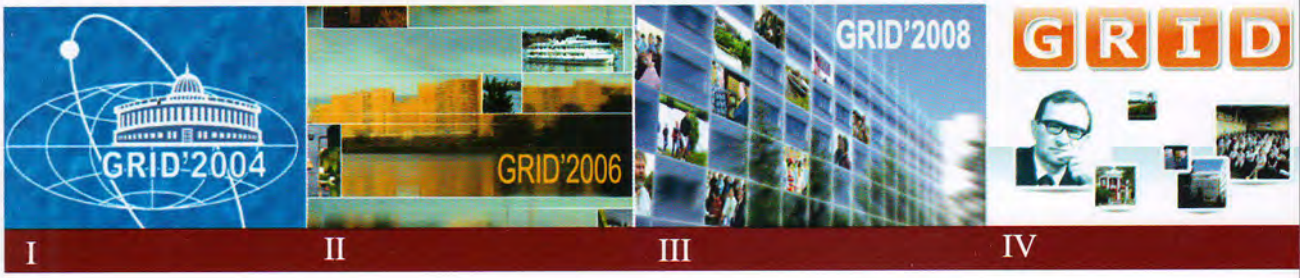
DISTRIBUTED COMPUTING AND GRID-TECHNOLOGIES IN SCIENCE AND EDUCATION

Proceedings of the 5th International Conference

Dubna, July 16–21, 2012



РАСПРЕДЕЛЕННЫЕ ВЫЧИСЛЕНИЯ И ГРИД-ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ

Труды 5-й международной конференции

Дубна, 16–21 июля 2012 г.

GRID'2012

PARALLEL.RU

РФФИ

NIAGARA
Distribution Company

SUPERMICRO
SUPER MICRO COMPUTER INC.

Joint Institute for Nuclear Research
Laboratory of Information Technologies

DISTRIBUTED COMPUTING AND GRID-TECHNOLOGIES IN SCIENCE AND EDUCATION

Proceedings of the Fifth International Conference

Dubna, July 16–21, 2012

РАСПРЕДЕЛЕННЫЕ ВЫЧИСЛЕНИЯ И ГРИД-ТЕХНОЛОГИИ В НАУКЕ И ОБРАЗОВАНИИ

Труды пятой международной конференции

Дубна, 16–21 июля 2012 г.

Дубна • 2012

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

Media partner
PARALLEL.RU

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 Fifth International Conference (Dubna, July 16–21, 2012). — Dubna: JINR, 2012. — 419 p.

ISBN 978-5-9530-0345-2

The Proceedings of the 5th International Conference «Distributed Computing and Grid-Technologies in Science and Education» (GRID'2012) include the reports presented at the GRID'2012, which was held in Dubna on July 16–21, 2012. The Conference is held every two years by the JINR Laboratory of Information Technologies. This is a unique conference conducted in Russia on the issues relating to the use of Grid-technologies in various areas of science, education, industry and business. The tentative timetable of this conference foresees in-depth analysis of hot topics such as the development of the global grid-infrastructure providing the optimal use of computing and data storage resources through national grid-infrastructures (National Grid Initiatives, NGIs) as well as the development of the Russian Grid Network. Special sections have been devoted to the emerging direction of the cloud computing and the desktop grids. Presentations of the Conference are available at the Conference web-page http://grid2012.jinr.ru/program.php

Распределенные вычисления и грид-технологии в науке и образовании: Труды пятой международной конференции (Дубна, 16–21 июля 2012 г.). — Дубна: ОИЯИ, 2012. — 419 с.

ISBN 978-5-9530-0345-2

Труды 5-й международной конференции «Распределенные вычисления и грид-технологии в науке и образовании» (GRID'2012) содержат доклады, представленные на «GRID'2012», которая проходила 16–21 июля 2012 г. в Дубне. Конференция проводится раз в два года Лабораторией информационных технологий ОИЯИ. Это единственная в России конференция, посвященная проблемам, связанным с использованием грид-технологий в различных областях науки, образования, промышленности и бизнеса. На конференции было проанализировано развитие глобальной грид-инфраструктуры, обеспечивающей оптимальное использование вычислительных ресурсов и ресурсов хранения данных, на основе национальных грид-инфраструктур (National Grid Initiatives, NGIs) и развитие российской грид-сети. Специальные секции были посвящены «облачным» вычислениям и грид-системам из персональных компьютеров. Презентации докладов, представленные на конференции, размещены на веб-странице конференции http://grid2012.jinr.ru/program.php

