностях метода компактных аппроксимаций:

Во-первых, метод компактных аппроксимаций (метод Нумерова) обладает более высокой точностью вычисления производных высших порядков по сравнению с методами численного дифференцирования на основе интерполяционных полиномов. Как следствие, этот метод позволяет достаточно точно определить положение нулей $k - jet$ и их количество.

Во-вторых, при использования этого метода значения производных определенного порядка вычисляются не в окрестности точки, а сразу на всем интервале.

В-третьих, в зависимости от точности вычисления производных задается величина исследуемого интервала и шаг сетки.

В-четвертых, метод компактных аппроксимаций обладает невысокой вычислительной сложностью, в чем превосходит другие методы численного дифференцирования. Его реализация на ЭВМ довольно проста. Такой метод может быть использован в режиме реального времени, то есть прямо в ходе проведения некоторого эксперимента.

В-пятых, метод компактных аппроксимаций легко распараллеливается и реализуется в мультипроцессорной среде.

В-шестых, к недостаткам метода компактных аппроксимаций необходимо отнести возрастание погрешности величин производных на концах исследуемого интервала. Краевой эффект проявляется тем сильнее, чем выше порядок производной. Чтобы уменьшить и/или избежать неточностей на границах, необходимо либо увеличить интервал, на котором рассматривается функция, либо уменьшить шаг сетки и увеличить число точек исследуемой функции.

Таким образом, применение методов параллельной обработки при вычислении производных методом компактных аппроксимаций позволяет существенно сократить время вычислений при их общей невысокой стоимости.

Литература

[1] Гостев И.М., Севастьянов Л.А. Об идентификации гладких пространственных кривых. // Сообщение Объединенного института ядерных исследований. Р11-2007-102. Дубна, 2007.

[2] Gostev I.M., Sevastianov L.A. About the Identification of Flat Unclosed Curves. Physics of Particles and Nuclei Letters 2008 – Vol.5 N. 3 (145). P. 502-507.

[3] Lele K. Compact finite difference schemes with spectral-like resolution Journal of computation physics. V.103. P. 16-42. 1992.

[4] Stefan Bondeli. Divide and conquer: a parallel algorithm for the solution of a tridiagonal linear system of equations. Parallel Computing. 17(1991). P. 419-434.

[5] Иордан В.И., Родионов К.Ю., Соловьев А.А. MPI-вычисления сеточных решений уравнения теплопроводности для исследования режимов горения в нелинейной пористой среде СВ-синтеза // Материалы XV Всероссийской научно-методической конференции "Телематика'2008".

РЕГИОНАЛЬНАЯ СЕТЬ ДЛЯ НАУКИ И ОБРАЗОВАНИЯ ChANT КАК ИНФРАСТРУКТУРА ДЛЯ ГРИД-ПРИЛОЖЕНИЙ

М. В. Григорьева[2], С. А. Крашаков[1,2], А. Ю. Меньшутин[1,2],
С. К. Шикота[2], Л. Н. Щур[1,2]

[1] Институт теоретической физики им. Л.Д. Ландау РАН
Россия, 142432, Московская область, г. Черноголовка, пл. ак. Семенова, 1а
тел./факс (+7 495) 702-93-1, office@itp.ac.ru
[2] Научный центр РАН в Черноголовке
Россия, 142432, Московская область, г. Черноголовка, Институтский просп., 8
тел./факс (+7 495) 993-58-17, sveta@chg.ru

Основной задачей компьютерной сети Научного центра РАН в Черноголовке ChANT является информационно-технологическое обеспечение фундаментальных научных исследований. На сегодняшний день основное внимание уделяется решению следующих задач:

- Доступ к электронным изданиям;
- Удаленный доступ к научным приборам;
- Совместная работа в удаленном режиме;
- Проведение конференций и семинаров;
- Разработка и развитие информационных систем.

1. История

Опорная сеть Научного центра РАН в Черноголовке создана в 1992 году с названием Chg-FREEnet. С 1 мая 2007 года сеть стала полностью автономной и получила название ChANT (Chernogolovka Academic Network).

Проект Chg-FREEnet начат осенью 1992 г. сотрудниками ИТФ им. Л.Д. Ландау РАН [1]. Существенную роль в развитии сети сыграла поддержка проекта Российским фондом фундаментальных исследований (проект 93-07-22858), а также Министерством науки РФ в рамках межведомственной программы "Создание национальной сети компьютерных телекоммуникаций для науки и высшей школы" в 1995-1997 годах. В 2003-2004 гг. проведена реконструкция центра управления сетью по программе РАН "Информатика". В 2005-2006 годах в рамках программы РАН "Телекоммуникации и информационные системы" были проведены работы по строительству сети ChANT.

Разработка проекта сети ChANT началась в 2004 году. Основная цель проекта - создание инфраструктуры для внедрения новых информационных технологий в научный процесс. Проект состоял из трех основных шагов.

Первый, это реконструкция оптоволоконных линий опорной сети НЦЧ РАН для возможности увеличения пропускной способности до 10 гигабит в секунду.

Второй шаг, это организация оптоволоконного канала емкостью 155 Мбит/с с резервированием волокон для возможности установки собственного канального оборудования между НЦЧ РАН и Москвой и увеличения пропускной способности канала.

Третий шаг состоял в создании новой организационно-юридической схемы работы сети. НЦЧ РАН в конце 2006 года получил статус LIR. Технически это означает самостоятельную маршрутизацию потоков информации сети ChANT с другими сетями, составляющими глобальную сеть Интернет. В первой половине 2007 года были смонтированы три основных канала сети ChANT. Первое, это соединение с научными сетями России (RBnet, RUNNet, EmNet, RSSI, Radio-MSU, FREEnet, MSUnet, JINR, KIAE), Европейского союза (GEANT, CERN) и США в рамках проекта NAP (Network Access Point). Второе, это пиринговое соединение с примерно двумя сотнями сетей общего пользования в России (в рамках проекта MSK-IX). Третье, это соединение с сетями общего пользования.

2. Пользователи

Сеть ChANT объединяет научные организации, входящие в структуру НЦЧ РАН. Пользователи имеют подключение по волоконно-оптическим линиям связи на скорости 1 Гбит/с и 10 Гбит/с. Опорная сеть построена с использованием топологии кольца (Рис. 1). В 2009 году начата работа по внедрению на опорной сети технологии передачи данных на скорости 10 Гбит/с. Для этих целей установлено 3 коммутатора в ключевых опорных узлах: НЦЧ, Инновационный центр РАН, ИТФ им. Л.Д. Ландау РАН.

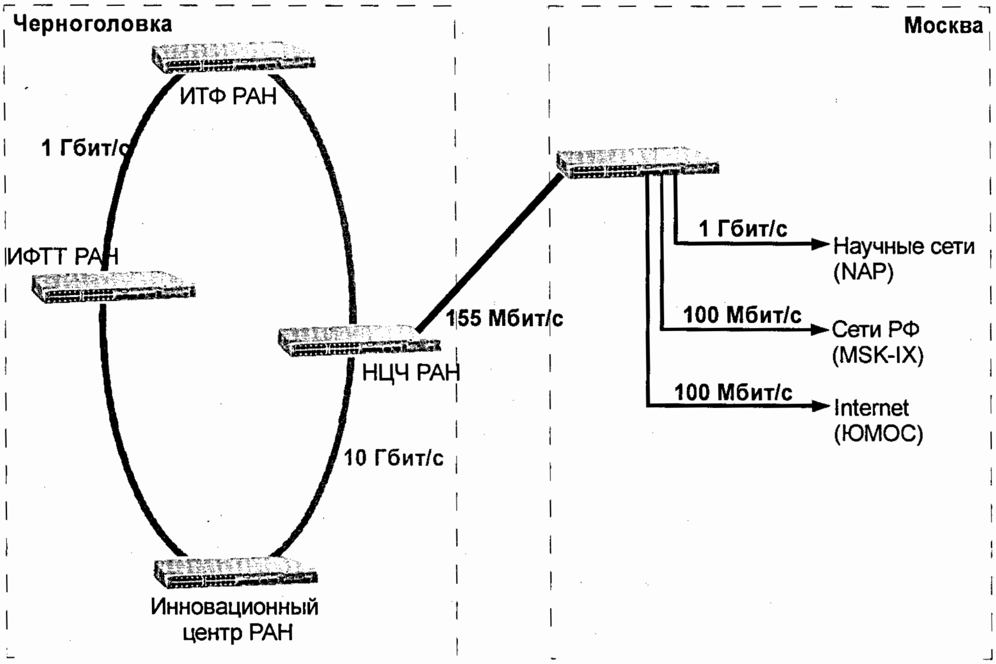


Рис. 1: Схема опорного кольца и внешних подключений

3. Внешняя коннективность

В настоящее время внешнюю коннективность сети НЦЧ РАН обеспечивает оптоволоконный канал связи Черноголовка – Москва (М-9). Специально для этих целей в 2006 году был построен узел связи ОАО «Ростелеком», расположенный в помещении Центра управления сетью ChANT. В настоящее время емкость канала составляет 155 Мбит/сек. Услугу по аренде канала связи предоставляет ОАО «Ростелеком», имеется также резервный канал (back-up) небольшой емкости, арендуемый у организации, имеющей альтернативный канал связи.

Сеть ChANT имеет следующие подключения на узле связи в г. Москва (М-9):

1. Научные сети (точка обмена трафиком NAP) – 1 Гбит/с.

2. Сети Интернет РФ (точка обмена трафиком MSK-IX) – 100 Мбит/с.
3. Сети Интернет общего доступа – 100 Мбит/с (Рис. 2).

Основной канальный протокол - IPv4. Одновременно с этим используется IPv6, по которому работают основные сетевые ресурсы (NS, WWW, FTP) и ведутся эксперименты в ряде институтов. Сеть ChANT является одним из основных генераторов IPv6 трафика в России. В 2009 году начаты работы по созданию параллельной сети на скорости 10 гигабит в секунду с элементами технологии ГРИД. Это позволит предоставлять сотрудникам учреждений Научного центра научных сервисов современного уровня с повышенной пропускной способностью – Грид-компьютинг, Грид видео-конференции, Грид распределенные лаборатории и т.п.

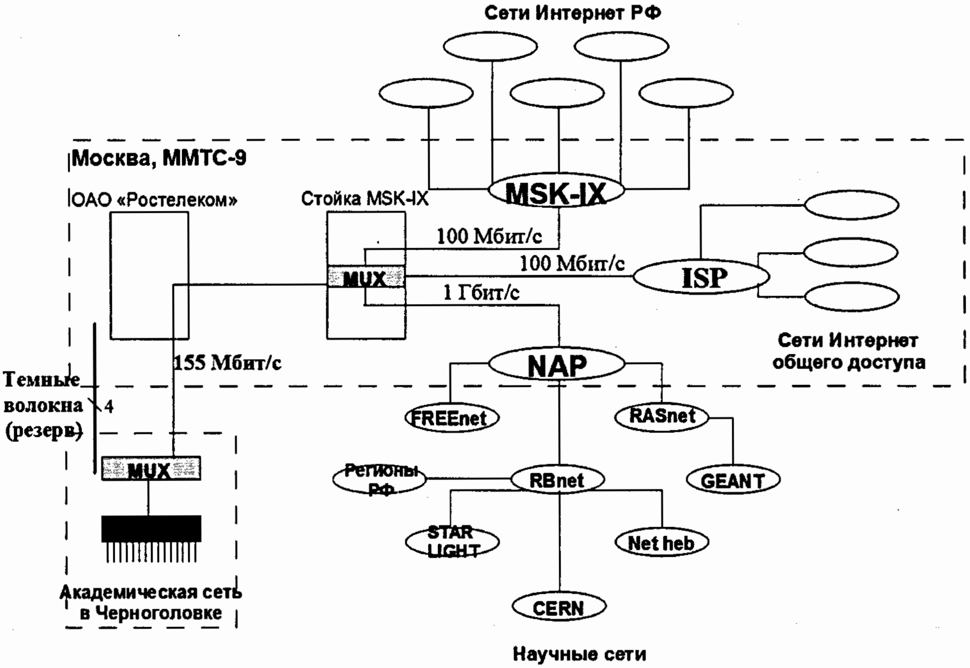


Рис. 2: Организация внешних подключений в сети ChANT

4. Сетевые сервисы

В сети функционируют следующие сервисы.

4.1. Система контроля и учета трафика

В сети ChANT действует автоматизированная система по учету трафика. Система позволяет вести учет всех классов трафика и выдавать отчет за разные промежутки времени по каждому классу трафика, по дням и по IP-адресам.

Использование в системе такого понятия как «класс трафика» позволяет вести учет трафика из различных сетей. Таким образом, имеется возможность анализировать трафик по каждому внешнему подключению (NAP, MSK-IX, UMOS).

Система имеет два интерфейса: администратора и пользователя. Интерфейс администратора совместим с основными операционными системами: Linux, FreeBSD, Windows.

Пользовательский интерфейс реализован в виде «личного кабинета» и позволяет ответственным представителям организаций-участников сети отслеживать объем входящего и исходящего трафика по своим организациям. Для обеспечения безопасного доступа к данным используется шифрование SSL.

4.2. Мониторинг устройств и сервисов сети

Для обеспечения высокого качества предоставляемых услуг, а также для осуществления контроля за использованием ресурсов внедрена система мониторинга Zenoss. Данная система является программным продуктом с открытым исходным кодом, написанная на языке Python. Имеются подробные руководства разработчика, что позволяет с легкостью модернизировать функционал системы. Основной протокол мониторинга, который используется в сети НЦЧ РАН - snmp. Кроме того, данная система поддерживает сбор информации с удаленных узлов по протоколам telnet, ssh, mysql и т.д.

На данный момент реализован сбор и хранение следующих метрик – загрузка всех портов сетевых устройств, мониторинг количества ошибок приема-передачи на сетевых устройствах, загрузка cpu и load average всех узлов, включая сетевое оборудование, мониторинг использования серверов Mysql, Squid, Apache, Mail.

Сотрудниками ОПСИ НЦЧ РАН разработан дополнительный модуль для системы Zenoss - модуль ShowGraph для отображения любых графических отчетов системы на главном экране. Выпущено несколько патчей, исправляющих различные недоработки системы.

Для визуализации общей загрузки основных каналов в систему Zenoss интегрирована система построения карт сети NetworkWeatherMap. Данная карта охватывает все основные сетевые устройства, отображает их статус и загрузку каналов между ними.

Система мониторинга сети автоматически генерирует тревожные события и посылает уведомление по почте/смс администраторам, а также подает голосовые сигналы оператору.

5. Информационные ресурсы

В сети Научного центра функционируют уникальные информационные ресурсы.

5.1. Архив ftp.chg.ru

Для установки и обновления программного обеспечения создан распределенный архив свободно-распространяемого программного обеспечения http://ftp.chg.ru [3]. Архив предоставляет пользователям уникальный набор операционных систем и программных средств [2]. В частности, такая операционная система, как Linux Mandrake, может быть получена в РФ только из нашего архива. Заметим, что Linux Mandrake является единственной сертифицированной операционной системой со свободной лицензией, которая разрешена для использования в государственных учреждениях России.

Для доступа к архиву программного обеспечения ftp.chg.ru используются следующие сетевые протоколы: HTTP, FTP, RSYNC. Для анализа данных по доступу к архиву разработана специальная программа, производящая вычисление количества запросов и количества переданной информации с привязкой к виртуальным именам ресурсов. Полученная информация сохраняется в базе данных mysql, и доступна для просмотра через веб-интерфейс по адресу http://archive.chg.ru. Веб-интерфейс также позволяет администратору ввести описание каждого виртуального ресурса, которое отображается при нажатии на ссылку, изображенную знаком «?» в общем дереве статистики. Сайт http://archive.chg.ru/ позволяет получить данные о посещаемости сервера ftp.chg.ru за определенный день, неделю, месяц, или за любой другой промежуток.

В дополнение к основному серверу ftp.chg.ru, расположенному на площадке ЮМОС (г. Москва), имеется сервер ftp.chant.ru в г. Черноголовка. Для повышения надежности функционирования распределенных серверов, реализуются механизмы синхронизации серверов, и балансировки нагрузки, позволяющие избегать перегрузок сервера независимо от протоколов доступа, используемых пользователями.

Среднее число пользователей в каждый момент от 50 до 10 тысяч. Общий объем дискового пространства – 35 Тб, полезный объем – 29 Тб. Архив существует в двух экземплярах, расположенных в разных геометрических точках сети. Средний размер исходящего трафика 80 мегабит в секунду. Он является крупнейшим архивом России и третьим в мире.

Приоритет в организации и развитии сети Научного центра ChANT сделан на внедрении сервисов, способствующих успешному выполнению научной деятельности сотрудниками и организациями НЦЧ РАН.

5.2. Электронный читальный зал

Работает электронный читальный зал на 12 мест в БНЦ РАН с доступом к научным журналам по подписке БЕН РАН, а также к другим электронным журналам. Проводится большая организационная работа по увеличению числа доступных журналов, как зарубежных, так и отечественных. В рамках той же модернизации осуществлен доступ к системе из филиалов Библиотеки НЦЧ РАН, расположенных ИПХФ, ИФТТ, ведутся работы по доступу других институтов НЦЧ РАН.

5.3. Система для распределенной коллективной работы Видео-Грид

Система для распределенной коллективной работы Видео-Грид использует инструментарий AccessGrid, разработанный в Аргонской национальной лаборатории для проведения распределенных видео-конференций и семинаров, а также совместной работы. Она позволяет принимать и передавать аудио и видео информацию, совместно просматривать файлы, презентации, web-документы [6]. Данная система может быть использована на разных масштабах:

- персональный уровень (точка-точка),
- уровень лаборатории,
- уровень учреждения,
- уровень конференции.

К другим преимуществам системы могут быть также отнесены:

- невысокая стоимость, которая складывается из стоимости компьютера, видеокамеры, аудио гарнитуры, а также стоимости проектора и экрана, если планируется использование AG в помещениях, где собирается большая аудитория;
- одновременное участие практически неограниченного числа пользователей (число пользователей ограничивается пропускной способностью каналов связи);
- многообразие применений (наука, образование, медицина, бизнес и т.д.);
- открытый программный код, позволяющий расширять возможности системы;
- параллельная и распределенная обработка потоков данных;
- высокий уровень безопасности и защищенности (используется протокол SSL);
- работает с разными операционными системами: Windows, Linux, MacOS.

5.4. 4-К видео

Другая система, которая может быть использована для совместной работы виртуальных коллективов, реализованная Отделом ПСИ — это мозаичная видеостена, построенная с применением ПО Sage. 20 мониторов с диагональю 27 дюймов объединены в один виртуальный монитор с общим разрешением 46 Мпикс [6]. Каждая пара мониторов обслуживается одним из 10 узлов вычислительного/графического кластера, соединенных по технологии GigabitEthernet. Суммарная производительность кластера на тестах Linpack составляет около 640 Гфлопс, что составляет около 75% от пиковой производительности.

Распределенная архитектура Sage позволяет создавать графические приложения, в которых задача по построению изображения может выполняться на всех узлах кластера [4, 5]. Для этих целей на кластере установлена реализация MPI — OpenMPI и система управления заданиями Torque совместно с планировщиком Maui. Распределенная задача по построению изображений требует общего доступа к исходным данным на всех узлах кластера. Каждый узел кластера, помимо загрузочного диска, содержит два диска объемом 500Гб, которые объединены в виртуальный диск большего размера. В свою очередь все жесткие диски со всех узлов кластера объединены в единый раздел с использованием распределенной сетевой файловой системы GlusterFS. Данный подход позволяет, во-первых, получить сетевое хранилище данных значительного объема, во-вторых, значительно увеличить производительность файловых операций, что особенно важно для такой ресурсоемкой задачи как обработка видео и графики сверхвысокого разрешения.

Система Sage содержит набор готовых приложений для совместной работы: с использованием специального Bridge сервера можно выводить одинаковые изображения на несколько видеостен, расположенных в разных организациях, имеются приложения для захвата изображения рабочего стола (VNC) и вывода данных изображений на сервер. Также имеются модифицированные версии программы mplayer для показа видео и набор средств для вывода графических файлов. В Отделе ПСИ разработано собственное прикладное приложение для вывода на видеостену результатов численного моделирования двумерных структур роста.

5.5. Научные базы данных

В сети НЦЧ РАН работает 8 научных баз данных, в том числе система учета результатов интеллектуальной деятельности Российской академии наук, основные функции которой заключаются в:

- создании программ финансирования и заявок на поддержку проектов,
- экспертизе заявок и запуске проектов,
- контроле и мониторинге поддержанных проектов,
- экспертизе отчетов по проектам,
- учете результатов научно-технической деятельности.

6. Вычислительные мощности

6.1. Кластер Wall

Расположен в Инновационном центре РАН, в одном из ключевых узлов сети ChANT. Кластер используется для параллельной обработки и визуализации изображения сверхвысокого разрешения. Производительность 650 Гфлопс. Кластер обслуживает систему видео 4К (46Мегапикселей).

6.2. Кластер "Парма"

Расположен в ИТФ им. Л.Д. Ландау РАН. Проект начат в конце 2006 года. Кластер был создан для решения задач гидродинамики, для ученых, занимающихся расчетами мембран, моделированием кластеров и методом Монте-Карло в применении к физике твердого тела, глобальными задачами космологической эволюции.

6.3. Кластеры CLI, CLI-X, CLICP

Кластеры расположены в лаборатории газодинамики Института проблем химической физики. Используются для решения задач в области вычислительной физики и молекулярной динамики.

Литература

[1] Крашаков С.А., Щур Л.Н. Основные принципы создания региональной информационно-вычислительной среды, положенные в основу реализованной в

научном центре в Черноголовке современной компьютерной сети для науки и образования //Тезисы докл. Всерос. конф. "Научный сервис в сети ИНТЕРНЕТ", г. Новороссийск, 18-23 сентября 2000. С. 144-145.

[2] Крашаков С.А., Щур Л.Н. FTP.Chg.RU - 10-летняя история крупнейшего в России архива свободно распространяемого программного обеспечения и дальнейшие перспективы //Тр. Всероссийской научной конференции "Научный сервис в сети Интернет ' 04", г. Новороссийск, 20-25 сентября 2004 г. С. 250-254.

[3] ftp.chg.ru

[4] Renambot L., Rao A., Singh R., Jeong B., Krishnaprasad N., Vishwanath V., Chandrasekhar V., Schwarz N., Spale A., Zhang C., Goldman G., Leigh J., Johnson A. SAGE: the Scalable Adaptive Graphics Environment //Proc. of WACE 2004, Sept. 23-24, 2004.

[5] Березин С.Б., Войцеховский Д.В., Жижин М.Н., Мишин Д.Ю., Новиков А.М. Многомасштабная визуализация окружающей среды на видеостенах. Научная визуализация, 2009. Т. 1. №1. С. 100-107, http://sv-journal.com/2009-1/04/index.html

[6] Aldoshin S.M., Krashakov S.A., Menshutin A.Y., Shikota S.K., Shchur V.L., Shchur L.N. Grid-facility for business incubator of Russian academy of science in Chernogolovka // Proc. Int. Conf. "Distributed computing and Grid technologies in science and education" (GRID'2008), JINR, Dubna, Russia, 30 June – 4 Aug., 2008.

АРХИТЕКТУРА ГРИДА ДЛЯ НАЦИОНАЛЬНОЙ НАНОТЕХНОЛОГИЧЕСКОЙ СЕТИ[1]

А. П. Демичев, В. А. Ильин, А. П. Крюков, Л. В. Шамардин

НИИ ядерной физики им. Д.В. Скобельцына
МГУ им. М.В.Ломоносова, 119992, Москва, Россия
kryukov@theory.sinp.msu.ru

Проект грида для Национальной нанотехнологической сети (ННС) [1,2] – российский проект начатый в 2008 году. Эта грид инфраструктура ориентирована на приложения в области нанотехнологии и нанонаук, которые требуют для своего решения параллельных вычислений на суперкомпьютерах средней мощности. Многие университеты и исследовательские центры имеют в своем распоряжении подобные суперкомпьютеры. Поэтому объединение подобных ресурсов в единую грид-инфраструктуру позволит увеличить эффективность использования дорогого оборудования и предоставит пользователям увеличить возможности использования суперкомпьютеры для своих исследований.

В процессе проектирования ГридННС участники проекта постарались решить проблемы, с которыми столкнулись другие гриды, такие как EGI/WLCG[3,4]. Одна из базовых идей – это разделение потока управляющих команд, таких как граф задания, от потока данных в том числе и исполняемых программ. Например, сервис распределения нагрузки, который распределяет задачи между компьютерами, не управляет потоком данных. Все данные непосредственно передаются со специальных серверов хранения на компьютерные ресурсы и обратно. Другим важным отличием от традиционных решений в гриде является использование REST архитектурного подхода для проектирования грид-сервисов. Например, сервис распределения нагрузки спроектрован и реализован как RESTful-грид-сервис[5,6]. Это позволило значительно упростить протоколы обмена между центральными сервисами по сравнению с традиционными решениями, основанными на WSRF стеке. В представленной работе мы приведем более детальное описание архитектуры ГридННС.

Основной задачей, которую решает программный комплекс ГридННС является интеграция различных компьютерных ресурсов, распределенных по территориально разделенным сайтам, в единый пул грид-ресурсов. При этом обеспечивается возможность выполнения вычислительных заданий на удаленных компьютерных ресурсах, выполнение композитных заданий (в том числе, выполнение отдельных задач композитного задания на разных ресурсах), а также выполнение задач, требующих параллельных вычислений на высокопроизводительных ресурсах. Промежуточное программное обеспечение (ППО) ГридННС позволяет распределять вычислительные задания по сайтам и принимать их там, возвращать результаты пользователю, контролировать права пользователей на доступ к тем или иным ресурсам, получать информацию о грид-инфраструктуре, осуществлять мониторинг ресурсов, учет их использования и осуществлять ряд других действий. Общедоступные ресурсы на основе сайта могут включать вычислительные узлы и/или узлы хранения и передачи данных, собственно данные и прикладное программное обеспечение.

В самом общем виде грид-архитектуру ГридННС можно представить как систему, имеющую три базовых слоя (см. Рис. 1):

- слой интерфейсов пользователей (ИП), связанный с доступом к гриду пользователей,

[1] Работа была частично выполнена при финансовой поддержке грантами РФФИ (10-07-00332), контрактами Министерства образования и науки РФ (№ 01.647.11.2004, №. 02.740.11.0388 и НШ № 4142.2010.2).

администраторов и менеджеров;

- слой общих грид-сервисов, отвечающих за работу ГридННС в целом;
- слой сайтов с грид-шлюзами - сервисами, обеспечивающими доступ к локальным ресурсам.

Интерфейсы пользователя предназначены для формирования задания пользователя и запуск его на обработку. После этого пользователь может отключиться от системы. Подключаясь время от времени к ГридННС, пользователь может контролировать ход выполнения задания; по окончанию задания, пользователь может получить результат выполнения задания на свой компьютер.

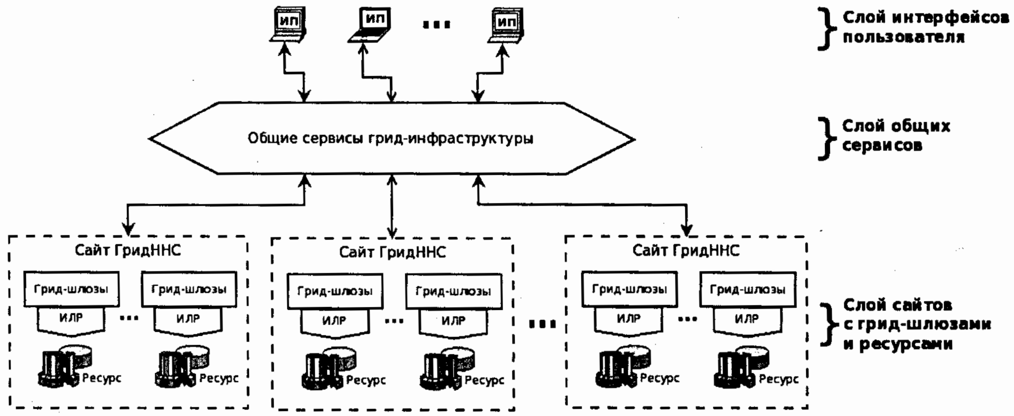

ИП
ИП
ИП
Слой интерфейсов пользователя
Общие сервисы грид-инфраструктуры
Слой общих сервисов
Сайт ГридННС
Грид-шлюзы
ИЛР
Ресурс
Слой сайтов с грид-шлюзами и ресурсами

Рис. 1: Общая архитектура ГридННС

Слой сайтов с грид-шлюзами и ресурсами: грид-шлюзы это сервисы, обеспечивающие, вместе с интерфейсами к локальным ресурсам (ИЛР), доступ к локальным ресурсам; благодаря грид-шлюзам любые локальные ресурсы ГридННС представлены для остальных компонент грид-инфраструктуры в виде грид/веб-сервисов.

Слой общих грид-сервисов обеспечивает работу и управление всем гридом в целом. Группы общих грид-сервисов и грид-шлюзов структурно объединяются по их функциональному назначению в ряд систем ГридННС. Эти системы и сервисы ГридННС перечислены в Таблице 1, а ниже дано краткое описание каждой из систем и некоторых ключевых сервисов.

Веб-интерфейс ГридННС (ВИГ) предназначен для взаимодействия пользователя с грид-средой посредством обычного веб-браузера. ВИГ позволяет пользователю получать информацию о грид-ресурсах, создавать прямо в окне веб-браузера описания заданий с помощью специальных редакторов и запускать их на обработку. После этого пользователь может отключиться от системы. Затем, подключаясь время от времени к ГридННС, пользователь может контролировать ход выполнения задания, а по окончанию задания, пользователь может получить результат выполнения задания на свой компьютер. Таким образом, ВИГ позволяет пользователю работающему за компьютером с любой операционной системой взаимодействовать с ГридННС без установки какого-либо дополнительного программного обеспечения.

Конечно, в некоторых случаях использование ВИГ, как и любого графического интерфейса, может оказаться неудобным: например, если пользователю необходимо осуществлять автоматизированный запуск набора заданий с помощью специально написанного сценария (скрипта). В таких случаях пользователь может использовать интерфейсы командной строки системы управления выполнением заданий (ИКС СУВЗ), а также инструментария для работы с электронными сертификатами (ИКС утилиты proxytool).

Таблица 1. Системы и сервисы ГридННС

Система	**Грид-сервисы**	**Сокращенное название сервиса**
Интерфейсы ГридННС	Сервер веб-интерфейса ГридННС	ВИГ
Система управления выполнением заданий (СУВЗ)	Сервис распределения и контроля заданий «Пилот»	СРКЗ "Пилот"
	Сервис передачи данных	GridFTP
	Вычислительный элемент с интерфейсом к локальному ресурсу ***(грид-шлюз)***	ВЭ, ИЛР
	Сервис надежной передачи файлов ***(грид-шлюз)***	СНПФ
Система мониторинга и учета ресурсов (СМУР)	Сервис сбора данных мониторинга	ССДМ
	Сервис сбора данных учета использования ресурсов	ССДУ
	Сервис регистрации ресурсов и грид служб	СРРГС
Информационная система (ИС)	Центральный информационный сервис	ЦИС
	Локальный информационный сервис ***(грид-шлюз)***	ЛИС
Система безопасности и контроля прав доступа (СБКПД)	Удостоверяющий центр	УЦ
	Сервис управления прокси-сертификатами	СУПС
	Сервис проверки статуса сертификатов	СПСС
	Сервис управления виртуальными организациями	СУВО

Веб-интерфейс ГридННС (ВИГ) предназначен для взаимодействия пользователя с грид-средой посредством обычного веб-браузера. ВИГ позволяет пользователю получать информацию о грид-ресурсах, создавать прямо в окне веб-браузера описания заданий с помощью специальных редакторов и запускать их на обработку. После этого пользователь может отключиться от системы. Затем, подключаясь время от времени к ГридННС, пользователь может контролировать ход выполнения задания, а по окончанию задания, пользователь может получить результат выполнения задания на свой компьютер. Таким образом, ВИГ позволяет пользователю работающему за компьютером с любой операционной системой взаимодействовать с ГридННС без установки какого-либо дополнительного программного обеспечения.

Конечно, в некоторых случаях использование ВИГ, как и любого графического интерфейса, может оказаться неудобным: например, если пользователю необходимо осуществлять автоматизированный запуск набора заданий с помощью специально написанного сценария (скрипта). В таких случаях пользователь может использовать интерфейсы командной строки системы управления выполнением заданий (ИКС СУВЗ), а также инструментария для работы с электронными сертификатами (ИКС утилиты proxytool).

Система управления выполнением заданий (СУВЗ) обеспечивает распределение и контроль за выполнением заданий на грид-ресурсах, которая относится к слою общесистемных служб. Главная задача сервиса «Пилот» - это запуск задач композитного задания в соответствии с требованиями задач и состоянием вычислительных ресурсов, которые публикуется в информационной системе.

Грид-шлюзы. Помимо общесистемных служб важную роль играют грид-шлюзы. Грид-шлюз к вычислительному ресурсу — это совокупность сервисов, таких как сервис запуска заданий, локальный информационный сервис, сервис передачи данных. Функционально к грид-шлюзам относятся еще интерфейсы к локальным ресурсам (ИЛР) - модули, которые обеспечивают трансляцию "языка" грид-среды на "язык", понятный конкретному ресурсу. Для каждого типа ресурсов (точнее - типа менеджера локального ресурса (МЛР)) должен быть свой

ИЛР. В настоящее время ГридННС поддерживает следующие МЛР: PBS/Torque, Cleo и Slurm. Грид-шлюз к ресурсу хранения данных (GridFTP сервис) используется для передачи входных данных задач на рабочие узлы ресурсов, а результатов задач - на серверы GridFTP, откуда они могут быть забраны пользователем на свой компьютер.

Система безопасности и контроля прав доступа (СБКПД) обеспечивает безопасный доступ к ресурсам в незащищенных сетях общего доступа (Интернет) с учетом прав данного пользователя и правил обслуживания пользователей данным ресурсным центром. СБКПД предоставляет такие сервисы, как сервис аутентификации, сервис конфиденциальной передачи информации, сервис делегирования прав и управления виртуальными организациями, включая права отдельных пользователей в грид-среде.

Информационная система (ИС) решает задачу сбора и управления данными о состоянии грида, получая информацию от множества распределенных источников – поставщиков. Она предназначена для постоянного контроля функционирования грид-системы, обеспечения своевременного реагирования на возникающие проблемы и работы сервиса распределения и контроля заданий. В эту систему входит как центральный информационный сервис (ЦИС), агрегирующий информацию, поступающую от всех ресурсов, так и локальные информационные сервисы, являющиеся частью грид-шлюза для информационных потоков от ресурсов в грид-среду.

Система мониторинга и учета ресурсов (СМУР) предназначена для отслеживания текущего состояния ресурсов, заданий и других объектов в системе. Инструментарий СМУР предоставляет как статическую так и динамическую информацию о функционировании ГридННС (примером динамической информации может служить состояние очередей на вычислительном кластере).

Служба регистрации ресурсов и грид-сервисов **(СРРГС)** обеспечивает доступ пользователей и грид-сервисов к реестру всех ресурсных грид-сайтов и сервисов ГридННС. Используется также для управлением ГридННС (например, для отключения менеджерами грида каких-либо сервисов или ресурсов по соображениям безопасности всей грид-инфраструктуры).

С организационной точки зрения основной структурной единицей является *виртуальная организация* (ВО). ВО — это совокупность людей и организаций, находящаяся под единым административным управлением, которые согласованным образом используют грид-сервисы на основе политики ВО. Основная задача ВО — это обеспечение доступа членов ВО к конкретным сервисам и ресурсам на основе договоренности с владельцем сервиса/ресурса. Причем внутри ВО разные пользователи могут получить разные права доступа к ресурсам (например, одна группа пользователей внутри ВО может получить доступ к большему набору ресурсов, чем другая). ВО организуются, как правило, по принципу общности научно-производственных целей и области исследований ее членов. Конечно, может оказаться, что исследователь, желающий работать в среде ГридННС не найдет подходящей ВО, членом которой он может стать. В этом случае ему будет предоставлена возможность вступить в специальную ВО с ограниченными ресурсами. Один пользователь может быть членом нескольких ВО.

Вместо логина/пароля, которые обычно используются при прямом доступе к ресурсам, в грид-среде, в частности, в ГридННС, используется цифровой сертификат - совокупность двух файлов (открытый и закрытый ключи). Этот сертификат выдается пользователю специальной грид-службой - Удостоверяющим центром. С его помощью, а также с помощью сервиса управления виртуальными организациями, пользователь создает так называемый расширенный прокси-сертификат, который делегируется другим грид-сервисам, что позволяет выполнять задания пользователя на ресурсах от его имени в соответствии с его правами.

Еще один шаг, помимо создания прокси-сертификата, который пользователь должен сделать перед отправкой задания в грид-среду, это создать описание задания. В этом описании пользователь указывает исполнимые файлы, которые необходимо выполнить, параметры, с которыми они должны исполняться, входные данные, которые нужны в процессе выполнения задания, и т.д. В частности, пользователь может явно указать ресурсы, на которых должно выполняться задание. Помимо унификации способов доступа к ресурсам при этом появляется еще одна возможность - задание может быть композитным, то есть состоять из набора отдельных связанных между собой задач. Соответствующий грид-сервис (а именно, СРКЗ «Пилот») возьмет на себя запуск и контроль выполнения отдельных задач в соответствии с описанием задания. В описании задач указывается какие файлы и куда должны быть переданы, чтобы обеспечить выполнения задания в целом. После запуска задания и получения от грида его идентификатора пользователь *может* выключить свой компьютер, а затем, подключившись снова, с помощью полученного идентификатора узнать статус запущенного задания и, если оно успешно завершилось, получить результат. Таким образом, режим работы ГридННС - пакетный режим. Интерактивный режим работы не предусмотрен.

Шаги, которые должен выполнить пользователь для использования компьютерных ресурсов в грид-среде являются следующими.

Предварительные шаги:

1. получение цифрового сертификата,
2. регистрация в виртуальной организации.

Шаги, выполняемые при запуске заданий в грид-среду (не все являются обязательными; некоторые выполняются один раз для ряда запусков):

1. создание прокси-сертификата с расширениями, указывающими на права пользователя,
2. создание описания задания,
3. передача данных с локального компьютера пользователя на сервер хранения данных (GridFTP), который указан в задании,
4. запуск задания,
5. отслеживание хода выполнения задания,
6. получение результатов на компьютер пользователя.

При отслеживании хода выполнения задания и задач основной контрольной информацией является их текущее состояние. Основные из них - это running, abort и finish. В случае аварийного завершения одной из задач задания, то сервис «Пилот» продолжит запускать те задачи, которые не зависят от аварийно завершившейся.

Подробную информацию о функционировании ГридННС, а также инструкции для пользователей и администраторов ресурсов ГридННС можно получить на сайте проекта: http://ngrid.ru

В заключение отметим, что ГридННС является полнофункциональным гридом, который включает все необходимые службы и сервисы. При его создании впервые был использован архитектурный стиль REST для разработки грид-сервисов. Это позволило значительно упростить протоколы обмена между сервисами и, тем самым, сделать эти сервисы более удобными для реализации и надежными в эксплуатации. Второй важной особенностью ГридННС — это разделение потока управления и потока данных при обработки заданий. Все это позволило создать гибкую систему, которая легко приспосабливается для подключения суперкомпьютеров, работающих под различными ОС и использующих различные локальные менеджеры ресурсов.

Литература

[1] GridNNN project web site, http://www.ngrid.ru/

[2] Крюков А., Ильин В., Добрецов В., Кореньков В., Рябов Ю. Проектирование и реализация грид-инфраструктуры для национальной нанотехнологической сети. В данном сборнике.

[3] Проект EGI, http://egi.eu/

[4] Worldwide LHC Computing Grid, http://lcg.web.cern.ch/lcg/

[5] Демичев А., Крюков А., Шамардин Л. Принципы построения грид с использованием Restful-веб-сервисов. Программные продукты и системы, г.Тверь, 2009. №4.

[6] Демичев А., Ильин В., Крюков А., Шамардин Л. Реализация программного интерфейса грид-сервиса Pilot на основе архитектурного стиля REST. Вычисленные методы и программирование, 2010. Т. 11. С. 62-65.

ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА ГРИД-ИНФРАСТРУКТУРЫ ДЛЯ НАЦИОНАЛЬНОЙ НАНОТЕХНОЛОГИЧЕСКОЙ СЕТИ[1]

В. Ю. Добрецов[2], В. А. Ильин[1], В. В. Кореньков[3], А. П. Крюков[1], Ю. Ф. Рябов[4]

[1]НИИЯФ МГУ, [2]РНЦ «Курчатовский институт»,[3]ОИЯИ, [4]ПИЯФ РАН

Grid-infrastructure for National nanotechnology network (GridNNN) is a new Russian Grid-project which is developing by SINP MSU, JINR, RRC "Kurchatov Institute" and PNPI in 2008.

The main target of the project is provide the users unified remote access to Russian supercomputer centers to solve wide set of application in nanotechnology and other science and engineering area.

The GridNNN is a full scale Grid infrastructure which base on the original MW and Globus Toolkit (v. 4.2). Most core Grid-services are original development. They base on modern REST architecture style in grid-service programming. Now GridNNS integrated ten resource centers which accumulated more then eight thousand slots (cores) in total.

The current status and the future of GridNNN project will be presented.