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

ISBN 978-5-9530-0345-2

General Information

The 5th International Conference "Distributed Computing and Grid-technologies in Science and Education" was held from 16 – 21 July, 2012 in the Laboratory of Information Technologies of the Joint Institute for Nuclear Research (Dubna, Russia). This event was the fifth one held by LIT in this subject area every two years. The program of the Conference included not only the questions related to creation and operation of Grid-infrastructures and Grid-applications, but also some theoretical and practical aspects of utilizing distributed computing environments, distributed data processing, etc. This time the heightened interest to the Conference was aroused by the creation in Russia of a Tier1 level data-processing centre at JINR and at NRC "Kurchatov Institute" as well as by vigorous activity in applying the so-called "cloud computing". The Conference was attended by 256 participants from 22 countries: Azerbaijan, Belarus, Bulgaria, Great Britain, Germany, Georgia, Italy, Kazakhstan, China, Cuba, Moldova, Mongolia, Myanmar, Russia, Romania, USA, Uzbekistan, Ukraine, France, Czechia, Switzerland, Sweden as well as CERN and JINR. Russia was presented by participants from 40 universities and research centers. The Conference program included daily plenary sessions and 8 sections: Grid-infrastructures, Clouds and Grid, Grid-applications, Desktop grids, Systems of distributed information resources, WLCG (Worldwide LHC Computing Grid), GridNNN (Grid of the National Nanotechnology Network), Distributed computing: methods and algorithms as well as poster presentations. A workshop on computing for ATLAS experiment and a round-table discussion devoted to the creation of a Tier-1 level data processing centre for LHC experiments in Russia were organized in frames of the Conference.

Advisory Committee

Abdinov O. (IoP, Baku, Republic of Azerbaijan), Afanasiev A.P. (ISA RAS, Moscow, Russian Federation), Antoniou I. (Aristotle University of Thessaloniki, Greece), Andreeva J. (CERN), Bird I. (CERN), Bogatencov P. (RENAM, Moldova), Bogdanov A.V. (IHPCIS, St.Petersburg, Russian Federation), Burtebaev N. (Institute of Nuclear Physics, Kazakhstan), Buša J. (TU, Košice, Slovakia), Buzatu F. (Institute for Atomic Physics, Magurele, Romania), Chetverushkin B.N. (Keldysh Institute of Applied Mathematics, Moscow), De K. (University of Texas at Arlington, USA), Dimitrov V. (Sofia University, Republic of Bulgaria), Dulea M. (IFIN-HH, Romania), Elizbarashvili A. (Tbilisi State University, Georgia), Gusev V.V. (IHEP, Protvino, Russian Federation), Ilyin V.A. (SINP MSU / NRC "Kurchatov Institute", Moscow, Russian Federation), Kitowski J. (CYFRONET, Republic of Poland), Klementov A. (BNL, USA), Kryukov A.P. (SINP MSU / NRC "Kurchatov Institute", Moscow, Russian Federation), Kudrov N.I. (Ministry of Communications and Mass Media of the Russian Federation), Kudryashov N.A. (NRNU MEPhI, Moscow, Russian Federation), Lakhno V.D. (IMPB Russian Academy of Sciences, Russian Federation), Lokajicek M. (PI AS CzR, Czech Republic), Matveev V.A. (JINR), 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), Ratnikova N. (Karlsruhe Institute of Technology, Germany), Ryabov Yu.F. (PNPI, Gatchina, Russian Federation), Sahakyan V.G. (IIAP NAS, Armenia), Savin G.I. (JSCC RAS, Moscow, Russian Federation), Shumeiko N.M.(NC PHEP, Minsk, Republic of Belarus), Smirnova O.G. (NDGF/University of Lund, Sweden), Vaniachine A.V. (Argonne National Laboratory, USA), Velikhov V.E. (NRC "Kurchatov Institute", Moscow, Russian Federation), Voevodin V.V. (SRCC Moscow State University, Russian Federation), Zhizhin M.N. (CGDS RAS, Russian Federation), Zhizhchenko A.B. (JSCC RAS, Moscow, Russian Federation), 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., Korotchik O.M., Lukyanov S.O., Podgainy D.V., Prikhodko A.V., Rudneva E.M., Rumyantseva O.Yu., Streltsova O.I., Tikhonenko E.A., Torosyan Sh., Zrelov P.V.

Общая информация

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 16 по 21 июля 2012 г. Настоящая конференция была пятой по данной тематике. Программа конференции включала не только вопросы, связанные с созданием и эксплуатацией грид-инфраструктур и грид-приложений, но и теоретические и практические аспекты применения распределенных вычислительных сред, распределенной обработки данных и др. В этом году повышенный интерес к конференции был связан с созданием в России центра обработки данных уровня Tier1 на базе ОИЯИ и НИЦ «Курчатовский институт» и активной деятельностью в области применения «облачных вычислений». Конференция собрала 256 участников из 22 стран: Азербайджана, Белоруссии, Болгарии, Великобритании, Германии, Грузии, Италии, Казахстана, Китая, Кубы, Молдавии, Монголии, Мьянмы, России, Румынии, США, Узбекистана, Украины, Франции, Чехии, Швейцарии, Швеции, а также ЦЕРНа и ОИЯИ. Россия была представлена участниками из 40 университетов и исследовательских центров. На конференции была организована работа ежедневных пленарных заседаний и 8 секций: Грид-инфраструктуры, «облачные» вычисления и грид, Грид-приложения, Грид-системы из персональных компьютеров, системы распределенных информационных ресурсов, WLCG – Всемирный грид для обработки данных с Большого адронного коллайдера в ЦЕРН, ГридННС – грид национальной нанотехнологической сети, распределенные вычисления (методы и алгоритмы) и стендовые доклады. В рамках конференции было проведено рабочее совещание по компьютингу для эксперимента ATLAS и круглый стол, посвященный созданию в России центра обработки данных уровня Tier-1 для экспериментов на БАК.

Программный комитет

Абдинов О. (Институт физики, Баку, Азербайджанская Республика), Афанасьев А.П. (ИСА РАН, Москва, Россия), Антониу И. (Университет имени Аристотеля, Салоники, Греция), Андреева Ю. (ЦЕРН), Берд Я. (ЦЕРН), Богатенков П. (RENARM, Молдова), Богданов А.В. (ИВВИС, Санкт-Петербург, Россия), Буртебаев Н. (Институт ядерной физики, Казахстан), Буша Я. (Технический университет, Кошице, Словакия), Бузату Ф. (IFIN-HH, Румыния), Ваняшин А.В. (Аргоннская национальная лаборатория, США), Велихов В.Е. (НИЦ «Курчатовский институт, Россия), Воеводин В.В. (НИВЦ МГУ, Россия), Гусев В.В. (ИФВЭ, Протвино, Россия), Де К. (Техасский университет, Арлингтон, США), Димитров В. (Софийский университет, Республика Болгария), Дулеа М. (IFIN-HH, Румыния), Жижин М.Н. (Центр геофизических данных РАН, Россия), Жижченко А.Б. (Межведомственный Суперкомпьютерный Центр, Россия), Зиновьев Г. (ИТФ, Киев, Украина), Ильин В.А. (НИИЯФ МГУ/НИЦ «Курчатовский институт, Россия), Китовски Я. (CIFRONET, Республика Польша), Климентов А. (БНЛ, США), Крюков А.П. (НИИЯФ МГУ/НИЦ «Курчатовский институт, Россия), Кудров Н.И. (Минкомсвязь, Россия), Кудряшов Н.А. (НИЯУ «МИФИ», Москва, Россия), Лахно В.Д. (ИМПБ РАН, Россия), Локайчик М.(Институт физики, Чехия), Матвеев В.А. (ОИЯИ), Метакидис Г. (Университет Патрас, Греция), Мусял Г. (ИФ Унив. Им. А.Мицкевича, Познань, Польша), Нгуен Мань Шат (ОИЯИ), Нэргуй Б. (Институт информатики МАН, Монголия), Ратникова Н. (Институт технологий Карлсруэ, Германия), Рябов Ю.Ф. (ПИЯФ РАН, Россия), Саакян В.Г. (ИПИА, Армения), Савин Г.И. (Межведомственный Суперкомпьютерный Центр, Россия), Смирнова О.Г. (NDGF/Университет Лунда, Швеция), Четверушкин Б.Н. (ИПМ им. М.В.Келдыша, Москва, Россия), Шумейко Н.М. (НЦ ФЧВЭ БГУ, Минск, Белоруссия), Элизбарашвили А. (Тбилисский государственный университет, Грузия).

Организационный комитет (ОИЯИ)

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

CONTENTS / СОДЕРЖАНИЕ

IMAGE PROCESSING BY CORE CLUSTERIZATION ALGORITHM

F.T. Adilova, R.R. Davronov
Laboratory of Medical Informatics,
Institute of Mathematics of National University of Uzbekistan,
100125, Tashkent, Uzbekistan

A graph-based approach to image segmentation is presented which can be applied to either grayscale or color images. The assumption is that nearby pixels with similar colors or grayscale intensities may belong to the same region or segment of the image. A graph representation for an image is derived from the similarity between the pixels, and then partitioned by a computationally efficient graph clustering method, which first identifies representative nodes for each cluster and then expands them to obtain complete clusters of the graph. A comparison with the well known normalized cut method shows that this approach can be faster and produces segmentations that are in better agreement with visual assessment of the original images.

Introduction

Image segmentation is a challenging problem in computer vision and usually applied for finding objects and their boundaries in images. Depending on the objectives, many definitions and criteria have been proposed and employed for image segmentation. Here, a problem of segmenting a digital image into a set of disjoint regions such that each region is composed of nearby pixels with similar colors or intensities, is considered.

An image can be represented by a proximity graph in which nodes represent image pixels and edges reflect pair wise similarities between the pixels. Weights of edges are computed by a similarity function of properties of corresponding pixels such as location, brightness and color. Considering that, the segmentation task can be solved by graph clustering methods. Let us we have an undirected graph G = (*V*, *E*, *W*), where the set of nodes *V* represents a set of data objects, the set of edges *E* represents relationships between data objects, and *W* is a symmetric matrix where the entry $w_{ij} \in [0,1]$ is the weight of the edge between nodes *i* and *j*. As *G* is a proximity graph, the edge weight w_{ij} represents the degree of similarity between the objects corresponding to *i* and *j*, a higher value of w_{ij} implies a higher similarity degree between *i* and *j*. In graph clustering, a graph is partitioned into subgraphs such that nodes of a subgraph are strongly or densely connected while nodes belonging to different subgraphs are weakly or sparsely connected. Therefore each subgraph corresponds to a group of similar objects, which are dissimilar to objects of groups corresponding to other subgraphs.