Основная цель проекта ГридННС [1] — это создание грид-инфраструктуры для Национальной нанотехнологической сети (ННС)[2]. Данный проект был начат в 2008 году и был рассчитан на три года. Головной исполнитель проекта — Научно-исследовательский институт ядерной физики имени Д.В. Скобельцына Московского государственного университета имени М.В. Ломоносова. В проекте также принимают участие сотрудники ОИЯИ, ПИЯФ РАН, РНЦ «Курчатовский институт». Руководитель работ, д.ф.-м.н. В.А. Ильин.

ГридННС — это комплексное грид-решение, которое включает промежуточное ПО, грид-сервисы и службы, регламенты развертывания и работы и так далее. В частности сюда входят следующие сервисы и службы:

- поддержка пользователей, виртуальных организаций, системных администраторов;
- мониторинг и учет ресурсов, заданий;
- служба регистрации сервисов;
- сервис управления выполнением задач;
- грид-шлюзы доступа к ресурсам;
- пользовательские интерфейсы.

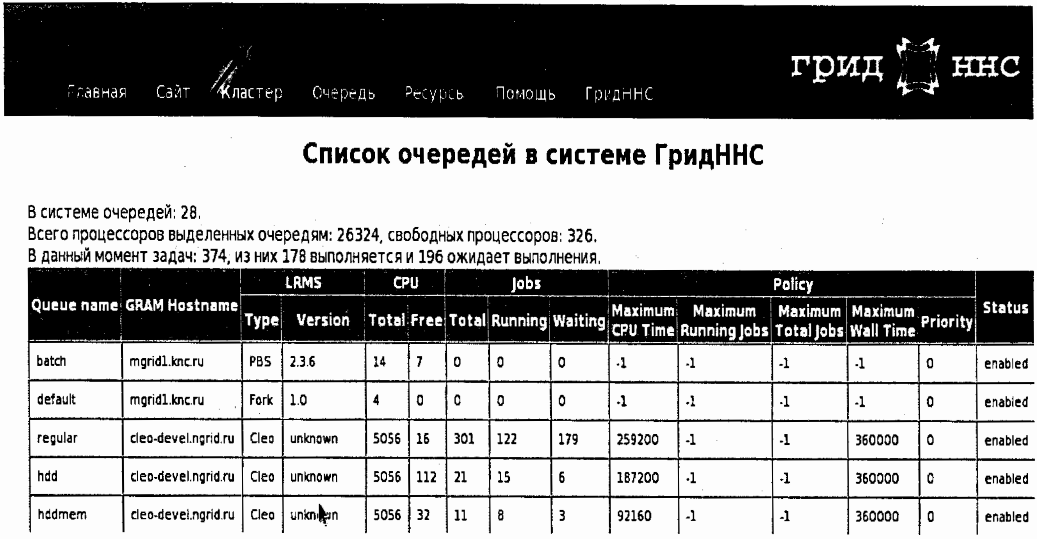
Таким образом ГридННС представляет собой полнофункциональный грид.

С самого начала, одной из целей проекта было объединение суперкомпьютерных ресурсов участников ННС в единую грид систему. Учитывая специфику работы суперкомпьютеров, важным требованием к разрабатываемому промежуточному грид-ПО было минимальное вмешательство в работу суперкомпьютерных центров. В частности, не допускалась установка специального ПО на узлы суперкомпьютерных кластеров. В этом существенное отличие данного проекта от грида WLCG [3].

Большинство сервисов и программ были разработаны участниками проекта. Однако, ряд сервисов используют имеющиеся открытые разработки. Например грид-шлюз основан на Globus Toolkit-4.2 [4], а информационная система построена на WS-MDS [5] на основе Glue

[1] Работа была частично выполнена при финансовой поддержке грантами РФФИ (10-07-00332), контрактами Министерства образования и науки РФ (№ 01.647.11.2004, №. 02.740.11.0388 и НШ № 4142.2010.2).

Scheme-1.3 [6] с некоторыми модификациями. На Рис. 1 приведена одна из страниц Веб-серсиса Центральной информационной системы ГридННС.



Список очередей в системе ГридННС

В системе очередей: 28.
Всего процессоров выделенных очередям: 26324, свободных процессоров: 326.
В данный момент задач: 374, из них 178 выполняется и 196 ожидает выполнения.

Queue name	GRAM Hostname	LRMS		CPU		Jobs			Policy					Status
		Type	Version	Total	Free	Total	Running	Waiting	Maximum CPU Time	Maximum Running Jobs	Maximum Total Jobs	Maximum Wall Time	Priority	
batch	mgrid1.knc.ru	PBS	2.3.6	14	7	0	0	0	-1	-1	-1	-1	0	enabled
default	mgrid1.knc.ru	Fork	1.0	4	0	0	0	0	-1	-1	-1	-1	0	enabled
regular	cleo-devel.ngrid.ru	Cleo	unknown	5056	16	301	122	179	259200	-1	-1	360000	0	enabled
hdd	cleo-devel.ngrid.ru	Cleo	unknown	5056	112	21	15	6	187200	-1	-1	360000	0	enabled
hddmem	cleo-devel.ngrid.ru	Cleo	unknown	5056	32	11	8	3	92160	-1	-1	360000	0	enabled

Рис. 1: Страница центральной информационной системы ГридННС

Основным объектом, с которым работает пользователь в ГридННС является задание [7]. Оно представляет собой набор задач, которые образуют направленный ациклический граф. Такая структура задания позволяет представить достаточно широкий класс композитных заданий, практически покрывающих все потребности пользователей. В качестве языка описания заданий используется JSON [8]. На Рис. 2 приведен пример простого задания ГридННС.

В описании задания и задач, пользователь может указать дополнительные требования, которые необходимо удовлетворить, чтобы выполнение задач прошло успешно. Такая схема позволяет пользователю достаточно гибко указывать не необходимые ему ресурсы.

Управлением выполнения заданиями осуществляет специальный грид-сервис — Pilot [7,9]. Основная его задача принять задание от пользователя, зарегистрировать его и обеспечить запуск задач на ресурсы, которые соответствуют требованиям пользователя. В процессе выполнения задач, сервис Pilot осуществляет контроль за выполнением задач и всего задания в целом. В частности, он обеспечивает правильный порядок запуска задач в соответствии с логикой задания.

Важной особенностью модели управления данными в ГридННС является то, что данные на ресурсы передаются на него непосредственно перед запуском задач (модель P2P). Для этого могут быть использованы любые GridFTP серверы, зарегистрированные в системе. Эта модель существенно отличается от того, что принято в проекте gLite [10].

Каждый суперкомпьютер работает под управлением некоторой системы пакетного запуска задач. Для наиболее популярных систем в GT-4 есть необходимое ПО, обеспечивающее стык между собственно гридом и системой пакетного запуска. К ним в первую очередь относятся PBS/Toque [11,12]. В рамках проекта ГридННС были дополнительно разработаны стыки для Slurm [13] и российской системы Cleo [14]. При необходимости, этот список может быть достаточно легко расширен.

```
{"version": 2,
 "description": "тестовое задание",
 "tasks": [ { "id": "a",
  "description": "задача #1",
  "definition": { "version": 2,
  "executable": "/bin/ls"
            }
        }
      ],
 "requirements": {
  "hostname": ["host.name.ru"],
  "lrms": "lrms_name",
  "queue": "queue_name"
            }
}
```

Рис. 2: Пример описания задания ГридННС

В настоящее время в ГридННС интегрировано в экспериментальном режиме 11 сайтов (Рис. 3).

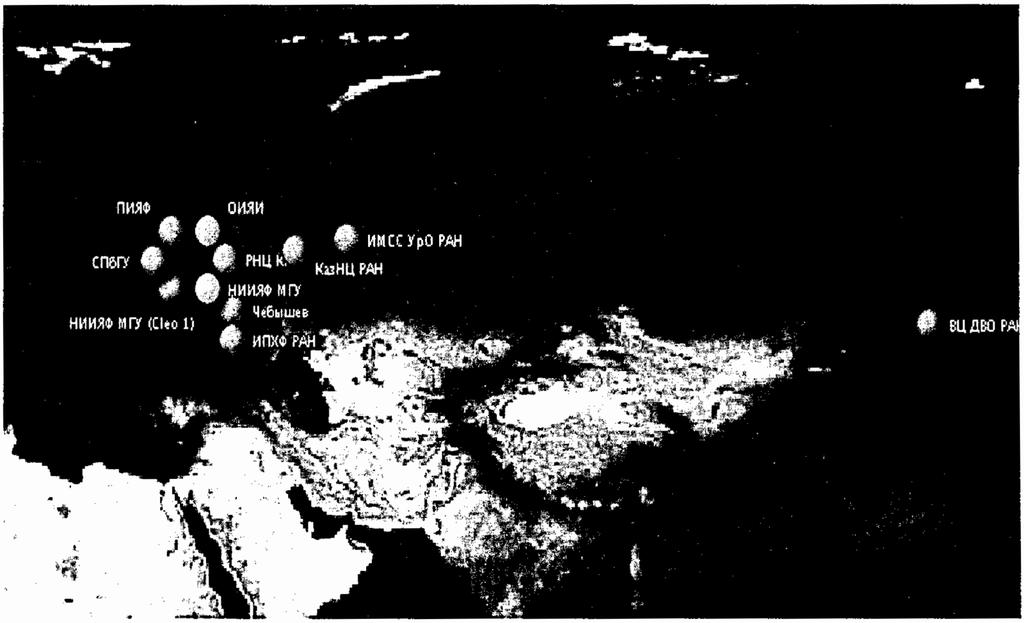
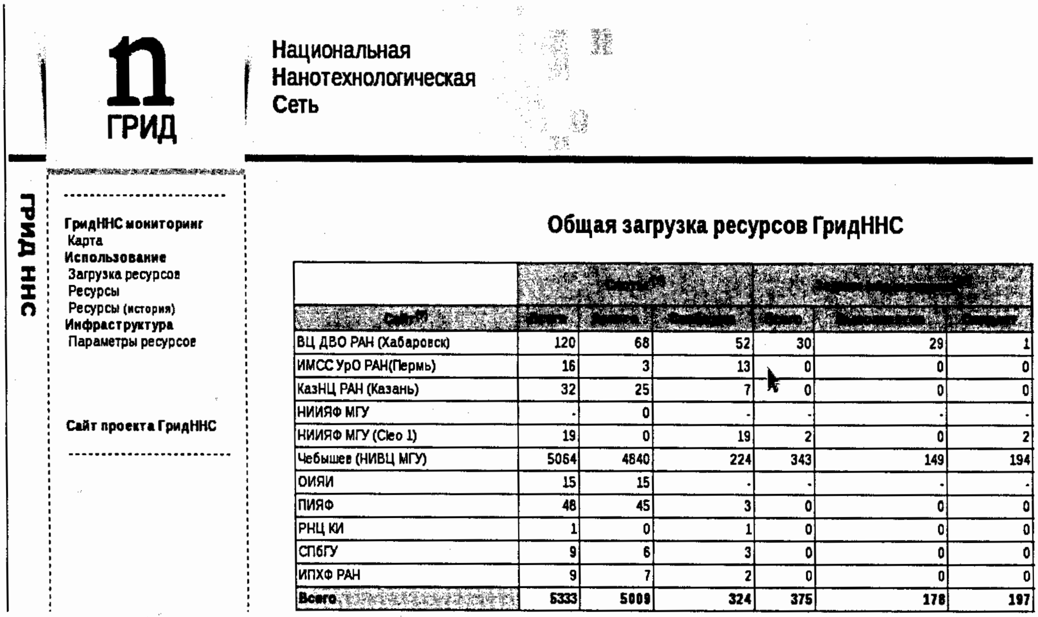


Рис. 3: Географическое распределение сайтов ГридННС

Общее количество ЦПУ составляет более пяти тысяч.

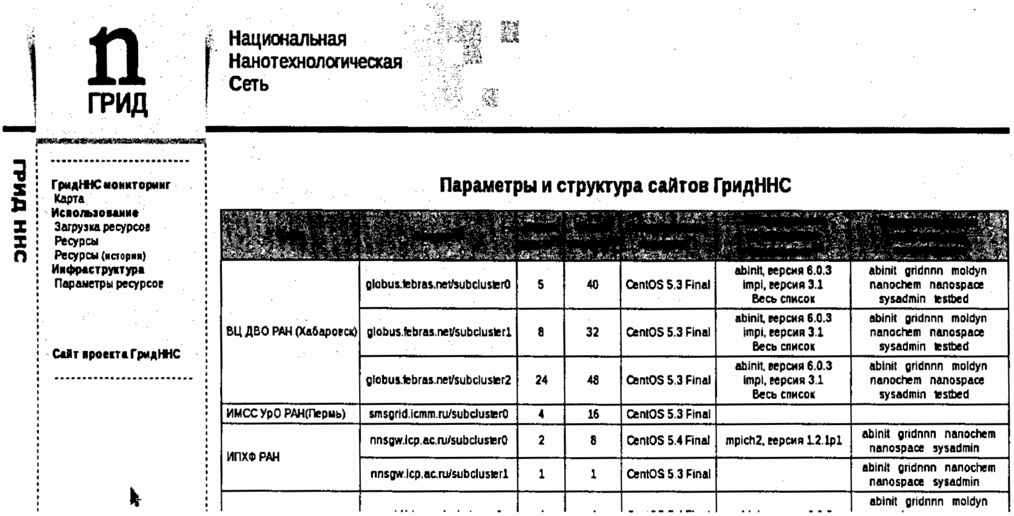


Общая загрузка ресурсов ГридННС

	[illegible]			[illegible]		
[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
ВЦ ДВО РАН (Хабаровск)	120	68	52	30	29	1
ИМСС УрО РАН(Пермь)	16	3	13	0	0	0
КазНЦ РАН (Казань)	32	25	7	0	0	0
НИИЯФ МГУ	-	0	-	-	-	-
НИИЯФ МГУ (Cleo 1)	19	0	19	2	0	2
Чебышев (НИВЦ МГУ)	5064	4840	224	343	149	194
ОИЯИ	15	15	-	-	-	-
ПИЯФ	48	45	3	0	0	0
РНЦ КИ	1	0	1	0	0	0
СПбГУ	9	6	3	0	0	0
ИПХФ РАН	9	7	2	0	0	0
Всего	5333	5009	324	375	178	197

Рис. 4: Список сайтов, подключенных к ГридННС

Каждый ресурс, подключаемый к ГридННС обязан публиковать информацию о имеющемся оборудовании, его параметрах, установленных операционных системах и прикладных программах и программных окружениях (Рис. 5). Это обеспечивает подбор ресурсов в соответствии с требованиями задач и обеспечивает необходимое окружение в процессе выполнения задач на ресурсе.



Параметры и структура сайтов ГридННС

[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
ВЦ ДВО РАН (Хабаровск)	globus.febras.net/subcluster0	5	40	CentOS 5.3 Final	abinit, версия 6.0.3 impi, версия 3.1 Весь список	abinit gridnnn moldyn nanochem nanospace sysadmin testbed
	globus.febras.net/subcluster1	8	32	CentOS 5.3 Final	abinit, версия 6.0.3 impi, версия 3.1 Весь список	abinit gridnnn moldyn nanochem nanospace sysadmin testbed
	globus.febras.net/subcluster2	24	48	CentOS 5.3 Final	abinit, версия 6.0.3 impi, версия 3.1 Весь список	abinit gridnnn moldyn nanochem nanospace sysadmin testbed
ИМСС УрО РАН(Пермь)	smsgrid.icmm.ru/subcluster0	4	16	CentOS 5.3 Final		
ИПХФ РАН	nnsgw.icp.ac.ru/subcluster0	2	8	CentOS 5.4 Final	mpich2, версия 1.2.1p1	abinit gridnnn nanochem nanospace sysadmin
	nnsgw.icp.ac.ru/subcluster1	1	1	CentOS 5.3 Final		abinit gridnnn nanochem nanospace sysadmin
						abinit gridnnn moldyn

Рис. 5: Параметры ресурсов ГридННС

Важной чертой ГридННС является обслуживание пользователей по виртуальным организациям (ВО) [15]. Технология ВО позволяет не только упростить процедуру получения доступа пользователя на ресурс, но и организовать учет потребления ресурсов по ВО.

Очень важно обеспечить удобный доступ пользователей к ресурсам. Для это в ГридННС существует два способа. Первый из них — это интерфейс командной строки (ИКС). Второй — специализированный графический веб-интерфейс (Рис. 6).

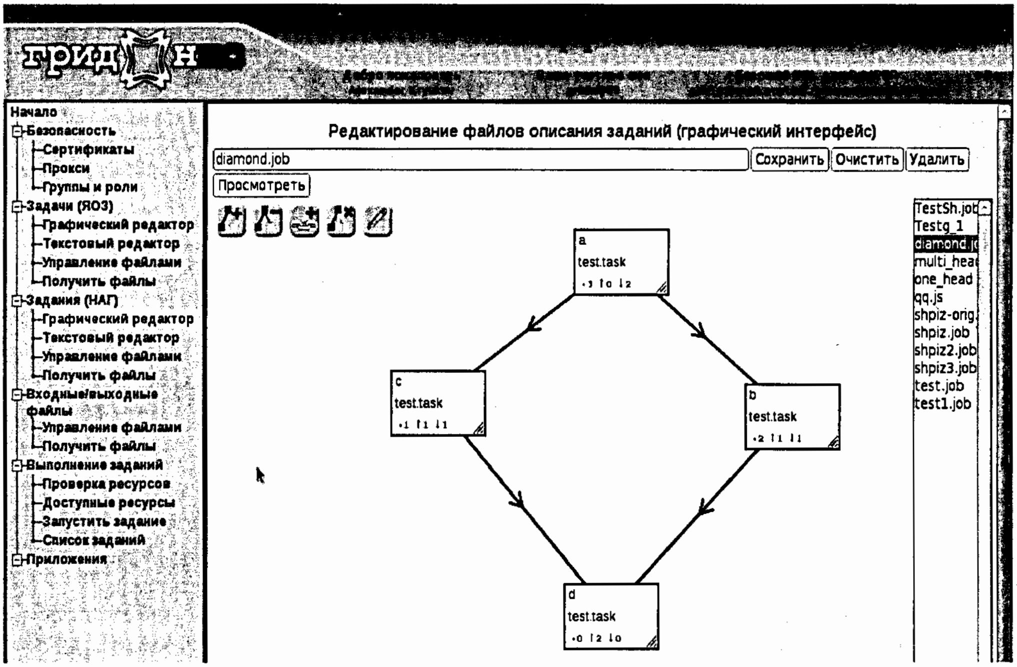


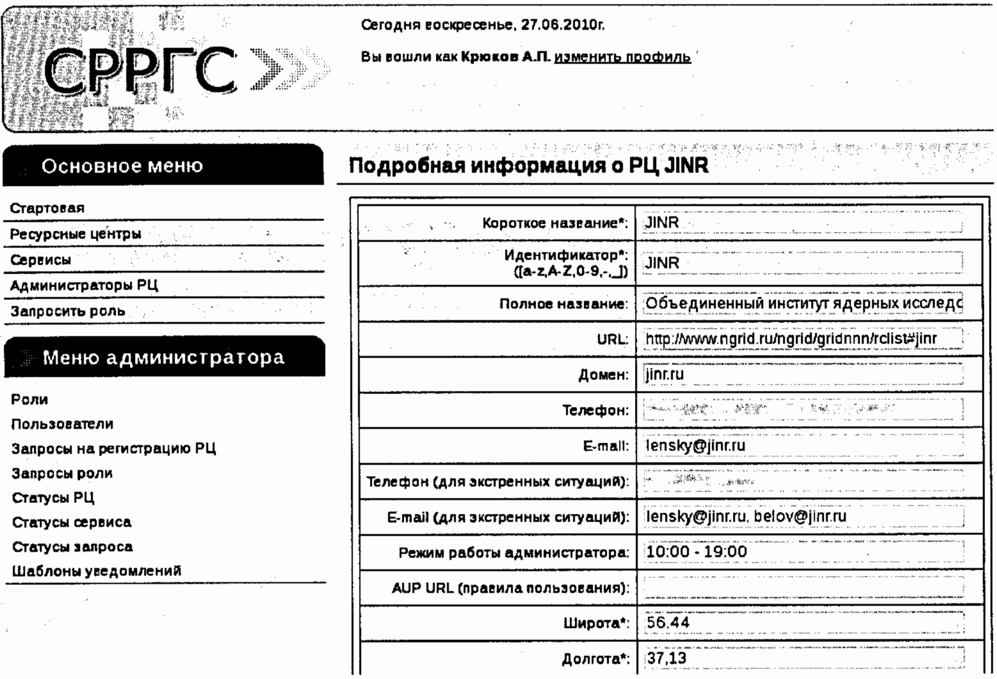
Рис. 6: Графический интерфейс пользователя

Интерфейс командной строки особенно удобен, если пользователь использует скрипты для автоматизации запуска заданий. ИКС представляет собой набор следующих утилит (Рис. 7).

```
pilot-job-submit [options] job_definition.js
pilot-job-status [options] URI
pilot-task-status [options] URI
pilot-job-pause [options] URI
pilot-job-cancel [options] URI
pilot-query-jobs [options]
pilot-job-resume [options] URI
```

Рис. 7: Интерфейс командной строки

Для управления функционированием инфраструктурой, а также для автоматического обнаружения ресурсов сервисами ГридННС (Рис. 8), реализован специальный сервис регистрации всех компонент ГридННС. Доступ пользователей и/или сервисов разрешен только для зарегистрированный служб и сервисов. Этим обеспечивается дополнительный уровень безопасности и исключает возможность подстановки поддельных (fake) сервисов.



Сегодня воскресенье, 27.06.2010г.

Вы вошли как Крюков А.П. изменить профиль

Подробная информация о РЦ JINR

Короткое название*:	JINR
Идентификатор*: ([a-z,A-Z,0-9,-,_])	JINR
Полное название:	Объединенный институт ядерных исслед
URL:	http://www.ngrid.ru/ngrid/gridnnn/rclist#jinr
Домен:	jinr.ru
Телефон:	[redacted]
E-mail:	lensky@jinr.ru
Телефон (для экстренных ситуаций):	[redacted]
E-mail (для экстренных ситуаций):	lensky@jinr.ru, belov@jinr.ru
Режим работы администратора:	10:00 - 19:00
AUP URL (правила пользования):	
Широта*:	56,44
Долгота*:	37,13

Рис. 8: Сервис регистрации

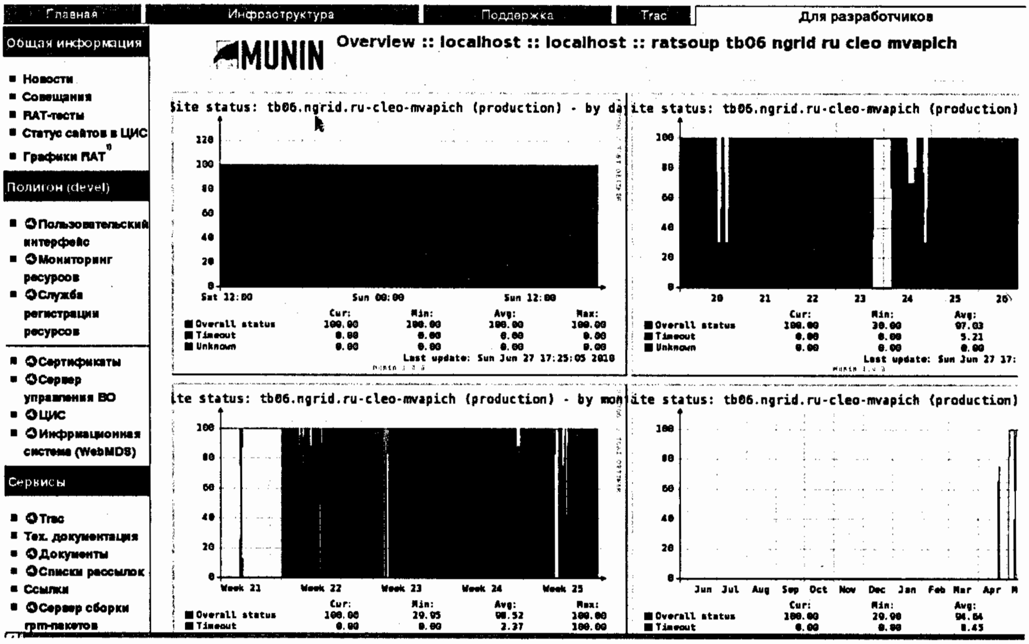
Большое внимание при разработке ГридННС было уделено организационным вопросам. Сюда входят вопросы регистрации пользователей, ВО, сайтов. Были также разработаны регламенты, регулирующие вопросы взаимодействия различных служб и сервисов, а также документация по установке и эксплуатации специального ПО ГридННС. Со всем этим можно ознакомиться на сайте проекта http://www.ngrid.ru. Отдельно стоит отметить службу поддержки пользователей. Она включает списки рассылки, через которые пользователи могут самостоятельно обсудить свои проблемы, система работы с билетами и имеет специальные страницы ЧАВО на сайте проекта.

Для контроля работы ГридННС регулярно производятся запуски тестовых задач (Рис. 9), а также запущен сервис Nagios [16], который позволяет системным администраторам сервисов проанализировать состояние своего сервиса или сайта более детально.

На текущем этапе основными задачами проекта являются

- подготовка дистрибутива,
- расширение числа сайтов,
- улучшение поддержки пользователей и системных администраторов,
- повышение стабильности работы сайтов.

В заключение отметим, что дальнейшее улучшение функциональности ГридННС будет проводиться на основе обобщения практического опыта эксплуатации системы. Но уже сейчас видно, что необходима разработка проблемно-ориентированных интерфейсов для подготовки композитных заданий, требующих использования сложных программных систем и комплексов.


Главная
Инфраструктура
Поддержка
Trac
Для разработчиков
Общая информация
Новости
Совещания
RAT-тесты
Статус сайтов в ЦИС
Графики RAT
Полигон (devel)
Пользовательский интерфейс
Мониторинг ресурсов
Служба регистрации ресурсов
Сертификаты
Сервер управления ВО
ЦИС
Информационная система (WebMDS)
Сервисы
Trac
Тех. документация
Документы
Списки рассылок
Ссылки
Сервер сборки rpm-пакетов
MUNIN
Overview :: localhost :: localhost :: ratsoup tb06 ngrid ru cleo mvapich
Site status: tb06.ngrid.ru-cleo-mvapich (production) - by day
ite status: tb06.ngrid.ru-cleo-mvapich (production)
ite status: tb06.ngrid.ru-cleo-mvapich (production) - by mon
ite status: tb06.ngrid.ru-cleo-mvapich (production)
120
100
80
60
40
20
0
Sat 12:00
Sun 00:00
Sun 12:00
20
21
22
23
24
25
26
Week 21
Week 22
Week 23
Week 24
Week 25
Jun
Jul
Aug
Sep
Oct
Nov
Dec
Jan
Feb
Mar
Apr
M

	Cur:	Min:	Avg:	Max:
■ Overall status	100.00	100.00	100.00	100.00
■ Timeout	0.00	0.00	0.00	0.00
■ Unknown	0.00	0.00	0.00	0.00

Last update: Sun Jun 27 17:25:05 2010

	Cur:	Min:	Avg:
■ Overall status	100.00	30.00	97.03
■ Timeout	0.00	0.00	5.21
■ Unknown	0.00	0.00	0.00

Last update: Sun Jun 27 17:

	Cur:	Min:	Avg:	Max:
■ Overall status	100.00	29.95	98.52	100.00
■ Timeout	0.00	0.00	2.37	100.00

	Cur:	Min:	Avg:
■ Overall status	100.00	29.90	94.64
■ Timeout	0.00	0.00	8.45

Рис. 9: Контроль прохождения тестовых задач на сайтах

Литература

[1] Grid for National Nanotechnology Network, http://ngrid.ru/ngrid

[2] О национальной нанотехнологической сети. Постановление Правительства РФ от 23 апреля 2010 г., № 282.

[3] WLCG, http://lcg.web.cern.ch/lcg/

[4] Globus Toolkit 4.2, http://globus.org

[5] GT 4.0 Release Notes: WS MDS Aggregator Framework, http://www.globus.org/toolkit/docs/4.0/info/aggregator/WS_MDS_Aggregator_Release_Notes.html

[6] [PDF] GLUE Schema Specification version 1.3 Final - 16 Jan 2007, http://forge.gridforum.org/sf/docman/do/.../projects.glue.../doc14185

[7] Демичев А., Ильин В., Крюков А. и Шамардин Л. Реализация программного интерфейса грид-сервиса Pilot на основе архитектурного стиля REST. Вычисленные методы и программирование, т. 11, 2010. С. 62-65.

[8] JSON, http://www.json.org/, K. Zyp. A JSON Media Type for Describing the Structure and Meaning of JSON Documents. Technical report, IETF Network Working Group, March 2010. draft-zyp-json-schema-02; D. Crockford. The application/json Media Type for JavaScript Object Notation (JSON). Technical report, IETF Network Working Group, July 2006. RFC4627.

[9] Демичев А., Крюков А. и Шамардин Л. Принципы построения грид с использованием Restful-веб-сервисов. Программные продукты и системы. №4. г. Тверь, 2009.

[10] gLite, http://www.glite.org

[11] PBS Professional, http://www.pbsworks.com/Product.aspx?id=1

[12] TORQUE Resource Manager, http://www.clusterresources.com/products/torque-resource-manager.php

[13] SLURM: A Highly Scalable Resource Manager, https://computing.llnl.gov/linux/slurm/

[14] Cleo batch system, http://parcon.parallel.ru/cleo-eng.html

[15] Foster I., Kesselman C., Tuecke S. The Anatomy of the Grid. Enabling Scalable Virtual Organizations, www.globus.org/alliance/publications/papers/anatomy.pdf
Virtual Organization Membership Service, http://glite.web.cern.ch/glite/.

[16] Nagios - The Industry Standard In Open Source Monitoring, http://www.nagios.org/

ВОПРОСЫ СТАНДАРТИЗАЦИИ И ОБЕСПЕЧЕНИЯ ИНТЕРОПЕРАБЕЛЬНОСТИ В GRID-СИСТЕМАХ[1]

Е. Е. Журавлёв[1], В. Н. Корниенко[2], А. Я. Олейников[2]

[1]Физический институт РАН им. П.Н. Лебедева
[2]Институт радиотехники и электроники им. В.А. Котельникова РАН

Введение

Grid-системы представляют собой сугубо гетерогенную среду, узлы которой (вычислительные информационные ресурсы) заведомо реализованы на разных программно-аппаратных платформах. В такой среде естественно возникает проблема взаимодействия разнородных платформ. Эта проблема получила название «проблемы интероперабельности» и должна решаться на основе принципов открытых систем. Основной принцип открытых систем состоит в использовании согласованных наборов стандартов информационных технологий, называемых профилями. За рубежом разработкой стандартов и профилей для GRID-систем занимается международная организация Open Grid Forum, в рамках которого разработано к настоящему времени около 170 документов. В нашей стране тематика стандартизации GRID-систем включена по инициативе авторов в одно из научных направлений работ Отделения нанотехнологий и информационных технологий РАН, в Программу Президиума РАН №1 и Программу фундаментальных научных исследований государственных академий наук на 2008-2012 годы. К сожалению, масштаб работ пока далеко не соответствует важности проблемы. Авторы в течение нескольких лет ведут проект, касающийся вопросов применения принципов открытых систем к созданию GRID-систем, сосредоточив в настоящее время внимание на проблеме интероперабельности. Из-за недостаточного финансирования работы сводятся в основном к составлению обзоров достижений Open Grid Forum, а тем временем с участием ОИЯИ и других научных центров постепенно создается национальный сегмент GRID-системы, создается масштабная GRID-система национальной нанотехнологической сети, есть понимание того, что для управления энергетической системой страны также понадобится GRID-система. Т.е., работы по GRID-системам переходят в практическую область, а вопросы стандартизации и обеспечения интероперабельности находятся в самой начальной стадии. Пока не создано ни одного национального стандарта и профиля GRID-систем.

В настоящей работе, в основу которой положен наш доклад на 4 международной конференции (Дубна 28 июня - 3 июля 2010), сформулирована проблема обеспечения интероперабельности в GRID-системах, описан подход к обеспечению интероперабельности в рамках Open Grid Forum, описаны результаты, полученные авторами, новые возможности по созданию национальных стандартов информационных технологий, гармонизированных с международными.

Проблема интероперабельности в GRID-системах

GRID – относительно новая компьютерная технология, основанная на объединении в единую инфраструктуру разнородных (гетерогенных) информационных и вычислительных ресурсов, которые могут находиться на значительном удалении друг от друга и являются административно независимыми. Одной из основных особенностей GRID-системы является ее

[1] Работа выполнена в рамках проекта РФФИ 09-07-00171-а и Программы Президиума РАН №1.

динамичность: отдельные элементы среды могут быть подключены в общую систему или выведены из нее, вообще говоря, в любой момент времени. Это свойство налагает серьезные требования как на внутреннюю программно-аппаратную структуру каждого элемента, так и на коммуникационную среду, используемую для создания GRID.

Решение названных вопросов невозможно без построения наиболее общей модели рассматриваемой системы, основанной на технологии открытых систем [1]. Такая модель позволяет провести продуманную и согласованную стандартизацию функционирования отдельных узлов системы и их взаимодействия. При создании такой модели необходимо провести детальный анализ системы с целью выделения ее функциональных особенностей, формирования обособленных групп сервисов и служб, отвечающих за реализацию общей работоспособности системы. В результате такого анализа становятся понятными назначения отдельных компонент системы и происходит согласование взаимодействия между ними. Таким образом, в основе модели будут лежать описание собственно свойств системы и требования к интерфейсам, через которые осуществляется взаимодействие элементов системы и собственно самой системы с внешней средой (например, пользователем). Основным отличием модели, построенной при использовании технологии открытых систем, от других моделей состоит в том, что налагаемые ею требования на службы и интерфейсы используют наборы открытых стандартов. По определению, открытые спецификации и стандарты – документы, получившие признание и применение и утверждённые полномочными организациями на национальном и международном уровне как результат консенсуса экспертов. Открытые стандарты являются общедоступными, и реализация той или иной структуры (системы) может быть выполнена любым разработчиком. Такой подход близок по своей сути к способам реализации GRID-систем: отдельные элементы GRID создаются различными группами разработчиков, которые могут использовать, вообще говоря, различные подходы к реализации необходимых свойств.

Из сказанного следует, что создание набора требований, налагаемых как на отдельные элементы GRID-систем, так и на способ их взаимодействия с внешней средой, должно осуществляться на основе технологии открытых систем. Отметим, что, согласно нормативному документу IS0/IEC TR 14252:1996 [2], открытая система - система, реализующая достаточно открытые спецификации или стандарты для интерфейсов, служб и форматов, с тем, чтобы облегчить должным образом созданному приложению:

- перенос с минимальными изменениями в широком диапазоне систем, полученных от одного или нескольких поставщиков;
- интероперабельность с другими приложениями, расположенными на местных или удаленных системах;
- взаимодействие с операторами в стиле, облегчающем «переносимость» интерфейса пользователя.

Основным документом, содержащим полное и документальное описание открытой системы, является документ, представляющий собрание открытых спецификаций и стандартов на функциональность системы. Этот документ (являющийся, в случае его утверждения в качестве стандарта, функциональным стандартом) получил название профиля системы, в данном случае профилем GRID-системы.

Одним из свойств открытости является свойство интероперабельности: (согласно ISO/IEC FCD 24765-Systems and Software Engineering-Vocabulary) способности двух или более систем или элементов к обмену информацией и к использованию информации, полученной в результате обмена. Как можно видеть, это свойство должно быть присуще любой GRID-системе. Эталонная модель интероперабельности приведена в [3].

Подход к обеспечению интероперабельности в рамках Open Grid Forum

С целью решения проблемы интероперабельности была создана международная организация, занимающаяся созданием GRID-систем, которая получила название Open GRID Forum (http://www.ogf.org). Деятельность форума направлена на ускоренное развитие прикладных распределенных вычислений и систем, позволяющих проводить такие вычисления. Участники форума направляют свои усилия на проведение глобальной стандартизации в области GRID-систем.

Одним из основных результатов деятельности форума являются нормативные документы, созданные в результате длительного неформального обсуждения предложений, вносимых на рассмотрение участниками форума. Эти нормативные документы имеют либо информационный, либо рекомендательный статус.

Open GRID Forum [4] принял для GRID-вычислений детальную последовательность создания открытой архитектуры служб на основе стандартов (Open GRID Services Architecture - сокращенно OGSA). Здесь слово «открытая» относится к процессу разработки стандартов, которые позволят достигнуть интероперабельности. GRID-среда «сервис – ориентирована», поскольку она обеспечивает функциональность служб как спаренных взаимодействующих служб, согласованных со стандартами на Web – службы. Слово «архитектура» характеризует определённые компоненты, их организацию и взаимодействие, и применяемые принципы конструирования.

Подход Open Grid Forum к общему методу создания интероперабельной GRID – среды можно записать в виде следующей последовательности:

Концепция – Архитектура – Модель – Проектирование -Реализация
(*Framework- Architecture-Model- Design- Solution*)

Деятельность OGSA-WG можно разделить на три направления:

1) архитектурный процесс,
2) спецификации и профили,
3) программное обеспечение.

Эти направления деятельности нацелены на координацию усилий по установлению соответствия всех документов OGSA и стандартов GRID.

Спецификации и профили OGSA являются нормативными документами. Спецификации - это документы, содержащие точные технические требования с включением интерфейсов, протоколов и поведения, отвечающих требованию конформности компонентов оборудования и программного обеспечения. В профиль OGSA входит некий набор широко распространённых признанных технических спецификаций программного обеспечения, которое будет управлять интероперабельной GRID-средой.

Программное обеспечение OGSA согласуют с нормативными документами и профилями OGSA, давая тем самым возможность запускать интероперабельные решения GRID, даже если они базируются на программно-аппаратных решениях, созданными разными производителями или взяты из разных источников. На рис.1 показана структура документов OGSA в их взаимосвязи, особенно связи информационных документов, профилей и действующих нормативных спецификаций.

Документы рабочей группы (OGSA – WG documents) содержат основополагающий документ (Base document), включающий:

а) OGSA Use Cases – условия пользователей, собранные рабочей группой;
б) OGSA Architecture – архитектуру, разработанную рабочей группой;
в) OGSA Roadmap – дорожную карту разработки документов архитектуры.

Внимательное рассмотрение приведенной структуры показывает, что ее можно рассматривать как развитие известных эталонных моделей применительно к GRID-среде.

На основе упомянутого документа ведутся разработки профилей с использованием следующих документов:

Guidelines (Profile Definition, Modelling Guidelines) – Руководства (контекст профиля, руководство по моделированию);

Documents produced by other OGF WGs or other SDOs – документов, выпущенных соседними рабочими группами или родственными организациями по стандартизации (информационные модели, необходимые спецификации).

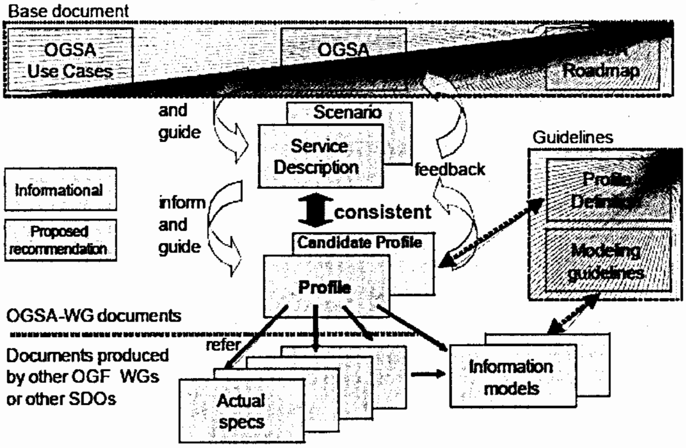


Рис.1: Структура и взаимосвязь документов OGSA по обеспечению интероперабельности в GRID-среде

К настоящему времени существуют более 170 документов Open Grid Forum, относящихся к различным проблемам GRID-технологий, метакомпьютингу и др. Как это следует из вышесказанного, из этого набора в отдельную группу можно выделить документы, касающиеся архитектуры сервисов открытых GRID-систем. Номера и названия документов OGSA, имеющих рекомендательный статус, приведены в Табл. 1.

Таблица 1. Список нормативных документов Open Grid Forum, касающихся OGSA

№	Номер документа	Название
1	GFD.072- P- Rec.	Архитектура сервисов открытых GRID. Базовый профиль: концептуальная основа ресурсов WEB сервисов (версия 1.0)
2	GFD. 087 P -Rec.	Спецификация байтного ввода/вывода, версия 1.0
3	GFD. 101 P -Rec.	Спецификация сервиса пространства имен ресурсов
4	GFD. 108 -Rec.	Архитектура сервисов открытых GRID. Базовый сервис выполнения заданий, версия 1.0
5	GFD.111 P -Rec.	Расширение языка описания заданий для высокопроизводительных вычислений, версия 1.0
6	GFD. 114 – P- Rec.	Базовый профиль высокопроизводительных вычислений, версия 1.0
7	GFD.115 P- Rec.	Расширение языка описания заданий для задач «один процесс – множество данных»
8	GFD. 129-Rec.	Спецификация интерфейса менеджера ресурсов хранения данных, версия 2.2
9	GFD. 131- P - Rec.	Профиль защищенной адресации, версия 1.0
10	GFD. 132- P- Rec.	Профиль защищенной связи, версия 1.0
11	GFD.135 P- Rec.	Профиль файловой платформы для высокопроизводительных вычислений, версия 1.0
12	GFD.136 Rec.	Спецификация языка описания представления задач, версия 1.0

№	Номер документа	Название
13	GFD. 138 P- Rec.	Основной профиль безопасности, версия 2.0
14	GFD. 144 – P- Rec.	Расширение простого интерфейса прикладных программ GRID
15	GFD. 147 - P -Rec.	Спецификация однородной GRID среды, версия 2.0
16	GFD.149 P-Rec.	Расширение языка описания представления задач для задач с изменяемыми параметрами

Результаты проводимой форумом работы дают основание полагать, что создание гармонизированного набора стандартов в области GRID-технологий позволит, в какой-то мере, решить задачу интероперабельности отдельных узлов, входящих в состав динамической системы, которой является GRID.

Как уже отмечалось выше, работу по созданию группы стандартов, а затем и профиля GRID-системы следует начинать с построения адекватной модели среды GRID, как открытой системы. Для этого необходимо провести детальный анализ минимального набора служб и сервисов, обеспечивающих работоспособность GRID.

Обеспечение полномасштабного функционирования GRID-структуры основывается на реализации целого ряда сервисов, к которым, в частности можно отнести: службу управления вычислениями (в том числе и составления расписаний), службу контроля вычислительных ресурсов, службы внешних интерфейсов (в том числе и WEB-сервисов) и др.

Так, к ключевым службам, определяющим интероперабельность , относятся:

– Job Submission;
- Information Services;
- Storage Management;
- Accounting;
- Job Monitoring;
- Database Access;
- Virtual Organization Management.

В основе интероперабельности для Job Submission лежит строгое следование языку JSDL и стандартам основ службы исполнения (**GFD.108** OGSA® Basic Execution Service Version 1.0 REC I. Foster, A. Grimshaw, P. Lane, W. Lee, M. Morgan, S. Newhouse, S. Pickles, D. Pulsipher, C. Smith, M. Theimer 2007-08-07 **Compute OGSA-BES-WG**).

Одним из сервисов выступает служба приема задания пользователя. В качестве такого задания может выступать запрос на проведение вычислений (с конкретным вычислительным заданием), на доступ к информации распределенной базы данных и т.д. Генерироваться такой запрос может как пользователем, так и автоматически (например, одним из составляющих GRID-систему узлов). В виду существенной гетерогенности GRID-структуры, форма представления задания пользователя должна быть стандартизована.

А одним из возможных способов представления задания пользователя является разработанный в рамках Open Grid Forum язык описания задания JSDL (Lob Submission Description Language).

Результаты, полученные авторами

Авторы, начиная с 1993 г., проводили систематизированные работы в области открытых систем и их применения в ряде областей (здравоохранение, образование, промышленность, военное дело и др.). Журнал «Информационные технологии и вычислительные системы» четыре своих выпуска (1997, 2003, 2006, 2009 г.г.) посвящал проблеме открытых систем, в том числе в GRID [6]. Нами строились проекты профилей для Grid-систем, в частности профиль интероперабельности. Однако наполнение профилей выполнялось только англоязычными стандартами, что делает невозможным их применение при построении национального сегмента GRID-системы. Дело в том, что согласно ФЗ «О техническом регулировании» в нашей стране действуют национальные стандарты, гармонизированные с международными. Превратить 170 стандартов, разработанных в Open Grid Forum, в стандарты РФ за достаточно короткое время невозможно, да и бессмысленно. Поэтому мы начали с первоначального набора из девяти документов, которые входят в перечень из 9 стандартов, названных в Open Grid Forum «полностью одобренными» (Full Recommendation) и назвали его базовым профилем GRID-системы. И в качестве первого шага провели адаптацию языка описания задания JSDL (Рис. 2).

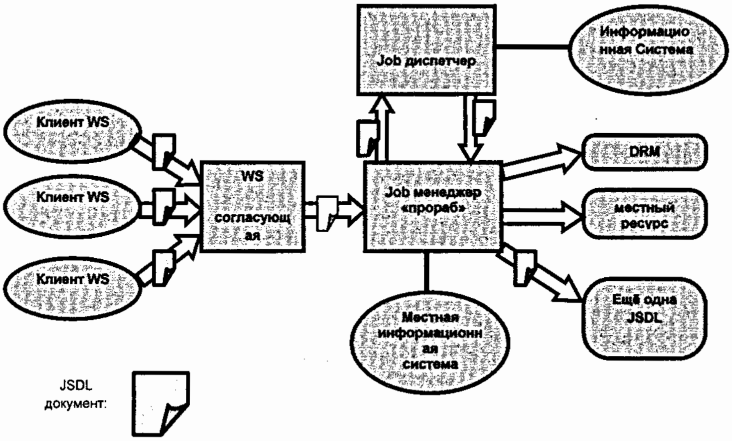


Рис. 2: Роль языка JSDL в GRID-среде

Язык JSDL основан на языке XML, содержит в себе описание элементов (и их допустимых значений), при помощи которых можно формировать как собственно задание, так и требования на программно-аппаратную платформу, на которой это задание может быть выполнено. JSDL хорошо зарекомендовал себя в мировой практике и, в виду наличия открытого стандарта, может быть успешно реализован разработчиками GRID-систем.

Для обеспечения интероперабельности различных GRID-систем или отдельных узлов внутри GRID-структуры, имело бы смысл организовать поток заданий, основываясь на едином подходе описания пользовательских задач. JSDL предоставляет такую возможность. Существенным его преимуществом является XML-подобная структура, что делает документ, написанный на JSDL, читаемым даже без наличия специального интерпретатора. Ниже приведен пример функционального JSDL-документа (http://www.gria.org/)

```
<?xml version="1.0" encoding="UTF-8"?>
<JobDefinition xmlns="http://schemas.ggf.org/jsdl/2005/11/jsdl">
<JobDescription>
<JobIdentification>
<JobName>http://it-innovation.soton.ac.uk/grid/imagemagick/blend 1</JobName>
</JobIdentification>
<Application>
<ApplicationName>http://it-
innovation.soton.ac.uk/grid/imagemagick/blend</ApplicationName>
</Application>
<DataStaging name="inputImage-0">
<FileName>inputImage-0</FileName>
<CreationFlag>overwrite</CreationFlag>
<DeleteOnTermination>true</DeleteOnTermination>
```

```
</DataStaging>
<DataStaging name="inputImage-1">
<FileName>inputImage-1</FileName>
<CreationFlag>overwrite</CreationFlag>
<DeleteOnTermination>true</DeleteOnTermination>
</DataStaging>
<DataStaging name="outputImage">
<FileName>outputImage</FileName>
<CreationFlag>overwrite</CreationFlag>
<DeleteOnTermination>true</DeleteOnTermination>
</DataStaging>
</JobDescription>
</JobDefinition>
```

В виду широкого распространения и относительной простоты восприятия, представляется возможным рекомендовать JSDL в качестве стандартного языка описания заданий пользователя. В качестве первого шага на пути внедрения JSDL целесообразно провести адаптацию ряда документов GFD для российских пользователей и разработчиков GRID-структур.

Новые возможности по созданию национальных стандартов информационных технологий

До последнего время в нашей стране в области ИТ-стандартизации имелось значительное отставание от мирового уровня. Российские специалисты практически не участвовали в работах международных организаций по стандартизации. Однако, в последнее время наметились определенные сдвиги в этом направлении:

- создан Межотраслевой Совет по техническому регулированию, стандартизации и оценке соответствия в сфере информационных технологий (http://www.msovit.ru);

- реорганизован технический комитет ТК22 «Информационные технологии» Росстандарта, являющийся национальным «зеркалом» международного технического комитета JTC1 ISO/IEC;

- принята поправка к ФЗ «О техническом регулировании», кардинально упрощающая процедуру адаптации международных стандартов;

- разрабатывается система машинного перевода ИТ-стандартов.

Институт радиотехники и электроники им. В.А,Котельникова РАН активно участвует в работе ТК22.

6-7 октября проведены конференции «ИТ-стандарт 2010» и СИТОП2010, отражающие имеющиеся сдвиги.

Все перечисленное открывает новые возможности для создания национальных стандартов, обеспечивающих интероперабельность в Grid-системах, конечно при соответствующем финансировании.

Заключение

Таким образом, на основании изложенного можно сделать следующие выводы:

- дальнейшее развитие работ по созданию российского сегмента GRID-системы, его интеграции в мировую GRID-систему требует форсированной разработки национальных стандартов и профилей, обеспечивающих интероперабельность в GRID-среде.

- имеющийся задел, опыт и новые возможности в области ИТ-стандартизации позволяют решить названную задачу при соответствующем финансировании.

- деятельность должна вестись в тесном сотрудничестве с международным сообществом в первую очередь с Open Grid Forum и при активном обсуждении с российскими специалистами, работающими в области GRID-систем.

Литература

[1] Технология открытых систем/ Под ред. А.Я. Олейникова. М.: Янус-К, 2004, 288 с.

[2] ISO/IEC TR 14252:1996.

[3] Батоврин В.К., Гуляев Ю.В., Олейников А.Я. Обеспечение интероперабельности – основная тенденция открытых систем // ИТ и ВС. №5, 2009. С. 7.

[4] GFD 141[1] Independent Software Vendors (ISV) Remote Computing Usage Primer, INFO, S. Newhouse, A. Grimshaw, 2008-10-07, Architecture, OGSA-WG.

[5] OGF Document Series, http://www.ogf.org/documents/ all//www.eu-egee.org

[6] Каменщиков М.А., Корниенко В.Н. GRID и технология открытых систем // Информационные технологии и вычислительные системы, 2003. №3. С.45.

РАСПРЕДЕЛЕННАЯ ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА УЧЕБНОГО НАЗНАЧЕНИЯ НА БАЗЕ КОГНИТИВНЫХ И РЕАКТИВНЫХ ПРОГРАММНЫХ АГЕНТОВ

Е. И. Зайцев

Московский государственный университет приборостроения и информатики
Россия, 107846, Москва, ул. Стромынка, 20
zei@tsinet.ru

В настоящее время в образовательный процесс высших учебных заведений внедряются новые методы обучения с использованием передовых информационных технологий, в частности, интеллектуальных систем базирующихся на знаниях (СОЗ, KBS – Knowledge-Based Systems). Основанные на знаниях системы учебного назначения, такие как интеллектуальные учебные среды (Intelligent Learning Environments) и интеллектуальные обучающие системы (Intelligent Tutoring Systems) увеличивают эффективность труда преподавателей и повышают качество подготовки специалистов. Также активно создаются и эксплуатируются распределенные системы учебного назначения (Distributed Teaching Systems), дальнейшее развитие которых связано с интеграцией распределенных вычислений и методов искусственного интеллекта с целью разработки распределенных интеллектуальных систем учебного назначения, таких как многоагентные банки знаний (МБЗ, MKB – Multi-agent Knowledge Banks) [1].

Примером интеллектуальных учебных сред являются статические банки знаний (СБЗ, SKB – Static Knowledge Banks) [2], которые формируют ответы на запросы пользователей посредством выполнения специализированных процедур поиска и логической обработки знаний. В СБЗ знания о моделируемых в базах статических предметных областях (ПО) представляются с помощью специальных фреймов-прототипов. Фреймы-прототипы описывают объекты и их состояния, действия и события, а также процессы, под которыми понимаются упорядоченные совокупности событий и/или других процессов, реализуемых в целях решения тех или иных проблем. В процессе обработки в СБЗ знаний о ПО формируются ответы на запросы пользователей о значениях различных характеристик объектов и событий, о сравнении и анализе событий, выявлении связей между событиями, а также запросы о синтезе планов действий для решения тех или иных задач, то есть о формировании упорядоченных совокупностей событий, обеспечивающих эти решения.

Развитием концепции статических банков знаний являются многоагентные банки знаний, которые не только выполняют функции интеллектуальных учебных сред, но также выступают в качестве интеллектуальных обучающих систем. МБЗ включают общие и специальные знания о предметной области, о процессе обучения и модели обучаемого, ассоциируя их с реактивными и когнитивными программными агентами [3], которые реализуют процедуры обработки этих знаний, выдают ответы на запросы пользователей и строят рациональную стратегию обучения, совершенствующуюся по мере накопления данных. В отличие от статических банков знаний, успех которых целиком зависит от мотивации учащихся и их самодисциплины, в МБЗ осуществляется проверка действий обучаемых с использованием динамичной обратной связи для адаптивной реакции на действия учащихся, а также отложенной обратной связи для периодической оценки их знаний.

Для формирования ответов на запросы пользователей о значениях различных характеристик объектов и событий, о сравнении и анализе событий, выявлении связей между событиями, а также запросы о формировании упорядоченных совокупностей событий, обеспечивающих решения тех или иных задач, в МБЗ вместо одного интеллектуального решателя используется многоуровневая сеть программных агентов. Агенты реализуют логические выводы на основе знаний о моделируемых в их базах предметных областях, которые, как и в СБЗ, могут быть представлены с помощью фреймов-прототипов.

Ответы на запросы пользователей формируются когнитивными агентами МБЗ в результате спецификации свойств сущностей (событий и их субъектов), вычисления каузальных, временных и других отношений на множестве сущностей, а также в результате планирования решения задач. При этом вычисление отношений и синтез плана действий для решения некоторой задачи происходят не только в процессе выполнения когнитивным агентом продукционных, редукционных или трансформационных правил, но также в процессе его переговоров с другими агентами. Многоуровневая архитектура МБЗ предполагает использование как горизонтальных, так и вертикальных связей между агентами. В ней присутствуют уровни ответственные за кооперативное поведение, локальное планирование, формирование намерений, восприятие и исполнение действий, реактивное поведение и обучение агента. Каждый агент функционирует в соответствии с кооперативными обязательствами, которые возложены на агента другими агентами МБЗ.

Одним из принципиальных преимуществ МБЗ является слабая связность, продиктованная подходом на основе контрактов. Контракты представляют собой спецификации требований к предоставляемым и требуемым агентом интерфейсам, реализующим протокол взаимодействия. Разбиение приложения на автономные агенты, чьи взаимодействия управляются правильно специфицированными контрактами на основе сообщений, облегчает идентификацию естественного параллелизма, который существует в контексте предметной области, и понимание того, как следует проводить декомпозицию работ, которые можно выполнять одновременно. Разработка банка знаний на основе агентов дает разработчику возможность сосредоточиться на корректном моделировании задач, выполняемых агентами, а не на стремлении явно управлять параллелизмом в программе.

Использование для формирования ответов на запросы пользователей одновременно нескольких программных агентов не только повышает производительность системы за счет параллелизма, но и расширяет возможности банков знаний по предоставлению пользователям обобщенной информации. Агенты, распределенные в узлах локальной или глобальной вычислительной сети, способны предоставлять или рекомендовать обучающие материалы на основе обобщения предпочтений, поведения и представлений определенных групп пользователей системы. При этом, протоколы, используемые в МБЗ, должны включать возможность реализации мобильных процессов, коммуникации между которыми могут быть прерваны и восстановлены.

Реализация мобильных программных агентов позволяет динамически перераспределять вычислительную нагрузку в зависимости от состояния сети. Если выполнение вычислений на одном из узлов стало не эффективно, программный агент может приостановить свою работу, переместиться на менее загруженный компьютер и продолжить работу на нем. Мобильный агент может последовательно посещать интересующие его вычислительные узлы, либо клонировать множество дочерних агентов, которые будут выполнять свои функции параллельно. Таким образом, мобильные агенты должны поддерживать “сильную” (strong mobility) модель, при которой вместе с сегментом кода переносится также сегмент исполнения, что позволяет работающему процессу после приостановки и перенесения на другую машину продолжить его выполнение с того места, на котором этот процесс был приостановлен.

При наличии в вычислительных узлах специального программного обеспечения промежуточного уровня, называемого агентной платформой, мобильные агенты могут работать на разных аппаратных платформах под управлением различных операционных систем. Агентная платформа отвечает за обеспечение жизнедеятельности агентов и представляет собой систему промежуточного уровня (middleware), которая находится между агентами и операционной системой. Основные функции агентной платформы состоят в управлении агентами, обеспечении передачи сообщений между агентами, в поиске агентов и данных о них внутри системы, поддержке онтологий. Платформа агентов позволяет передавать, принимать, регистрировать агентов, обеспечивает безопасность узла и устойчивость, то есть способность агентов восстанавливать свое состояние после аварийного завершения.

Важным отличием МБЗ от СБЗ является лежащая в их основе нейрологическая модель, в которой объединены семиотический и коннекционистский подходы в искусственном интеллекте. В нейрологической модели МБЗ интегрированы механизмы логического вывода с нейросетевыми моделями, а также методами обработки информации, основанными на нечеткой логике, которые реализованы как в механизме немонотонного вывода когнитивных агентов, так и в структуре нейронных сетей реактивных агентов. Благодаря этому, агенты МБЗ способны решать плохо формализуемые задачи в открытых, динамических проблемных областях, в которых данные и знания, описывающие сущности и связи, как правило, неполны, противоречивы, неточны и неопределенны.

Реактивные агенты способны извлекать знания из поступающих выборок, интерпретируя их как обучающие выборки, а также формулировать и реализовывать нечеткие запросы к базе данных. Когнитивные агенты, не обладая полным знанием о своем окружении и имея лишь частичное представление о проблеме, проводят неточные, предположительные рассуждения, которые подвергаются пересмотру (belief revision) при получении агентом дополнительной информации, несовместимой с полученными ранее заключениями, а также при изменении модели мира в результате обновления убеждений агентов (belief update). Когнитивные агенты, используют специальные механизмы, позволяющие им оперировать нечеткими понятиями и реализовывать прямые либо обратные нечеткие выводы. При прямом нечетком выводе (fuzzy forward-chaining reasoning) отдельные факты базы знаний когнитивных агентов преобразуются в конкретные значения функций принадлежности антецедентов нечетких продукций и находятся значения функций принадлежности заключений по каждому из нечетких правил. Процесс обратного нечеткого вывода (fuzzy backward-chaining reasoning) заключается в подстановке отдельных значений функций принадлежности заключений и нахождении функций принадлежности условий, которые принимаются в качестве очередных подцелей и далее могут использоваться как функции принадлежности новых заключений.

Создание многоагентных банков знаний является сложной задачей, которая требует от разработчиков опыта проектирования и реализации как концептуальных, так и технических решений в таких областях как представление и обработка знаний, сетевые коммуникации и протоколы взаимодействия, нейронные сети и нечеткая логика. Проектирование и реализация МБЗ значительно упрощается при использовании специальных инструментальных средств, поддерживающих процесс создания экземпляров и сборки агентов, обеспечивающих автоматическую генерацию частичных или полных реализаций на основе спецификации. Специализированные библиотеки и инструментальные средства разработки многоагентных систем учебного назначения, подобные проблемно-ориентированной среде AgentITS (МГУПИ), поддерживают процессы проектирования и реализации обучающих агентов с заданным поведением.

Для описания программных систем теория агентов предлагает такие высокоуровневые понятия как роли агентов, знания, убеждения, желания агентов, планы, цели, протоколы общения и ведения переговоров. Повышение уровня абстракции облегчает разработку программного обеспечения, при этом оно ограничивает область действия абстракций и объем контроля деталей реализации, порученного разработчикам. Агенты, как специфичные абстракции, содержат больше знаний о предметной области, по сравнению с общими и, следовательно, меньше знаний требуется от разработчика для решения проблемы. Однако, чем больше знаний сконцентрировано в абстракции, тем уже ее область приложения. Чтобы быть многократно используемой, высокоуровневая абстракция должна допускать адаптацию, как через некоторые внутренние механизмы изменчивости, так и через внешние адаптеры. В процессе создания МБЗ в среде AgentITS при реализации абстракции агента пользователь имеет возможность конфигурировать визуальное представление решения, а затем генерировать исходный код, который создаст экземпляры библиотечных классов и настроит их соответствующим образом.

Система AgentITS включает инструментальную среду разработки многоагентных банков знаний, а также агентную платформу, обеспечивающую создание сетевых соединений между агентами, поиск нужных агентов. Данный инструментарий, состоящий из интерактивных мастеров и панелей свойств, оптимизирован для создания интеллектуальных систем учебного назначения, которые должны осуществлять адаптивное обучение с использованием персональных обучающих агентов. Персонализация в обучении достигается за счет представления в МБЗ метазнаний для проведения индивидуального подбора и формирования учебных материалов. Непосредственный доступ к учебным материалам осуществляют агенты обучающих ресурсов. Технология МЗБ позволяет перемещать этих агентов к удаленным ресурсам и выполнять анализ полученной информации параллельно на нескольких вычислительных узлах. При разработке МБЗ проектируются и реализуются агенты мониторинга индивидуальной траектории обучения, агенты тестирования и контроля правильности действий обучающихся, агенты обратной связи МБЗ с обучающимися и преподавателями.

Литература

[1] Зайцев Е.И. Многоагентные банки знаний: архитектура и методы реализации// Сб. трудов XIII Международной научно-практической конференции “Фундаментальные и прикладные проблемы приборостроения, информатики и экономики”. М: МГУПИ, 2010.

[2] Миронов А.С. Представление и обработка знаний в статических банках знаний// Сб. трудов VIII Международной научно-практической конференции “Фундаментальные и прикладные проблемы приборостроения, информатики и экономики”. М: МГУПИ, 2005.

[3] Зайцев Е.И. Методология представления и обработки знаний в распределенных интеллектуальных информационных системах. М: Автоматизация и современные технологии. №1, 2008.

АСИНХРОННЫЕ АВТОМАТНЫЕ СХЕМЫ – МОДЕЛЬ РАСПРЕДЕЛЕННОЙ ОБРАБОТКИ ДАННЫХ

Г. А. Калинина, Ю. Е. Мороховец

Московский энергетический институт (технический университет)
Россия, 111250, Москва, Красноказарменная 14
KalininaGA@mpei.ru, MorokhovetsYY@mpei.ru

Под обработкой данных вообще и распределенной обработкой данных в частности будем понимать процесс непрерывного приема, преобразования, накопления и выдачи данных, направленный на решение определенного, часто фиксированного набора задач. Обработка данных – потенциально бесконечный во времени процесс, в общем случае не являющийся алгоритмическим, то есть обеспечивающим однозначность получаемых результатов. Вычисление, в силу данного определения, является самостоятельной смысловой категорией. Вычисление – конечный во времени процесс преобразования исходных данных в результат, основанный на применении алгоритма. Вычисление заканчивается, если результат получен.

Процесс обработки данных – это процесс выполнения схемы обработки данных системой обработки данных. Учитывая этот факт, понятие распределенной обработки данных может быть определено с двух точек зрения – логической и физической. С логической точки зрения свойство распределённости определяется особенностями синтаксиса и семантики схемы, задающей процесс обработки данных. В распределенном случае схема представляется множеством информационно связанных логических компонентов, не имеющих общего управления и работающих независимо друг от друга. Совместная работа компонентов координируется посредством обмена данными, причем этот обмен может носить как синхронный, так и асинхронный характер. С физической точки зрения свойство распределённости определяется организацией системы обработки данных, а также способом отображения логических компонентов схемы на физические компоненты системы. В распределенном случае система представляется множеством независимо работающих информационно связанных физических компонентов, каждый их которых выполняет отображенный на него функционально завершенный фрагмент схемы.

В докладе речь идет о логической модели, формализующей схемы распределенной обработки данных в виде асинхронных автоматных схем.

Асинхронную автоматную схему a, как элемент множества A, зададим структурой

$$a = \langle P, \Sigma, lnk \rangle,$$

где P и Σ – множества автоматных компонентов и их шаблонов; lnk – структурообразующее соответствие, устанавливающее информационные связи между выходами и входами компонентов схемы.

Множество автоматных компонентов P – конечное непустое множество. Каждый элемент $p \in P$ характеризуем шаблоном $\sigma(p) \in \Sigma$, определяющим его структуру и функционирование.

Базис автоматной схемы $\mathbf{\Sigma}$ определим как конечное множество структур вида

$$\sigma = \langle \boldsymbol{X}, \boldsymbol{Y}, \boldsymbol{M}, \boldsymbol{S}, F, \theta \rangle,$$

где $\boldsymbol{X}$ и $\boldsymbol{Y}$ – множества входов и выходов, обеспечивающих информационное взаимодействие компонентов схемы; $\boldsymbol{M}$ – множество состояний памяти компонента; $\boldsymbol{S}$ – множество собственных состояний компонента; F – множество процедур обработки данных, выполняемых компонентом; θ – функция выбора процедур обработки данных.

Каждый вход компонента из $\boldsymbol{X}$ характеризуем типом принимаемых сообщений. Тип самого входа обозначим через $\tau(x)$, $x \in \boldsymbol{X}$. Будем считать, что входы компонента имеют встроенные буферы, в которых хранятся сообщения до того, как компонент приступит к их обработке. Буфер входа $\boldsymbol{x}$ обозначим через $\boldsymbol{b}(\boldsymbol{x})$, а его тип положим равным типу соответствующего входа – $\tau(\boldsymbol{b}(\boldsymbol{x})) = \tau(\boldsymbol{x})$.

Для буфера $\boldsymbol{b}$ типа τ множество

$$Q(\boldsymbol{b}) = (D(\tau))^0 \cup (D(\tau))^1 \cup (D(\tau))^2 \cup \ldots \cup (D(\tau))^n \cup \ldots$$

положим в качестве множества его возможных состояний. Здесь $D(\tau)$ – множество сообщений типа τ, а $(D(\tau))^n$ – множество последовательностей указанных сообщений длины n, $n \geq 0$. Для любого типа τ будем считать, что $(D(\tau))^0 = \omega$, где символ «ω» обозначает пустую последовательность сообщений.

Пусть $\boldsymbol{b}$ – буфер типа τ, а q – текущее состояние этого буфера. Введем функции *append*, *head* и *tail* следующим образом:

$$append(d, q) = \begin{cases} (d), \text{ если } q = \omega \\ (d, d_1, \ldots, d_n), \text{ если } q = (d_1, \ldots, d_n), \end{cases}$$

$$head(q) = \begin{cases} \text{не определена, если } q = \omega \\ d_n, \text{ если } q = (d_1, \ldots, d_n), \end{cases}$$

$$tail(q) = \begin{cases} \text{не определена, если } q = \omega \\ (d_1, \ldots, d_{n-1}), \text{ если } q = (d_1, \ldots, d_n). \end{cases}$$

Функция *append* моделирует запись сообщения $d \in D(\tau)$ в буфер $\boldsymbol{b}$, находящийся в состоянии q, функции *head* и *tail* – чтение сообщения $d \in D(\tau)$ из буфера $\boldsymbol{b}$, находящегося в указанном состоянии.

Каждый выход из $\boldsymbol{Y}$ характеризуем типом. Тип выхода обозначим через $\tau(\boldsymbol{y})$, $\boldsymbol{y} \in \boldsymbol{Y}$.

В множествах $\boldsymbol{M}$ и $\boldsymbol{S}$ выделим состояния m_0 и s_0, которые примем в качестве начального состояния памяти и начального собственного состояния автоматного компонента, соответственно.

Произвольную процедуру обработки данных $f \in F$ зададим посредством структуры

$$f = \langle \varphi, fy, fm, fs \rangle,$$

где $\varphi = (\varphi_1, \ldots, \varphi_{nof(X)})$ – двоичный вектор маски входов процедуры; fy – функция выходов; fm – функция состояний памяти; fs – функция собственных состояний компонента. Здесь и далее nof – функция, возвращающая в качестве значения число элементов в множестве-аргументе.

Взаимосвязанные функции fy, fm и fs зададим как соответствия вида

$$\begin{aligned} fy\colon & (\varphi_1 \circ D(\tau(\boldsymbol{x}_1))) \times \ldots \times (\varphi_{nof(X)} \circ D(\tau(\boldsymbol{x}_{nof(X)}))) \times \boldsymbol{M} \times \boldsymbol{S} \to \\ & \to (\{\lambda\} \cup D(\tau(\boldsymbol{y}_1))) \times \ldots \times (\{\lambda\} \cup D(\tau(\boldsymbol{y}_{nof(Y)}))), \qquad (1) \\ fm\colon & (\varphi_1 \circ D(\tau(\boldsymbol{x}_1))) \times \ldots \times (\varphi_{nof(X)} \circ D(\tau(\boldsymbol{x}_{nof(X)}))) \times \boldsymbol{M} \times \boldsymbol{S} \to \boldsymbol{M}, \\ fs\colon & (\varphi_1 \circ D(\tau(\boldsymbol{x}_1))) \times \ldots \times (\varphi_{nof(X)} \circ D(\tau(\boldsymbol{x}_{nof(X)}))) \times \boldsymbol{M} \times \boldsymbol{S} \to \boldsymbol{S}, \end{aligned}$$

где λ – неопределенный блок данных. Символ «$\circ$» обозначает операцию, которую для двоичной переменной φ_j и множества $D(\tau(\boldsymbol{x}_j))$, $1 \leq j \leq nof(\boldsymbol{X})$ определим следующим образом:

$$\varphi_j \circ D(\tau(\boldsymbol{x}_j)) = \begin{cases} \{\lambda\}, \text{ если } \varphi_j = 0 \\ D(\tau(\boldsymbol{x}_j)), \text{ если } \varphi_j = 1. \end{cases}$$

Из сказанного следует, что вектор маски входов процедуры указывает входы компонента, данные буферов которых используются при ее выполнении.

Множество $U(\boldsymbol{s}) \subseteq \{0, 1\}^{nof(X)}$ назовем множеством возможных ситуаций на входах автоматного компонента в состоянии $\boldsymbol{s} \in \boldsymbol{S}$. При $nof(\boldsymbol{X}) > 0$ двоичный вектор ситуации $u = (u_1, \ldots, u_{nof(X)}) \in U(\boldsymbol{s})$ определяет наличие данных в буферах входов компонента: $u_j = 1$ означает наличие сообщений в буфере входа $\boldsymbol{x}_j$, а $u_j = 0$ – отсутствие сообщений в указанном буфере. При $nof(\boldsymbol{X}) = 0$ множество ситуаций на входах автоматного компонента не определено.

Функция выбора процедур обработки данных θ имеет вид

$$\theta\colon \boldsymbol{S}\times U\to F,$$

где U – множество допустимых ситуаций на входах компонента, являющееся объединением множеств $U(\boldsymbol{s})$ по всем $\boldsymbol{s}\in\boldsymbol{S}$.

Обозначим через $dom_s(\theta)$ сечение области определения функции выбора в точке $\boldsymbol{s}\in\boldsymbol{S}$. Множество $dom_s(\theta)\subseteq U(\boldsymbol{s})$ назовем множеством рабочих ситуаций на входах компонента, обрабатываемых в состоянии $\boldsymbol{s}$. Обозначим через $val_s(\theta)$ сечение области изменения функции выбора в точке $\boldsymbol{s}\in\boldsymbol{S}$. Множество $val_s(\theta)\subseteq F$ назовем множеством процедур обработки данных подчиненных состоянию $\boldsymbol{s}$.

Множество ситуаций $\Delta(s, f)\subseteq dom_s(\theta)$ назовем дискриминантом процедуры f в состоянии s, если и только если для любой ситуации $u\in\Delta(s,f)$ выполняется равенство $f=\theta(\boldsymbol{s}, u)$.

Маска входов φ процедуры f и дискриминант $\Delta(\boldsymbol{s}, f)$ должны быть связаны соотношением

$$\varphi = inf(\Delta(\boldsymbol{s}, f)),$$

где функция inf возвращает наименьшую точную грань множества-аргумента.

Функцию выбора процедур обработки данных θ для состояния $\boldsymbol{s}$ и ситуации на входах u конструктивно зададим следующим образом:

$$\theta(\boldsymbol{s}, u) = \begin{cases} f, \text{ если существует } \Delta(\boldsymbol{s}, f) \text{ такой, что } u\in\Delta(\boldsymbol{s}, f) \\ \text{не определена, в противном случае.} \end{cases}$$

Состояние $\boldsymbol{s}\in\boldsymbol{S}$ назовем заключительным состоянием в том случае, если сечение области определения функции выбора θ в точке s пусто, то есть $dom_s(\theta)=\varnothing$. Автоматный компонент, не имеющий заключительных состояний, назовем компонентом непрерывного действия.

Группой состояния $\boldsymbol{s}\in\boldsymbol{S}$ назовем структуру, включающую состояние $\boldsymbol{s}$ вместе с подчиненными ему процедурами из $val_s(\theta)\in F$. Впредь, без ограничения общности, будем рассматривать лишь компоненты с непересекающимися группами состояний, то есть такие, у которых пересечение множеств $val_s(\theta)$ по всем $\boldsymbol{s}\in\boldsymbol{S}$ пусто. Процедуры, образующие группу состояния, назовем ветвями обработки данных.

Взаимосвязь элементов множеств $\boldsymbol{S}$ и F, устанавливаемую функциями θ и fs, может быть представлена графически в виде диаграммы переходов компонента. Диаграмма переходов – двудольный ориентированный граф, одни вершины которого отображают состояния компонента, а другие – процедуры обработки данных. Дуги, идущие от состояний к процедурам, взвешены дискриминантами процедур – непустыми множествами ситуаций на входах компонента, обрабатываемых в соответствующих состояниях. Дуги, идущие от процедур к состояниям, показывают направления переходов компонента, реализуемые как результат выполнения процедур.

Для иллюстрации введенных понятий рассмотрим пример шаблона компонента непрерывного действия, реализующего унарное преобразование, зависящее от двух параметров:

$$\text{if } pi(x_1) \text{ then } y_1{:=}E(c_1, x_1) \text{ else } y_2{:=}E(c_2, x_1), \qquad (2)$$

где pi – булевская функция, а E – функция, при вычислении которой, в зависимости от значения $pi(x_1)$, применяется либо параметр c_1, либо параметр c_2.

Шаблон имеет пять входов $X=\{x_1, x_2, x_3, x_4, x_5\}$ и два выхода $Y=\{y_1, y_2\}$. Основной режим работы реальных компонентов, отвечающих шаблону, заключается в приеме сообщений из буфера входа x_1, их преобразовании и выдаче результатов на выходы y_1 или y_2, а также в приеме сообщений, поступающих в буферы входов x_2 и x_3, x_4 и x_5, обеспечивающих вычисление параметров c_1, c_2, соответственно. Значения параметров преобразования хранятся во внутренней памяти компонента. Начальные и все последующие значения параметров c_1, c_2 вычисляются операторами

$$c_1{:=}\,E_1(x_2, x_3),\quad c_2{:=}\,E_2(x_4, x_5), \qquad (3)$$

где E_1 и E_2 – заданные функции. Выполнение операторов (3) обусловливается наличием сообщений в буферах входов x_2 и x_3, x_4 и x_5, причем частота поступления сообщений в эти буферы значительно меньше частоты поступления сообщений в буфер входа x_1. Сложность вычисления функций E_1 и E_2 превосходит сложность вычисления функции E.

В начале работы вычисляются значения параметров c_1, c_2 и только затем компонент переходит в основной режим – периодически выполняя преобразование (2) при поступлении сообщений в буфер входа x_1, осуществляя пересчет значений параметров c_1, c_2, при поступлении сообщений в буферы соответствующих входов.

Вход x_1 имеет самый низкий приоритет. Сообщения принимаются из его буфера лишь тогда, когда данные на входах x_2 и x_3 или x_4 и x_5 отсутствуют. Взаимный приоритет пар входов x_2 и x_3, x_4 и x_5 не фиксирован – их приоритеты меняются таким образом, что обслуженной паре входов назначается более низкий приоритет.

Диаграмма переходов для рассматриваемого случая показана на рис. 1.

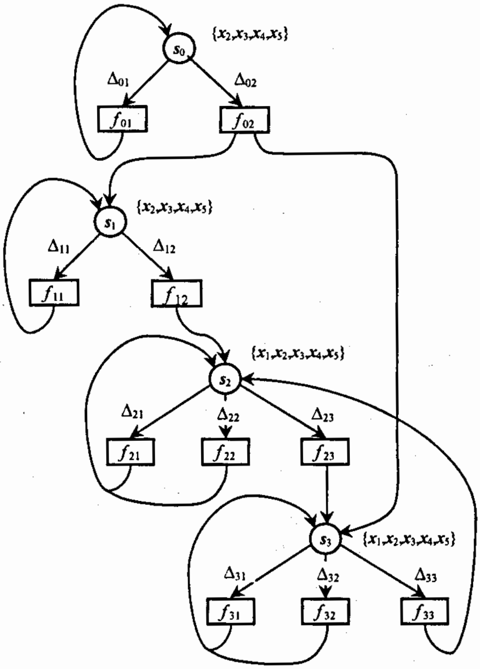


Рис. 1: Диаграмма переходов автоматного компонента

Начальное состояние внутренней памяти компонента характеризуется следующими значениями переменных: $c_1 = c_2 = 0$; $g = \mathit{false}$, где g – переменная булевского типа.

Дискриминанты и алгоритмы выполнения ветвей обработки данных для состояний, изображенных на рис. 1, имеют следующий вид.

Состояние s_0:

$\Delta_{01} = \Delta(s_0, f_{01}) = \{01100, 01101, 01110, 11100, 11101, 11110\}$;

ветвь f_{01}: $\varphi_{01} = 01100$; $c_1 := E_1(x_2, x_3)$; $g := \mathit{true}$; $\mathit{next}(s_0)$;

$\Delta_{02} = \Delta(s_0, f_{02}) = \{00011, 00111, 01011, 01111, 10011, 10111, 11011, 11111\}$;

ветвь f_{02}: $\varphi_{02} = 00011$; $c_2 := E_2(x_4, x_5)$; if $\mathit{not}(g)$ then $\mathit{next}(s_1)$ else $\mathit{next}(s_3)$;

Состояние s_1:

$\Delta_{11} = \Delta(s_1, f_{11}) = \{00011, 00111, 01011, 10011, 10111, 11011\}$;

ветвь f_{11}: $\varphi_{11} = 00011$; $c_2 := E_2(x_4, x_5)$; $\mathit{next}(s_1)$;

$\Delta_{12} = \Delta(s_1, f_{12}) = \{01100, 01101, 01110, 01111, 11100, 11101, 11110, 11111\}$;

ветвь f_{12}: $\varphi_{12} = 01100$; $c_1 := E_1(x_2, x_3)$; $\mathit{next}(s_2)$;

Состояние s_2:

$\Delta_{21} = \Delta\,(s_2, f_{21}) = \{10000, 10001, 10010, 10100, 10101, 10110, 11000, 11001, 11010\}$;

ветвь f_{21}: $\varphi_{21} = 10000$; if $pi(x_1)$ then $y_1 := E(c_1, x_1)$ else $y_2 := E(c_2, x_1)$; $next(s_2)$;

$\Delta_{22} = \Delta(s_2, f_{22}) = \Delta_{01} = \{01100, 01101, 01110, 11100, 11101, 11110\}$;

ветвь f_{22}: $\varphi_{22} = 01100$; $c_1 := E_1(x_2, x_3)$; $next(s_2)$;

$\Delta_{23} = \Delta(s_2, f_{23}) = \Delta_{02} = \{00011, 00111, 01011, 01111, 10011, 10111, 11011, 11111\}$;

ветвь f_{23}: $\varphi_{23} = 00011$; $c_2 := E_2(x_4, x_5)$; $next(s_3)$;

Состояние s_3:

$\Delta_{31} = \Delta(s_3, f_{31}) = \{10000, 10001, 10010, 10100, 10101, 10110, 11000, 11001, 11010\}$;

ветвь f_{31}: $\varphi_{31} = 10000$; if $pi(x_1)$ then $y_1 := E(c_1, x_1)$ else $y_2 := E(c_2, x_1)$; $next(s_3)$;

$\Delta_{32} = \Delta(s_3, f_{32}) = \Delta_{11} = \{00011, 00111, 01011, 10011, 10111, 11011\}$;

ветвь f_{32}: $\varphi_{32} = 00011$; $c_2 := E_2(x_4, x_5)$; $next(s_3)$;

$\Delta_{33} = \Delta(s_3, f_{33}) = \Delta_{12} = \{01100, 01101, 01110, 01111, 11100, 11101, 11110, 11111\}$;

ветвь f_{33}: $\varphi_{33} = 01100$; $c_1 := E_1(x_2, x_3)$; $next(s_2)$.

Сечение области определения функции θ, соответствующее состоянию s_3, показано на рис. 2.

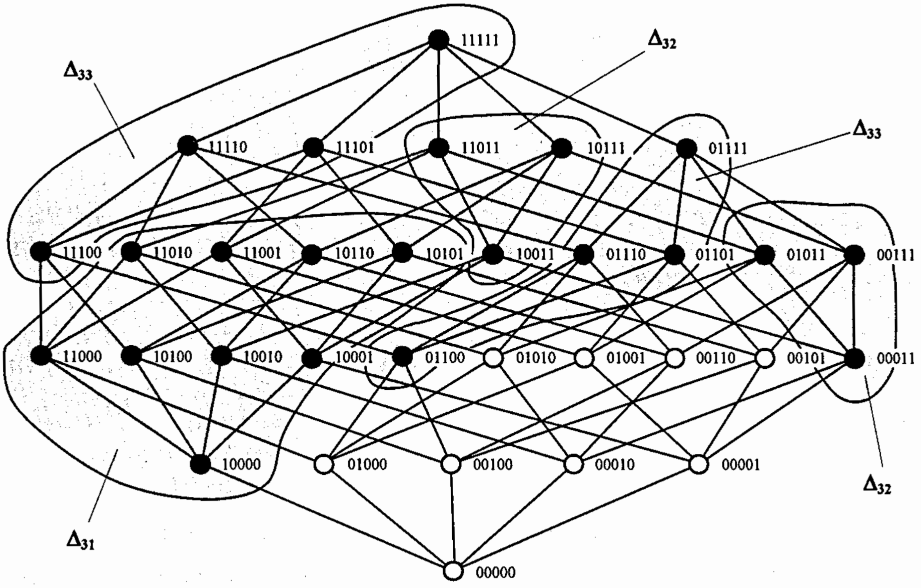


Рис. 2: Сечение области определения функции θ

Темными точками изображены рабочие ситуации, для которых функция выбора ветвей θ определена. Множества Δ_{31}, Δ_{32}, Δ_{33} – дискриминанты ветвей f_{31}, f_{32}, f_{33}, подчиненных состоянию s_3, представлены на рис. 2 в виде затемненных областей.