In this paper, we give the results of computing of the clustering method described in [1] to segmenting grayscale and color images in two variants, - sequential and parallel modes. The results of computing were compared with the known normalized cut method**.**

Image segmentation method and algorithm

In task of image segmentation we group nearby pixels that have a similar intensity/ color, the weights of graph edges are computed by a likelihood function based on the location and intensity/color of neighboring pixels. For grayscale images, we use the weight function described in [2]:

$$w_{ij} = \begin{cases} e^{-\left(\frac{I(i)-I(j)}{\sigma_I}\right)^2 - \left(\frac{dist(i,j)}{\sigma_d}\right)^2} & \text{if } dist(i,j) < r, \\ o & \text{otherwise,} \end{cases} \quad (1)$$

where $I(i) \in [0,1]$ is the intensity of pixel *i*, *dist*(*i*, *j*) is the Euclidean distance in pixels between *i* and *j*. For color images, we replace the difference of intensity in (1) by the normalized Euclidean distance between pixel colors:

$$w_{ij} = \begin{cases} e^{-\left(\frac{\|C(i)-C(j)\|_2}{\sqrt{3}\sigma_I}\right)^2 - \left(\frac{dist(i,j)}{\sigma_d}\right)^2} & \text{if } dist(i,j) < r, \\ 0 & \text{otherwise}, \end{cases} \quad (2)$$

where $C(i)$ is a vector of three features, - red, green, and blue color components of pixel i, so $\|C(i) - C(j)\|_2 \in [0, \sqrt{3}]$.

There is an edge linking two nodes only if the distance between the corresponding pixels is less than r pixels, so each node is connected to approximately πr^2 other nodes. The proximity graph is very sparse as $\pi r^2 << |V|$. By using these weight functions, strong edges exist between nodes whose corresponding pixels have a similar intensity/color and are close to each other. Therefore, pixels inside a segment with a homogeneous intensity/color (i.e., the inner or core region of a segment) have their nodes in the proximity graph strongly connected. On the other hand, pixels at boundaries of segments often have neighbor pixels with dissimilar intensity/color, so their corresponding nodes are usually weakly connected to one another. Using the weight functions (1) and (2) above, one can easily evaluate and validate segmentation results by visual inspection.

To group image pixels, we apply the coring method [1], which is a general clustering method to find clusters in an arbitrary proximity graph. The main idea is that cores of clusters should be obtainable by analysis of neighborhood relationships between objects in a particular space which reflects object similarities. In most practical problems, direct analysis is unrealistic due to the high dimensionality of the space, but a heuristic allows reducing this to a one-dimensional ordinal sequence of density variation for a set of objects contained in a graph G. In the coring method, a local density at node i with respect to $H \subseteq V$ is measured by the function $d(i, H)$:

$$d(i,H) = \frac{1}{|H|} \sum_{j \in H} w_{ij} \quad (3)$$

The minimum density of H is defined by function $D(H)$:

$$D(H) = \min_{i \in H} d(i,H) \quad (4)$$

The node $m = \arg\min_{i \in H} d(i,H)$. is called the weakest node of H as it has the minimum density. The method computes the variation of minimum density D values while the weakest node is iteratively removed from the graph. If clusters of the graph have a dense core, we can identify nodes belonging to cluster cores by analyzing the sequence of D values. Specifically, if there is a significant drop in D value after the removal of a node, this node is highly connected with other nodes in a dense region and it is potentially a core node because its elimination drastically reduces the density of the region around it.

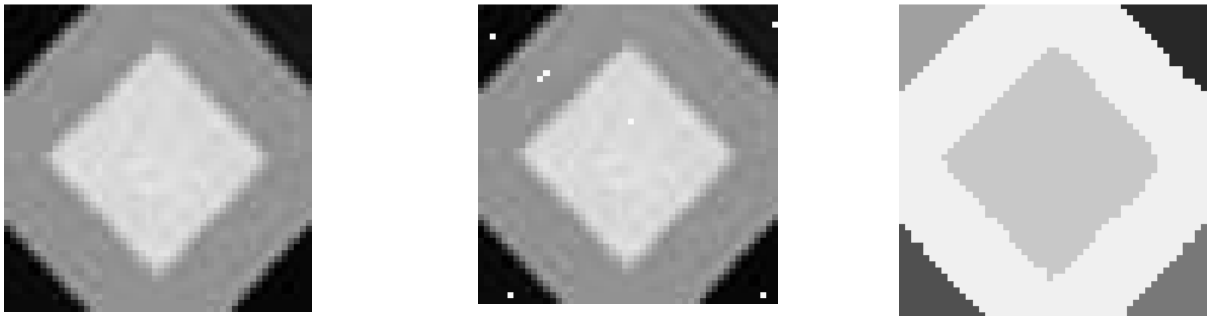
In [1], the local density at a node is defined with the normalized term $|H|$ as shown in (3). This term is necessary if the graph G is densely connected because the sum $\sum_{j \in H} w_{ij}$ for any node i depends on most of the nodes of the graph. In other words, this sum tends to decrease if we remove some nodes from the graph. The normalized term makes the local density estimation more accurate when parts of the graph are removed. However, in cases where graph G is sparse, which almost always applies when G is the proximity graph of an image, each node is connected to only a small and fixed number of neighbors. For any node i, the sum $\sum_{j \in H} w_{ij}$ depends on only several neighbor nodes and therefore this sum for most of the nodes does not change when we remove parts of the graph. Thus, here we eliminate this normalized term and estimate the local density at node i of H by:

$$d(i,H) = \sum_{j \in H} w_{ij} \quad (5)$$

Image segmentation algorithm steps are outlined in the following procedure.
Input: An image *I*.
Output: Segmentation of the image *I*.
1. Build a proximity graph *G* for the input image *I*.
2. Compute the sequence of density variation for *G*.
3. Identify a set of core pixels based on the sequence of density variation.
4. Partition the set of core pixels into groups.
5. Expand the groups of core pixels to get the image segmentation.

Computing experiments and discussion

In computing experiment we focused on study of a core clusterization algorithm, particularly, on its two free arguments,- α , β. For example, Figure 1 and Table 1 present the image segmentation results, when α=0.4, β=1.



a) b) c)

Fig.1: (a) initial gray-scale picture; (b) basic pixels; (c) segmented picture

Table 1

G	V	E	σ_1	σ_d	r	α	Basic pixels nodes
50x50	2500	102908	0.1	4	4	0.4	480,532,594,679,1394,2493,2494

Since the number of nodes is finite, there is a limited number of possible settings for β and α. Parameter β is an integer and its role is to eliminate some isolated noisy pixels at the top of the list of pixels with high rates of decrease in *D* values. Usually we fix β to be 3 or 4. The main parameter of the clustering method is α, changing it can result in increasing or decreasing the number of segments.

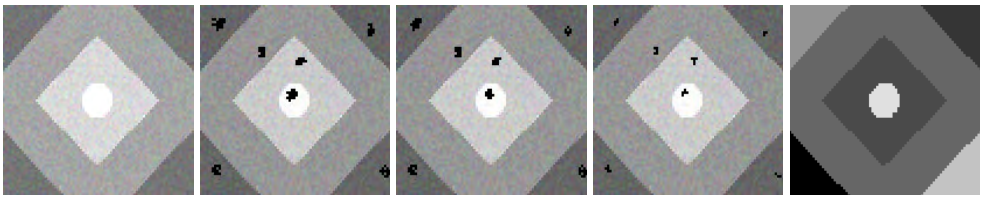
In (b), (c), and (d) of Fig. 2, we illustrate the effect of parameter α on the set of core pixels. The cores of segments shrink when α increased, conversely they are enlarged if α is decreased. Yet the segmentation results remain the same as shown in (e). In general, we avoid setting α to a value lower than 90% because it may produce core pixels that are not very reliable, especially in noisy images.

In images which contain a mixture of segments of different sizes or levels, by increasing or decreasing α, one can obtain a coarse or fine-grained segmentation. Figure 2(a) shows a gray scale image, (b), (c), and (d) show segmentation results of (a) with different values of α. Higher α produces coarser segmentation while lower α yields finer segmentation. This illustrates the point that core pixels are arranged in an order such that the pixels having high rates of decrease in their *D* values belong to the cores of strong segments.

(a) (b) (c) (d)

Fig. 2: (a) is a grey scale image. (b), (c), and (d) shows segmentations of the image with d =97%, 98%, and 99%, respectively

Figure 3 presents the result of segmentation, when parameter α ≥0.9. The set of basic pixels consists of 7 components, representing the cores of 7 segments that exactly correspond to the number of areas in initial image.



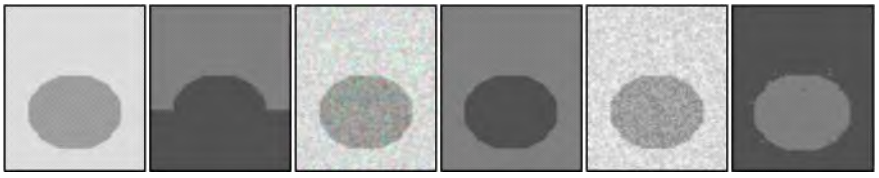
(a) (b) (c) (d) (e)

Fig.3: (a)- Initial grey-scale image; (b), (c), (d) show basic pixels at β=3, α=0.9,0.96 и 0.99 accordingly. (e)-different segments have different intensities

Spectral clustering methods in general and normalized cuts in particular are well known graph clustering approaches [2, 4, 5]. Application of normalized cut to image segmentation has been shown in [2]. An implementation of this method developed by its authors is available on the web [3]. The method basic idea is to partition a graph $G = (V, E, W)$ into k subgraphs $A1, A2, ..., A_k$ based on the minimum k-way normalized cut, which is defined by:

$$Ncut_k = \sum_{i=1}^{k} \frac{\sum_{u \in A_i, v \in V - A_i} w_{uv}}{\sum_{u \in A_i, v \in V} w_{uv}} \qquad (6)$$