Структурообразующее соответствие *lnk* – биективное соответствие, обеспечивающее соединение выходов и входов компонентов «один к одному». Для определения соответствия *lnk* введем понятия реальных, не шаблонных входов и выходов компонентов схемы, а также распространим на них рассмотренное ранее понятие типа. Множество реальных, присутствующих в структуре схемы, входов компонента $p \in P$ обозначим через X, считая при этом, что $X \equiv \langle p, \sigma(p).X\rangle$. Для реального входа $x \in X$ определим тип $\tau(x) = \tau(\sigma(p).x)$, буфер входа обозначим через $b(x)$. Множество реальных, присутствующих в структуре схемы, выходов компонента $p \in P$ обозначим через Y, считая при этом, что $Y \equiv \langle p, \sigma(p).Y\rangle$. Для реального выхода $y \in Y$ определим тип $\tau(y) = \tau(\sigma(p).y)$. Множества реальных входов и выходов всех компонентов асинхронной автоматной схемы $a \in A$ обозначим через X^* и Y^*, соответственно.

Соответствие lnk, устанавливающее информационные связи между выходами и входами компонентов схемы, имеет вид:

$$lnk: Y^* \to X^*,$$

причем для любых y и x таких, что, $x = lnk(y)$ выполняется равенство типов $\tau(y) = \tau(x)$.

Выход y одного компонента и вход x в общем случае другого компонента, удовлетворяющие условию $x = lnk(y)$, назовем ассоциированными. Сообщения передаются в схеме от выходов к ассоциированным с ними входам автоматных компонентов.

Для определения функционирования асинхронной автоматной схемы рассмотрим функционирование ее произвольного компонента $p \in P$, имеющего шаблон σ.

Обозначим через $M \equiv \langle p, \sigma(p).M\rangle$ множество состояний реальной памяти, а через $S \equiv \langle p, \sigma(p).S\rangle$ – множество реальных собственных состояний компонента. Функцию выбора ветви θ, а также функции fy, fm, fs распространим на множества M и S естественным образом.

Пусть $s \in S$ текущее состояние компонента, а $u \in U(s)$ – ситуация на его входах. Элементы вектора u формально зададим так:

$$u_j = \begin{cases} 0, \text{ если } q(b(x_j)) = \omega \\ 1, \text{ если } q(b(x_j)) \neq \omega, \end{cases}$$

где $1 \leq j \leq nof(X)$.

Условием перехода компонента p, находящегося в состоянии s, в следующее состояние s' является требование

$$u \in dom_s(\theta).$$

Если условие истинно, компонент выполняет ветвь $f = \theta(s, u)$. Применяя статически определенную маску входов φ, согласно выражениям (1) он вычисляет значения функций fy, fm, fs и формирует маску выходов ψ:

$$\psi_k = \begin{cases} 0, \text{ если } fy_k(z_1, \ldots, z_{nof(X)}, m, s) = \lambda \\ 1, \text{ если } fy_k(z_1, \ldots, z_{nof(X)}, m, s) = d, \end{cases}$$

где $d \in D(\tau(y_k))$, $1 \leq k \leq nof(Y)$, и

$$z_j = \begin{cases} \lambda, \text{ если } \varphi_j = 0 \\ head(q(b(x_j))), \text{ если } \varphi_j = 1 \end{cases}$$

для $1 \leq j \leq nof(X)$.

В процессе выполнения ветви f компонент p изменяет состояния буферов своих незамаскированных входов и буферов входов, ассоциированных с его незамаскированными выходами.

Правило, в соответствии с которым определяется новое состояние буфера входа x_j компонента, имеет вид:

$$q'(b(x_j)) = \begin{cases} q(b(x_j)), \text{ если } \varphi_j = 0 \\ tail(q(b(x_j))), \text{ если } \varphi_j = 1. \end{cases}$$

Правило, в соответствии с которым определяется новое состояние входа x, ассоциированного с выходом y_k компонента, имеет вид:

$$q'(b(x)) = \begin{cases} q(b(x)), \text{ если } \psi_k = 0 \\ ((fy_k(z_1, \ldots, z_{nof(X)}, m, s), q(b(x))), \text{ если } \psi_k = 1, \end{cases}$$

где x такой вход, что $x = lnk(y_k)$

Новые состояние памяти компонента и его новое собственное состояние определяется согласно следующим правилам:

$$m' = fm(z_1, \ldots, z_{nof(X)}, m, s),$$

$$s' = fs(z_1, \ldots, z_{nof(X)}, m, s).$$

Завершение выполнения ветви обработки данных означает завершение перехода компонента в следующее состояние.

Функционирование асинхронной автоматной схемы складывается из независимого (в смысле отсутствия централизованного управления) функционирования ее компонентов.

В заключение еще раз отметим, что схемы, содержащие непустое множество компонентов непрерывного действия, могут явиться эффективным инструментом для представления процессов асинхронной обработки данных в распределенных системах различного назначения.

ИСПОЛЬЗОВАНИЕ ГРИД-СРЕДЫ ДЛЯ ПРИЛОЖЕНИЙ КВАНТОВОЙ ХИМИИ И МОЛЕКУЛЯРНОЙ ДИНАМИКИ

Д. С. Кастерин, М. М. Степанова, К. С. Шефов

*Санкт-Петербургский государственный университет,
физический факультет, кафедра вычислительной физики
dmk.pre@gmail.com, mstep@mms.nw.ru, k.s.shefov@gmail.com*

Большая часть современного ПО для моделирования химических соединений и молекулярных систем ориентирована на работу в параллельном режиме на вычислительных кластерах. В рамках данной работы тестировалась возможность использования пакетов квантовой химии (Firefly/PC GAMESS) и молекулярной динамики (LAMMPS) в среде ГридННС.

Программно-аппаратная среда

Полигон

Установка программного обеспечения выполнена на сегменте ГридННС физического факультета СПбГУ, включающем шлюз gt3.phys.spbu.ru (2CPU, 1GB RAM) и кластер из 2-х вычислительных узлов (4CPU, 4GB RAM). Все узлы объединены сетью Gigabit Ethernet и работают под управлением CentOS 5.5, ядро версии 2.6.18. На кластере используется система очередей PBS Torque 2.3.6 со стандартным планировщиком. Узлам доступен общий NFS-раздел. Для подключения к грид на шлюзе установлено базовое ПО Globus Toolkit 4.2.1.1 и дополнительные пакеты ГридННС [1].

На кластере установлено следующее программное обеспечение:

- пакет MPICH2 1.2.1p1 (сборка из исходных кодов),
- утилита OSC Mpiexec 0.83 (сборка из исходных кодов),
- пакет Intel Compiler Suite Professional Edition for Linux 11.1.069 (установка из бинарных пакетов),
- библиотека FFTW 2.1.5 (сборка из исходных кодов),
- библиотеки BLAS и LAPACK 3.0.3 (установка из бинарных пакетов).

Из специализированных прикладных пакетов доступны:

- Firefly 7.1.G с поддержкой MPICH2 (установка из бинарных пакетов) [2],
- LAMMPS 18Apr10 с поддержкой дополнительных библиотек ATC, MEAM, POEMS, REAXFF (сборка из исходных кодов) [3].

Конфигурация кластера имеет несколько особенностей. Во-первых, для запуска MPI-заданий через PBS используется специальная утилита Mpiexec [4] вместо стандартных средств запуска параллельных задач (mpirun/mpiexec) и демона mpd из пакета MPICH2. Утилита позволяет обеспечить строгий контроль ресурсов системы и её устойчивость к сбоям.

Вторая особенность заключается в том, что программе Firefly требуются временные каталоги на локальной файловой системе. При их создании по умолчанию не обеспечивается уникальность имён каталогов и их удаление после завершения работы. В нашей конфигурации для имён каталогов используются идентификаторы задачи в системе PBS. Помимо этого, разработана схема "сборки мусора", которая может быть использована для удаления каталогов уже отработавших заданий.

В-третьих, используются разработанные прототипы скриптов, которые предоставляют пользователю простой интерфейс к установленному ПО, что позволит упростить запуск исполняемых файлов прикладных пакетов.

Обеспечение работы в ГридННС

В инфраструктуре ГридННС запуск заданий выполняется с помощью пользовательского интерфейса (CLI или Web). Пользователь формирует задание и передает его системе управления Pilot, которая направит задание на конкретный ресурс. Прием, обработка и передача на выполнение задания на ресурсе обеспечивается модулем GRAM.

Сайт сконфигурирован с поддержкой softenv-расширений, доступных в ПО ГридННС и Globus, для установки среды окружения, специфичной для конкретного задания. Для включения поддержки расширений на сайте нужно:

- разрешить механизм стандартных расширений в конфигурации GRAM,
- установить пакет SoftEnv 1.6.2, создать и внести в базу описания расширений для пакетов Firefly и LAMMPS.

В нашем случае также потребовалось исправить ошибку в коде GRAM, которая не позволяла использовать расширения при запуске параллельных (MPI) задач и внести изменения в код GRAM-адаптера для PBS, чтобы задействовать утилиту Mpiexec.

Непосредственно запуск задач из грид-среды включает следующие действия:

- Подготовка файлов с входными данными и размещение их на доступном GridFTP-хранилище,
- Составление описания задания на языке JSON (например, см. приложение),
- Запуск задания (pilot-job-submit),
- Получение результатов вычислений на локальную машину (globus-url-copy) по факту завершения задания (pilot-job-status).

Моделирование Al_nH_{3n} систем

Длительное хранение водорода является на сегодняшний день актуальной задачей химической физики. Задачей данной части работы является исследование условий и процессов поглощения водорода металлами магнием и алюминием с примесями титана и ванадия и его испускания их гидридами.

Исследование предполагается проводить посредством компьютерного моделирования, основанного на молекулярно-динамических (МД) расчетах с использованием потенциала ReaxFF, способного моделировать химические реакции. Этот потенциал на сегодняшний день при надлежащем выборе параметров дает наилучшие результаты при моделировании больших (от 1000 атомов) химически реактивных систем, которые не могут быть рассчитаны с помощью более точных методов квантовой химии (ab initio) ввиду своих размеров.

Общий вид потенциала [5,6]:

$$E_{\mathrm{Re}\mathit{ax}\mathrm{FF}}\left(\{r_{ij}\},\{r_{ijk}\},\{r_{ijkl}\},\{q_i\},\{BO_{ij}\}\right) = E_{\mathrm{bond}} + E_{\mathrm{lp}} + E_{\mathrm{over}} + E_{\mathrm{under}} +$$
$$+ E_{\mathrm{val}} + E_{\mathrm{pen}} + E_{\mathrm{coa}} + E_{\mathrm{tors}} + E_{\mathrm{conj}} + E_{\mathrm{hbond}} + E_{\mathrm{vdWaals}} + E_{\mathrm{Coulomb}}$$

Потенциал описывает взаимодействие между N атомами, которое включает как ковалентное (характеризуется порядком химической связи BO), так и нековалентное (между атомами, не образующими связь). Полная энергия является функцией относительных положений пар атомов $\boldsymbol{r}_{ij}$, троек $\boldsymbol{r}_{ijk}$ и четверок $\boldsymbol{r}_{ijkl}$, а также атомных эффективных зарядов q_i и порядков связей BO_{ij} между двумя атомами.

Для проведения расчетов требуется предварительно подобрать оптимальные значения параметров для этого потенциала с помощью однопараметрической подгонки [7]. Данные, по которым производится подгонка, берутся из расчетов простых соединений водорода с металлами, кристаллических решеток этих металлов и их гидридов методами квантовой химии (теория функционала плотности, ТФП).

За основу для подгонки из квантово-химических расчетов берутся геометрические характеристики молекул (расстояния между атомами, валентные углы), колебательные частоты, эффективные заряды атомов и энергии связи соединений, а также энергия гидрирования поверхности кристаллов. Квантовые расчеты проводились с использованием пакетов Firefly [2] и GAUSSIAN 03 [8], расчеты молекулярной динамики – пакетом LAMMPS[3].

Реализация потенциала ReaxFF в пакете LAMMPS уже включает в себя параметры для алюминия и водорода. На первом этапе работы выполнялось сравнение МД-расчетов с этими параметрами с результатами квантовых ТФП-расчетов гидридов алюминия. Вычисления проводились для небольших кластеров Al_nH_{3n} и Al_n при значениях n от 1 до 6 (Рис. 1).

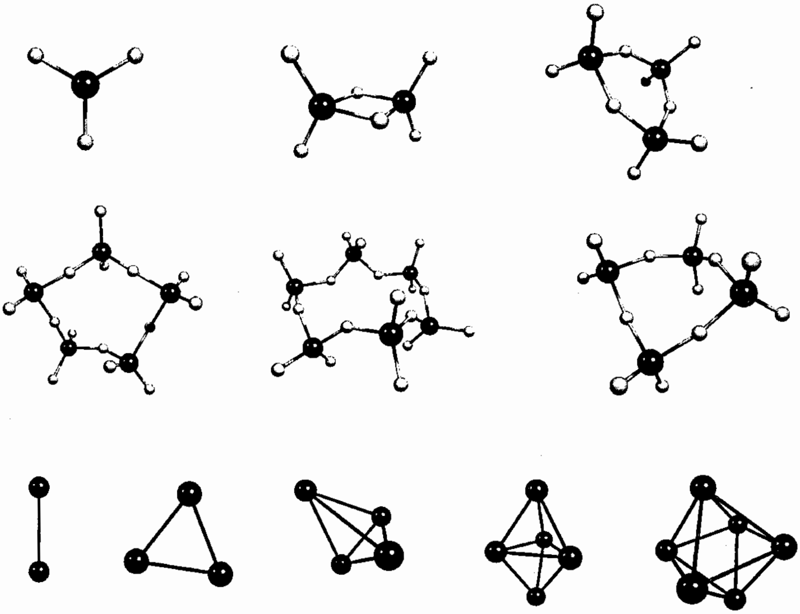


Рис.1: Кластеры Al_nH_{3n} и Al_n, для которых производился расчет. Атомы алюминия показаны серым цветом, водорода – белым

Для квантовых расчетов (Firefly) был использован метод B3LYP с базисным набором 6-31+G(d). По данным вычислений ТФП и молекулярной динамики построены кривые диссоциации AlH_3(Рис. 2), энергия рассчитана относительно энергии равновесной длины связи.

Минимум энергии достигается на расстоянии между алюминием и водородом в 1,6Å,(Рис.2а), при угле 120^0 (Рис. 2б) как в ТФП, так и в МД. Результаты расчетов энергии полной диссоциации AlH_3 также хорошо согласуются друг с другом (Рис. 2в), различия наблюдаются лишь на удалении от минимума (Рис. 2а, б).

Для длин связей и валентных углов кластеров Al_nH_{3n} при значениях n от 2 до 6 разница при расчетах методами МД и ТФП составляет от 10% до 20%.

Выполнено сравнение энергий связи (табл. 1), отнесенных к числу атомов водорода, для шести кластеров (Рис. 1), которые рассчитывались по формуле:

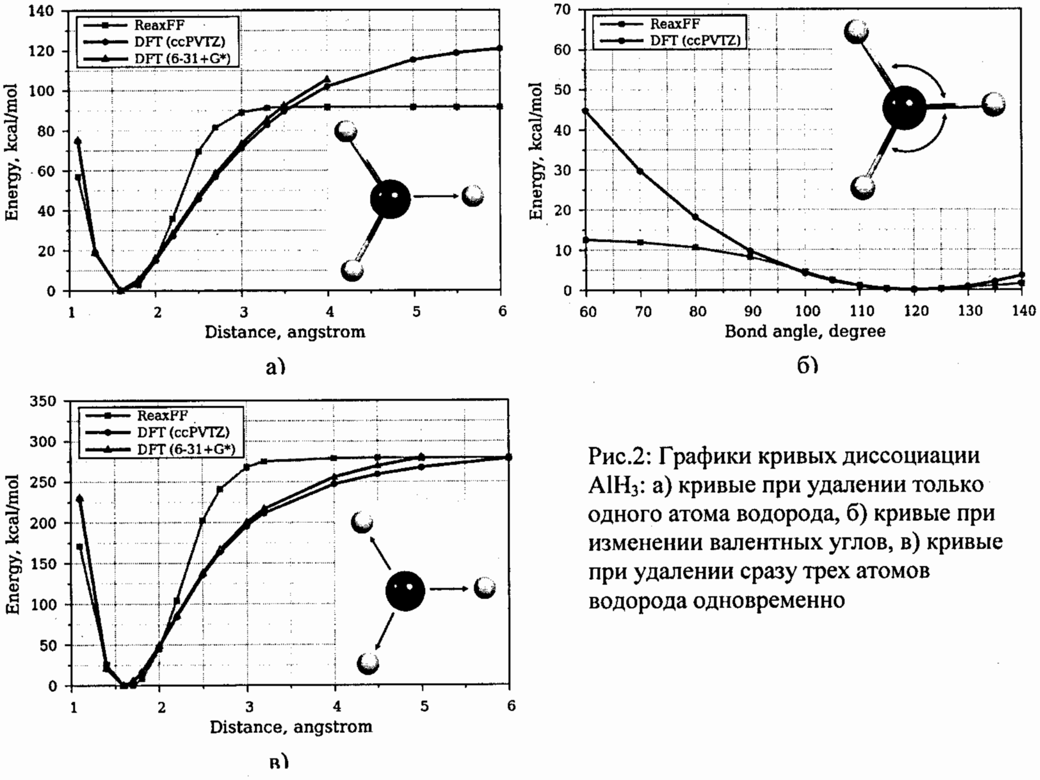
$$BE_1 = -[E(Al_nH_m) - E(Al_n) - mE(H)]/m,$$

где E(X) – полная энергия частицы X в основном состоянии. Также сравнивались энергии связи по отношению к числу молекул водорода, приходящихся на кластер (так как ReaxFF считает энергию по отношению к энергии изолированных атомов):

$$BE_2 = -[E(Al_nH_{2m}) - E(Al_n) - mE(H_2)]/m.$$

Таблица 1. Сравнение энергий связи кластеров Al_nH_{3n}

Кластер	BE_1 (kcal/mol H)		BE_2(kcal/mol H)	
	ReaxFF	ТФП	ReaxFF	ТФП
AlH_3	94.17	70.36	73.97	30.95
Al_2H_6	104.42	72.10	94.47	34.43
Al_3H_9	102.23	68.71	90.09	27.65
Al_4H_{12}	98.09	67.50	81.82	25.23
Al_5H_{15}	88.52	65.40	62.67	21.02
Al_6H_{18}	93.12	64.33	71.86	18.88



а)

б)

в)

Рис.2: Графики кривых диссоциации AlH_3: а) кривые при удалении только одного атома водорода, б) кривые при изменении валентных углов, в) кривые при удалении сразу трех атомов водорода одновременно

Как видно из таблицы 1, результаты расчетов ТФП и МД значительно отличаются. Квантовый расчет показывает, что при увеличении размеров Al_nH_{3n}-кластера энергия связи, отнесенная к числу как атомов, так и молекул водорода, медленно падает. Полученные результаты для энергии связи, отнесенной к числу атомов водорода, согласуются с работой [9] (разница не превышает 8%). Однако результаты МД расчета с использованием выбранной параметризации ReaxFF не согласуются с ТФП, вместо монотонного спада энергии с ростом размера кластеров наблюдается ее скачок вверх в конце. Это означает, что необходимо будет проводить подгонку параметров и в случае алюминия. Хотя стоит заметить, что энергия связи молекулы водорода, рассчитанная с ReaxFF, отличается от ТФП-результата всего на 4% (109,5 и 114,4 ккал/моль соответственно).

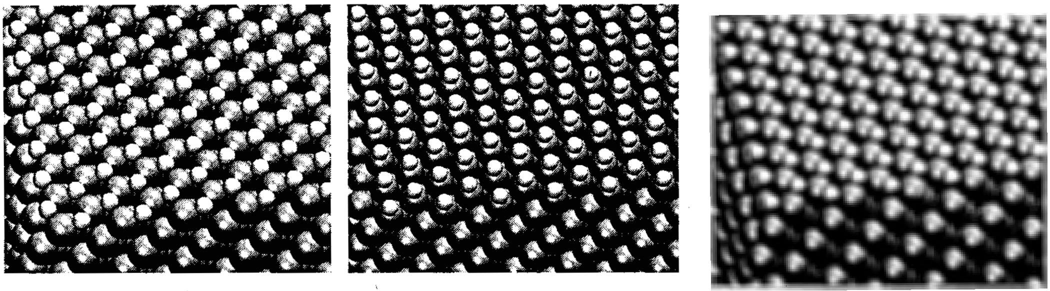


Рис. 3: Варианты расположения атомов водорода над атомами алюминия, *слева направо*: в узлах ГЦК-решетки одного слоя вместо атомов алюминия; точно над атомами металла в один слой; в узлах одного слоя гексагональной плотной упаковки типа ABAB

Также было выполнено сравнение энергий гидрирования поверхности ГЦК-решетки алюминия (111):

$$E_{hyd} = [E_{(fcc+H)} - E_{fcc} - mE_H]/m,$$

где $E_{(fcc+H)}$ – энергия пластины алюминия с атомами водорода на поверхности, E_{fcc} – энергия алюминиевой пластины без водорода, E_H - полная энергия атома водорода, m – число атомов водорода в системе. Для этого проведен квантовый расчет элементарной ячейки Al в GAUSSIAN 03 (базис 8-511G для Al и 5-111G для H [10]) с 2D-периодическими условиями. Рассчитывалось пять слоев атомов Al с индексами Миллера (111) толщиной 9,3Å, атомы трех нижних слоев фиксировались в узлах решетки, а положения атомов двух верхних слоев с водородом на поверхности оптимизировались для достижения минимума энергии. Аналогичный расчет с использованием ReaxFF производился в LAMMPS моделированием 19 слоев Al (111) (Рис. 3). Результаты расчетов ТФП и МД (табл. 2) различаются на 20%, что снова говорит о необходимости подгонки параметров. Исходя из полученных данных, наименьшей энергией обладает случай расположения водорода в узлах ГЦК-решетки.

Таблица 2. Сравнение энергий гидрирования

Метод	E_{hyd}, kcal/mol		
	FCC	TOP	HCP
ТФП	-110.7	-104.4	-107.8
ReaxFF	-91.6	-79.1	-91.4

Также были проведены квантовые ТФП-расчеты (базис 6-31++G**) для кластеров гидрида магния MgH, MgH_2, Mg_2H_2, Mg_2H_4, Mg_4H_8. Выполнена оптимизация геометрии, рассчитаны колебательные частоты, энергии связи, эффективные заряды атомов. Полученные результаты квантовых расчетов для гидридов алюминия и магния будут использованы в качестве подгоночного набора для поиска параметров потенциала ReaxFF, что необходимо для моделирования больших систем.

Заключение

Расчеты, выполненные в данной работе, требовали настройки ряда параллельных пакетов и устойчивой работы грид-системы. Полученные за короткий срок результаты позволяют сделать вывод, что среда ГридННС достаточно эффективна для ресурсоемких вычислений с использованием специализированного ПО.

Приложение. Пример описания задания на языке JSON

```
{ "version": 2,
 "description": "LAMMPS Job",
 "default_storage_base": "gsiftp://gt3.phys.spbu.ru/tmp/user/",
 "tasks": [
   { "id": "friction",
    "description": "LAMMPS MPI task",
    "definition":
      { "version": 2,
       "extensions" : { "softenv": "+lammps" },
       "executable": "lammps",
       "arguments": [ "-i", "in.alumane", "-l", "log.alumane" ],
       "stdout": "stdout.txt",
       "stderr": "stderr.txt",
       "input_files": { "in.friction": "in.friction" },
       "output_files":
         { "dump.friction": "dump.friction",
          "log.friction": "log.friction",
          "screen.friction": "screen.friction" },
       "count": 2
      }
   }    ],
 "requirements": { "hostname": [ "gt3.phys.spbu.ru" ],
            "lrms": "PBS",
            "queue": "unic" } }
```

Литература

[1] ГридННС, http://ngrid.ru/ngrid/

[2] Пакет квантовохимических расчётов Firefly, http://classic.chem.msu.su/gran/gamess/

[3] Пакет молекулярной динамики LAMMPS, http://lammps.sandia.gov/

[4] Утилита Mpiexec, http://www.osc.edu/~djohnson/mpiexec/

[5] A.C.T. van Duin, Dasgupta S., Lorant F., Goddard W.A. Journal of Physical Chemistry. A. 105 (2001) 9396.

[6] Nomura K., Kalia R. K., Nakano A., Vashishta P. Computer Physics Communications. 178 (2008) 73.

[7] A.C.T. van Duin, J.M.A. Baas, B. van de Graaf. J. Chem. Soc. Faraday Trans. 90(19) (1994) 2881.

[8] Пакет квантовохимических расчётов Gaussian, http://www.gaussian.com/

[9] Kawamura H., Kumar V., Sun Q., Kawazoe Y. Phys. Rev. A 67 (2003) 063205.

[10]Dovesi R, Saunders V.R., Roetti C., Orlando R., Zicovich-Wilson C.M., Pascale F., Civalleri B. , Doll K., Harrison N.M., Bush I.J. et al. CRYSTAL06 User's Manual, University of Torino, Torino, 2008.

СИСТЕМА ВЫПОЛНЕНИЯ МАССОВЫХ ЗАПРОСОВ НА МНОЖЕСТВЕ РАСПРЕДЕЛЁННЫХ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ[1]

В. Н. Коваленко, Е. И. Коваленко, А. Ю. Куликов

Институт прикладной математики им. М.В.Келдыша РАН
125047, Москва, Россия

В работе излагаются основные положения подхода к разработке программных средств для выполнения массовых распределённых поисковых запросов. Под распределённым понимается запрос, который выполняет поиск и доставку данных из множества пространственно распределённых автономных баз данных. В отличие от известных подходов, рассматривается выполнение запросов над большим количеством баз данных (десятки), поэтому запрос охарактеризован как массовый.

1. Введение

Произошедшая за последние двадцать лет трансформация социального устройства России настоятельно требует автоматизации многих видов производственной деятельности. Создание соответствующих программных систем существенно упрощается благодаря наличию стандартизированных способов представления структурированных данных – в реляционной, объектно-ориентированной, XML моделях – и систем управления базами данных (СУБД), в которых реализованы универсальные примитивы управления данными.

В настоящее время становятся актуальными сценарии профессиональной деятельности, для которых требуется оперативное получение достоверной информации из многочисленных источников, которые функционируют автономно и слабо связаны друг с другом. Такие виды деятельности обнаруживаются в сферах государственного и корпоративного управления, планирования и контроля, финансов, торговле, медицине и многих других. На практике для разработки систем автоматизации такого типа обычно используются оболочки интеграции приложений, интеграционные порталы, метапоисковые машины, но эти средства не дают решения ключевой проблемы – доступа к распредёленным данным – на уровне, сопоставимом с традиционными СУБД [1].

Между тем, проблема управления данными из распределённых источников – баз данных (БД) стала предметом исследований с начала 1980-х годов. Задача состоит в создании независимой от приложений платформы, которая интегрирует размещённые в разных БД данные таким образом, чтобы в рамках одного декларативного запроса, аналогичного запросу традиционных СУБД, могли выполняться операции над данными из различных БД. В этом направлении получены важные результаты по архитектуре интеграции, разработаны методы обработки декларативных запросов [2] и интеграции гетерогенных БД [3], которые различаются системами управления, протоколами доступа, способами представления данных.

2. Массовая интеграция

Известные средства интеграционного подхода (исследовательские K2/Kleisli [4], Garlic [5], TSIMMIS [6], DISCO [7] и коммерческие системы Microsoft SQL Server 2008 R2, Oracle Data Integrator, IBM InfoSphere Federation Server) позволяют создавать прикладные системы со стабильным составом информационных источников, так что область применения

[1] Работа выполнена при поддержке гранта РФФИ 08-07-00381, программы фундаментальных исследований Президиума РАН, гранта Президента РФ для ведущих научных школ НШ-2139.2008.9.

средств интеграции остаётся ограниченной распределёнными организациями и устоявшимися коалициями смежных организаций. Предлагаемое в настоящей работе развитие направлено на поддержку динамического формирования широкомасштабных информационных инфраструктур из большого числа (несколько десятков) автономных БД.

Из различных вариантов интеграции рассматриваемая постановка задачи в наибольшей степени соответствует архитектуре *федеративного объединения* [8], которое не предполагает перемещение данных в централизованное хранилище, сохраняет автономию составляющих БД, но позволяет представить всю их совокупность в виде одной виртуальной.

Отличие нашей постановки задачи федеративного объединения выражается в следующем:

- количество интегрируемых БД может быть большим (десятки и сотни единиц);
- состав БД, доступных приложениям и образующих информационное пространство федеративного объединения, может динамически меняться в ходе функционирования;
- запросы могут содержать большое количество операторов, выполняющихся над многими БД, – такие запросы будем называть *массовыми*;
- средства интеграции должны поддерживать разработку приложений, которые способны обрабатывать данные как из отдельных БД, так и из их совокупности.

3. Применения массовой интеграции

В качестве основной цели образования информационных инфраструктур с большим количеством БД рассматривается поиск и получение информации из множества источников, которые содержат однотипные данные. Такие условия характерны, прежде всего, для производственной сферы: любое учреждение ведёт финансовую деятельность и деятельность по учёту персонала – соответствующие данные семантически эквивалентны, то есть имеют одинаковый смысл, хотя и могут быть представлены в отдельных БД по-разному. Каждое учреждение имеет свою специфику, но учреждения с близкой специализацией собирают данные, которые также семантически эквивалентны.

Прикладная обработка семантически эквивалентных данных не зависит от того, в каком месте они расположены, хранятся ли они в одной или в нескольких БД. Однако для реализации приложений в инвариантном относительно размещения данных виде нужен специальный аппарат управления данными.

В условиях больших информационных инфраструктур представляется необходимым наделять приложения следующими свойствами.

1. *Способность работать с любой БД информационной инфраструктуры*. Хотя концепция интеграции БД исходит из того, что приложения не должны зависеть от расположения данных, соотнесение данных учреждениям, в которых эти данные порождаются, является естественным. Поэтому одно и то же приложение должно обладать способностью обрабатывать данные из любой отдельной БД по выбору пользователя. Используя такую возможность, пользователь может с любой периодичностью и без дополнительных действий со стороны персонала учреждений получать необходимые данные, например, агрегированные данные для составления отчётов или детальные данные справочного характера.

2. *Способность работать со многими БД*. Укажем два вида задач, в которых требуется поиск информации по множеству БД. Первый вид – поиск информации по косвенным данным, когда её местонахождение неизвестно. Например, таким способом могут быть найдены дополнительные данные о человеке, если известно только его ФИО. Результативность поиска в таких задачах зависит от полноты охвата БД учреждений, которые содержат искомый тип данных.

Второй вид задач, в которых необходим аппарат для работы с множествами баз данных, – это получение интегральной информации по совокупности БД, такой, например, как номенклатура выпускаемой продукции определённого назначения, общая численность работников отрасли, средняя зарплата работников. Важно, чтобы приложения могли работать с произвольной совокупностью БД из информационной инфраструктуры: критериями выделения могут служить тип учреждений, регион, в котором они расположены, ведомственная принадлежность.

3. *Совместная обработка неявно связанных БД.* Хотя автономные БД не имеют прямых связей (подобных тем, которые реализуются через суррогатные ключи в рамках одной БД) друг с другом, тем не менее, связи могут существовать в виде одинаковых значений данных, в качестве которых выступают общепринятые обозначения, отраслевые стандарты. Таким образом, и в случае автономных БД может применяться бинарный оператор связывания JOIN. Это позволяет получать всестороннюю информацию об одном объекте, даже если она содержится в разных БД. Так, приложение может получить из БД нескольких организаций паспортные данные, сведения о состоянии здоровья, трудовой деятельности, имея в качестве входных данных ФИО некоторого лица.

Для поддержки описанных выше свойств приложений далее предлагается развитие аппарата управления данными, включающее: расширение формы поисковых запросов, ориентированное на операции с множеством БД, и средства определения информационного пространства. Такое развитие даёт возможность разрабатывать параметризованные приложения, пользователи которых могут определять состав обрабатываемых данных непосредственно в ходе работы.

В реализации системной поддержки новой функциональности мы исходим из хорошо известных архитектурных решений и методов обработки распределённых запросов [2], опробованных на практике. В то же время специфика массовых запросов, в которых становится значительным объём обрабатываемых данных и сетевых передач, даёт основание для применения современных перспективных подходов, основанных на концепции грида [9]. В качестве основы используется комплекс OGSA-DAI/DQP [10], в котором реализованы предложения по базовым стандартам информационного грида (OGSA-DAI) и поддерживается обработка распределённых декларативных запросов (OGSA-DQP). OGSA-DQP ограничен реляционной моделью БД и реализует часть языка SQL-92, и далее будет рассматриваться вариант интеграции для этих условий.

4. Расширение языка управления запросами

Информационные приложения обычно работают с одной или с фиксированным числом БД, причём идентификаторы БД и таблиц задаются либо явным образом, либо посредством определяемых статически пользовательских представлений (View). В запросах, поддерживаемых OGSA-DQP, участвуют распределённые БД, поэтому они обозначаются глобальным идентификатором – именем ресурса, в котором определяется сетевой адрес (URL).

Представим структуру оператора Select в следующем виде:

```
SELECT [distinct_condition] select_expressions
 [ FROM table_expression
 [ WHERE select_conditions] [ GROUP BY grouping_conditions]
 [ HAVING select_conditions] [ ORDER BY order_conditions]
 ]
```

Интерес представляет конструкция table_expression, выделенная ключевым словом FROM: именно в ней указываются имена обрабатываемых таблиц. В DQP они задаются парой: {RDB, LT}, где RDB – это идентификатор реляционного ресурса, соответствующего некоторой БД, LT – локальное имя таблицы в БД RDB.

В условиях, когда работа происходит с большим количеством БД, едва ли следует предполагать, что пользователь знает, где расположены интересующие его данные. Более гибкий подход, позволяющий создавать приложения, которые без изменения кода способны обрабатывать данные из любой совокупности БД, основан на том, чтобы, во-первых, интегрировать данные в общее информационное пространство и, во-вторых, дать возможность выделения рабочего пространства по содержательным признакам.

4.1. Создание общего информационного пространства – интеграция данных

Вопрос интеграции данных на основе их семантической эквивалентности хорошо изучен в многочисленных исследованиях. Согласно [3] интеграция данных направлена на обеспечение доступа к данным из разных источников путём определения унифицированного представления этих данных, которое называется *глобальной схемой*. Интеграция определяется заданием соответствия между глобальной схемой и схемами интегрируемых БД.

Среди различных методов определения соответствия наиболее известны два: Global as View (GAV) и Local as View (LAV). Более прост для реализации метод GAV, в котором таблицы глобальной схемы ассоциируются с запросами к схемам источников, то есть каждая таблица глобальной схемы представляется как одно или более пользовательских представлений схем источников. В реляционной модели на языке SQL соответствие задаётся операторами:

CREATE VIEW view [(column_name_list)] AS SELECT query

Здесь view – имя глобальной таблицы, column_name_list – поля глобальной таблицы view, query – запрос SELECT к локальной БД, порождающий таблицу с атрибутами column_name_list.

Таким образом, образование общего пространства данных производится путём определения глобальной схемы и представлений, отображающих её в локальные БД. Естественно полагать, что данные локальных БД, которые отображаются в одинаковые элементы (таблицы, атрибуты) глобальной схемы, являются семантически эквивалентными.

4.2. Определение состава обрабатываемых данных

Интегрируя множество распределённых БД, глобальная схема открывает доступ из приложений ко всей совокупности содержащихся в них данных. Для этого базовый способ OGSA-DQP адресации таблиц дополняется возможностью задания имён таблиц глобальной схемы вместо адресов конкретных БД и имён их таблиц.

Для работы с подмножествами интегрированных данных нужны новые конструкции языка. Предлагаемое расширение состоит в том, что во всех вхождениях оператора Select таблицы могут идентифицироваться парой: {GDB, GT}, где GDB – группа БД, из которых выбираются данные, соответствующие глобальной таблице GT. Все вхождения имён вида GDB_GT в конструкции FROM интерпретируются как:

RDB1_GT ∪ RDB2_GT ... ∪ RDBn_GT,   где RDBi∈GDB (i=1,2,...n) – БД, входящие в группу GDB.

Например, оператор: SELECT select_expressions FROM GDB_GT заменяется на:

SELECT select_expressions FROM (RDB1_GT ∪ RDB2_GT ... ∪ RDBn_GT)

Каждая ссылка на таблицу вида RDBi_GT представляет собой заданный с помощью представления View запрос, который выполняется в локальной БД RDBi, а результаты всех таких запросов объединяются. Заметим, что возможны два варианта объединения: Union и Union ALL.

5. Определение групп баз данных

Понятие группы БД позволяет параметризовать приложения так, что задавать состав групп может пользователь непосредственно в процессе работы. Определяя группу, он может исходить только из содержательной модели информационного пространства и не располагает сведениями об адресах и составе данных отдельных БД. При этих условиях отбор БД для образования группы может осуществляться по содержательным критериям: названиям организаций-владельцев, адресу организации, описанию тематики данных, то есть по метаданным, описывающим тот или иной источник.

Хотя способ отбора БД зависит от приложения и способа его использования, в системной поддержке нуждается представление метаданных. В качестве одного из возможных вариантов мы рассматриваем реляционное представление. Реляционная база метаданных содержит одну таблицу, атрибуты которой содержат характеристики организации, а уникальное название организации-владельца является первичным ключом. В этом случае отбор БД для включения в группу реализуется поисковыми запросами типа Select с условиями, определяющими требуемые характеристики.

Заметим, что один и тот же аппарат для работы с данными и метаданными позволяет использовать последние в обычных запросах – таким способом могут быть выражены дополнительные ограничения на область поиска.

6. Программирование запросов с использованием групп баз данных

Для параметризации приложений необходимы две функции: функция отбора БД для формирования групп и функция, связывающая отобранный список БД с конкретной группой. В целом, схема программирования запросов в приложении выглядит следующим образом.

- Обращение к функции отбора БД. В обращении указывается список групп, состав которых определяется или модифицируется. Если, например, отбор БД реализован в интерактивной форме, пользовательский интерфейс должен представлять содержательное описание определяемых групп. В результате функция отбора возвращает списки БД для каждой определяемой группы.
- Результат функции отбора передаётся функции определения групп.
- При выполнении запросов используется текущее значение состава групп.

Заключение

На первом этапе реализации описанного подхода создан прототип, который позволил оценить возможности применения комплекса OGSA-DAI в качестве средства выполнения массовых запросов. Соответствующие исследования проводились в программно-аппаратной инфраструктуре, развёрнутой в локальной сети ИПМ им. М.В. Келдыша РАН. Целью исследования являлась оценка времени выполнения массового запроса в зависимости от его сложности – числа БД по которым ведётся поиск.

При небольшом числе компьютеров (10), задействованных в инфраструктуре, моделировались запросы, выполняющие поиск данных на 100-1000 БД. Экспериментально показано, что такой способ моделирования даёт хорошее приближение для оценки времени выполнения запросов в реальной ситуации, когда каждая БД размещена на отдельном компьютере.

Основной результат состоит в том, что время выполнения возрастает почти линейно при увеличении сложности запроса и составляет 800 - 1000 Мс для сложности запроса 100 БД, 8350 до 9400 Мс для 1000 БД (при выборке 1 строки из каждой БД). Такие показатели представляются приемлемыми для практики.

Литература

[1] Haas L. M., Lin E. T., Roth M. A. Data integration through database federation. IBM Systems Journal, Volume 41 , Issue 4 (October 2002), p. 578 – 596.

[2] Kossmann D. The State of the Art in Distributed Query Processing, ACM Computing Surveys, 32(4):422-469, December 2000. http://www.db.fmi.uni-passau.de/~kossmann

[3] Lenzerini M. Data Integration: A Theoretical Perspective, PODS 2002. pp. 233–246. http://www.dis.uniroma1.it/~lenzerin/homepagine/talks/TutorialPODS02.pdf

[4] Davidson S. B., Crabtree J., Brunk B. P., Schug J., Tannen V., Overton G. C., Stoeckert C. J. K2/Kleisli and GUS: Experiments in Integrated Access to Genomic Data Sources. IBM Systems Journal, 40(2):512–531, 2001.

[5] Carey M. et al. Towards Heterogeneous Multimedia Information Systems: The Garlic Approach, technical report, IBM Almaden Research, 1995.