Finding the exact minimum normalized cut is a NP-hard problem, so an approximate solution is estimated using eigenvectors of the normalized Laplacian matrix $L=I-D^{-1}W$, where D is the diagonal matrix of vertex degrees. The complexity of solving this approximation is relatively high, about $O(|V|2.5)$ for sparse graphs [2]. As is true of many clustering methods, the normalized cut method requires that the number of clusters k be pre-specified, which is especially problematic for image segmentation, since the number of segments is highly variable depending on the scene in the images. The number of segments in an image is not something that we would think about in the first place, but rather is a natural result of the perception process. Yet a more significant problem with minimizing normalized cuts is that the normalizing factors in the criterion function (6) make the cut favor clusters of similar sizes. As a result, small clusters are very easily omitted while large clusters are often split up into small parts.



(a) (b) (c) (d) (e) (f)

Fig. 4: (b) is the segmentation by the normalized cut on the image in (a). Images (c) and (e) are created by adding noise to (a). (d) and (f) show the segmentations by core clusterization method on (c) and (e), respectively

Fig. 4(b) shows the segmentation of the image in (a) by the two-way normalized cut. The cut partitions the image into two similar size segments. It fails to find the oval as one segment because of the disparity between the number of pixels inside and outside the oval. Clearly changing k dramatically changes the segmentation, and for any k, the image is always partitioned into segments of roughly similar sizes.

Experiments show that core clusterization is much faster than the normalized cut method implemented to run on the same proximity graphs. It can be seen that the running time of the coring method is roughly linear to the number of edges of the proximity graphs. Running times shown for the normalized cut are the average time for partitioning the graph into 2, 3, and 4 segments. For the cases

that the graphs contain more than 105*103 nodes and 17.6*106 edges, the normalized cut method fails to execute because of an 'out of memory' error. An additional advantage of the coring method is that one can change β or α and quickly obtain a new result by re-executing fast steps 3, 4 and 5 of the method. In contrast, for the normalized cut method, changing *k* will result in re-computing the cut from scratch.

In Table 2 , we show the approximate execution times on different proximity graphs of the three methods using a PC with a CPU of Core 2 Duo 2.4GHz and 2GB RAM.

Table 2. Execution times of the normalized cut and coring methods for clustering graphs

Proximity graphs		Normalized cut (sec)	Coring method (sec)		k-means (sec)
#nodes	#edges		Sequential execution	Parallel execution	
$30*10^3$	$4.7*10^6$	7	0.3	0.2	7
$45*10^3$	$7.1*10^6$	12	0.4	0.3	12
$60*10^3$	$10.7*10^6$	22	0.7	0.5	25
$75*10^3$	$12.2*10^6$	50	0.9	0.65	40
$90*10^3$	$14.8*10^6$	91	1.0	0.8	80
$105*10^3$	$17.6*10^6$	116	1.2	0.93	100
$120*10^3$	$18.7*10^6$	NA	1.5	1.1	NA
$500*10^3$	$50*10^6$	NA	4.2	3.2	NA
$1000*10^3$	$70*10^6$	NA	5.5	4.1	NA

Conclusion

We have evaluated a graph-based clustering method for image segmentation. Using proximity graphs, we partition images into segments of nearby pixels with similar intensities or colors. The method is simple and fast. It is also stable and robust to noise because it identifies and uses the pixels in cores of segments In addition to speed and robustness, an advantage is that parameters can be adjusted to yield a range of results from a coarse to very fine-grained segmentation.

References

[1] Le, T., Kulikowski, C., Muchnik, I.: Coring method for clustering a graph, DIMACS Technical Report (2008)

[2] Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Trans. on Pattern Analysis and Machine Intelligence (2000) 888-905

[3] Shi, J.: Normalized cut image segmentation code, http://www.cis.upenn.edu/~jshi/software

[4] Luxburg, U.: A tutorial on spectral clustering. Statistics and Computing (2007) 395-416

[5] Kannan, R., Vempala, S., Vetta, A.: On Clusterings: Good, Bad and Spectral. Proc. 41st Annual Symposium on the Foundation of Computer Science (2000) 367-380

GRID TECHNOLOGIES IN SPbSU LONG-RANGE CORRELATIONS ANALYSIS AND MC SIMULATIONS FOR ALICE

I.G. Altsybeev, G.A. Feofilov, M.V. Kompaniets, V.N. Kovalenko, V.V. Vechernin, I.S. Vorobyev, A.K. Zarochentsev

Laboratory of Ultra-High Energy Physics, St. Petersburg State University
198504 Ulianovskaya st., 1, Petrodvorets, St. Petersburg, Russia
feofilov@hiex.phys.spbu.ru

Studies of long-range correlations within the physics program of ALICE [1] require a high accuracy statistical analysis of experimental data. In this report, examples of Grid technologies used for the analysis of several types of correlations in proton-proton and Pb-Pb collisions in ALICE are presented. The main stages of software development and debugging on the basis of the AliAnalysis Manager [2] platform are described, allowing one to make calculations with both local and distributed computing systems (PROOF [3, 4], Grid). Examples of Monte-Carlo calculations are also given. A distributed storage and data processing system (ALICE Environment Grid [5, 6]) providing access to experimental data and results of modeling of proton-proton and Pb-Pb collisions is described. The analysis of large data (tens of Tb) allowed one to obtain results with statistics of more than 20 million events as well as to investigate a topological structure of long-range correlations.

Introduction

ALICE experiment [1] at the Large Hadron Collider gives us a unique opportunity to study heavy ion and proton collisions at high energies. In such collisions nuclear matter is believed to undergo a phase transition to quark-gluon plasma, a state of matter which is thought to have existed just moments after the Big Bang. Since the start of operation in 2008 ALICE collects data with rate of about 4 PB a year, which means that one has to develop powerful and adaptable computing system to handle with it. Grid technologies and distributed computing are an integral part of any modern high energy physics experiment and ALICE is not an exception.

Physical program

Our physics motivation is based on the String-Fusion Model predictions [7] where the effects of string interaction are taken into account in the form of fusion or percolation. As quark-gluon string is an extended object in the rapidity space, i.e. contributes by fragmentation to wide rapidity range, one can expect appearance of the correlations between observable quantities such as charged particles multiplicity n, transverse momentum pt, net charge q etc., in distant rapidity intervals. The SPbSU team now actively searches for the long-range n-n, pt-n, pt-pt and qq correlations. Such types of correlations are investigated in pp and AA collisions, string fusion effect is also investigated in Monte Carlo calculations using Grid and PROOF facilities. The presence of such long-range correlations can be also considered as one of the signatures of the early stages of quark-gluon plasma formation. Experimental studies of the long-range correlations in pp and AA collisions are proposed for ALICE at LHC and NA61 at SPS experiments, CERN [8, 9].

Distributed data processing in PROOF and Grid

The main goal of our analysis is the event-by-event processing of reconstructed or simulated ALICE experimental data in ESD format (Event Summary Data). To complete physical program with high statistics of the data Grid calculations are essential. In order to get the reliable results we have to deal with analysis at the level of about not less than 10^6 high-multiplicity events. Meanwhile Grid computing takes a lot of time and it is very unpractical to wait for hours for each debugging run. The most convenient case for code debugging is local analysis, but it implies processing of very small

number of events, especially in case of AA collisions. PROOF calculations take an intermediate place. It is suitable for an intermediate statistics and allows working in an interactive mode. So the 3-stages data processing strategy "local → PROOF → Grid" was accepted. The key issue here is AliAnalysisManager class [2] in AliROOT, which provides the unified access to all three ways of analysis. It makes possible to run one code containing physics logic in all execution systems transparently, one just needs to change execution scripts. The same benefit appears in Monte-Carlo calculations which were also performed in our analysis. Dwell on each stage of analysis more detailed.

Stage 1 – Local analysis

At this first stage a new code is created and a first debugging is performed in order to compile and run the code. Local analysis is very convenient for debugging due to the absence of latency, providing real-time feedback. Most of code writing is done at this stage; however, it is not possible to debug the physics analysis code using the local PC, especially in case of AA collisions. One can't debug physics logic using the statistics of tens or hundreds events.

Stage 2 – data analysis using PROOF

When the primary debugging of code is done, we need to ensure that physical ideas are implemented correctly. The second stage should be applied with the debugging output in several runs using a significant statistics dataset. These runs are needed to check if the program is calculating the results in line with the initial physics ideas. There could be always some errors leading to non realistic results, and to see those errors one needs a statistics that can provide at least preliminary conclusions. Analysis using the parallel ROOT facility (PROOF) [3, 4] suits these requirements perfectly. PROOF is a software system enabling ROOT-based analysis and processing in parallel on distributed resources. This system optimizes the execution time by implementing data parallelism at event-level and provides real-time feedback, which is very important for code developing. At the same time PROOF allows one to get results with statistics up to 10^6 events on different runs, which is quite enough to get preliminary results. At this stage the tuning and checking of main parameters is also performed (cutting thresholds, histogram limits etc.). Additional software required by the algorithm to be run can be loaded directly to the system in optimized way.

Nevertheless, the PROOF analysis has some drawbacks, and unfortunately main of them is insufficient stability of operation. At this stage it is harder to debug code, and some PROOF specific errors can appear like merging problems. Also there is still insufficient statistics to get final results.

List of ALICE PROOF clusters, their operation status, current workload and other parameters can be found at http://alimonitor.cern.ch/stats?page=PROOF/list

Stage 3 – Grid analysis

At this stage one performs last checks of physics logic and gets final results using full statistics of data available in AliEn system.