[6] Hammer J., Garcia-Molina H., Ireland K., Papakonstantinou Y., Ullman J., and Widom J. Information Translation, Mediation, and Mosaic-Based Browsing in the TSIMMIS System, Proc. ACM SIGMOD Int'l Conf. Management of Data, 1995.

[7] Tomasic A., Raschid L., Valduriez P. Scaling Access to Heterogeneous Data Sources with DISCO. IEEE Trans. Knowl. Data Eng. 10(5): 808-823 (1998).

[8] Sheth A., Larson J. A. Federated Database Systems for Managing Distributed, Heterogeneous and Autonomous Databases. ACM Computing Survey, 1990, 22(3): 183- 236.

[9] Arenas A., Bilas A., Luna J., Marazakis M., Comito C., Talia D., Dikaiakos M.D., Gounaris A., Massonet P., Naqvi S., Smith J., Watson P., Stagni F. Knowledge and Data Management in Grids: Notes on the State of the Art. CoreGRID White Paper Number WHP-0002. May 2, 2008. http://www.coregrid.net/mambo/images/stories/WhitePapers/whp-0002.pdf

[10] Lynden S., Mukherjee A., Hume A.C., Fernandes A.A.A., Paton N. W., Sakellariou R., Watson P. The design and implementation of OGSA-DQP: A service-based distributed query processor. Future Generation Computer Systems, Volume 25, Issue 3, March 2009, Pages 224-236. http://www.ogsadai.org.uk/

ПРОБЛЕМЫ ПОСТРОЕНИЯ СЕМАНТИЧЕСКОГО ОПИСАНИЯ СИСТЕМОЗАВИСИМЫХ ЭЛЕКТРОННЫХ УЧЕБНЫХ МАТЕРИАЛОВ С ПРИМЕНЕНИЕМ РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ АЛГОРИТМОВ

М. А. Михеев

*ГОУ ВПО МО Международный университет природы, общества и человека «Дубна»
Россия, 141980, г. Дубна Московской обл., ул. Университетская 19*

На сегодняшний день системы дистанционного обучения (СДО) проникают во все сферы образования. Удобный способ получить образование привлекает людей. В результате на рынке появляются различные реализации СДО поддерживаемые как высшими учебными заведениями, так и коммерческими учреждениями. Соответственно разрабатывается множество инструментальных решений, предназначенных для поддержания учебного процесса и взаимодействия обучающихся и преподавателей.

Часто цели, поставленные при организации СДО, требуют взаимодействия с другими СДО для обмена электронными учебными материалами (ЭУМ). При условии, что инструментальные средства разных СДО произведены либо самостоятельно ВУЗом, либо сторонним разработчиком, возникает проблема интероперабельности при обмене данными — ЭУМ одной системы не могут быть применены в другой, т.е. требуется их преобразование к внутреннему стандарту конкретной системы.

Существуют методы решения проблемы интероперабельности СДО или систем управления обучением (Learning Management System) через общие стандарты описания ЭУМ, например SCORM (Sharable Content Object Reference Model) [2] или через семантическое описание [1] на базе онтологий и языка OWL (Web Ontology Language) [3]. При этом не исключается вариант создания конверторов импорта/экспорта в требуемый формат.

Указанные технологии и методики решения проблемы взаимодействия СДО ориентированы на создание *новых* ЭУМ. В тоже время в каждой отдельно организованной СДО, не поддерживающей международных стандартов, существуют крупные базы ЭУМ со своей внутренней структурой и форматами описания, обычно это гипертекст (HTML) или другая внутренняя нотация. Для преобразования данных ЭУМ к выбранному стандарту или к семантически описанному виду требуется создание конверторов, разработка которых трудоемка.

Для создания семантического описания ЭУМ нужна онтологическая основа, создавать которую без участия экспертного сообщества не представляется возможным, к тому же это требует больших временных затрат. К указанным аспектам так же можно причислить обработку большого объема разнородной информации, из которой могут состоять ЭУМ. При этом для работы с конкретной базой ЭУМ, реализованной по внутрисистемным стандартам СДО, требуется разработка нового инструментария.

Решение всей совокупности проблем семантического описания можно разделить на несколько этапов. Первый этап — это анализ структуры ЭУМ в конкретной СДО, создание программного слоя для взаимодействия с содержимым ЭУМ. Второй этап — определение терминологического базиса или «локальных онтологий» на основе словарей и глоссариев СДО. Третий этап — логическое разделение ЭУМ по предметным областям и оценка степени их пересечения.

На первом этапе сконцентрирована вся работа по созданию или корректировке инструментария для работы с ЭУМ конкретной СДО. Чаще всего содержание ЭУМ храниться в формате HTML, целостность которого требуется привести к общепринятому виду, что требует также создания дополнительного программного обеспечения.

Для уменьшения временных затрат по созданию онтологий при построении семантического описания, предлагается создавать «локальные онтологии» (второй этап), основанные на глоссариях и словарях присутствующих в СДО. Отсутствие подобных объектов в развитых СДО не подразумевается. Определяя базовые термины как онтологии внутри СДО, можно определить принадлежность ЭУМ к предметным областям и соответственно перейти к третьему этапу. Дальнейший анализ и детализация содержимого ЭУМ должны быть направлены на выявление недостающих онтологий [7].

Построение семантического описания не ограничивается определением «локальных онтологий» предметных областей множества ЭУМ. Основной фокус работы направлен на работу с содержимым ЭУМ. Удобным для целевого применения в данном случае будет модель описания метаданных на основе формата RDF (Resource Description Framework) [5], в частности его формат RDFa (RDF in attributes) [6].

Согласно спецификации RDF метаданные представляются в виде так называемых триплетов: субъект — предикат — объект. Триплеты лежат в основе графовой структуры, с помощью которой описываются все значимые элементы содержимого ЭУМ. Т.е. субъект — это вершина, из которой выходит ребро. Предикат — само ребро, а объект — вершина, конец ребра (рис. 1).

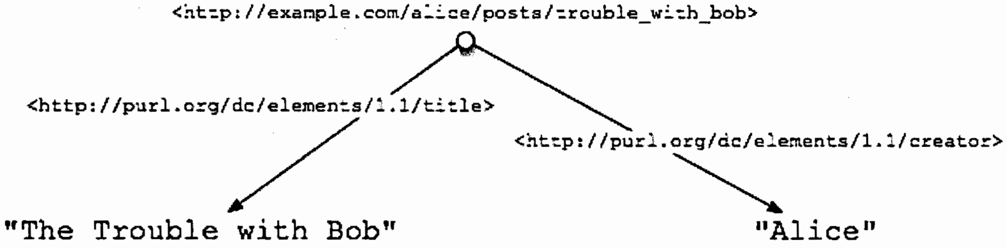

<http://example.com/alice/posts/trouble_with_bob>
<http://purl.org/dc/elements/1.1/title>
<http://purl.org/dc/elements/1.1/creator>
"The Trouble with Bob"
"Alice"

Рис. 1: Пример RDFa графового представления триплета

Для выявления связей «субъект — предикат — объект» требуется применения алгоритмов обработки текстов, которые можно разделить на группы согласно способу обработки текстовой информации (рис.2).

По сложности почти все алгоритмы работы с текстом относятся к NP-сложным, т.к. время их работы напрямую зависит от объемов обрабатываемых данных. Этот факт определяет требования к оборудованию, которое будет использоваться для обработки базы ЭУМ.

Далее алгоритмы анализа текста можно разделить по базовому принципу. Алгоритмы, направленные на морфологический и синтаксический анализ, позволяют определить содержательные элементы текста, относящиеся к определениям онтологий, т.е. можно построить первый уровень связи термина и текста его раскрывающего. Графометрические алгоритмы определяют структуру текста и взаимосвязи его частей, что полезно при анализе ЭУМ, содержащих данные из разных предметных областей, а также позволяют находить триплеты. Сами семантические описания призваны строить алгоритмы ассоциативного анализа [4].

Алгоритмы анализа естественных языков зависят от объема обрабатываемых данных. В контексте обозначенных проблем подразумеваются большие объемы данных в базе СДО. Следовательно, работа алгоритмов потребует больших вычислительных ресурсов. Это ведет к необходимости распараллеливания как задач в целом, так и отдельных частей алгоритмов, таких как: лингвистический анализ, разбор на основе грамматик и т.д.

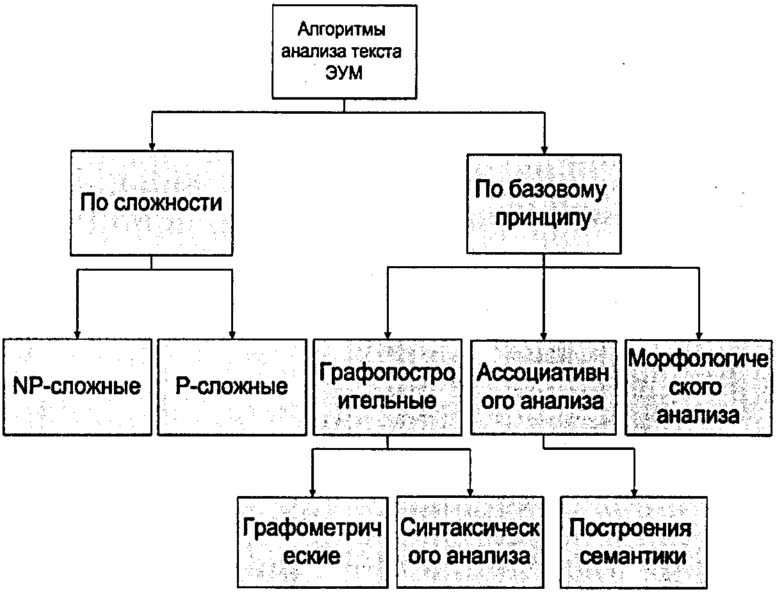


Рис. 2: Группы алгоритмов анализа текстов

При работе с текстом выгодно использовать распараллеливание частей алгоритмов, для эффективной обработки блока данных, а сам текст распределять по узлам, таким образом, предоставляется возможность обработать большую базу данных ЭУМ за приемлемое время.

Реализация семантического описания ЭУМ на базе «локальных онтологий» имеет как положительные стороны, так и отрицательные.

К положительным результатам построения такого описания можно отнести:

- *Решение проблемы интероперабельности как между системами, так и с внешней средой.* Помимо взаимодействия с другими СДО, у которых ЭУМ будут семантически описаны, появится возможность получать информацию также от ресурсов расположенных в Интернет и поддерживающих семантическое описание.
- *Упрощение построения баз знаний интеллектуальных обучающих систем.* ЭУМ должен быть структурирован таким образом, чтобы его изучение было максимально удобным и эффективным. Чтобы ЭУМ отвечал заданным свойствам, в рамках интеллектуальных обучающих систем создаются ассоциативный ЭУМ, который должен иметь семантическую структуру, т.е. содержать в себе гипертекстовую семантическую сеть [8].
- *Возможность использования базы ЭУМ в качестве базы знаний.* В результате структуризации базы ЭУМ СДО, посредством семантического описания, появится возможность получать новые учебные материалы и знания, конструируя их по запросу.

К отрицательным факторам описанного в статье решения можно отнести:

[1] *Оторванность «локальных онтологий» от общепринятых онтологий.* Очевидно, что «локальные онтологии», представляющие предметные области внутри СДО, будут иметь определенную степень различия с онтологиями, созданными с участием экспертного сообщества.

[2] *Исправление ошибок описания при сопоставлении «локальных онтологий» с общепринятыми онтологиями.* Потребуется провести работы по доведению «локальных онтологий» до общепринятого вида, либо следует заменить их общепринятыми онтологиями.

Данная проблема может возникнуть только при взаимодействии с семантически описанными ресурсами, которые поддерживает сообщество экспертов в соответствующих предметных областях.

Литература

[3] Жыжырий Е.А., Щербак С.С. Применение web-онтологий в задачах дистанционного обучения. [Электронный ресурс]. URL: http://shcherbak.net/dist/ (дата обращения 12.04.10).

[4] Advanced Distributed Learning. Sharable Content Object Reference Model (SCORM) 2004. / Перевод с англ. Е.В. Кузьминой. – М.: ФГУ ГНИИ ИТТ «Информика». – 2005, 29 с.

[5] OWL Web Ontology Language Semantics and Abstract Syntax , Peter F. Patel-Schneider, Patrick Hayes, and Ian Horrocks, Editors. Рекомендация W3C, 10 февраля 2004. [Электронный ресурс]. URL: http://www.w3.org/TR/2004/REC-owl-semantics-20040210/ (дата обращения 18.04.10).

[6] Акобир Шахиди. Введение в анализ ассоциативных правил. [Электронный ресурс]. URL: http://www.basegroup.ru/library/analysis/association_rules/intro/ (дата обращения 15.09.10).

[7] Resource Description Framework (RDF) — Спецификации RDF W3C [Электронный ресурс]. URL: http://www.w3.org/RDF/ (дата обращения 15.09.10).

[8] Щербак С. Начальное руководство по RDFa (перевод) [Электронный ресурс]. URL: http://shcherbak.net/translations/ru_rdfa_primer_shcherbak_net.html (дата обращения 15.09.10).

[9] Ермаков А.Е. Автоматизация онтологического инжиниринга в системах извлечения знаний из текста. // Компьютерная лингвистика и интеллектуальные технологии: По материалам ежегодной Международной конференции «Диалог» (Бекасово, 4–8 июня 2008 г.). Вып. 7 (14).– М.: РГГУ, 2008. – С. 154- 159.

[10] Голенков В.В., Тарасов В.Б., Елисеева О.Е. и др. Интеллектуальные обучающие системы и виртуальные учебные организации: Монография / Под ред. В.В. Голенкова, В.Б. Тарасова — Мн.: БГУИР, 2001.

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ ПРИ ОБРАБОТКЕ КВАТЕРНИОННЫХ СИГНАЛОВ[1]

А. А. Роженцов, А. А. Баев

Марийский государственный технический университет,
кафедра радиотехнических и медико-биологических систем, Россия, 424000
krtmbs@marstu.net

В статье рассмотрены два метода распознавания 3D изображений, не требующие нумерации отметок. Предложены подходы к повышению их быстродействия посредством применения параллельных вычислений на центральном и графическом процессорах. Показаны условия, при которых обеспечивается выигрыш от 10 до 57 раз в быстродействии при использовании графического сопроцессора. Приведен сравнительный анализ быстродействия и эффективности распознавания на основе теста Princeton Benchmark.

Введение

В настоящее время разработчиками систем машинного зрения особое внимание уделяется вопросам обработки 3D изображений ввиду их более высокой информативности, по сравнению с плоскими, появлению относительно недорогих систем 3D сканирования, росту вычислительной мощности систем обработки. Для распознавания 3D изображений разработано большое количество методов: это методы, основанные на согласованной фильтрации [1, 2], требующие упорядочения точек объекта; методы, основанные на отображении из пространства модели в некоторое N-мерное векторное пространство, с последующим определением меры схожести по соответствующим коэффициентам [2, 3, 4]. При этом, сравнение выполняется между коэффициентами искомой модели с заранее вычисленными коэффициентами в базе данных. Этот подход включает в себя методы, основанные как на статистическом [3, 4], так и на аналитическом [2] описании модели. Хотя представление моделей с помощью коэффициентов позволяет ускорить процесс распознавания, большинство методов требуют значительных временных затрат для вычисления коэффициентов искомой модели, что неприемлемо для систем реального времени. С другой стороны эти подходы могут обладать хорошими показателями распознавания. Одним из решений задачи повышения быстродействия алгоритмов является применение параллельных вычислений, основанных на принципах многопоточности в рамках одной вычислительной машины, как на многоядерных CPU, так и на GPU.

В данной статье рассматривается два метода, не требующих упорядочения точек, первый из которых основан на статистическом описании модели, а второй – на аналитическом. Исследуются подходы к повышению быстродействия путем распараллеливания алгоритмов на центральном многоядерном процессоре и на графическом сопроцессоре с использованием технологии NVIDIA CUDA [5].

Оптимизация метода, основанного на аналитическом описании модели

Как показано в работе [1], для обработки объемных изображений могут использоваться методы кватернионного анализа. В этом случае векторы, проведенные в пространстве к точкам,

[1] Работа выполнена при финансовой поддержке РФФИ, проект №10-01-00445-а, и по программе «Развитие научного потенциала высшей школы», проект 2.1.2/2204.

задающим поверхность объекта, описываются векторными кватернионами, а их набор представляет собой кватернионный сигнал.

Использование аппарата кватернионного анализа позволяет связать поверхность, заданную в пространстве, с функцией кватернионного переменного, например, отображающей ее отсчеты на сферу [2]. Для этого применяется полиномиальная функция вида:

$$\sum_{m=0}^{M-1} q_n^m a_m = p_n, \qquad (1)$$

где a_m - коэффициенты полинома, также являющиеся кватернионами, задающие отображение пространственной фигуры на поверхность сферы, q_n - кватернионы, соединяющие точки поверхности объекта с началом координат, p_n- проекции кватернионов q_n на сферу.

Согласно формуле (1) можно вычислить коэффициенты полинома ***a***, связывающего поверхность исследуемого объекта с поверхностью сферы. При использовании метода наименьших квадратов, для вычисления коэффициентов полинома степени М, следует решить систему линейных кватернионных уравнений, элементы которой определяется из соотношений:

$$q_{r,m} = \sum_{n=0}^{N-1} \overline{q_n^r} q_n^m, \quad p_r = \sum_{n=0}^{N-1} \overline{p_n^r} q_n, \qquad (2)$$

где N – количество элементов исходного сигнала.

Решение системы уравнений, например, методом Гаусса, позволяет найти значения коэффициентов a_m полиномиальной функции, выполняющей отображение пространственной фигуры на сферу.

Если рассмотреть последовательный алгоритм решения задачи вычисления коэффициентов полинома (Рис. 1), то здесь каждый последующий блок ждет завершения предыдущего.

При анализе алгоритма на возможность распараллеливания, следует обратить внимание на участки, где проходит обработка массива данных одинаковой последовательностью команд, так как здесь можно применить параллельные вычисления наиболее эффективно.

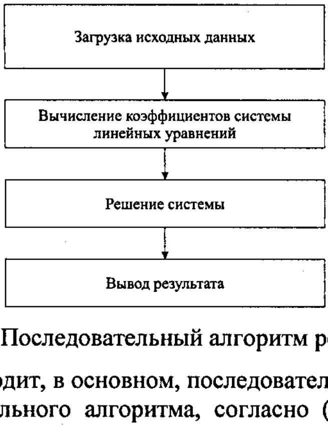


Рис. 1: Последовательный алгоритм решения

Загрузка данных происходит, в основном, последовательно и распараллелить ее сложно. Во второй секции последовательного алгоритма, согласно (2) вычисляются коэффициенты системы линейных уравнений. Однако, суммы (2) можно разложить на K частей, размерностью $N_K=N/K$. Таким образом, система линейных уравнений раскладывается на K независимых друг от друга подсистем, которые могут формироваться параллельно с последующим объединением в одну. Учитывая то, что размер системы линейных уравнений относительно мал, секцию решения системы уравнений можно оставить последовательной.

Для дальнейшей оптимизации на уровне потоков следует рассмотреть формулу (2), где присутствует возведение в степень, что с точки зрения вычисления является трудоемкой задачей. Так как умножение намного проще, наиболее эффективно будет составить таблицу степеней и затем использовать табличные значения.

Реализация, ранее представленного алгоритма, на GPU (Рис.2) подобна CPU, но будет отличаться тем, что код, выполняемый в параллельных секциях, останется параллельным.

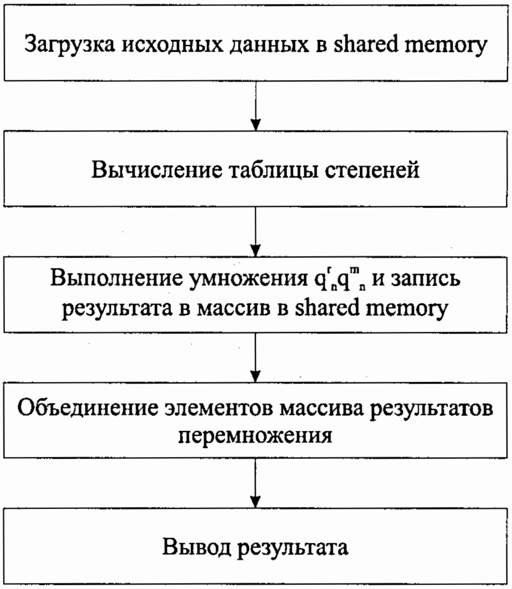

Загрузка исходных данных в shared memory
Вычисление таблицы степеней
Выполнение умножения $q^r_n q^m_n$ и запись результата в массив в shared memory
Объединение элементов массива результатов перемножения
Вывод результата

Рис. 2: Алгоритм работы вычислительного ядра GPU

Для оптимизации кода GPU, для вычислений используется разделяемая память, ввиду наиболее высокого быстродействия. Исходный массив разбивается на блоки, так чтобы их размер с учетом использования временной таблицы степеней не превышал 16 Кб. При использовании двойной точности, наилучшим будет размер блока равный 64 элементам. Ввиду особенностей выполнения кода на GPU, после каждой секции следует выполнять синхронизацию потоков. Так как в блоке выполняется сразу несколько потоков, то в данном алгоритме объединить умножение с суммированием невозможно, и наиболее эффективным способом является использование иерархического суммирования [6].

По завершении работы каждого блока, временная система линейных уравнений копируется в оперативную память. Как только все блоки закончили вычисления, на CPU производится окончательное объединение временных систем линейных уравнений в одну, с последующим ее решением.

Описание и оптимизация метода, основанного на статистическом описании модели

В [3] представлен метод Shape Distribution (D2), генерирующий гистограмму расстояний между парами точек на поверхности модели. Описатель D2 – одномерное, инвариантное к вращению представление трехмерных форм. Кроме того, сохранение расстояний между парами точек на поверхности модели приводит к представлению формы, которое является также инвариантным к трансляции. Однако вычислительная сложность метода резко возрастает с увеличением количества точек объекта. Структура алгоритма распознавания приведена на рис. 3.

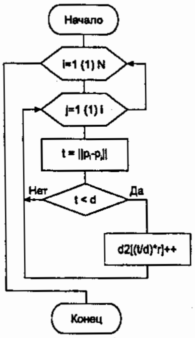


Рис. 3: Алгоритм работы описателя D2

Для оптимизации данного метода для CPU необходимо равномерно распределить нагрузку между вычислительными потоками так, чтобы они заканчивали свою работу за одинаковый промежуток времени, что позволит облегчить организацию их работы. Это достигается путем деления итераций внешнего цикла на несколько частей так, чтобы суммарное количество итераций обоих циклов в потоке было одинаково.

Допустим в системе K решателей, суммарное количество итераций обоих циклов равно N*N, тогда начальные индексы внешнего цикла определятся в соответствии с:

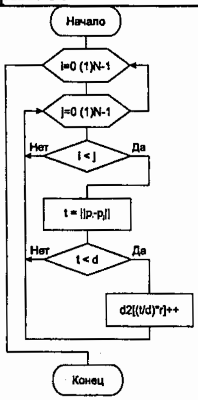
$$n_i = ceil\left(\sqrt{m-i}\right),$$

где $m = N*N/K$, $i = 0..K\text{-}1$, n_i – начальный индекс i-го решателя.

Так, для $N = 1024$ и $K = 4$, $n_0 = 0$, $n_1 = 512$, $n_2 = 725$, $n_3 = 887$.

Однако, ввиду ограничений на время выполнения ядер, на GPU этот алгоритм не будет работать с большими значениями N.

Одним из средств позволяющих работать с GPU является технология NVIDIA CUDA. Для оптимизации алгоритма для GPU следует учесть, что архитектура CUDA позволяет создавать двумерные сетки потоков с фиксированными размерами строк и столбцов и в циклах исходного алгоритма формируется матрица расстояний, можно выделить по потоку на каждую ячейку этой матрицы, тем самым разложив циклы. Однако, в исходном алгоритме, для исключения повторного вычисления расстояний между точками, граница внутреннего цикла плавающая, и чтобы сохранить это свойство при ее фиксации, достаточно добавить условие $i<j$ в алгоритм на рис. 3.



а)

	0	1	2	3	4	5	6	7	8	9
0		1	1	1	1	1	1	1	1	1
1			1	1	1	1	1	1	1	1
2				1	1	1	1	1	1	1
3					1	1	1	1	1	1
4						1	1	1	1	1
5							1	1	1	1
6								1	1	1
7									1	1
8										1

б)

Рис. 4: а) Алгоритм с фиксированными границами цикла и, формируемая в нем, б) матрица расстояний

На рис. 4б изображена матрица расстояний, формируемая в алгоритме при фиксировании границ циклов (рис. 4а), где единичные значения обозначают факт вычисления расстояния между *i*-й и *j*-й точками, а соответствующие ячейки удовлетворяют условию $i<j$. Видно, что заполнена только половина матрицы, следовательно, половина потоков сетки не будет выполнять вычислений.

Предлагаемый подход для оптимального использования ресурсов GPU основан на связи сетки потоков с элементами матрицы, которые необходимо обработать. Если допустить, что x – индекс столбца и y - индекс строки в сетке потоков, то индексы матрицы расстояний определятся в соответствии с:

$$\begin{matrix} i=y \\ j=x \end{matrix}\ if\ y<x \quad \text{и} \quad \begin{matrix} i-x+\dfrac{N}{2} \\ j-y+\dfrac{N}{2} \end{matrix}\ if\ y>x\ .$$

Таким образом, вычисляются все требуемые расстояния при использовании сетки потоков размером $N*N/2$.

Согласно программной архитектуре CUDA, все потоки группируются в блоки. Потоки между собой могут взаимодействовать только в пределах блока. В связи с этим, на каждый блок в глобальной памяти выделяется область под временную гистограмму (рис.5б). Учитывая это, в блоке выполняется 32 потока, для каждого из которых в разделяемой памяти выделена временная гистограмма. Для уменьшения требуемой памяти, гистограммы объявлены как массив типа byte. Таким образом, если каждый поток вычисляет в цикле, по 256 элементов таблицы, то при этом не возникнет переполнения ячейки даже при одинаковых значениях расстояния. Так же при подобном подходе размер необходимой памяти не превысит 16 кбайт, что позволит выполнять программу на всех устройствах с поддержкой CUDA.

Следует отметить, что каждый блок обрабатывается вычислительным ядром, общий алгоритм его работы показан на рисунке 5а. Вычислительное ядро разделено синхронизацией на два блока, так как суммирование ячеек гистограммы производится только после их вычисления. Как говорилось ранее, для суммирования данных из параллельно выполняющихся потоков можно использовать иерархическое суммирование, однако, для данного метода алгоритм, изображенный на рис. 5в, позволяет заменить его на суммирование с накоплением, при котором нет необходимости использовать барьерную синхронизацию, что исключит «холостой ход» потоков.

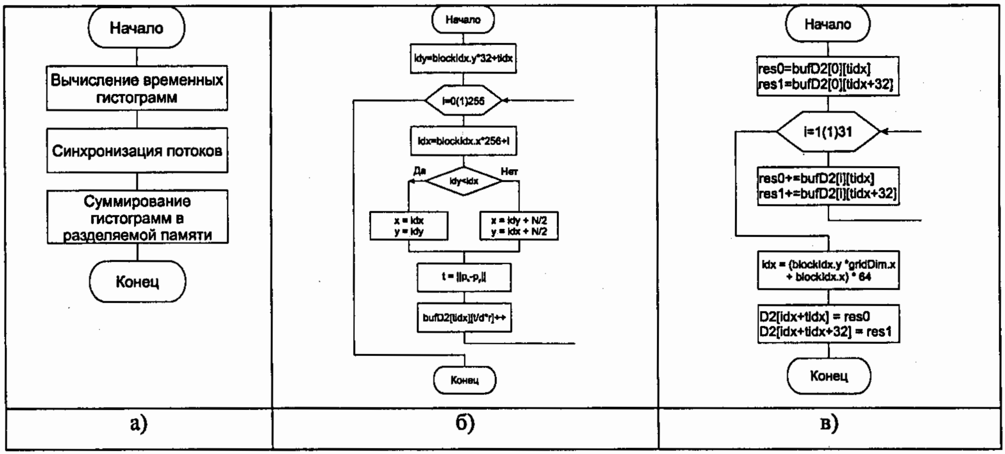


Рис. 5: Вычислительное ядро: а) общий алгоритм работы; б) блок вычисления временных гистограмм; в) блок суммирования и вывода результата

На рис. 5 blockIdx.x и blockIdx.y – индексы блоков потоков, idx и idy – индексы ячеек матрицы расстояний, bufD2 – массив временных гистограмм.

Данный подход позволяет организовать практически независимую друг от друга работу потоков и свести до минимума использование барьерной синхронизации.

Сравнительный анализ быстродействия реализаций на CPU и GPU

Для получения адекватной оценки выигрыша в производительности при вычислении коэффициентов отображающей функции средствами CPU и GPU, выполнен сравнительный анализ результатов исполнения при различных размерах исходных данных N (таблицы 1 и 2). В качестве критерия быстродействия принято время выполнения вычислений в зависимости от размера входных данных.

Для каждого метода выполнены замеры для реализаций на одном и на всех ядрах центрального процессора, а также на видеокарте. В эксперименте использовались 4-х ядерный процессор Intel Corei7 и видеокарта NVIDIA GTX480 с поддержкой технологии CUDA.

Таблица 1. Сравнение быстродействия алгоритмов для CPU и GPU для метода, основанного на аналитическом описании объекта

	Время выполнения (мс)			Соотношения		
N	CPU	MCPU	GPU	CPU/MCPU	CPU/GPU	MCPU/GPU
1280	7	7	0,44	1	15,91	15,91
5000	15	15	1	1	15	15
10000	15	16	1,64	0,93	9,15	9,76
20000	16	16	2,56	1	6,25	6,25
50000	47	31	5,74	1,52	8,19	5,40
100000	125	47	10,51	2,66	11,89	4,47
128000	140	63	12,76	2,22	10,97	4,94
150000	172	78	16,12	2,20	10,67	4,84

Таблица 2. Сравнение быстродействия алгоритмов для CPU и GPU для метода, основанного на статистическом описании объекта

	Время выполнения (мс)			Соотношения		
N	CPU	MCPU	GPU	CPU/MCPU	CPU/GPU	MCPU/GPU
1280	47	32	0,975346	1,47	48,19	32,81
5000	249	125	4,6912	1,99	53,08	26,65
10000	951	312	16,3707	3,05	58,09	19,06
20000	3681	936	62,35221	3,93	59,04	15,01
50000	22667	5444	393,5066	4,16	57,6	13,83
100000	91245	22167	1594,377	4,12	57,23	13,9
128000	150182	33853	2618,04	4,44	57,36	12,93
150000	206889	46395	3592,869	4,46	57,58	12,91

Получено, что для метода, основанного на аналитическом описании, применение параллельных вычислений позволило сократить время вычислений в 12 раз, при относительно малом времени вычислений. Для второго метода применение параллельных вычислений на CPU позволило снизить время расчета в 10 раз, а на GPU более чем в 57 раз.

Заключение

Проведенное сравнение качества результатов распознавания на эталонных тестах Princenton Benchmark показало небольшое преимущество метода, основанного на статистическом описании модели. Однако при анализе быстродействия и помехоустойчивости, наилучшим образом проявил себя метод, основанный на аналитическом описании модели. В ходе проведения эксперимента по распознаванию отмечено, что, при формировании алфавита коэффициентов для 907 объектов, время вычислений при использовании одного ядра центрального процессора для первого метода составило 15 секунд, в то время как для второго метода потребовалось 25 секунд при использовании GPU.

Следует отметить, что применение параллельных вычислений на GPU позволило снизить время вычислений для метода, основанного на статистическом описании модели более чем в 57 раз, в то время как для метода, основанного на статистическом описании модели – всего в 11 раз по сравнению с реализацией последовательных алгоритмов на CPU. Выявлено, что первый метод требует меньше вычислительных затрат и значительно превосходит второй даже при применении параллельных вычислений.

Также на различных этапах отладки программы для вычисления на GPU было отмечено, что при больших объемах входных данных и малой вычислительной нагрузке (незначительное число действий на один элемент массива) вычисления на графических ускорителях становятся нерентабельными из-за копирования данных в память видеокарты.

Литература

[1] Фурман Я.А., Кревецкий А.В., Передреев А.К., Роженцов А.А., Егошина И.Л., Леухин А.Н. Введение в комплексный анализ и его приложения к обработке изображений и сигналов/ Под общей редакцией Я.А. Фурмана. М.: Физматлит, 2002. С. 592.

[2] Роженцов А.А. Оценка параметров и распознавание изображений трехмерных объектов с неупорядоченными отсчетами / Баев А.А., Наумов А.С.//Автометрия, 2010. 46. №1. С. 57 – 69.

[3] Osada R., Funkhouser T., Chazelle B., Dobkin D. Matching 3D models with shape distributions. Shape Modeling International. 2001. P. 154–166.

[4] Kazhdan M., Funkhouser T., and Rusinkiewicz S. Rotation invariant spherical harmonic representation of 3D shape descriptors// In Symposium on Geometry Processing, 2003.

[5] Боресков А.В., Харламов А.А. Основы технологии CUDA. М:ДМК Пресс, 2010. С. 232.

ГРИД-ИНФРАСТРУКТУРА МСЦ РАН ДЛЯ СУПЕРКОМПЬЮТЕРНЫХ ПРИЛОЖЕНИЙ

Г. И. Савин[1], В. В. Корнеев[2], Б. М. Шабанов[1], П. Н. Телегин[1],
Д. В. Семёнов[2], А. В. Киселев[2], А. В. Баранов[2], О. И. Вдовикин[1],
О. С. Аладышев[1], А. П. Овсяников[1]

[1] МСЦ РАН, Россия, 119991, Москва, Ленинский проспект, 32а,
[2] ФГУП «НИИ «Квант», Россия, 125438, *Москва, 4-й Лихачевский пер., д.15*

В докладе представлен опыт создания и эксплуатации грид инфраструктуры МСЦ РАН [1] в рамках проекта «Распределенная инфраструктура для суперкомпьютерных приложений» (РИСП), реализуемого сотрудниками МСЦ РАН и ФГУП «НИИ «Квант».

Цель проекта – интеграция в состав грид-инфраструктуры географически распределенных ресурсов вычислительных систем (ВС) организаций - участников проекта для:

- предоставления пользователям РИСП «прозрачного» доступа к вычислительным ресурсам, превышающим объемы вычислительных ресурсов своих организаций,
- увеличения реальной производительности совокупности ресурсов за счет исключения простоя одних ресурсов при перегруженности других,
- повышения отказоустойчивости и обеспечения требуемого качества обслуживания пользователей, характеризующегося такими параметрами, как время отклика, гарантированная пропускная способность, доступность.

В основу РИСП положены принципы:

- не отчуждаемости ресурсов - ВС организаций, включаемые в состав РИСП, доступны также и для «локальных» пользователей организации, не являющихся пользователями РИСП;
- множественности политик администрирования вычислительных ресурсов – администраторы ВС, включаемых в состав РИСП, определяют права пользователей РИСП в соответствии с реализуемой политикой администрирования, задавая отображения идентификаторов пользователей РИСП в локальные учетные записи пользователей соответствующих ВС;
- прозрачного доступа к требуемым вычислительным ресурсам – пользователь может определить требования к вычислительным ресурсам, необходимым для выполнения задания, не конкретизируя ВС РИСП;
- балансировки вычислительной нагрузки на ВС РИСП – для эффективного использования РИСП, назначение заданий пользователей на ВС осуществляются с учетом требований к ресурсам и текущей загруженности ВС РИСП;
- объединения ресурсов ВС для решения крупномасштабных задач – MPI–программа может быть выполнена на ресурсах нескольких ВС РИСП;
- поддержки отказоустойчивых вычислений – пользователю предоставляется API для формирования контрольных точек, обеспечивается возможность рестарта заданий с контрольной точки.

Управление ресурсами и заданиями в РИСП осуществляется разработанной авторами доклада программной системой (ПС) «Градиент».

Принципы построения и функционирования ПС «Градиент» изложены в [2, 3].

Основные функции, реализуемые ПС «Градиент»:

- мониторинг и визуализация состояния ресурсов РИСП и заданий пользователей;
- идентификация субъектов (пользователи, задания пользователей, управляющие процессы ПС «Градиент») и объектов (вычислительные системы) РИСП. Каждый

объект и субъект РИСП имеет глобальный идентификатор и пару ключей: закрытый и открытый. Соответствие между идентификатором и открытым ключом устанавливается сертификатом формата X.509, подписанным доверенным центром сертификации. Все субъекты и объекты РИСП доверяют Центру Сертификации, развернутому в МСЦ РАН;

- аутентификация субъектов и объектов РИСП. Взаимодействующие стороны осуществляют взаимную аутентификацию на основе протокола SSL с использованием ключей, содержащихся в предъявляемых сторонами сертификатах;
- авторизация субъектов на ВС РИСП. Авторизация осуществляется путем отображения глобального идентификатора субъекта в локальную учетную запись ВС, которая определяет права пользователя на данной ВС;
- прием заданий пользователя и поиск требуемых для его выполнения вычислительных ресурсов в соответствии с одной из реализуемых стратегий назначения заданий (задается при конфигурировании ПС «Градиент») на ресурсы РИСП;
- планирование запуска заданий (при взаимодействии с локальной системой планирования заданий ВС). На стадии планирования задание может быть переназначено на другую ВС для выравнивания вычислительной нагрузки на ВС РИСП;
- управление выполнением задания пользователя (запуск, останов, продолжение, завершение) от имени и с правами пользователя;
- поддержка запуска и выполнения распределенной параллельной задачи, отдельные ветви которой выполняются на разных ВС;
- формирование пользовательской контрольной точки, рестарт заданий с набора данных контрольной точки;
- передача пользователю результатов выполнения задания и файлов системного вывода.

ПС «Градиент» использует сервисы безопасности, пересылки файлов и запуска заданий пакета Globus Toolkit 2.4.3 и взаимодействует с локальными системами планирования заданий ВС. Компоненты ПС «Градиент» устанавливаются на управляющие машины (УМ) ВС, включаемых в состав РИСП.

В ПС «Градиент» в отдельные домены выделены ресурсы каждой ВС грид-среды и создан еще один домен, включающий ресурсы всех ВС, с образованием иерархической очереди. В этой очереди возможно согласованное локальное распределение ресурсов между заданиями в каждом домене при выполнении следующего алгоритма работы с очередями: передача заданий между очередями нижнего уровня иерархии возможна только через очередь верхнего уровня, используемую для планирования заданий между очередями нижнего уровня.

Каждый выделенный домен управляется компонентом ПС «Градиент» — менеджером. Менеджер имеет набор структур данных, необходимых для локального планировщика:

— информационную систему (ИС), содержащую таблицу ресурсов управляемого домена и описание, возможно редуцированное, состояния грид-среды в целом;

— очередь заданий, подлежащих планированию.

Основными функциональными процессами менеджера служат:

— собственно локальный планировщик, вырабатывающий решения о назначении заданий на ресурсы домена или пересылке их в другую очередь на основе данных ИС и очереди заданий;

— процесс, поддерживающий актуальное состояние данных ИС, когерентное с ИС других локальных планировщиков;

— служебные процессы, обеспечивающие отказоустойчивость и информационную безопасность (защиту от несанкционированного доступа к ресурсам).

Менеджер, локальный планировщик которого принимает решение о назначении задания непосредственно на ресурсы ВС, называется менеджером 1-го уровня – M1. Менеджер, планировщик которого распределяет задания между локальными планировщиками, называется менеджером 2-го уровня – М2. Менеджеры М1 передают задания в очередь системы пакетной обработки ВС или менеджеру М2. На УМ каждой ВС обязательно запущен менеджер М1 этой ВС. Количество менеджеров М2 может быть от одного и более в зависимости от требуемых показателей надежности и пропускной способности грид-среды. Менеджеры могут выполняться на управляющих машинах ВС или на специально выделенных компьютерах ВС. Информационные связи между менеджерами образуют граф - дерево. Взаимодействие между менеджерами выполняется по IP-адресам и номерам портов, используемых менеджерами.

В локальных планировщиках менеджеров могут применяться различные алгоритмы выделения ресурсов: от решения оптимизационных задач до эвристических алгоритмов, что позволяет учитывать специфику неоднородности компонентов грид-среды.

В настоящее время (рис.1) в РИСП включены ВС МВС 100К (1460 двухпроцессорных узлов на базе четырехядерных процессоров Intel Xeon - 3 ГГц) и компоненты распределенной ВС МВС-15000ВМD (574 двухпроцессорных узлов на базе процессоров IBM PowerPC 970FX - 2,2ГГц), размещенные в МСЦ РАН (г. Москва), филиале МСЦ РАН (г. Санкт-Петербург) и КНЦ РАН (г. Казань). Продолжаются работы по расширению географии и наращиванию вычислительных мощностей РИСП.

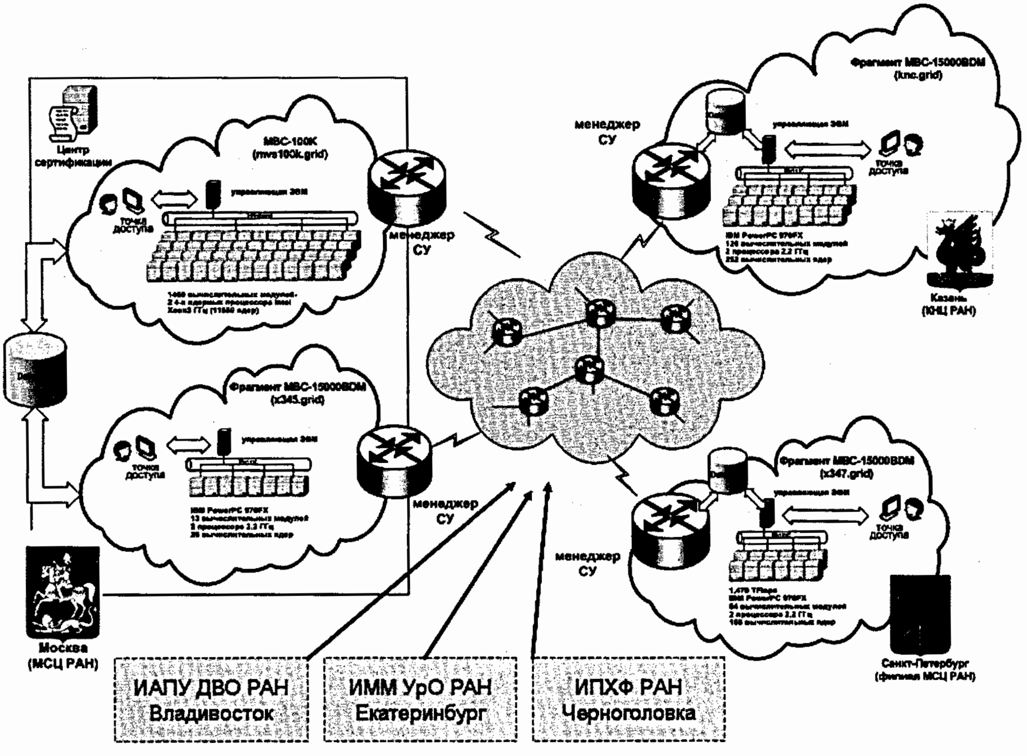


Рис. 1: Структура РИСП

Грид инфраструктура МСЦ РАН используется для решения задач как сотрудниками организаций - участников проекта РИСП, так и представителями других научно-исследовательских организаций и вузов.

В результате проведенных работ удалось решить следующие задачи:

1) развернуть сегмент РИСП на базе распределенных вычислительных ресурсов МСЦ РАН, представленных четырьмя кластерами, размещенными в разных городах и связанными сетью Internet;
2) обеспечить доступность для пользовательских заданий всех территориально распределенных вычислительных ресурсов РИСП;
3) удовлетворить все требования к безопасности, надёжности и отказоустойчивости, предъявляемые к организации вычислений в РИСП;
4) сохранить привычную модель организации вычислений и привычный пользовательский интерфейс.

Литература

[1] Савин Г.И., Корнеев В.В., Шабанов Б.М. и др. Создание распределенной инфраструктуры для суперкомпьютерных приложений. Программные продукты и системы. 2008 №2, с. 2-7.

[2] Корнеев В.В, Киселёв А.В., Семёнов Д.В., Сахаров И.Е. Управление метакомпьютерными системами. // М. «Открытые системы» №2, 2005 г.

[3] Корнеев В.В., Киселев А.В., Баранов А.В., Семенов Д.В., Кузнецов А.В. Сетевая среда распределенных вычислений на базе ресурсов МСЦ РАН. Распределенные вычисления и Грид-технологии в науке и образовании: Труды третьей международной конференции (Дубна, 30 июня – 4 июля 2008 г.). – Дубна: ОИЯИ, 2008. - стр. 118 – 122.

ИНФОРМАЦИОННАЯ ОБРАЗОВАТЕЛЬНАЯ СРЕДА НА ОСНОВЕ WEB 2.0, СЕРВИС-ОРИЕНТИРОВАННОЙ АРХИТЕКТУРЫ И ТЕХНОЛОГИЙ IBM

Д. В. Седова

Университет «Дубна», 141980, Дубна, Россия, dsedova@yandex.ru

В статье представлен краткий анализ эволюции применения информационных и телекоммуникационных технологий (ИКТ) в дистанционном обучении (ДО), предложена методика организации процесса дистанционного обучения с применением социальных сервисов Web 2.0 и SOA, представлена обобщенная модель взаимодействия участников учебного процесса, представлена методика контроля активности участников виртуального коллектива, приведен пример реализации методики.

Информатизация образования и растущие требования к качеству и количеству высококвалифицированных специалистов [1] приводят к необходимости разработки и внедрения инновационных образовательных методик и технологий, способствующих формированию новых форм обучения, не ограниченных пространственно-временными рамками. Поэтому последние тридцать лет характеризуются бурным развитием дистанционных образовательных методов и технологий.

Стремительное развитие информационных технологий подталкивает развитие системы образования в сторону все большего их привлечения в образовательный процесс, особенно это касается дистанционного обучения, ведь в этом случае основным способом «доставки» учебного материала до студентов являются именно ИКТ.

По мере развития ИКТ и, в частности, сети Интернет, изменяются и принципы построения информационных систем для дистанционного обучения (ИСДО). В эволюции применения ИКТ в ДО автор выделяет три этапа:

1) Процесс обучения на основе электронной почты в качестве основного средства взаимодействия «студент-преподаватель» и доставки учебных материалов [2, 3];
2) Внедрение автоматизированных систем управления учебной деятельностью (Learning Management Systems, LMS);
3) Внедрение технологий виртуализации и технологий, ориентированных на использование IT-сервисов [4].

Качество дистанционного обучения напрямую зависит от возможностей систем дистанционного обучения (СДО). Однако нужно учитывать, что СДО — лишь средство обеспечения взаимосвязи между преподавателем и обучающимся, а эффективность обучения в большей степени зависит от личностных и педагогических качеств преподавателя и качества реализации методик обучения, чем от программно-аппаратной платформы. Современный опыт в сфере дистанционного обучения позволяет формулировать требования [5] к функциональности эффективной ИСДО.

С учетом высокого внимания к вопросу об управлении знаниями, в дополнение к сказанному в [5] автор данной работы считает целесообразным сформулировать дополнительные требования программно-методологической платформе для организации ДО:

- широкий спектр форм взаимодействия участников образовательного процесса, в том числе и средства коллективной работы, для формирования горизонтальных связей внутри обучающегося коллектива;
- возможность извлекать знания из информационных источников сети Интернет, систематизировать и обрабатывать информацию, хранить и применять

полученные знания на практике с целью формирования профессиональных навыков;

- инструментарий для создания нового знания, доступного для других участников коллектива;
- доступ к учебному контенту (информации и программному обеспечению) в любое время независимо от местоположения.

Взгляд автора на возможность реализации указанных требований к ИСДО на разных этапах применения ИКТ в ДО представлен в Табл. 1. Нужно учитывать, что наличие соответствующих программных технологий гарантирует реализацию того или иного требования только при наличии соответствующих методик и правильном их использовании.

Таблица 1. Обеспечение функциональных требований к ИСДО
на разных этапах развития ИКТ в ДО

	E-mail	**LMS**	**Использование виртуализации и IT-сервисов**
Доступ к учебному контенту (программное обеспечение и материалы УМК)	Только УМК	Предоставляется только УМК. Дополнительное ПО устанавливается на компьютере учащегося автономно	В полной мере, доступ к ПО обеспечивается виртуализацией приложений и SaaS
Формирование профессиональных навыков работы с программно-аппаратным обеспечением	Невозможно	Изучаемое ПО устанавливается на компьютере учащегося автономно	В полной мере. Доступ к ПО обеспечивается виртуализацией приложений и SaaS, автономной установки не требуется
Формирование навыков профессионального общения (как минимум, освоение терминологии)	Невозможно	В ограниченном объеме из-за узкого набора средств взаимодействия (напр., группового голосового общения)	Широкий спектр инструментов (доступных в качестве веб-сервисов) для многостороннего взаимодействия в реальном времени, напр., вебинары.
Формирование навыков совместной работы	Невозможно	Невозможно, т.к. соответствующие технологии в современных LMS не реализуются	В полной мере обеспечивается возможностью подключения требуемых технологий (напр. wiki) в качестве веб-сервисов
Инструментарий для извлечения знаний из сети Интернет, их систематизации и создания знания	Невозможно	Ограниченный инструментарий	Широкий спектр инструментов
Синхронное/асинхронное взаимодействие с преподавателем и обратная связь	Только асинхронно	Преимущественно асинхронно	Синхронное и асинхронное взаимодействие
Интерактивная связь с преподавателем	Невозможно	Затруднительно	Любые соответствующие технологии и ПО, подключаемые как сервисы
Учет успеваемости и контроль знаний	Невозможно	Тестирование, ведение электронных дневников	Любые соответствующие технологии и ПО, подключаемые как сервисы

	E-mail	**LMS**	**Использование виртуализации и IT-сервисов**
Система идентификации учащегося и персонализации	Невозможно	Затруднительно	Биометрические системы идентификации, персонализация за счет портальных технологий и mashup
Мотивация учебной деятельности студента	Нет	Затруднительно	Высокая, обеспечивается возможностью работы с передовыми технологиями
Возможность быстрой модификации образовательного бизнес-процесса по мере появления новых задач	Невозможно	Невозможно	В полной мере обеспечивается SOA
Оперативное подключение новых курсов и образовательных сервисов	Невозможно	Только в случае поддержки соответствующих стандартов разработки материалов (напр. SCORM)	В полной мере обеспечивается SOA
Всестороннее взаимодействие между всеми участниками процесса обучения	Только «преподаватель-студент»	Преимущественно «преподаватель - студент»	В полной мере

Таким образом, внедрение виртуализации и SOA в программно-технологическую платформу ДО существенно расширяет функциональные возможности традиционных LMS и дает возможность приблизить дистанционное обучение к очной форме обучения по характеру взаимодействия между преподавателями и учащимися. Тем не менее, организация такого образовательного процесса требует новых подходов к управлению и методов обучения, которые будут решать проблемы эффективного использования средств совместной работы в процессе обучения и управления географически-распределенным обучающимся коллективом.

В качестве технологического инструментария для решения вышеуказанных проблем автор данной работы видит применение социальных сервисов в рамках концепции Web 2.0, основными идеями которой являются [6]:

- ориентация на использование веб-сервисов (социальные сети, блоги, форумы, wiki-страницы, теги, закладки и пр.) и распределенное использование ресурсов;
- социализация, что подразумевает формирование сообществ и поддержку общения, применение «коллективного разума» к решению проблем;
- роль пользователя трансформируется из пассивного читателя в создателя контента (знания);
- «фолксономия» — систематизация информации с помощью ключевых слов (тэгов);
- применение специализированных технических средств (синдикация контента (RSS, Atom), технология AJAX, mash-up).

Социальные сервисы Web 2.0 активно применяются в бизнесе. Согласно исследованиям компании Cisco [7], 75% опрошенных компаний используют социальные сети в бизнес-целях, а 50% активно применяют микроблоги. Инструментарий Web 2.0 становится неотъемлемым атрибутом деятельности современного предприятия. Что касается внедрения социальных сервисов в процесс обучения, то это обеспечит широкий спектр форм гибкого всестороннего взаимодействия всех участников процесса обучения, а также мотивирует учащихся к активному процессу получения знаний.

Таким образом, с учетом вышеперечисленных требований к организации ИСДО и, принимая во внимание основные концепции виртуального университета, сформулированные в [1], применение средств Web 2.0 в качестве основы образовательного процесса в ВОС и управления виртуальным коллективом становится оправданным.

На сегодняшний день уже есть опыт применения социальных сетей в образовании как в России (например, социальная сеть для школьников www.dnevnik.ru), так и за рубежом (http://www.educationalnetworking.com/List+of+Networks). Появился термин eLearning 2.0, связанный с применением социальных сервисов для решения образовательных задач в виртуальной среде.

Однако проблемы управления распределенным обучающимся коллективом недостаточно проработаны и методик организации эффективного взаимодействия участников таких коллективов практически нет. В данной статье предлагается методика управления, включающая следующие этапы:

1. Объединение всех участников в социальную сеть,
2. Построение графической модели сети с применением специализированного ПО,
3. Анализ степени и характера активности участников, определение основных параметров социальной сети,
4. На основе полученной информации принятие того или иного управленческого решения.

Эксперимент по построению образовательной социальной сети был проведен в Международном университете природы, общества и человека «Дубна». Социальная сеть была построена на платформе IBM Lotus Connections, которая включает все основные сервисы Web 2.0. Доступ к содержанию экспериментального курса осуществляется по ссылке https://greenhouse.lotus.com/communities/service/html/communityview?communityUuid=5cc0839a-649b-41e0-a635-1eac366e43fc (для зарегистрированных пользователей). В качестве инструментария для взаимодействия преподавателя и студентов использовались следующие сервисы указанной платформы [9]: профили, форумы, блоги, система мгновенных сообщений, загрузка и хранение файлов, активности (activities), обмен закладками (dogear). Эти инструменты могут применяться как персонально, так и для коллективной работы. Весь контент может быть снабжен информационными тегами для быстрой навигации и поиска информации.

В Табл. 2 сделана попытка привести в соответствие элементы учебного процесса и информационные ресурсы, служащие средством обеспечения образовательного процесса в социальной сети.

Таблица 2. Элементы учебного процесса и соответствующий инструментарий Web 2.0

Элементы учебного процесса	**Инструментарий социальной сети**
Лекции	Веб-конференции в Lotus Sametime
	Видеозаписи лекций
Практические занятия	Сессии коллективного принятия решений на форуме Lotus Connections
	Вебинар в Lotus Sametime
	Wiki-страницы
Контроль качества обучения	Тестирование с применением чата Lotus Sametime
	Контрольные вопросы в форуме Lotus Connections
	Мониторинг блогов учащихся
	Загрузка студентами выполненных заданий через сервис Файлы

Элементы учебного процесса	**Инструментарий социальной сети**
	(Files) Lotus Connections
Консультации	Форум Lotus Connections
	Блоги
	Комментарии к работам учащихся в сервисе Файлы (Files) Lotus Connections
	Lotus Sametime
Ведение плана учебного курса (дисциплины)	Активности (Activities) в Lotus Connections
Учебно-методический комплекс	Загрузка, хранение и скачивание учебно-методических материалов через сервис Файлы (Files) Lotus Connections
	Закладки (Bookmarks) для размещения ссылок на полезные ресурсы в Интернете
	Привязка материалов для семинаров и лекций к соответствующим записям в Событиях (Activities)
	Wiki-страницы
Дополнительное взаимодействие участников учебного процесса	Система мгновенных сообщений Lotus Connections

Стоит заметить, что эффективность процесса обучения обусловлена возможностью тщательного контроля и анализа процессов социальной сети. Для осуществления контроля в первую очередь нужен механизм визуализации и анализа, который позволит наглядно представить структуру сети и проанализировать характер взаимодействия ее участников. В случае образовательной социальной сети подобный механизм позволит преподавателю оперативно руководить работой виртуального студенческого коллектива и своевременно принимать соответствующие педагогические решения. Анализ социальной сети можно проводить с применением метода SNA (Social Network Analysis) [9], который позволяет выявить основные показатели социальной сети, характеризующие интенсивность, плотность и направленность связей между участниками сети, а представление социальной сети в виде графа позволяет увидеть сеть и сделать выводы о характере взаимодействия участников сети. Существует ряд программных продуктов [9] для построения графических моделей и анализа сетей.

Для визуализации (т.е. построения модели сети в виде графа) экспериментальной образовательной социальной сети применяется пакет Condor [10]. Анализ степени и характера поведения участников ведется по индексу вклада (contribution index), который определяется так [10]:

$$\text{contribution index} = \frac{\text{полученные сообщения - отправленные сообщения}}{\text{полученные сообщения + отправленные сообщения}}$$

То есть, индексвклада стремится к –1 в том случае, если участник сети преимущественно получает сообщения, к 1– если участник в основном отправляет сообщения, 0 – если количество принятых и отправленных сообщений равно.

В данном эксперименте учитывались только сообщения на форуме, так как основная активность участников сети происходила именно там.

На рис. 1 представлен график, где по оси ординат представлен индекс вклада, а по оси ординат активность участников, равная сумме полученных и отправленных сообщений.

Как видно из рис. 1, основная группа студентов проявляет небольшую активность (порядка 25 сообщений), при этом в основном отправляет сообщения. Некоторые участники сети имеют индекс вклада более приближенный к 0, они посылают сообщений немногим больше чем получают, что говорит о сбалансированном поведении этих участников в сети. Выделяются из общей массы два участника, индекс вклада которых стремится к –1, т.е. они в основном получают сообщения. Согласно исследованиям COIN [10], такая манера поведения свойственна лидерам сети: лидер задает новые актуальные темы для обсуждений, интересные остальным участникам, которые активно поддерживают тему, отвечая автору.

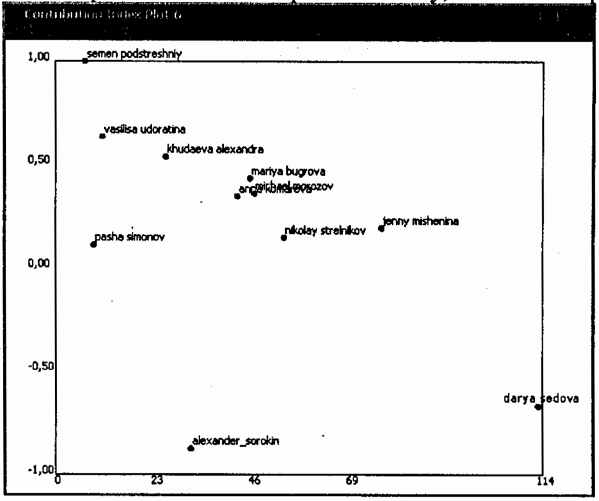


Рис. 1: Индекс вклада

Визуализация социальной сети может быть выполнена на разных этапах обучения, что позволит получить динамическую картину активности участников и своевременно принимать соответствующие педагогические решения. В процессе проведения эксперимента построение и анализ графических моделей социальной сети проводились по мере прохождения основных контрольных точек курса, указанных в учебном плане сервиса «Активности» Lotus Connections.

Общая модель взаимодействия участников учебного процесса схематично изображена на рис. 2:

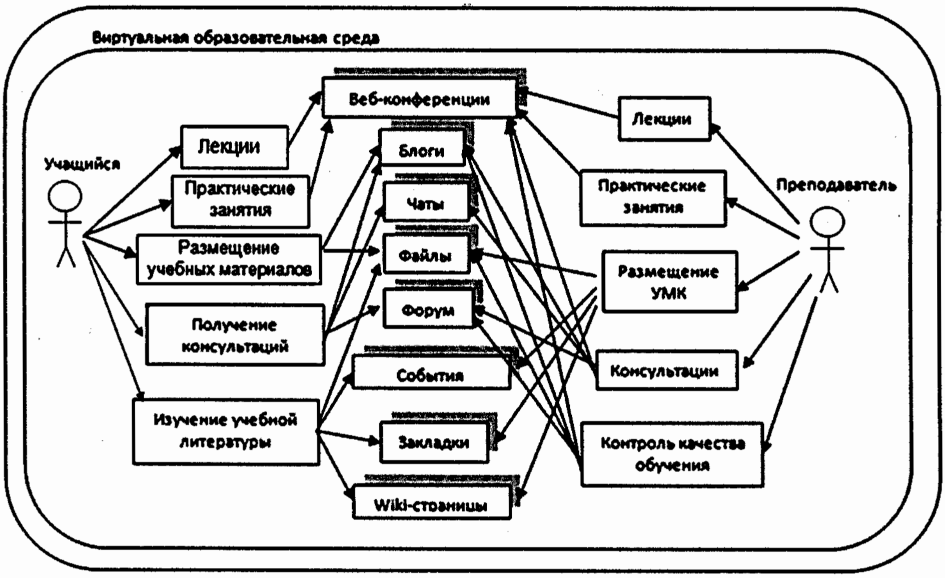


Рис. 2: Модель взаимодействия участников учебного процесса в социальной сети

Таким образом, в совокупности с функциональными возможностями инструментария Web 2.0, технология визуализации дает возможность качественного и оперативного контроля активности всех участников виртуального образовательного коллектива, что является одним из показателей при оценке качества обучения в целом. Представленная концепция построения виртуальной образовательной среды решает указанные задачи программно-методологической платформы ВОС и обеспечивает комплексную поддержку виртуального образования, включая обучение, управление образовательным процессом и контроль его качества.

Литература

[1] Седова Д.В. Виртуальный университет: предпосылки возникновения и перспективы развития // Материалы 15-й научной конференции студентов, аспирантов и молодых специалистов. Дубна, 2008. С. 91– 93.

[2] Lucas-Smith A. E-learning – Who?What?Where? // Tenth International Seminar: Scientific and Technical Information in the Countries of Central and Eastern Europe. Zakopane, Poland, May 9-12, 2001.

[3] Pillemer J. E-mail as a teaching tool. Режим доступа, http://www.etni.org.il/jack.htm

[4] Осознание значения SOA и современных ИТ-инноваций. Режим доступа, http://www-01.ibm.com/software/ru/soa/entrypoints/WSW14032-USEN_Rus_N.pdf

[5] Минзов А.С. Профессиональное высшее и корпоративное образование: образовательные модели и механизмы их реализации / А.С. Минзов. Дубна : Междунар. Ун-т природы, о-ва и человека «Дубна», 2008. С. 138: ил.

[6] O'Reilly Tim. What is Web 2.0 / Tim O'Reilly. 2005. URL: http://oreilly.com/web2/archive/what-is-web-20.html (дата обращения 13.03.2010).

[7] Global Study Reveals Proliferation of Consumer-Based Social Networking Throughout the Enterprise and a Growing Need for Governance and IT Involvement. URL: http://investor.cisco.com/releasedetail.cfm?ReleaseID=437376 (дата обращения 13.03.2010).

[8] Lotus Connections - Social software for business // IBM (сайт). URL: http://www-01.ibm.com/software/lotus/products/connections/ (дата обращения 13.03.2010).

[9] Прохоров А., Ларичев Н. Компьютерная визуализация социальных сетей // КомпьютерПресс, 2006. №9. URL: http://www.sapr.ru/article.aspx?id=16593&iid=771 (дата обращения 13.03.2010).

[10] Gloor P.A. Net creators: Unlocking the swarm creativity of cyberteams through collaborative innovation networks / P. A. Gloor. Oxford University Press, 2004.

[11] Сорокин А.В. Предприятие и Интернет следующего поколения / А.В. Сорокин // Системы и средства информатики, Вып. 18 (дополнительный выпуск). М.: Наука, 2008. С. 86-117.

ИНТЕГРАЦИЯ СИСТЕМЫ МЕТАКОМПЬЮТИНГА X-Сом С СИСТЕМАМИ УПРАВЛЕНИЯ ПРОХОЖДЕНИЕМ ЗАДАНИЙ СУПЕРКОМПЬЮТЕРНЫХ КОМПЛЕКСОВ[1]

С. И. Соболев

НИВЦ МГУ им. М.В. Ломоносова
sergeys@parallel.ru

Система метакомпьютинга X-Com [1], разрабатываемая в НИВЦ МГУ имени М.В. Ломоносова, представляет собой программный инструментарий для организации распределенных неоднородных вычислительных сред и проведения расчетов в таких средах. В качестве вычислительных ресурсов, составляющих основу распределенных сред, могут использоваться компьютеры практически любого типа, от домашних и офисных машин до высокопроизводительных многопроцессорных серверов. Часто распределенные среды строятся на основе узлов и сегментов суперкомпьютерных комплексов. Это позволяет задействовать сразу несколько высокопроизводительных систем для работы над "тяжелыми" задачами в тех случаях, когда ресурсов одного комплекса оказывается недостаточно. Однотипная структура программно-аппаратной платформы таких узлов упрощает их использование в составе распределенной среды как в плане установки программного обеспечения, так и при организации непосредственно вычислительного процесса.

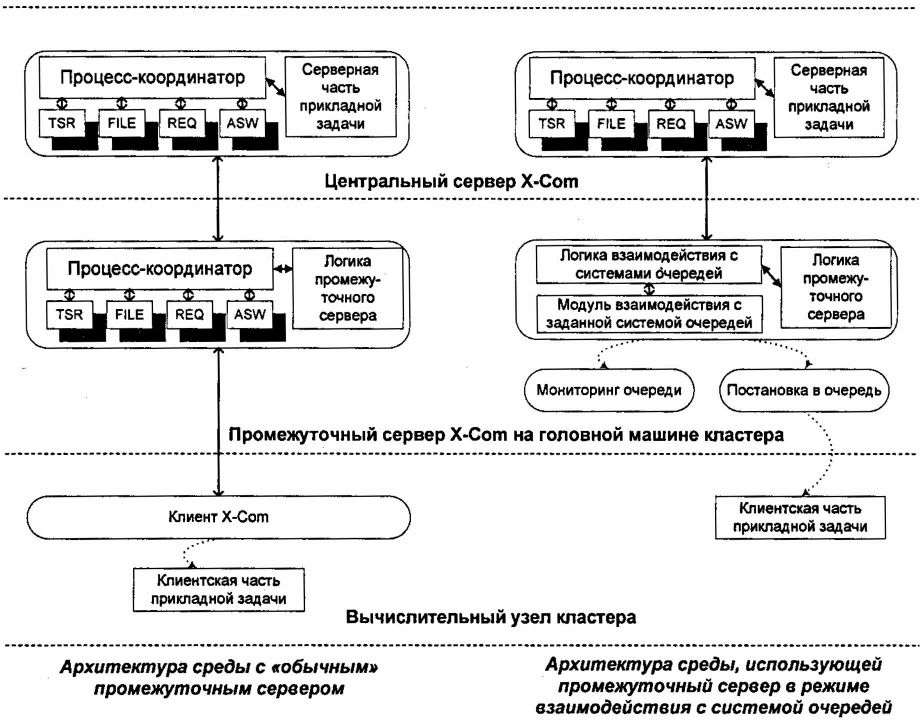
Изначально в X-Com было заложено несколько базовых сценариев использования вычислительных узлов в рамках кластерных систем. В простейшем случае предполагалось, что узлы для распределенного расчета выделяются монопольно. Такой способ оправдан в случае относительно небольшой загрузки вычислительной системы. Однако в настоящее время таких ситуаций практически не бывает – большинство суперкомпьютерных комплексов работают со стопроцентной загрузкой. По той же причине оказывается малоприменим и запуск распределенных расчетов на узлах в моменты простоя, когда они не используются для других приложений. Кроме того, оба способа неявно предполагают проведение определенных действий или получение разрешений со стороны администрации вычислительных комплексов. Запуск клиентской части X-Com через штатные системы управления прохождением заданий (СУПЗ) также имеет свои ограничения – возможность запуска только однопроцессорных задач, необходимость отслеживать состояние очереди заданий сторонними средствами, повышенный расход трафика (если несколько клиентов работают на одном узле, то каждый из них независимо от других скачает с сервера исполнимый код задачи и т.д.).

Тем не менее, очевидно, что работать все-таки нужно через штатные СУПЗ – это единственный способ прозрачно и эффективно использовать доступные ресурсы. Кроме того, использование СУПЗ позволило бы запускать в рамках X-Com не только однопроцессорные задачи, но и приложения, использующие MPI и другие технологии параллельного программирования. Поэтому в систему X-Com были добавлены модули взаимодействия и интеграции с наиболее распространенными СУПЗ кластерных систем Torque, Cleo, LoadLeveler, а также с ПО промежуточного уровня Grid-сред Unicore. Последний модуль дает возможность подключить к расчетам сегменты Grid-сред, использующие соответствующее ПО для доступа к удаленным вычислительным ресурсам.

С точки зрения архитектуры X-Com разработанные модули используют интерфейсы, механизмы и функции промежуточных серверов X-Com [2]. В частности, модули осуществляют буферизацию входящих и исходящих порций между центральным

[1] Работа выполняется при поддержке научно-технической программы Союзного государства СКИФ-ГРИД и гранта Президента РФ для молодых ученых МК-3040.2009.9.

(вышестоящим) сервером и целевой системой. Сервер X-Com может быть запущен в режиме взаимодействия с СУПЗ путем указания соответствующих настроек в файле инициализации, при этом запуск осуществляется на головной машине целевого кластера (Рис. 1). Отметим, что клиент X-Com при такой организации расчетов не используется – его функциональность фактически берет на себя промежуточный сервер.



Процесс-координатор
Серверная часть прикладной задачи
TSR
FILE
REQ
ASW
Процесс-координатор
Серверная часть прикладной задачи
TSR
FILE
REQ
ASW
Центральный сервер X-Com
Процесс-координатор
Логика промежуточного сервера
TSR
FILE
REQ
ASW
Логика взаимодействия с системами очередей
Модуль взаимодействия с заданной системой очередей
Логика промежуточного сервера
Мониторинг очереди
Постановка в очередь
Промежуточный сервер X-Com на головной машине кластера
Клиент X-Com
Клиентская часть прикладной задачи
Клиентская часть прикладной задачи
Вычислительный узел кластера

Архитектура среды с «обычным» промежуточным сервером

Архитектура среды, использующей промежуточный сервер в режиме взаимодействия с системой очередей

Рис. 1: Архитектура распределенной среды X-Com с использованием "обычного" промежуточного сервера и режима взаимодействия с кластерной СУПЗ

Для непосредственного взаимодействия с СУПЗ модули X-Com вызывают утилиты командной строки соответствующей СУПЗ. Так, для Torque мониторинг и постановка задачи в очередь осуществляется командами qstat и qsub, для LoadLeveler – командами llq и llsubmit, для Unicore во всех случаях используется вызов клиента ucc с нужными опциями. Постановка задач в очередь Cleo производится с помощью команды mpirun, мониторинг с целью экономии ресурсов реализован через чтение служебных файлов, автоматически создаваемых этой СУПЗ. Для каждой СУПЗ генерируется файл описания задания в нужном формате.

Для любой поддерживаемой СУПЗ применимы следующие настройки:

- максимально допустимое время счета задачи;
- число процессоров, на которых будет запущено задание;
- интервал обновления данных о состоянии очереди;
- число попыток поставить задание в очередь, по истечении которого будет осуществлен переход к следующей порции.

Также для каждой поддерживаемой СУПЗ существует набор собственных настроек. Например, для Cleo можно указать название очереди, для Unicore – целевую систему и т.д.

Имеется ряд настроек, отвечающий за буферизацию входящих и исходящих порций. Эти настройки применяются как для "обычного" промежуточного сервера, так и для режима работы совместно с СУПЗ:

- максимальное число входящих порций в одном запросе, буферизуемых промежуточным сервером. Это же значение в настоящий момент используется как максимально допустимое число заданий в очереди;
- минимальное число порций во входном буфере, после достижения которого промежуточный сервер пополнит буфер;
- максимальный размер буфера исходящих порций, после достижения которого начнется отправка порций вышестоящему серверу;
- число исходящих порций, отправляемых вышестоящему серверу в одном запросе.

Для апробации разработанных модулей и новых режимов работы X-Com проводились серии распределенных экспериментов с использованием приложения John the Ripper [3], осуществляющего перебор паролей, зашифрованных стандартными средствами UNIX. При наличии достаточно большого набора зашифрованных паролей каждый из них может расшифровываться отдельно и независимо от других, поэтому задача элементарно делится на порции достаточно высокой вычислительной сложности, при этом размеры передаваемых данных крайне невелики (десятки байт на каждую порцию). Таким образом, выбранная задача идеально отвечает схеме метакомпьютерных вычислений.

Для проведения расчетов были задействованы 4 суперкомпьютерные системы:

- СКИФ МГУ "Чебышев" (г. Москва);
- СКИФ Урал (г. Челябинск);
- СКИФ Cyberia (г. Томск);
- вычислительный кластер УГАТУ (г. Уфа).

Центральный сервер X-Com был запущен на головной машине кластера СКИФ МГУ "Чебышев". На головных машинах каждого из кластеров запускался промежуточный сервер X-Com в режиме взаимодействия с системой очередей кластера. На кластере СКИФ МГУ "Чебышев" использовался модуль взаимодействия с Cleo, на кластерах СКИФ Cyberia и СКИФ Урал – модули взаимодействия с Torque, на кластере УГАТУ – модуль взаимодействия с LoadLeveler.

На всех кластерах были установлены одинаковые настройки буферизации порций. Размер окна входящих порций составлял 20 порций, размер окна исходящих порций – 5 порций. Согласно этим параметрам, изначально каждый промежуточный сервер запрашивал у центрального сервера по 20 порций и далее всегда поддерживал запас в 20 порций к обработке, при этом число задач в очереди кластера независимо от их состояния также поддерживалось равным 20. При превышении числа готовых порций, равного 5, производилась отправка результатов пакетами по 5 порций. Отметим, что число порций в пакетных запросах на обработку было плавающим (от 1 до 20), число же готовых порций в ответных пакетах всегда было фиксированным (5).

Было проведено две серии вычислительных экспериментов на двух различных наборах паролей. В качестве первого набора использовался файл со словарем, размещенный на сайте John the Ripper и составляющий основу его внутренней базы [4]. Ввиду того, что подбор таких паролей, очевидно, не мог представлять большой сложности, каждая порция содержала 50 паролей, а время обработки одной порции было ограничено 15 минутами. Ограничения были заданы как при постановке задачи в очереди кластеров, так и непосредственно в скрипте, вызывающем John the Ripper. В случае неудачи (невозможности за указанное время полностью расшифровать заданный набор) в качестве результата возвращался пустой файл. Также 15-минутное ограничение позволило использовать очередь test на кластере СКИФ МГУ "Чебышев", которая, как правило, более свободна, чем другие очереди на этой машине. Первая серия экспериментов продолжалась 15 часов, из 129000 паролей было расшифровано 95.7%.

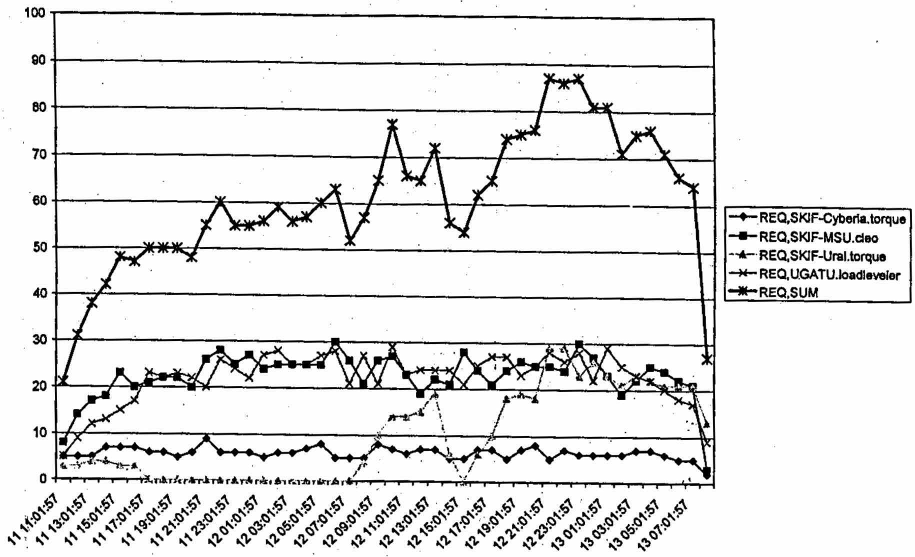


Рис. 2: График запросов исходных данных промежуточными серверами в распределенном запуске John the Ripper

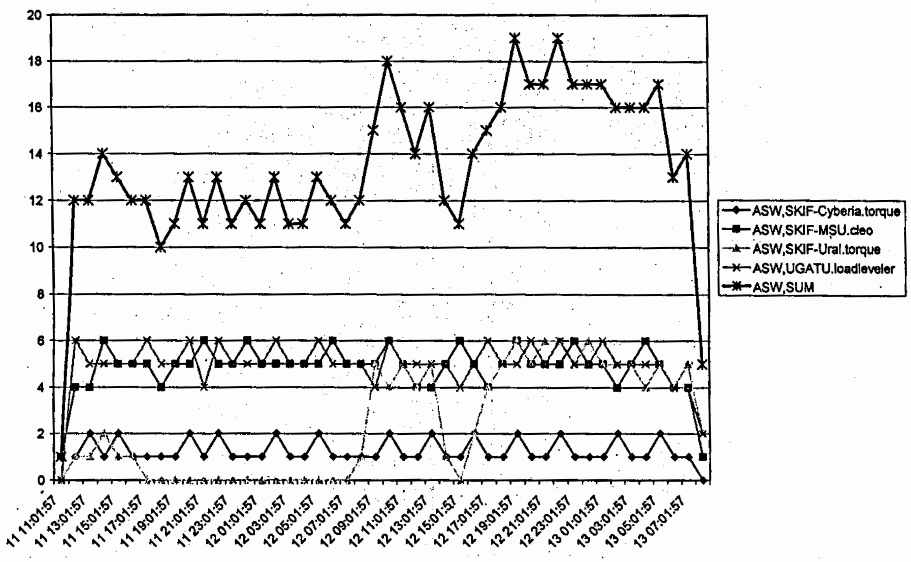


Рис. 3: График возвращения результатов промежуточными серверами в распределенном запуске John the Ripper

Более интересной оказалась вторая серия экспериментов. Набор паролей для нее был сгенерирован с помощью программы на языке Си. Генерировались пароли длиной от 4 до 10 символов, дополнительным условием при генерации была указана возможность легкого произнесения и запоминания паролей, при этом они не должны были совпадать со словарными выражениями. В каждой порции содержался только один пароль, а время подбора было ограничено 1 часом. Эксперимент продолжался более 45 часов, при этом из 3000 паролей было подобрано всего 728, т.е. 24.3%.

Графики интенсивности запросов второй серии экспериментов приведены на рис. 2 и 3. По горизонтальной оси отложено время, по вертикальной – число запросов за данный интервал времени. Видно, что в данном эксперименте сложность обработки каждой порции была примерно одинаковой. Рост суммарных графиков начиная со 2-й половины эксперимента можно объяснить освобождением ресурсов суперкомпьютера СКИФ Урал. Регулярный "пилообразный" характер линий на графике выходных результатов (рис. 3) вызван фиксированным размером выходного окна. На рис. 4 изображен вклад каждого из вычислительных комплексов в решение задачи (по числу обработанных и полученных сервером порций).

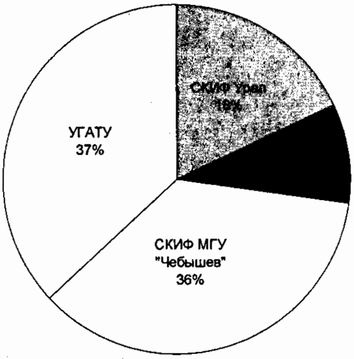


Рис. 4: Вклад каждого из вычислительных комплексов в задаче распределенного запуска John the Ripper

Безусловно, в рамках проведенных экспериментов расшифровка паролей имела лишь академический характере. Тем не менее, решение такой задачи может быть полезно на практике для проверки стойкости паролей пользователей больших систем коллективного доступа в тех случаях, когда пароли фактически являются единственным методом защиты доступа. Стойкость паролей имеет крайне важное значение при обеспечении безопасности таких систем.

Использование модулей взаимодействия X-Com с системами очередей, как уже говорилось выше, в настоящее время видится как основной способ подключения к распределенным расчетам ресурсов высокопроизводительных вычислительных комплексов. Однако такая модель подключения ресурсов ставит перед системой X-Com ряд новых вопросов и задач. Одна из таких задач – поиск новых оценок эффективности проведения распределенных расчетов. Долгое время в качестве таковых в X-Com применялись две характеристики, условно называемые "серверной" и "клиентской" эффективностью. "Серверная" эффективность позволяла оценить накладные расходы системы метакомпьютинга и вычислялась как отношение времени расчета порции на узле ко времени, затраченному серверной частью X-Com на полную обработку этой порции, включая генерацию, обработку клиентских запросов, передачу данных. При работе через СУПЗ к накладным расходам системы метакомпьютинга добавляются накладные расходы системы очередей, включая время ожидания задания в очереди, которое может быть достаточно велико. Очевидно, что данная оценка при прочих равных будет принимать достаточно высокие значения для относительно свободной вычислительной системы и достаточно низкие значения для загруженной системы, однако об "эффективности" в таком контексте говорить будет уже некорректно.

Вторая используемая оценка – "клиентская эффективность" – вычислялась как отношение числа порций, розданных клиентам, к числу полученных результатов. Эта оценка характеризовала потери порций, вызванные, как правило, сбоями работы клиентской части X-Com на вычислительных узлах. В случае стабильной работы всех клиентов такая оценка имеет высокие значения, при наличии сбоев (перезапуск клиентов, обрывы связи и т.д.) значение ее снижается. Следует отметить, что вследствие особенностей стандартного алгоритма распределения порций в X-Com (после отдачи последней порции сервер X-Com начинает заново раздавать те порции, результаты обработки которых до сих пор не получены) данная оценка почти никогда не достигает 100%, хотя при грамотной организации вычислительного эксперимента ее значения достаточно высоки. При использовании механизмов взаимодействия с СУПЗ возникает новый потенциальный источник "потерь" порций – входные и выходные буферы промежуточных серверов.

Еще одна интересная задача – динамическое определение размеров буферов промежуточных серверов, работающих совместно с СУПЗ, размеров окон приема и передачи порций, а также числа одновременно поддерживаемых в очереди заданий, при которых может быть достигнут оптимум между накладными расходами на обмен данными, неизбежными потерями порций, оседающих в буферах, и загрузкой конкретной вычислительной системы. Решением этой задачи может быть увеличение числа ручных настроек параметров буферизации, возможность менять их в ходе работы промежуточного сервера, а также использование информации о доступных ресурсах, полученной от СУПЗ.

Литература

[1] Система метакомпьютинга X-Com (официальный сайт проекта), http://x-com.parallel.ru

[2] Воеводин Вл.В., Жолудев Ю.А., Соболев С.И., Стефанов К.С. Эволюция системы метакомпьютинга X-Com // Вестник Нижегородского государственного университета им. Н.И. Лобачевского. №4. 2009. С 157.

[3] John the Ripper, http://www.openwall.com/john/

[4] John the Ripper (библиотека стандартных паролей), http://download.openwall.net/pub/wordlists/all.gz

ИНФОРМАЦИОННАЯ СИСТЕМА ГРИДННС[1]

М. М. Степанова, О. Л. Стесик, Д. С. Кастерин, С. Л. Яковлев

Санкт-Петербургский государственный университет,
физический факультет, кафедра вычислительной физики
mstep@mms.nw.ru, stes@mms.nw.ru, dmk.pre@gmail.com, sl-yakovlev@yandex.ru

Стабильная, масштабируемая и динамичная информационная система необходима для устойчивого функционирования любой качественной грид-системы. В данной работе представлен модуль "Информационная система" (ИС), который разработан для использования в рамках проекта Национальной нанотехнологической сети (ГридННС) [1].

Назначение этого модуля - сбор, агрегирование и динамическое обновление информации о состоянии ресурсов сайтов, подключенных к грид-среде, и предоставление актуальных данных в ответ на клиентские запросы. Все основные сервисы ГридННС взаимодействуют с ИС для получения оперативной информации о текущем состоянии грид-инфраструктуры. Основным потребителем информации ИС является Система управления выполнением заданий (СУВЗ) в процессе поиска и выбора ресурса для выполнения конкретного задания. Кроме того, данные ИС необходимы для работы системы мониторинга, пользовательского интерфейса, системы передачи данных и других подсистем.

Большинство производственных грид-инфраструктур пока использует основанные на LDAP информационные системы, такие как BDII (gLite) [2] и MDS2 [3]. Однако, в качестве базового middleware проекта ГридННС был выбран сервис-ориентированный Globus Toolkit 4 (GT4) [4], поэтому основой реализации ИС стал компонент Monitoring and Discovery System (MDS4) из этого пакета. Следует также отметить, что MDS4 успешно используется в ряде других крупных проектов, например, TeraGrid [5], APAC[6].

Основа реализации – MDS4

Информационная система ГридННС реализована на базе веб-сервисных компонентов с использованием спецификаций WSRF [7]. Использование MDS4 упрощает создание информационного грид-сервиса, поскольку он предоставляет работоспособный высокоуровневый сервис агрегирования данных, стандартный интерфейс и минимальную схему для доступа к информации о ресурсах. При этом он позволяет создавать собственных поставщиков, если требуется публиковать дополнительную информацию, которая отсутствует в исходной реализации.

MDS4 включает два высокоуровневых сервиса – Индекс (Index Service) и Сервис Событий (Trigger Service), набор поставщиков информации (InformationProviders), а также интерфейсы для представления данных. Индекс обеспечивает сбор данных от разных поставщиков и предоставление информации в виде свойств ресурса другим сервисам и клиентам. Возможно построение иерархической структуры Индексов с агрегированием данных на нескольких уровнях, при этом поддерживается регулярное обновление и кэширование самых последних версий. Сервис Событий может извлекать информацию из Индекса и других источников, а также инициировать некоторые действия, когда выполняются определённые условия.

Данные для регистрации в Информационном Сервисе (Индексе) поступают от поставщиков информации (InformationProviders). В качестве провайдера может выступать как WSRF-сервис, данные от которого поступают по запросу, так и внешняя программа, которая

[1] Работа была частично выполнена при финансовой поддержке Министерства образования и науки РФ (контракт № 01.647.11.2004).

периодически опрашивает произвольный ресурс и выводит информацию в XML-формате, соответствующем схеме публикации в ИС.

В ИС GT4 уже входит набор стандартных провайдеров для кластеров, ЛМР и некоторых не WSRF-сервисов. По умолчанию, реализация MDS4 может публиковать информацию только на основе встроенной схемы GT4. Эта схема, как правило, недостаточна для использования в реальном проекте, кроме того, не все возможности корректно реализованы в коде. Но в MDS4 существуют два интерфейса для внедрения новых провайдеров: UsefulRP и Execution Aggregator Source [8]. При использовании UsefulRP данные от каждого источника будут публиковаться, как отдельная группа (свойство ресурса), в то время как Execution Aggregator Source позволяет публиковать данные в одной группе Индекса. Такая расширяемая подсистема дает возможность динамически генерировать значения XML для одного или более свойств ресурса и публиковать любую информацию по любой схеме.

Информационная схема ГридННС

Для описания и представления информации о грид-ресурсах необходима стандартная схема. В грид принято использовать GLUE-схему, которая поддерживается рабочей группой OGF (GLUE Working Group). Она определяет объекты, которые описывают организацию и структуру сайта, кластера, гомогенных подкластеров, узлов, системы очередей, программного обеспечения и др. Смысл использования в различных проектах максимально стандартной схемы описания ресурсов состоит в том, чтобы упростить взаимодействие между производственными грид-системами, созданными на основе разного ПО.

В настоящее время для ГридННС разработана и используется XML-реализация модели информационного пространства на основе GLUE версии 1.3 [9]. На момент старта нашего проекта еще не была принята окончательная спецификация GLUE 2.0 [10], а встроенная в MDS4 XML-реализация GLUE 1.1 имеет существенные ограничения – в частности, в ней нельзя публиковать данные о сайте, виртуальных организациях и программном обеспечении, а также код встроенного провайдера GRAM4 не дает адекватной информации об очередях. Поэтому, как и большинство грид-систем, мы используем вариант GLUE1.3, модифицированный с учетом особенностей и текущих потребностей своего проекта.

На рис.1 представлены основные объекты схемы.

Объект SITE содержит общие административные сведения о грид-сайте и список всех сервисов, которые поддерживаются этим сайтом.

Важнейшей частью описания вычислительного ресурса является объект GLUECE (или ComputingElement), который, по сути, дает обобщенное представление очереди системы ЛМР. GLUECE содержит статические характеристики (Info), параметры, часто меняющие статус (State), и политики очереди (Policy). Также он может включать набор авторизованных пользователей или групп (ACL), атрибутов групп (VOViews) и текущие задания (Jobs).

Объект Service служит для представления атрибутов грид-сервисов. В зависимости от типа сервиса, можно использовать описание, наиболее полно отражающее его конкретную специфику.

Объект Cluster предназначен для комплексного описания вычислительного ресурса, включающего произвольное количество однородных подкластеров (SubCluster). В свою очередь, SubCluster определяет однородный набор узлов, объединенных одним ЛМР. Соответственно, он включает в себя объекты Queue, Host и Software. Следует отметить, что объект Queue нам пришлось ввести в описание подкластера, чтобы устранить неоднозначность в стандартной схеме GLUE 1.3, возникающую при описании сайтов с набором очередей и более чем одним подкластером. В исходной схеме не было связки между очередью и подкластером, на котором она работает. Объект Queue содержит доменное имя узла с ЛМР, название очереди, и ограничения доступа к очереди (ACL).

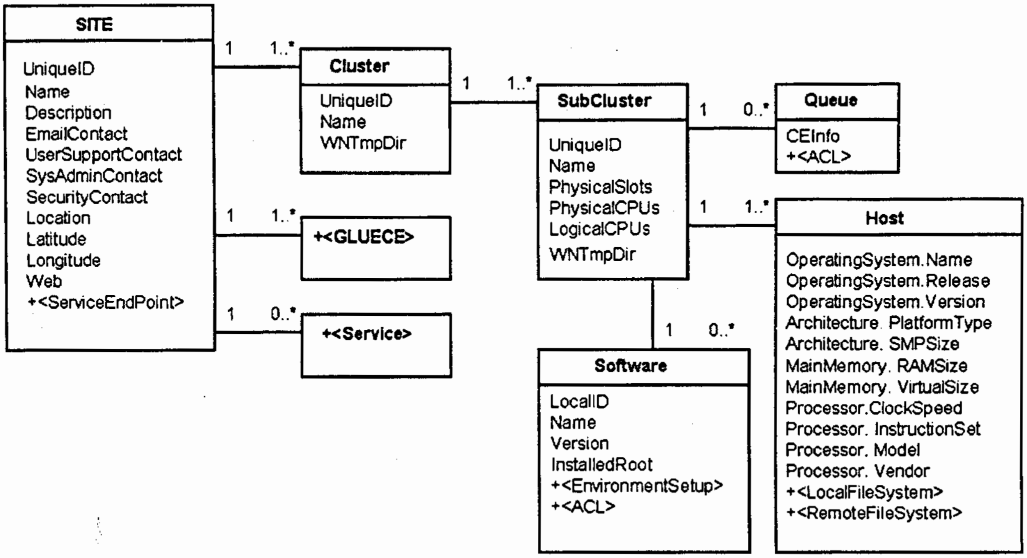


Рис. 1: Информационная схема ГридННС - модифицированный вариант GLUE 1.3

Отметим еще ряд особенностей структуры объекта Software, описывающего ПО на подкластере. Во-первых, это поддержка публикации названия исполняемого модуля, которая решает проблему сайтов, где присутствует несколько скомпилированных версий пакетов, установленных в разные каталоги. Во-вторых, он содержит вложенный объект EnvironmentSetup, который используется для установки среды окружения при запуске и выполнении программы, если GRAM сайта имеет поддержку модуля SoftEnv. Наконец, добавлен вложенный элемент ACL, поскольку приложения могут иметь лицензионные ограничения и к ним потребуется ограничение доступа в рамках VO.

Реализация Информационной системы в ГридННС

Архитектура. Функционирование ИС обеспечивается взаимосвязанной работой компонентов, поставляющих информацию, и компонентов, обеспечивающих ее сбор и объединение. Взаимодействие отдельных частей ИС основано на технологии веб-сервисов и происходит путем обмена сообщениями, которые содержат структурированные данные в виде XML документов. Обмен сообщениями осуществляется по протоколу SOAP, который в качестве транспортного протокола использует HTTPS.

Информационная система ГридННС имеет два иерархических уровня – нижний уровень ресурсного центра (соответствующий ему информационный сервис в дальнейшем будем называть Локальным информационным сервисом (ЛИС)) и верхний уровень, который агрегирует данные всех ЛИС (далее Центральный информационный сервис (ЦИС)). Информационные сервисы (как ЛИС, так и ЦИС) построены на основе MDS4, который запускается в Globus-контейнере.

ИС собирает информацию о грид-ресурсах и делает ее доступной в виде свойств ресурса. При этом поддерживает кэширование самых последних версий данных, используя механизм “самоочистки”. Опубликованный в ИС объект имеет определенное время жизни и запись автоматически удаляется, если за это время не произошло обновление.

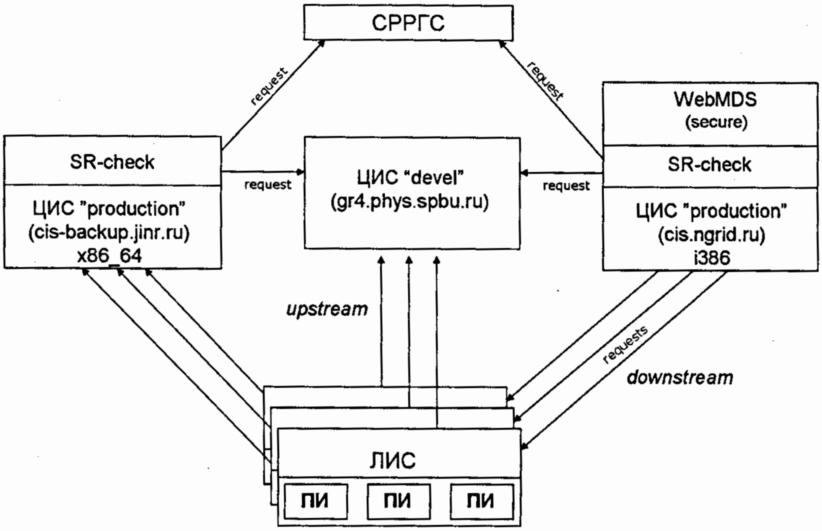


Рис. 2: Схема взаимодействия информационных сервисов ГридННС

Текущая схема работы ИС представлена на рис.2. Каждый сайт должен иметь ЛИС, который обеспечивает агрегирование данных о всех ресурсах сайта и публикацию в ЦИС. В настоящее время информационная инфраструктура ГридННС содержит три ЦИС: два из них обеспечивают работу production-зоны, один (ЦИС "devel") – публикует информацию о всех ресурсах, в том числе тестовых. ЛИС на всех сайтах настраиваются на публикацию в devel-ЦИС, используя механизм upstream. Чтобы сайт смог публиковать данные на сервере с установленным ЛИС достаточно иметь сертификат узла, подписанный удостоверяющим центром ГридННС. Production-ЦИС сам опрашивает ЛИС и забирает с них данные, используя механизм downstream. При этом возможна публикация данных только для тех ЛИС, которые предоставляют корректную информацию о ресурсах и сервисах сайтов, официально зарегистрированных в Сервисе регистрации (СРРГС). Такая фильтрация обеспечивается разработанным модулем SR-check.

Модуль взаимодействия с СРРГС. Каждый сайт ГридННС должен быть зарегистрирован в Сервисе регистрации. Модуль взаимодействия с СРРГС выполняет задачу коррекции данных production-ЦИС по данным сервиса регистрации. А именно, периодически делается проверка на соответствие данных, которые публикует ЛИС сайта в devel-ЦИС, с информацией о сайте из СРРГР, в том числе проверяется статус и состояние сайта (working/testing/downtime, etc). По результатам проверки формируется список ЛИС сайтов (CIS-downstream-list), которые будут опрашиваться ЦИСом. Диагностическая информация с результатами каждой проверки доступна для просмотра администраторам сайтов в виде html-страницы.

Провайдеры и публикация. Первичными источниками информации о состоянии ресурсов ГридННС являются а) локальные (кластерные) системы управления, работающие на грид-шлюзе и б) сервисы ГридННС, предоставляющие разные типы услуг (например, сервисы, обеспечивающие передачу данных, или сервис управления прокси-сертификатами). Учитывая гетерогенность реально существующих систем, каждая из которых поддерживает свои собственные интерфейсы и протоколы доступа к данным, необходима разработка специальных программных компонент — поставщиков информации (ПИ, InformationProvider). ПИ выполняют функции сбора данных о состоянии конкретных ресурсов, формируют валидный XML-документ и передают его службе агрегации.

В реализацию MDS4 входит набор провайдеров, но ни один из них в исходном виде не удовлетворял нашим требованиям. Основную сложность представлял провайдер локального менеджера ресурсов (ПИ GRAM), который собирает информацию из ЛМР и предоставляет ее в сервис GRAM4. Сервис использует эту информацию для запуска задач и регистрирует ее в ЛИС. При способе публикации, который используется в MDS по умолчанию, отключить этот ПИ нельзя, а часть программного кода, отвечающего за публикацию, неработоспособна и не позволяет, в частности, публиковать динамическую информацию о состоянии очередей. Поэтому потребовалось использовать другой способ публикации, который позволяет отключить встроенный GRAM4-провайдер, и разработать новый ПИ для ЛРМ. В результате, для ЛМР PBS/Torque [11] имеется возможность корректно публиковать всю динамическую информацию. На рис. 3 жирным шрифтом выделены атрибуты очереди, которые были добавлены.

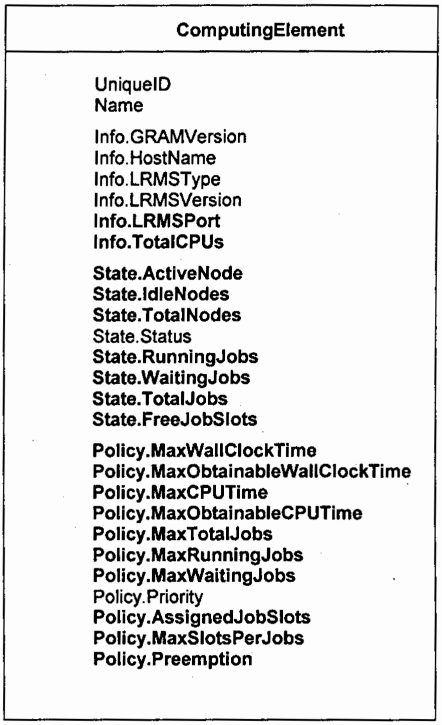


Рис. 3: Информация о CE

Создание собственного провайдера требуется для любого не WSRF-сервиса. Например, провайдер информации, разработанный для сервиса GridFTP, соединяется с GridFTP-сервером, читает заголовок его ответа и публикует в ЛИС. Если соединение не удается установить в течение 30 секунд или присутствуют ошибки, сервер помечается как нерабочий. Этот ПИ представляет собой скрипт на языке Perl и конфигурационный XML файл.

Кроме указанных выше двух основных типов источников динамической информации – очереди и сервисы, в ИС публикуется статическая информация, источником которой является текстовый конфигурационный файл. В частности, он содержит данные об организации, поддерживающей ресурс, программном и аппаратном обеспечении кластеров и др.

Веб-интерфейс Информационной системы

Веб-интерфейс Информационной Системы (ВИС) позволяет пользователю посмотреть текущую информацию о состоянии ресурсов. ВИС не является частью ИС - это просто веб-интерфейс для отображения информации о свойствах ресурсов, доступных в ЦИС.

Программная реализация ВИС представляет собой сервлет, функционирующий в рамках отдельного сервера приложений, который использует стандартные запросы свойств ресурсов к Центральному информационному сервису (ЦИС), форматирует их и отображает в виде динамических html-страниц. Разработанные алгоритмы получения данных из ИС реализованы в виде набора XSLT-преобразований.

Текущая реализация ВИС встраивается в контейнер Tomcat 6. Для обеспечения требуемого уровня безопасности выполнена настройка сервера приложений для работы с установкой защищенного SSL-соединения с использованием сертификатов удостоверяющего центра ГридННС.

В настоящее время ВИС ГридННС размещается по адресу https://cis.ngrid.ru:4443. Типичные запросы, позволяющие выяснить общую структуру грид-комплекса и получить полную информацию о состоянии сайтов, очередей и других ресурсов, вынесены в отдельные ссылки на стартовой странице. Кроме того, создан сайт, с динамически обновляемыми страницами и возможностью просмотра содержимого всех трех ЦИС, функционирующих в проекте.

Заключение

В результате проделанной работы получена устойчивая конфигурация Информационной системы, которая запущена и используется на полигоне ГридННС. По мере развития и наращивания функциональных возможностей проекта она, безусловно, будет дорабатываться и совершенствоваться, но уже сейчас это вполне работоспособная система, пригодная к эксплуатации.

Тем не менее, стоит сказать и о недостатках, присущих веб-сервисной реализации MDS. В первую очередь, это сложность и большой объем программного кода, а главное, очень сильная интеграция с Globus WS Core и другими компонентами GT4. MDS4 устанавливается вместе с GRAM4 в базовой установке и работает в Globus-контейнере. Весьма проблематично вычленить MDS из GT4 и собрать в виде независимого пакета. Это значит, что использовать MDS4 в грид-среде, где в качестве шлюза будет использоваться не WS GRAM4 вряд ли целесообразно – с одной стороны, придется ставить тяжелые библиотеки WS Core, а с другой – решать дополнительную проблему взаимодействия с клиентами и сервисами, у которых XML не является внутренним форматом данных. Также следует иметь в виду, что прекращена поддержка кода этого пакета разработчиками.

Литература

[1] Web-сайт проекта ГридННС, http://www.ngrid.ru

[2] Berkeley Database Information Index, https://twiki.cern.ch/twiki/bin/view/EGEE/BDII

[3] MDS 2.4 in the Globus Toolkit 2.4, http://www.globus.org/toolkit/docs/2.4/mds

[4] GT4.2.1: Information Services Webpage, http://www.globus.org/toolkit/docs/4.2/4.2.1/info/

[5] TeraGrid Information Servises, http://info.teragrid.org

[6] Galang G., Coddington P., Fraser R., Jones R., Sharpe A. Experiences in Deploying an MDS4 Grid Information System on the Australian National Grid, 2008. http://wiki.arcs.org.au/pub/APACgrid/PlanResource/APAC-DS4WithGLUEDeployment.pdf

[7] Web Services Resource Framework (WSRF) v1.2: OASIS WSRF Webpage, http://www.oasis-open.org/specs/index.php#wsrf

[8] RPProvider Framework (UsefulRP), http://www.globus.org/toolkit/docs/4.2/4.2.1/info/usefulrp/

[9] The GLUE Information model version 1.3, http://glueschema.forge.cnaf.infn.it/Spec/V13

[10] Andreozzi S., Burke S., Ehm F., Field L., Konya B., etc. GLUE Specification v. 2.0, 2009. http://www.ogf.org/documents/GFD.147.pdf

[11] Torque Resource Manger, http://www.clusterresources.com/pages/products/torque-resource-manager.php

[12] Веб-интерфейс ЦИС ГридННС, https://cis.ngrid.ru:4443

[13] Schopf J. et al. Monitoring and Discovery in a Web Services Framework: Functionality and Performance of Globus Toolkit MDS4, Argonne National Laboratory Technical Report ANL/MCSP12480405, 2005.

[14] Flechl M., Field L. Grid Interoperability: Joining Grid Information Systems. Journal of Physics: Conference Series 119, 2008. http://iopscience.iop.org/1742-6596/119/6/062030/pdf/1742-6596_119_6_062030.pdf

ПАРАЛЛЕЛЬНЫЕ ВЫЧИСЛЕНИЯ В ЗАДАЧАХ ДИСТАНЦИОННОЙ НАНОДИАГНОСТИКИ АТМОСФЕРЫ И ПОВЕРХНОСТИ ЗЕМЛИ ПОСЛЕ ИЗВЕРЖЕНИЯ ВУЛКАНОВ[1]

Т. А. Сушкевич, С. А. Стрелков, С. В. Максакова

Учреждение Российской академии наук
Институт прикладной математики им. М.В. Келдыша РАН, Россия, Москва
tamaras@keldysh.ru

Посвящается 100-летнему юбилею Главного Теоретика космонавтики академика М.В. Келдыша (10.02.1911-24.06.1978) и 35-летию программы «Союз-Аполлон».

М.В. Келдыш – идеолог и организатор космических исследований. **В 1955 году** для убеждения руководителей СССР в необходимости освоения космического пространства и запусков космических спутников и кораблей М.В. Келдыш выделил **две главные задачи: разведка и наблюдения Земли**, вокруг которых сформировались многие научно-исследовательские проекты. В Институте им. М.В. Келдыша один из авторов (Т.А. Сушкевич) работает с 1961 года, а с 1963 года участвует в космических исследований, в том числе в первых научных экспериментах по наблюдениям и дистанционному зондированию из космоса ореола Земли, земной поверхности и океана.

В настоящей работе речь идет о дистанционном зондировании стратосферных аэрозольных слоев и загрязнения окружающей среды, возникающих в результате извержений вулканов, мощных пожаров и последствий военных операций (война во Вьетнаме), которые учитываются при расчетах радиационных членов климатических моделей и оценках радиационного форсинга на климат. ВПЕРВЫЕ такую проблему Т.А. Сушкевич пришлось решать при математическом моделировании ореола Земли, ВПЕРВЫЕ сфотографированного с пилотируемых космических кораблей (ПКК) в июне 1963 года Валерием Федоровичем Быковским на ПКК «Восток-5» и Валентиной Васильевной Терешковой на ПКК «Восток-6»: которые ***впервые сфотографировали дневной и сумеречный горизонты Земли и провели первый научный эксперимент по исследованию Земли из космоса при участии космонавтов*** [1-11].

В 2010 году покорители космоса, научная и политическая общественность отметили **35-летие исторического события**, когда впервые в истории человечества космические корабли СССР и США осуществили сближение и стыковку, образовав единый орбитальный комплекс. 15 июля 1975 года в 15 часов 20 минут московского времени стартом корабля «Союз-19» с космодрома Байконур (СССР) начался ПЕРВЫЙ в истории пилотируемый космонавтами **международный космический полет по программе ЭПАС (Экспериментальный полет «Аполлон-Союз»).** В тот же день в 22 часа 50 минут с космодрома на мысе Канаверал (США) стартовал космический корабль «Аполлон-18» (англ. Apollo-Soyuz Test Project (ASTP)). 17 июля 1975 года, состоялась стыковка. Контактное взаимодействие «Союза» и «Аполлона»

[1] Работа выполняется при финансовой поддержке Российского фонда фундаментальных исследований (проекты 09-01-00071, 08-01-00024) и Программы фундаментальных исследований РАН (проект ПФИ № 3 (3.5) ОМН РАН).

было названо историческим и явилось прообразом будущих международных космических станций (МКС). «Союз-19» пилотировали командир корабля Герой Советского Союза, летчик-космонавт СССР полковник Алексей Архипович Леонов и бортинженер Герой Советского Союза, летчик-космонавт СССР, кандидат технических наук Валерий Николаевич Кубасов, «Аполлон» – астронавты США Томас Стаффорд, Вэнс Бранд и Дональд Слейтон.

По советской программе ЭПАС с 2 по 8 декабря 1974 года был осуществлён полёт ПКК «Союз-16» с экипажем – Анатолий Васильевич Филипченко (командир) и Николай Николаевич Рукавишников (бортинженер), на котором кроме испытаний стыковочного устройства, неоднократно испытанного и проверенного на земле, были проведены атмосферно-оптические научные эксперименты. В рамках ЭПАС на ПКК «Союз-16» и «Союз-19» ***эксперимент по наблюдениям последствий газовых и аэрозольных выбросов из вулкана и пожаров (военные действия во Вьетнаме в 1961-1975 гг.) в стратосферу*** подготовили Георгий Владимирович Розенберг и Анатолий Борисович Сандомирский, а теоретико-расчетные исследования и моделирование обеспечила Тамара Алексеевна Сушкевич [12].

Эти пионерские работы заложили фундаментальные основы в современные методы и средства дистанционного зондирования Земли из космоса и мониторинга последствий естественно-природных и антропогенных катастроф, а также подтвердили гипотезы о стратосферных аэрозольных слоях, их происхождении и релаксации.

Моделированием последствий извержений вулканов занимался академик Никита Николаевич Моисеев со своими учениками. Стратосферные аэрозольные слои вулканического происхождения обычно учитываются при расчетах радиационных членов в климатических моделях. Замети, что именно опыт, приобретенный при моделировании выбросов вулканов, оказался чрезвычайно полезным при моделировании последствий «локального ядерного удара» по мегаполису, вследствие которого возникают мощные пожары и термики, выносящие «сажу» в стратосферу и приводящие к эффекту, названному в США «ядерная ночь», и, как следствие, в итоге реализуется известный сценарий «ядерной зимы», рассчитанный в 1983 году Владимиром Валентиновичем Александровым (погиб в 1985 году). Аэрозоль, как дымная пелена, за два месяца может распространиться по всей Земле. В какой бы стране ни взорвались бомбы - все перемешается. Лучи Солнца почти не будут доходить до поверхности Земли, температура воздуха в разных местах упадет на 10-50 градусов. В США идею подобных исследований выдвинул астроном Карл Саган.

В хронологии пионерских работ советских ученых по дистанционному зондированию атмосферы и земной поверхности Земли особое место занимают достижения советской пилотируемой космонавтики, связанные с огромной ролью ПКК и ДОС (долговременные орбитальные станции) с экипажами космонавтов, которые проводили пионерские уникальные космические эксперименты в контролируемых условиях. США предпочтение отдавали искусственным спутникам Земли, работающим в автоматическом режиме.

Полет Ю.А. Гагарина 12 апреля 1961 г. на ПКК «Восток», который совершил один виток за 108 мин. вокруг Земли, - это был ***первый взгляд из космоса на Землю, т.е. первые визуальные наблюдения поверхности и ореола Земли, впервые увидели «космическую зарю».***

Полеты Г.С. Титова на ПКК «Восток-2» (август 1961 г.), А.Г. Николаева на ПКК «Восток-3» и П.Р. Поповича на ПКК «Восток-4» (август 1962 г.) расширили представления о возможностях ***визуальных наблюдений***. Г.С. Титов 6 августа 1961 г. в начале второго витка ПКК «Восток-2» ***впервые в мире провел киносъемку Земли из космоса.***

Валерий Федорович Быковский на ПКК «Восток-5» и Валентина Васильевна Терешкова на ПКК «Восток-6» (июнь 1963) ***впервые сфотографировали дневной и сумеречный горизонты (ореол) Земли - провели ПЕРВЫЙ научный эксперимент из космоса.*** Было положено ***начало инструментальным исследованиям оптически активных компонентов атмосферы***. Теоретическое обоснование этих экспериментов провел Г.В. Розенберг [5-7], а теоретико-расчетные результаты для анализа и интерпретации космических данных получила Т.А. Сушкевич. Этому достижению посвящена статья «К истории первого научного эксперимента по дистанционному зондированию Земли на пилотируемом космическом корабле» [2]. Статья посвящается также 45-летию осуществления в СССР в июне 1963 года ***ПЕРВОГО в истории земной цивилизации научного эксперимента по дистанционному зондированию Земли космонавтами на пилотируемых «кораблях-спутниках»,*** как называл космический корабль с космонавтами на борту Сергей Павлович Королев.

В этом научном эксперименте были ВПЕРВЫЕ обнаружены из космоса аэрозольные стратосферные слои, возникшие в результате мощного извержения вулкана Агунг в марте 1963 года (Гунунг Агунг – гора-вулкан на острове Бали, Индонезия).

14 апреля 2010 года началось извержение вулкана, расположенного в южной части ледника Эйяфьятлайокудль (Eyjafjallajoekull) в 200 километрах к востоку от Рейкьявика, Исландия. Это событие стало новостью номер один на длительное время и население планеты могло наблюдать визуально (в интернете и на экранах телевизоров) за развитием процесса извержения. Извержение привело к образованию большого облака пепла и вулканической пыли высотой 6 км, которое сначала понесло ветром на юго-восток к Британским островам, а затем сместилось в сторону Северной Европы и северо-западной части России, а 16 апреля достигло Москвы на высоте 10 км. Изображения со спутника TERRE/MODIS можно было увидеть на сайте NASA http://earthobservatory.nasa.gov. Государственное учреждение Научный центр аэрокосмического мониторинга АЭРОКОСМОС под руководством академика В.Г. Бондур оперативно представило космические снимки на своем сайте http://www.aerocosmos.info/ и достаточно скоро в АЭРОКОСМОС были проведены оценки шлейфа диоксида серы SO_2 для высот более 5 км. Газовые и аэрозольные выбросы достигли стратосферы и началась релаксация нового стратосферного аэрозольного слоя. ·

Вулканические продукты. *Лава* – это магма, изливающаяся на земную поверхность при извержениях, а затем затвердевающая. *Состав лавы:* твердые породы, образующиеся при остывании лавы, содержат в основном диоксид кремния, оксиды алюминия, железа, магния, кальция, натрия, калия, титана и воду. Обычно в лавах содержание каждого из этих компонентов превышает один процент, а многие другие элементы присутствуют в меньшем количестве.

Существует множество типов вулканических пород, различающихся по химическому составу. Чаще всего встречаются четыре типа, принадлежность к которым устанавливается по содержанию в породе диоксида кремния: базальт 48-53%, андезит 54-62%, дацит 63-70%, риолит 70-76%. Породы, в которых количество диоксида кремния меньше, в большом количестве содержат магний и железо. При остывании лавы значительная часть расплава образует вулканическое стекло, в массе которого встречаются отдельные микроскопические кристаллы. Исключение составляют т.н. фенокристаллы - крупные кристаллы, образовавшиеся в магме еще в недрах Земли и вынесенные на поверхность потоком жидкой лавы. Чаще всего фенокристаллы представлены полевыми шпатами, оливином, пироксеном и кварцем. Цвет вулканического стекла зависит от количества присутствующего в нем железа: чем больше железа, тем оно темнее. Таким образом, даже без химических анализов можно догадаться, что светлоокрашенная порода – это риолит или дацит, темноокрашенная - базальт, серого цвета - андезит. По различимым в породе минералам определяют ее тип. Так, например, оливин – минерал, содержащий железо и магний, характерен для базальтов, кварц - для риолитов.

Вулканические газы. После мощного извержения в верхние слои атмосферы поднимаются многие миллионы тонн вулканической пыли и газов. В основном это сернистые газы, которые образуют в стратосфере облако из мельчайших частичек серной кислоты. Газы, выделяющиеся из магмы до и после извержения, имеют вид белых струй водяного пара. Когда к ним при извержении примешивается тефра (от греч. τεφρα или *téphra — пепел, зола —* собирательный термин для отложений осевшего вулканического пепла), выбросы становятся серыми или черными. *Состав вулканических газов*: газ, выделяющийся из вулканов, на 50-85% состоит из водяного пара; свыше 10% приходится на долю углекислого газа, около 5% составляет сернистый газ, 2-5% - хлористый водород и 0,02-0,05% - фтористый водород.

Сероводород и газообразная сера обычно содержатся в малых количествах. Иногда присутствуют водород, метан и оксид углерода, а также небольшая примесь различных металлов. В газовых выделениях с поверхности лавового потока, покрытого растительностью, можно обнаружить аммиак.

Вулканические газы, выделяемые вулканами любого типа, поднимаются в атмосферу и обычно не причиняют вреда, однако частично они могут возвращаться на поверхность земли в виде кислотных дождей. Иногда рельеф местности способствует тому, что вулканические газы (сернистый газ, хлористый водород или углекислый газ) распространяются близ поверхности земли, уничтожая растительность или загрязняя воздух в концентрациях, превышающих предельные допустимые нормы. Вулканические газы могут наносить и косвенный вред. Так, содержащиеся в них соединения фтора, азота и серы захватываются пепловыми частицами, а при выпадении последних на земную поверхность заражают пастбища и водоемы, вызывая тяжелые заболевания скота. Таким же образом могут быть загрязнены открытые источники водоснабжения населения.

Вулканы и климат. Теоретические расчеты и непосредственные измерения показывают, что средняя температура на планете после мощного вулканического извержения повышается примерно на полградуса. Значит ли это, что в каждой точке земного шара зимой, летом, осенью и весной делается теплее? Чтобы ответить на этот вопрос, чтобы выяснить, как извержение меняет погодные условия отдельных регионов в различные сезоны года, необходимо оценить мощность того или иного извержения, нужны данные о количестве выброшенных в стратосферу газов, или, как принято говорить, о количестве *сернокислого аэрозоля.* На сегодняшний день существует лишь один косвенный метод «взвесить» аэрозоль. Этот метод позволяет оценить мощность даже тех извержений, которые произошли сотни лет назад. С этой целью анализируют годовые слои льда в кернах, взятых в Гренландии. Атмосферная циркуляция над центральными районами Гренландии имеет важную особенность: здесь область постоянного антициклона, индустриальные загрязнения здесь практически не влияют на состав атмосферы и осадков. Данные о кислотности определенного слоя льда, который соответствует тому или иному году, характеризуют количество сернокислого аэрозоля, который после извержения вулкана попал в стратосферу, а потом осел на поверхность льда. Естественно, что осаждение аэрозоля неравномерно по Земле и этот метод дает только приблизительные оценки последствий извержения вулканов.

Полагают, что после извержений вулканов средняя температура атмосферы Земли понижается за счет выброса мельчайших частиц (менее 0,001 мм) в виде аэрозолей и вулканической пыли (при этом сульфатные аэрозоли и тонкая пыль при извержениях попадают в стратосферу) и сохраняется таковой в течение 1–2 лет. Проведенный анализ климатологами показал, что у каждого региона своя индивидуальная реакция на извержение.

Математическое моделирование

Даже такой краткий обзор процессов загрязнений окружающей среды как последствий вулканических извержений показывает, насколько сложно идентифицировать компоненты загрязнений, а также их количественный состав и пространственное распределение. Тем более это сложно сделать оперативно с момента начала и в процессе и ближайшие сроки после извержения, когда ***нельзя взять пробы и провести измерения in situ***! На первичном этапе допустимы лишь косвенные (консервативные) методы. Уже создается и функционирует международная кооперация по организации оперативного обнаружения очагов и источников возгорания (миниспутники для наблюдения за пожарами). Аналогичные специализированные службы необходимы для мониторинга за вулканами и последствиями их извержений. Наиболее эффективным является подход, основанный на ***аэрокосмическом дистанционном зондировании атмосферы и земной поверхности*** с использованием ***гиперспектрального метода*** для идентификации компонент выбросов и загрязнений по их поглощательным или отражательным характеристикам, поскольку заранее химический состав извержения вулканов не известен. Это и есть ***нанодиагностика***.

Электромагнитное излучение, регистрируемое разными средствами, является основным источником информации о строении и физических свойствах планетных атмосфер и поверхностей при дистанционном зондировании. Для пассивных систем наблюдений источниками излучения являются внешний солнечный поток коротковолнового диапазона спектра (ультрафиолетовый, видимый, ближний инфракрасный) и собственное излучение планеты длинноволнового диапазона спектра (инфракрасный, миллиметровый), когда применимо квазиоптическое приближение теории переноса излучения.

Для космических проектов и аэрокосмических систем дистанционного зондирования и землеобзора с первых шагов освоения космического пространства необходимо было разрабатывать информационно-математическую систему и методологию решения двух основных классов многомерных задач теории переноса излучения [13-18] :

- для 3D или 2D сферической оболочки (сферическая Земля с атмосферой),
- для 3D плоского слоя (атмосфера над мозаичной земной поверхностью),

с двумя типами источников:

- внешний параллельный поток солнечного (коротковолнового) излучения,
- собственное (длинноволновое, инфракрасное) излучение.

О супервычислениях и параллельных алгоритмах

Даже на современных высокопроизводительных вычислительных системах и суперкомпьютерах разной архитектуры стоят проблемы скорости вычислений и оптимальной организации распараллеливания расчета при больших размерностях разностной сетки, а также передачи больших массивов результатов расчета по сетям от суперкомпьютера к рабочей станции оператора для последующей обработки. За основу принято численное решение краевой задачи для стационарного уравнения переноса монохроматического или квазимонохроматического излучения в рассеивающей, поглощающей, излучающей атмосфере сложной пространственной структуры, ограниченной неоднородной отражающей подстилающей поверхностью, роль которой могут играть земная поверхность (суша, океан), верхняя граница облачности или гидрометеоров (осадки).

Разработанные авторами метод функций влияния и теория передаточного оператора обладают удивительными свойствами распараллеливания вычислений и построения новых алгоритмов декомпозиции методом векторных функций влияния: исходную задачу с областью определения решения большой размерности и большим размером разностной сетки фазового пространства задачи можно факторизовать на ряд малоразмерных подзадач, определенных на подобластях и разностных сетках меньшей размерности. При этом подобласти могут отличаться радиационными режимами и в них можно использовать разные приближения и методы решения краевых задач теории переноса излучения.

Используются следующие приемы распараллеливания вычислений:

1) распределенные вычисления по физическим моделям:

- многоспектральные, в том числе гиперспектральные (по длине волны);
- по оптико-геофизической погоде (по коэффициентам общей краевой задачи);
- по источникам излучения;

2) распределенные вычисления на основе методического распараллеливания - декомпозиции краевых задач:

- по моделям переноса излучения, т.е. по приближениям теории переноса излучения;
- по подобластям;
- по параметрам функций влияния;
- по компонентам векторов функций влияния;
- по компонентам матричных функционалов – передаточных операторов;

3) алгоритмическое распараллеливание для многомерных моделей:

- однократное рассеяние по характеристикам;
- многократное рассеяние по интегралам столкновений;
- по квадрантам угловых разностных сеток;
- по подобластям с разными сеточно-характеристическими схемами.

Вычислительные модули «линеаризованы» настолько, насколько это возможно, что позволяет переносить программное обеспечение на разные типы и архитектуры многопроцессорных вычислительных кластеров, суперкомпьютеров, GRID-систем.

***Основные составные части математического обеспечения*:**

- банки данных по оптико-метеорологическим моделям атмосферы и земной поверхности (температуры, давления, влажность, концентрации компонент атмосферы и т.п.);
- система автоматизированного расчета спектро-энергетических и других радиационных характеристик атмосферы и Земли в различных диапазонах спектра от ультрафиолета (УФ) до миллиметровых волн (ММВ);
- банки данных радиационных характеристик (функции влияния локальных возмущений параметров или источников в атмосфере, дымах, облаках, гидрометеорах, океане и на земной поверхности, пространственно-угловые и спектральные распределения яркости системы Земля-атмосфера, функции пропускания и сферическое альбедо атмосферы и т.д.);
- пакеты программ обработки, визуализации и диагностики результатов численного эксперимента и аэрокосмических данных.

Библиотека программ численного решения краевых задач теории переноса излучения в рассеивающих, поглощающих и излучающих средах (атмосфера, океан, облачность, дымы, гидрометеоры, водные бассейны) составляется из набора программ на Fortran, каждая из которых позволяет рассчитывать радиационные характеристики при заданных модели и методике (краевая задача теории переноса, геометрия, численный метод и т.д.) в определенном диапазоне длин волн.

С учетом источников и процессов трансформации излучения выделяются *четыре основные физико-математические модели*, отвечающие спектральным диапазонам:

- оптический диапазон (источник - Солнце, многократное рассеяние);
- ближний ИК-диапазон (источники - Солнце и собственное излучение, многократное рассеяние);
- ИК-диапазон (источник - собственное излучение, без многократного рассеяния, сложная структура спектров поглощения);
- ММВ диапазон (источник - собственное радиоизлучение, многократное рассеяние в гидрометеорах и облаках, сложные спектры поглощения).

Программные комплексы создаваемой системы автоматизированного расчета, обработки и анализа радиационных характеристик Земли и решения задач дистанционного зондирования разрабатываются на многопроцессорных суперЭВМ с параллельными вычислениями под управлением через сеть с «рабочего места», организованного на PC.

Создаваемая система содержит *три группы программных комплексов*.

Первая группа программ - формирование оптико-метеорологических моделей среды: программы работы с архивом и базами данных моделей атмосферы, облаков, дымов, земной поверхности, океана; банк спектров поглощения атмосферных газов; банк характеристик аэрозольного рассеяния и поглощения; формирование модели атмосферы; пакеты данных к программам расчета радиационных характеристик и т.д.

Вторая группа программ - численное решение скалярного уравнения переноса излучения быстрыми приближенными и репрезентативными высокоточными методами для плоской геометрии: для системы свободная атмосфера-дымовая завеса, для системы атмосфера-океан, для системы атмосфера с многоярусными облаками, для функции влияния атмосферы, дымов, облачности, гидрометеоров, океана, для функции пропускания атмосферы, отягощенной многократным рассеянием, и т.д.

Третья группа программ - обработка и диагностика результатов расчетов: аналитическая аппроксимация и параметризация табличных функций; компьютерная графика и визуализация; решение обратных задач по восстановлению параметров среды и т.д.

Предложенная архитектура программного обеспечения с функциональным наполнением, ориентированным на решение задач мониторинга развития и оценки последствий воздействия техногенных аварий и природных катастроф, а также природно-ресурсных, экологических, геоэкоинформационных и т.п. задач, позволяет осуществлять модификацию и адаптацию вычислительно-информационной системы применительно к конкретным проблемам математического моделирования радиационных процессов в системе Земля-атмосфера (САП) или восстановления набора параметров зондируемой среды.

В настоящее время, в отличие от момента начала работ в 60-е годы, благодаря активному развитию теоретических и экспериментальных исследований по проблемам светорассеяния, а также систем космических наблюдений и актуальности тематики, в которую вовлечены более 100 стран, мы располагаем достаточно достоверными данными

- о тонкой структуре полос поглощения водяного пара и газовых компонент атмосферы и разных примесей (аэрозолей) и способах учета этих данных для математического моделирования радиационного переноса в поглощающей реальной атмосфере;

- о коэффициентах и индикатрисах рассеяния атмосферы с учетом аэрозольных примесей;

- об отражающих свойствах естественных поверхностей и разных объектов;

- о географических, сезонных, суточных распределениях, вариациях и статистических характеристиках влажности, давления, температуры, концентраций газовых и аэрозольных компонент и облачности, имеющих случайный характер и играющих основную роль в изменчивости радиационного поля Земли.

Каждая из этих моделей описывается совокупностью оптико-метеорологических (геофизических) характеристик атмосферы, облаков, подстилающей поверхности, которые являются входными физическими данными для уравнения переноса (через коэффициенты, граничные условия, источники). Степень близости расчетных полей яркости САП к реальным определяется, с одной стороны, адекватностью входных параметров фактическим, с другой стороны, - математической идеализацией процесса переноса излучения, реализованной в модели, методе, расчетном алгоритме. Современные модели достаточно адекватны.

Результаты единичных расчетов накапливаются в архивах решений, которые переоформляются в управляемые базы данных и используются в дальнейшем для расчета различных функционалов и для визуальной и графической обработки в интересах конкретных целевых приложений. Программы (вычислительные модули) для расчета «единичного» варианта реализованы на языке Fortran. Существенно, что в процессе счета варианта и при записи в архив решений используются операторы прямого доступа Fortran.

Реализация функции управления и сетевого взаимодействия «унаследованным» комплексом программ производится с помощью оболочек (wrapper's), написанных на языке описания сценариев Perl. Другими словами, производится упаковка Fortran-программ внутрь модулей на языке Perl (Perl scripts).

Заключение

Цель работы – на основе теоретико-расчетных исследований обосновать возможности новых перспективных гиперспектральных методик аэрокосмического и наземного дистанционного зондирования системы атмосфера-Земля по спектрам солнечного и собственного излучения в интересах нанодиагностики объектов окружающей среды и техносферы. Первые гиперспектрометры были использованы в 1981-1983 гг. «Спектр-256» испытывался на ДОС «Салют-7». В 80-ые годы такие работы проводились активно... В последние годы наметился всплеск интереса к подобным работам, поскольку появились новые возможности в связи с развитием нанотехнологий, новой элементной базы и усовершенствованных оптико-электронных средств регистрации излучения.

Научная идея основана на использовании существенных различий в спектральном ходе поглощения, рассеяния, излучения и пропускания основных компонент системы атмосфера-Земля и спектральных характеристик отражения объектами природно-техногенной сферы для выделения интервалов длин волн спектра многократно рассеянного солнечного и собственного излучения, информативных в отношении конкретных компонент, и на этой основе идентифицировать компоненты по их спектральным характеристикам.

Новые перспективные возможности математического моделирования атмосферной радиации Земли связаны с разработкой информационно-математической системы для широкой области приложений на суперкомпьютерах и кластерах с распараллеливанием вычислений и распределением ресурсов. В России, США, Японии, Китае, Индии, Германии, Испании, Англии, Франции, Бразилии и др. странах появились высокопроизводительные с мощными ресурсами памяти суперкомпьютеры нового поколения, ориентированные на массовый параллелизм и массовые супервычисления. Это позволяет ставить более сложные многомерные задачи, которые ближе и адекватнее натурным условиям.

Литература

[1] Сушкевич Т.А. О пионерских работах по математическому моделированию радиационного поля Земли при освоении космоса / Методы и алгоритмы обработки спутниковых данных // Современные проблемы дистанционного зондирования Земли из космоса. Физические основы, методы и технологии мониторинга окружающей среды, потенциально опасных явлений и объектов. Институт космических исследований РАН. Сборник научных статей. Выпуск 5. Том 1. М.: ООО "Азбука-2000", 2008. С. 165-180. ISSN 2070-7401.

[2] Сушкевич Т.А. К истории первого научного эксперимента по дистанционному зондированию Земли на пилотируемом космическом корабле / Вопросы создания и использования приборов и систем для спутникового мониторинга состояния окружающей среды // Современные проблемы дистанционного зондирования Земли из космоса. Физические основы, методы и технологии мониторинга окружающей среды, потенциально опасных явлений и объектов. Институт космических исследований РАН. Сборник научных статей. Выпуск 5. Том 1. М.: ООО "Азбука-2000", 2008. С. 315-322. ISSN 2070-7401.

[3] Sushkevich T.A. Pioneering remote sensing in the USSR. 1. Radiation transfer in the optical wavelength region of the electromagnetic spectrum // International Journal of Remote Sensing, Vol. 29. № 9. P. 2585-2597.

[4] Sushkevich T.A. Pioneering Remote Sensing in the USSR. 2. Global spherical models of radiation transfer // International Journal of Remote Sensing, Vol. 29. № 9. P. 2599-2613.

[5] Розенберг Г.В. О новом явлении в рассеянном свете сумеречного неба // Докл. АН СССР, 1942. Т. 36. № 9. С. 288-293.

[6] Розенберг Г.В. Сумерки. М.: ГИФМЛ, 1963. 380 с.

[7] Розенберг Г.В. О сумеречных исследованиях планетных атмосфер с космических кораблей // Изв. АН СССР. Физика атмосферы и океана, 1965. Т. 1. № 4. С. 377-385.

[8] Розенберг Г.В., Николаева-Терешкова В.В. Стратосферный аэрозоль по измерениям с космического корабля // Изв. АН СССР. Физика атмосферы и океана, 1965. Т. 1. № 4. С. 386-394.

[9] Сушкевич Т.А. Осесимметричная задача о распространении излучения в сферической системе // Труды ИПМ АН СССР. О-572-66. М.: ИПМ АН СССР, 1966. 180 с.

[10]Розенберг Г.В., Сандомирский А.Б., Сушкевич Т.А., Альтовская Н.П. Поле яркости зари, наблюдаемой с космических кораблей // Изв. АН СССР. Серия Физика атмосферы и океана, Т. 7. № 3. 1971. С. 279-290.

[11]Розенберг Г.В., Сандомирский А.Б., Сушкевич Т.А., Альтовская Н.П. Некоторые результаты фотометрических исследований дневного горизонта Земли с космических кораблей "Союз-4" и "Союз-5" // Изв. АН СССР. Серия Физика атмосферы и океана, Т. 7. № 6. 1971. С. 590-598.

[12]Розенберг Г.В., Сандомирский А.Б., Сушкевич Т.А., Матешвили Ю.Д. Исследование стратификации аэрозоля в стратосфере по программе "Союз-Аполлон" // Изв. АН СССР. Серия Физика атмосферы и океана, Т. 16. № 8. 1980. С. 861-864.

[13]Численное решение задач атмосферной оптики // Под ред. М.В. Масленникова, Т.А. Сушкевич. Научные труды ИПМ им. М.В.Келдыша АН СССР. М.: ИПМ им. М.В. Келдыша АН СССР, 1984. 234 с.

[14]Сушкевич Т.А. О решении задач атмосферной коррекции спутниковой информации // Исслед. Земли из космоса, 1999. № 6. С. 49-66.

[15]Сушкевич Т.А., Стрелков С.А., Иолтуховский А.А. Метод характеристик в задачах атмосферной оптики. М.: Наука, 1990. 296 с.

[16]Сушкевич Т.А., Максакова С.В. Обзор методов учета земной поверхности и задачах дистанционного зондирования в расчетах радиационного поля Земли – 2 // Препринт № 52-54. М.: ИПМ им. М.В.Келдыша РАН, 1999.

[17]Сушкевич Т.А. Математические модели переноса излучения. М.: БИНОМ. Лаборатория знаний, 2005. 661 с.

[18]Сушкевич Т.А., Стрелков С.А., Куликов А.К., Максакова С.В., Волкович А.Н. Глобальная сферическая модель переноса излучения в системе "Земля – атмосфера с многослойными облаками" // Современные проблемы дистанционного зондирования Земли из космоса. Физические основы, методы и технологии мониторинга окружающей среды, потенциально опасных явлений и объектов. Сборник научных статей. Выпуск 3. Том 1. М.: Российская академия наук, ИКИ РАН, ООО "Азбука-2000", 2006. С. 326-331.

ОБОБЩЕННЫЙ АУКЦИОН ВИКРИ ДЛЯ РАСПРЕДЕЛЕНИЯ ПРОЦЕССОРНОГО ВРЕМЕНИ В МНОГОМАШИННОЙ СИСТЕМЕ

А. Б. Хуторецкий[1], С. В. Бредихин[2], А. С. Белов[3]

[1] Новосибирский государственный педагогический институт,
ул. Вилюйская 28, 630126, Новосибирск, Россия, hab@dus.nsc.ru
[2] Институт вычислительной математики и математической геофизики СО РАН,
пр. Лаврентьева 6, 630090, Новосибирск, Россия, bred@nsc.ru
[3] Новосибирский государственный университет,
ул. Пирогова 2, 630090, Новосибирск, Россия, beal@ngs.ru

Введение

Спрос на платные услуги распределенных вычислительных систем (Грид-систем) растет. Чем больше интенсивность и неравномерность спроса, тем большее значение приобретает механизм распределения ресурсов системы [1]. Традиционные механизмы распределения ресурсов не учитывают готовность пользователей платить и затраты на эксплуатацию системы. Поэтому, обеспечивая вычислительную эффективность распределения, они не гарантируют его экономическую эффективность.

Грид-система, предоставляющая платные услуги, порождает рынок вычислительных ресурсов, агентами которого являются пользователи (потребители) и владельцы (поставщики) ресурсов. На этом рынке может действовать экономический механизм распределения ресурсов. Будем предполагать, что каждый агент рынка сообщает системе свой тип (потребности в ресурсах, отправные цены и пр). На основании этой информации механизм определяет распределение ресурсов и платежи. В работах [2, 3] перечислены следующие желательные свойства такого механизма.

(а) *Эффективность*: результирующее распределение должно максимизировать суммарный излишек (денежный эквивалент полезностей, полученных потребителями за вычетом затрат поставщиков).

(б) *Индивидуальная рациональность* (добровольность, условие участия): в результирующем распределении каждый агент рынка получает неотрицательную полезность.

(в) *Бюджетная сбалансированность*: механизм должен определять платежи так, чтобы поставщики в сумме получили в точности столько, сколько выплатили потребители.

(г) *Неманипулируемость* (самовыявление, совместимость стимулов): агент рынка не может получить выгоду от предоставления системе ложной информации о своем типе.

Из теоремы Майерсона–Саттерсвэйта [4] следует, что в рассматриваемой ситуации никакой механизм не удовлетворяет условиям (а) – (г) одновременно. Например, механизмы, определяющие распределение ресурсов и денег в соответствии с экономическим равновесием [5, 6], являются манипулируемыми. В этом случае агент рынка, стремясь увеличить результирующую полезность, может неверно указать свой тип, вследствие чего будет построено неэффективное распределение.

Для распределения времени одной машины можно использовать удовлетворяющий условиям (а) – (в) и неманипулируемый для потребителей механизм, предложенный в [7]. Аналогичный механизм при неделимых заданиях построен в [8]. В работе [9] для многомашинной системы предложен механизм, использующий повторный аукцион Викри (RVA). Авторы отмечают, что распределение, полученное с помощью RVA, может быть неэффективным. Обобщенный аукцион Викри (VGA) [10] удовлетворяет условиям (а), (б), (г), но не гарантирует бюджетную сбалансированность.

Чтобы построить приемлемый механизм, необходимо смягчить одно из требований (а) – (г). В [3] представлен использующий VGA механизм, который ослабляет условие (г). При выполнении условий (а) – (в) он строит платежи, минимально отличающиеся от платежей, соответствующих VGA. Мы предлагаем основанный на VGA механизм, который строит равновесное распределение ресурсов, обеспечивает выполнение условий (а) – (в) и неманипулируем для потребителей.

1. Постановка задачи

Мы рассматриваем рынок процессорного времени в периоде $[0, T]$. Пусть I и J — множества номеров потребителей и поставщиков соответственно, $I \cap J = \varnothing$; $K = I \cup J$ — множество номеров всех агентов рынка. Поставщик j владеет вычислительной машиной (с тем же номером), которая за единицу времени выполняет s_j единиц работы (например, миллионов инструкций) с удельными затратами c_j. Предоставление t единиц времени машины j некоторому потребителю эквивалентно выполнению соответствующей задачи в объеме ts_j. Поэтому вместо распределения времени в дальнейшем будем говорить о распределении объемов работ (мощности машин) между задачами.

Каждый потребитель в течение рассматриваемого периода хотел бы выполнить одну задачу. Задача может использовать различные машины, но не одновременно.

1.1. Модель потребителя

Потребитель с номером $i \in I$ имеет бюджет B_i и задачу объема Q_i (единиц работы). Вектор $\theta_i = (B_i, Q_i)$ назовем типом потребителя i. Потребительский набор пользователя i — это вектор $\mathbf{x}_i = (x_{ij} \mid j \in J)$, где x_{ij} — спрос потребителя на объем работы машины j за период T. Распределение объемов работы между потребителями задается вектором $\mathbf{x} = (\mathbf{x}_i \mid i \in I)$. Вектор $\mathbf{x}_i$ должен удовлетворять следующим условиям:

$$\sum_{j \in J} \frac{x_{ij}}{s_j} \le T; \ \sum_{j \in J} x_{ij} \le Q_i; \ x_{ij} \ge 0.$$

Первое условие следует из того, что задача в каждый момент может использовать не более чем одну машину, второе и третье — очевидны. Пусть X_i – множество потребительских наборов пользователя i, т.е. множество векторов $\mathbf{x}_i$, удовлетворяющих указанным выше условиям.

За единицу работы потребитель готов заплатить $a_i = B_i / Q_i$. Будем интерпретировать a_i как денежную оценку полезности, получаемой потребителем от единицы работы. Избыточная работа для потребителя бесполезна, поэтому денежную оценку полезности набора $\mathbf{x}_i$ (оценочную функцию потребителя i) можно записать следующим образом:

$$v_i(\theta_i, \mathbf{x}) = a_i \sum_{j \in J} x_{ij}, \text{ если} \sum_{j \in J} x_{ij} \le Q_i, \text{ в противном случае } v_i(\theta_i, \mathbf{x}) = a_i Q_i,\ i \in I. \tag{1}$$

Будем считать, что функция полезности потребителя квазилинейна (и линейна на X_i):

$$u_i(\theta_i, \mathbf{x}) = v_i(\theta_i, \mathbf{x}) - t_i,\ i \in I,$$

где t_i — плата за набор $\mathbf{x}_i$.

1.2. Модель поставщика

Поставщик с номером $j \in J$ за рассматриваемый период может поставить не более $s_j T$ единиц работы с удельными затратами c_j. Вектор $\theta_j = (c_j, s_j)$ назовем типом поставщика j. Оценочную функцию поставщика j при распределении $\mathbf{x}$ запишем следующим образом:

$$v_j(\theta_j, \mathbf{x}) = -c_j \sum_{i \in I} x_{ij}, \text{ если } \sum_{i \in I} x_{ij} \le s_j T,\ v_j(\theta_j, \mathbf{x}) = -\infty \text{ в противном случае},\ j \in J. \tag{2}$$

Функция полезности (прибыль) поставщика j имеет вид

$$u_j(\theta_j, \mathbf{x}) = v_j(\theta_j, \mathbf{x}) + t_j,\ j \in J,$$

где t_j — суммарный платеж, полученный поставщиком j.

Условие бюджетной сбалансированности (в) принимает вид

$$\sum_{i\in I} t_i = \sum_{j\in J} t_j .$$

2. Аукцион для распределения машинного времени

Как указано выше, распределение времени эквивалентно распределению мощностей входящих в систему машин между потребителями (задачами). Оценочные функции (1) и (2) полностью определяются типами агентов рынка.

2.1. Обобщенный аукцион Викри

По правилам GVA [10] каждый агент рынка объявляет аукционисту свой тип, при этом заявленный тип может отличаться от истинного. Пусть $\widetilde{\theta}_k$ для $k\in K$ — заявленные типы агентов рынка. Этим типам по формулам (1) и (2) соответствуют оценочные функции $v_k(\widetilde{\theta}_k, \mathbf{x})$.

Аукционист находит распределение $\mathbf{x}^0$, решая задачу

$$\sum_{k\in K} v_k\left(\widetilde{\theta}_k, \mathbf{x}\right) \to \max \quad \text{при } \mathbf{x}_i \in X_i \text{ для всех } i\in I. \tag{3}$$

Пусть $\widetilde{\theta}_i = (\widetilde{B}_i, \widetilde{D}_i)$ и $\widetilde{a}_i = \widetilde{B}_i / \widetilde{D}_i$ для $i\in I$, $\widetilde{\theta}_j = (\widetilde{c}_j, \widetilde{s}_j)$ для $j\in J$. Легко доказать, что задача (3) эквивалентна следующей задаче линейного программирования.

$$\sum_{i\in I}\sum_{j\in J} (\widetilde{a}_i - \widetilde{c}_j) x_{ij} \to \max \tag{4}$$

при условиях:

$$\sum_{j\in J} \frac{x_{ij}}{\widetilde{s}_j} \le T \text{ для } i\in I; \tag{5}$$

$$\sum_{j\in J} x_{ij} \le Q_i \text{ для } i\in I; \tag{6}$$

$$\sum_{i\in I} x_{ij} \le \widetilde{s}_j T \ \text{ для } j\in J; \tag{7}$$

$$x_{ij} \ge 0 \text{ для } i\in I, j\in J. \tag{8}$$

Для $k\in K$ задачу (4) – (8) с заменой I, J на $I \setminus \{k\}$ и $J \setminus \{k\}$ соответственно обозначим P_{-k}. Пусть $\mathbf{x}^0$ — оптимальное решение задачи (4) – (8), M^0_{-k} — оптимальное значение целевой функции в задаче P_{-k}. Платежи агентов рынка вычисляются следующим образом:

$$t_k = \left| \sum_{n\in K\setminus\{k\}} \widetilde{v}_n(\widetilde{\theta}_n, \mathbf{x}^0) - M^0_{-k} \right| . \tag{9}$$

Таким образом, GVA по заявленным типам агентов рынка определяет распределение ресурсов и платежи.

Механизм распределения ресурсов, использующий GVA, эффективен (по построению) и неманипулируем [10]. Поэтому можно считать, что каждый агент рынка сообщает свой истинный тип, то есть $\widetilde{\theta}_k = \theta_k$ для всех $k\in K$.

Утверждение 1. Механизм GVA индивидуально рационален, $u_k(\theta_k, \mathbf{x}^0) \ge 0$ для всех $k\in K$.

По теореме Майерсона–Саттерсвэйта [4] рассматриваемый механизм не гарантирует, вообще говоря, бюджетную сбалансированность. Легко построить примеры, в которых

$$\sum_{i\in I} t_i < \sum_{j\in J} t_j .$$

2.2. Модификация GVA

Для того, чтобы обеспечить бюджетную сбалансированность механизма, ослабим условие (г): потребуем, чтобы механизм был неманипулируем только со стороны потребителей. Заметим, что если платежи потребителей определены формулой (9), то, независимо от способа назначения платежей поставщикам, механизм эффективен и удовлетворяет условиям индивидуальной рациональности и неманипулируемости для потребителей. Поэтому можно считать, что потребители сообщают свои истинные типы, $\tilde{\theta}_i = \theta_i$ для $i \in I$.

Поставщик j, имеющий тип $\theta_j = (c_j, s_j)$, не может скрыть от системы истинное значение параметра s_j. Однако поставщику может быть выгодно сообщить завышенное значение удельных затрат. Поэтому будем считать в дальнейшем, что c_j есть отправная цена поставщика j (минимальная цена, по которой он согласен продавать ресурс).

Определим платежи поставщикам так, чтобы обеспечить для них индивидуальную рациональность при бюджетной сбалансированности. Пусть $\mathbf{x}^0 = (\,\mathbf{x}_i^0 \mid i \in I)$, $\mathbf{x}_i^0 = (\,x_{ij}^0 \mid j \in J)$.

Утверждение 2. $\sum\limits_{j \in J} c_j x_{ij}^0 \le t_i \le a_i \sum\limits_{j \in J} x_{ij}^0$ для всех $i \in I$.

Положим

$$\lambda_i = \frac{t_i - \sum\limits_{j \in J} c_j x_{ij}^0}{\sum\limits_{j \in J} (a_i - c_j) x_{ij}^0}.$$

Из утверждения 2 следует, что $\lambda_i \in [0, 1]$. Теперь определим платёж потребителя i поставщику j

$$t_{ij} = (1 - \lambda_i) c_j x_{ij}^0 + \lambda_i a_i x_{ij}^0 \tag{10}$$

и суммарный платеж поставщику j: $t_j^1 = \sum\limits_{i \in I} t_{ij}$.

Утверждение 3. Если платежи потребителя i поставщикам определены равенством (10), то справедливы следующие соотношения:

$$c_j x_{ij}^0 \le t_{ij} \le a_i x_{ij}^0 \text{ для всех } j \text{ и } t_i = \sum_{j \in J} t_{kj}\,.$$

Другими словами, платеж потребителя, определенный формулой (9), можно распределить между поставщиками так, чтобы компенсировать затраты каждого поставщика на полученный от него ресурс и не превысить готовность потребителя платить.

Утверждение 4. Описанная выше модификация GVA порождает эффективный, индивидуально рациональный, бюджетно сбалансированный и неманипулируемый для потребителей механизм распределения ресурсов.

Заключение

Из [6, теорема 1] следует, что если $\tilde{\theta}_k = \theta_k$ для всех $k \in K$, то задача (4) – (8) находит равновесное распределение ресурсов между потребителями, а цена единицы работы машины j в равновесии определяется по формуле $p_j = c_j + \pi_j$, где π_j — двойственная оценка ограничения (7). Если же платежи определены по формуле (10), то цена единицы работы машины j для потребителя i есть

$$p(i,j) = \frac{t_{ij}}{x_{ij}^0} = (1 - \lambda_i) c_j + \lambda_i a_i.$$

Следовательно, механизм, описанный в разделе 2.2, порождает равновесное распределение с неравновесными, вообще говоря, платежами.

Апробация механизма на условных примерах показала, что платежи потребителей не превышают платежи, определенные по ценам равновесия, $p(i,j) \le p_j$. Мы предполагаем, что последнее неравенство можно доказать. Это означало бы, что $p_j - p(i,j)$ есть «премия», которую система платит потребителю j за «правдивость».

Литература

[1] Broberg J., Venugopal S., Buyya R. Market-oriented Grids and Utility Computing: The state-of-the-art and future directions // J. Grid Computing, 2008, 6, 255 – 276.

[2] Schnizler B. Auction-based Resource Allocation. In: Market Oriented Grid and Utility Computing, Buyya R. and Bubendorfer K (eds.). Hoboken, NJ : Wiley, 2009. 673 p.

[3] Parkes D.C., Kalagnanam J., Eso M. Achieving Budget-Balance with Vickrey-Based Payment Schemes in Exchanges. In: Proc. 17th International Joint Conference on Artificial Intelligence (IJCAI-01), 1161 – 1168.

[4] Myerson R.B., Satterthwaite M.A. Efficient Mechanisms for Bilateral Trading // J. of Economic Theory, 1983, 28, 265 – 281.

[5] Wolski R., Plank J., Brevik J., Bryan T. Analyzing Market-Based Resource Allocation Strategies for the Computational Grid // International J. of High Performance Computing Applications, 2001, 15(3), 258 – 281.

[6] Бредихин С.В., Хуторецкий А.Б. Равновесные распределения процессорного времени при линейных функциях полезности // Сибирский журнал индустриальной математики, 2010, т. XIII, № 2(42), 46 – 53.

[7] Lazar A., Semret N. Auctions for network resource sharing. CTR Technical Report: CU/CTR/TR 468-97-02. Center for Telecommunications Research, Columbia Univ., 1997.

[8] Хуторецкий А.Б., Бредихин С.В., Голдобин И.А. Распределение процессорного времени посредством последовательного аукциона второй цены при неделимых спросах. В сб.: Параллельные вычислительные технологии (ПаВТ'2008): Труды международной научной конференции (Санкт-Петербург, 28 января – 1 февраля 2008 г.). Электронный ресурс (1 CD), ISBN 978-5-696-03720-2. Челябинск: Изд. ЮУрГУ, 2008.

[9] Regev O., Nisan N., The POPCORN market: Online markets for computational Resources // Decision Support Systems, 2000, 28(1-2), 177 – 189.

[10] Varian H.R. Economic Mechanism Design for Computerized Agents. In: Proceedings of the Usenix workshop on electronic commerce. NY, 1995.

[11] Gray J. Distributed Computing Economics. In: Computer Systems: Theory, Technology, and Applications. NY: Springer-Verl., 2003, pp. 93-101.

[12] Khutoretsky A., Bredikhin S. Distributions and schedules of CPU time in a multiprocessor system when the user's utility functions are linear. In: Parallel Computing Technologies (Lecture Notes in Computer Science, 5698). Berlin, Heidelberg: Springer, 2009, pp. 316–320.

[13] Данилов В.И., Сотсков А.И. Механизмы группового выбора. М.: Наука, 1991.

INDEX

Kekelidze D.	Laboratory of Information Technologies, JINR	Dmitry.Kekelidze@cern.ch	133
Kerpl L.	Institute of Physics of the AS CR, Prague, Czech Republic		125
Kholodkov K.I.	Institute of physics of the Earth RAS, Moscow, Russia	keir@ifz.ru	30
Kholmurodov Kh.T.	Joint Institute for Nuclear Research	mirzo@jinr.ru	90
Kiryanov A.K.	Petersburg Nuclear Physics Institute, Gatchina, Orlova Roscha, Russia	globus@pnpi.nw.ru	121, 137
Klopov N.V.	Petersburg Nuclear Physics Institute, Gatchina, Orlova Roscha, Russia	knv@omrb.pnpi.spb.ru	121, 137
Kodolova O.	Skobeltsyn Institute of Nuclear Physics, Moscow State University, Moscow, Russia		103
Korenkov V.V.	Laboratory of Information Technologies, JINR	korenkov@cv.jinr.ru	13, 25, 39, 43, 103, 109, 142, 148
Korkhov V.V.	St.Petersburg State University	vladimir@csa.ru	153
Koryagin V.N.	Institute of physics of the Earth RAS, Moscow, Russia	admin@ifz.ru	30
Kouba T.	Institute of Physics, Academy of Sciences of the Czech Republic	koubat@fzu.cz	159
Krejcova M.	Institute of Physics of the AS CR, Prague, Czech Republic		125
Kryukov A.P.	Skobeltsyn Institute of Nuclear Physics Lomonosov Moscow State University (SINP MSU), Moscow, Russia	kryukov@theory.sinp.msu.ru	191, 215
Kukushkin A.B.	Tokamak Physics Institute RRC "Kurchatov Institute", Moscow, Russia	kukushkin@nfi.kiae.ru	177
Kundrat J.	Institute of Physics of the AS CR, Prague, Czech Republic	kundratj@fzu.cz	125
Kutovskiy N.	Laboratory of Information Technologies, JINR	Nikolay.Kutovskiy@jinr.ru	39, 90, 148
La Min Htut	Saint-Petersburg state university	laminhtut@mail.ru	63, 69
Lazarev A.A.	Institute for High-performance Computing and Information Systems (IHPCIS), Saint-Petersburg, Russia		63, 69
Lensky I.	Laboratory of Information Technologies, JINR		43
Levin M.P.	Advanced Software Group, Samsung Research Center, Moscow, Russia	m.levin@samsung.com	33
Lokajicek M.	Institute of Physics, Academy of Sciences of the Czech Republic	lokajick@fzu.cz	159

Lovas R.	MTA SZTAKI, Budapest, Hungary	rlovas@sztaki.hu	165
Marusov N.L.	Tokamak Physics Institute RRC "Kurchatov Institute", Moscow, Russia		177
Matveev M.	Laboratory of Information Technologies, JINR		43
Matveeva E.	Laboratory of Information Technologies, JINR		43
Misiorny M.	Faculty of Physics, Adam Mickiewicz University, Poznan, Poland		173
Mitsyn S.V.	Laboratory of Information Technologies, JINR	svm@jinr.ru	43, 47
Mitsyn V.V.	Laboratory of Information Technologies, JINR	vvm@mammoth.jinr.ru	25
Musial G.	Faculty of Physics, Adam Mickiewicz University, Poznan, Poland	gmusial@amu.edu.pl	173
Musulmanbekov G.	Laboratory of Information Technologies, JINR	genis@jinr.ru	93
Myo Tun Tun	Saint-Petersburg state university	bonge21@mail.ru	63, 69
Nechaevskiy A.	Laboratory of Information Technologies, JINR	nechav@mail.ru	39
Nefedov Yu.A.	Joint Institute for Nuclear Research		81
Neukirchen H.	University of Iceland, Reykjavik, Iceland	helmut@hi.is	98
Neverov V.S.	Tokamak Physics Institute RRC "Kurchatov Institute", Moscow, Russia	vs-never@hotmail.com	177
Oleshko S.B.	Petersburg Nuclear Physics Institute, Gatchina, Orlova Roscha, Russia	oleshko@pnpi.nw.ru	121
Oleynik D.	Laboratory of Information Technologies, JINR	danila@jinr.ru	43
Petrosyan A.	Laboratory of Information Technologies, JINR	artem.petrosyan@jinr.ru	43
Placinta C.	Department of Elementary Particles and Information Technologies, National Institute of R&D for Physics and Nuclear Engineering 'Horia Hulubei', Magurele, Romania		183
Popeneciu G.	National Institute for Research and Development of Isotopic and Molecular Technology, Cluj-Napoca, Romania		25
Prikhodko N.V.	Yaroslav-the-Wise Novgorod State University (NovSU), Velikiy Novgorod, Russia	niko2004x@mail.ru	191
Ryabov Yu.F.	Petersburg Nuclear Physics Institute, Gatchina, Orlova Roscha, Russia	ryabov@pnpi.nw.ru	121, 137
Sayzhenkova S.S.	Institute for High Energy Physics, Moscow region, Protvino, Russia	Sofia.Sayzhenkova@ihep.ru	196
Schmelling M.	Max-Planck-Institute for Nuclear Physics, Heidelberg, Germany	michael.schmelling@mpi-hd.mpg.de	98

Semenov I.B.	Tokamak Physics Institute RRC "Kurchatov Institute", Moscow, Russia	i.semenov@iterrf.ru	177
Semenov R.	Laboratory of Information Technologies, JINR	roman@jinr.ru	43
Shabratova G.	Laboratory of High Energy Physics, JINR	G.Chabratova@cern.ch	202
Shamardin L.	Skobeltsyn Institute of Nuclear Physics Lomonosov Moscow State University (SINP MSU), Moscow, Russia	shamardin@theory.sinp.msu.ru	215
Sherstnev A.	Scobeltsyn Institute of Nuclear Physics, Moscow State University, Moscow, Russia, R. Peierls Centre for Theoretical Physics, University of Oxford, Oxford, UK		133
Shiyakova M.M.	Joint Institute for Nuclear Research	maria@jinr.ru	81
Shmatov S.V.	Joint Institute for Nuclear Research	shmatov@cern.ch	103, 109
Shogin A.N.	All-Russian Institute for Scientific and Technical Information RAS, Moscow, Russia	alex@viniti.ru	30
Shuvalov A.	Saint-Petersburg state university	anatoly.shuvalov@gmail.com	75
Smirnov S.A.	Moscow Institute of Physics and Technology, Dolgoprudny, Russia	sasmir@gmail.com	230, 257
Smirnova O.	NDGF / Lund University	oxana.smirnova@hep.lu.se	220
Soe Moe Lwin	St.Petersburg State Marine Technical University	mogokthar@gmail.com	51, 75
Stankevich V.G.	Kurchatov Center for Synchrotron Radiation and Nanotechnology, RRC "Kurchatov Institute", Moscow, Russia		177
Strizh T.A.	Laboratory of Information Technologies, JINR	strizh@jinr.ri	13
Sukhoroslov O.V.	Centre for Grid Technologies and Distributed Computing, Institute for Systems Analysis, RAS, Moscow, Russia	os@isa.ru	236
Svec J.	Institute of Physics, Academy of Sciences of the Czech Republic		159
Svechnikov N.Yu.	Kurchatov Center for Synchrotron Radiation and Nanotechnology, RRC "Kurchatov Institute", Moscow, Russia		177
Tarasov A.S.	Institute for System Analysis RAS, Moscow, Russia	tarasov.alexey@gmail.com	177
Thurein Kyaw Lwin	St.Petersburg State Marine Technical University	phothar83@mail.ru trkl.mm@mail.ru	51, 75
Tikhonenko E.	Laboratory of Information Technologies, JINR	eat@cv.jinr.ru	103
Trusca R.	National Institute for Research and Development of Isotopic and Molecular Technology, Cluj-Napoca, Romania		25
Uzhinskiy A.	Laboratory of Information Technologies, JINR	zalexandr@list.ru	39
Vaniachine A.V.	Argonne National Laboratory, USA	vaniachine@anl.gov	241

Богатенков П.П.	Институт математики и информатики ФНМ, Ассоциация RENAM, Кишинев, Молдова		279
Бредихин С.В.	Институт вычислительной математики и математической геофизики СО РАН, Новосибирск, Россия	bred@nsc.ru	438
Бунецкий О.О.	Национальный научный центр Харьковский физико-технический институт (ННЦ ХФТИ),Украина	kipt@kipt.kharkov.ua	303
Варламов Д.А.	Институт проблем химической физики РАН, Институт экспериментальной минералогии РАН, Черноголовка, Россия	dima@iem.ac.ru	324, 329
Васюкова Е.В.	Университет «Дубна», Дубна, Россия		279
Вдовикин О.И.	МСЦ РАН, Москва, Россия		406
Вовченко А.Е.	Институт проблем информатики РАН, Москва, Россия	itsnein@gmail.com	309
Войтиков К.Ю.	Филиал Кемеровского государственного университета в г. Анжеро-Судженске, Россия	kost_v@ngs.ru	319
Волохов А.В.	Институт проблем химической физики РАН, Черноголовка, Россия	vav@icp.ac.ru	324, 329
Волохов В.М.	Институт проблем химической физики РАН, Черноголовка, Россия	vvm@icp.ac.ru	324, 329
Волошинов В.В.	Институт системного анализа РАН, Москва, Россия, МФТИ	vv_vol@isa.ru	290
Гостев И.М.	Лаборатория информационных технологий ОИЯИ	igostev@gmail.com	334, 340
Григорьева М.В.	Научный центр РАН в Черноголовке		345
Демичев А.П.	НИИ ядерной физики им. Д.В.Скобельцына, МГУ им. М.В.Ломоносова, Москва, Россия	demichev@theory.sinp.msu.ru	352
Добрецов В.Ю.	РНЦ «Курчатовский институт»		357
Журавлёв Е.Е.	Физический институт РАН им. П.Н.Лебедева, Москва, Россия	euzhurav@pluton.lpi.troitsk.ru	364
Зайцев Е.И.	Московский государственный университет приборостроения и информатики, Москва, Россия	zei@tsinet.ru	372
Зуб С.С.	Национальный научный центр Харьковский физико-технический институт (ННЦ ХФТИ),Украина		303
Ильин В.А.	НИИ ядерной физики им. Д.В.Скобельцына, МГУ им. М.В.Ломоносова, Москва, Россия	ilyin@sinp.msu.ru	352, 357
Калинина Г.А.	Московский энергетический институт (технический университет), Москва, Россия	KalininaGA@mpei.ru	376
Калиниченко Л.А.	Институт проблем информатики РАН, Москва, Россия	leonidk@synth.ipi.ac.ru	284, 309

Посыпкин М.А.	Институт системного анализа РАН, МФТИ, Москва, Россия	mposypkin@mail.ru	284, 290
Приставка А.С.	Национальный научный центр Харьковский физико-технический институт (ННЦ ХФТИ),Украина	alexander@kipt.kharkov.ua	303
Радченко Т.Д.	Московский государственный институт электроники и математики, Москва, Россия	tanyar1987@mail.ru	340
Родин А.С.	Институт прикладной математики им. М.В.Келдыша РАН, Москва, Россия	rals@bk.ru	297
Роженцов А.А.	Марийский государственный технический университет, кафедра радиотехнических и медико-биологических систем, Россия	krtmbs@marstu.net	399
Рябов Ю.Ф.	Петербургский институт ядерной физики им. Б.П.Константинова, Гатчина, Россия	ryabov@pnpi.nw.ru	357
Савин Г.И.	МСЦ РАН, Москва, Россия		406
Седова Д.В.	Университет «Дубна», Дубна, Россия	dsedova@yandex.ru	410
Секриеру Г.В.	Ассоциация RENAM, Кишинев, Молдова	secrieru@cc.acad.md, secrieru@renam.md	279
Семёнов Д.В.	ФГУП «НИИ «Квант», Москва, Россия		406
Соболев С.И.	НИВЦ МГУ им. М.В. Ломоносова	sergeys@parallel.ru	417
Сорока Д.В.	Национальный научный центр Харьковский физико-технический институт,Украина		303
Степанова М.М.	Санкт-Петербургский государственный университет, физический факультет, кафедра вычислительной физики	mstep@mms.nw.ru	383, 423
Стесик О.Л.	Санкт-Петербургский государственный университет, физический факультет, кафедра вычислительной физики	stes@mms.nw.ru	423
Стрелков С.А.	Институт прикладной математики им. М.В.Келдыша РАН, Москва, Россия		429
Ступников С.А.	Институт проблем информатики РАН, Москва, Россия	ssa@synth.ipi.ac.ru	284, 309
Сурков Н.Ф.	Институт проблем химической физики РАН, Черноголовка, Россия		324, 329
Сухомлин В.А.	МГУ им. М.В. Ломоносова, Москва, Россия	sukhomlin@mail.ru	284
Сухорослов О.В.	Институт системного анализа РАН, Москва, Россия, МФТИ	oleg.sukhoroslov@gmail.com	284, 290
Сушкевич Т.А.	Институт прикладной математики им. М.В.Келдыша РАН, Москва, Россия	tamaras@keldysh.ru	429
Телегин П.Н.	МСЦ РАН, Москва, Россия		406
Тумаев П.Н.	Филиал Кемеровского государственного университета в г. Анжеро-Судженске, Россия	pavel.tumaev@gmail.com	319

Научное издание

Distributed Computing and Grid-Technologies in Science and Education

Proceedings of the Fourth International Conference

Распределенные вычисления и грид-технологии в науке и образовании

Труды четвертой международной конференции

Д11-2010-140

Ответственная за подготовку сборника к печати *Т. А. Стриж.*

Сборник отпечатан методом прямого репродуцирования
с оригиналов, предоставленных оргкомитетом.

Подписано в печать 08.02.2011.
Формат 70 × 100/16. Бумага офсетная. Печать офсетная.
Усл. печ. л. 36,44. Уч.-изд. л. 44,07. Тираж 200 экз. Заказ № 57211.

Издательский отдел Объединенного института ядерных исследований
141980, г. Дубна, Московская обл., ул. Жолио-Кюри, 6.
E-mail: publish@jinr.ru
www.jinr.ru/publish/