AliEn [5, 6] – ALICE Environment – is a set of middleware tools and services that implement a Grid infrastructure. The development of AliEn started in 2000 by ALICE collaboration, and it was deployed for distributed Monte Carlo productions on several remote computing sites. From 2005 AliEn has been used both for data production and end-user analysis. In spite of the fact that Grid technologies were rapidly evolving, AliEn has very successfully served its primary goal of hiding from the end user the complexity and heterogeneity of all underlying Grid services. The main AliEn components are as follows:

- File Catalogue
  - – UNIX-like file system interface
  - – Entries are retrieved by LFN (Logical File Name) or GUID (Globally Unique IDentifier)
  - – Mapping to physical files
  - – Powerful metadata catalogue
  - – Automatic Storage Element selection
  - – 4 storage technologies: CASTOR2, dCache, DPM, Scalla

– Multiple storage protocols: Xrootd, torrent, srm, file

• Authentification & authorization
• ROOT interface
• VO-box system
• Monitoring based on MonALISA software [10], package management
• Workload management using TaskQueue, Job Agents & pull model

AliEn system provides access to all Grid data and allows us to get the final reliable results with statistics up to 10^7 events. On the other hand usage of AliEn system implies high latency and queues. Sometimes Grid specific errors appear, mainly validation and execution problems for some jobs, which are very difficult to trace and debug.

ALICE Computing in SPbSU

Cluster RU-SPbSU is in stable operation since 2004 running AliEn middleware. In 2005 it was registered in RDIG and started its operation in the Large Hadron Collider Grid. In order to use the site as storage of Tier-2 level, the Xrootd middleware was installed. After the significant upgrades according to EGEE program in 2007, 2008 and 2010 RU-SPbSU cluster supports 4 Virtual Organizations (ATLAS, CMS, LHCb, ALICE). In 2011 control servers and then working nodes were moved to the virtual VMware Vsphere machines to form a part of University cloud. The POD (Proof On Demand) cluster is being prepared on the same cloud for ALICE experimental data processing by SPbSU researchers.

By 2012 cluster provides ALICE 62 TB of disk space and 146 computing cores, 96 of those are also available for 3 other VOs. Both the cluster and the LCG middleware demonstrate stable performance and functioning in the Worldwide LHC Computing Grid (WLCG).

Experimental data processing results

Correlations n-n, pt-n and pt-pt were investigated in pp collisions in ALICE at the energies of 0.9 and 7 TeV. Experimental results on the correlation coefficient behavior against the width of pseudorapidity windows and the gap between them are shown in Fig. 1 (plots were approved as ALICE Preliminary). Non-zero p_T-N and p_T-p_T correlations coefficients can be considered as a signature of quark-gluon strings fusion and QGP formation on the early stages of proton-proton collisions.

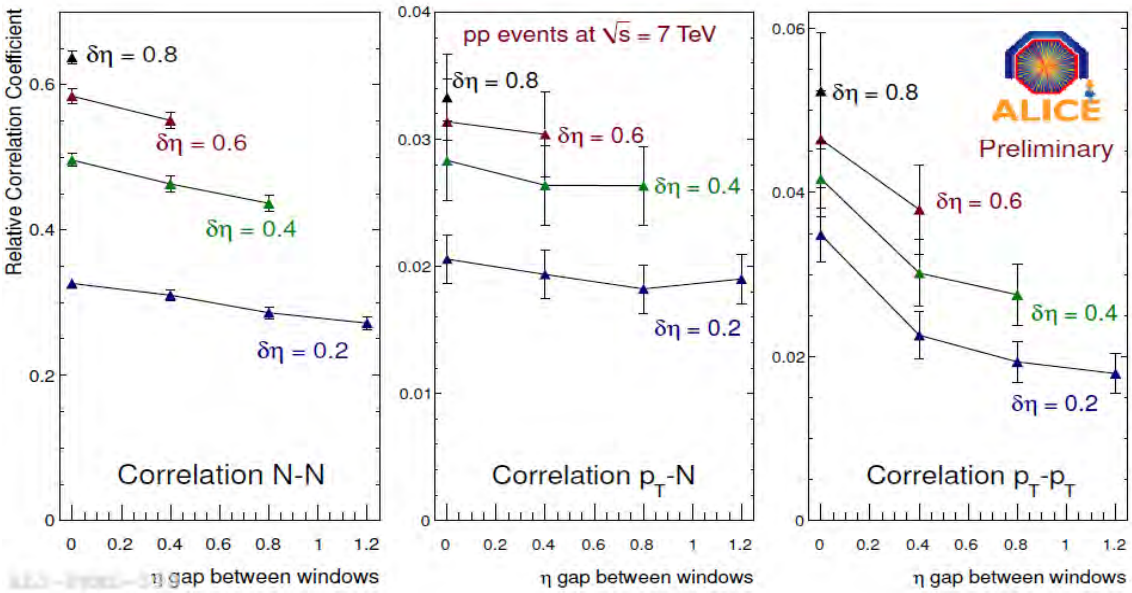


Fig. 1: The dependence of long-range N-N, p_T-N and p_T-p_T correlations on the pseudorapidity gap between windows in pp collisions at 7 TeV, measured for different widths δη of the observation windows [11]. Normalized observables. Lines are drawn to guide the eye

Additional studies in the Grid framework of long-range correlations were performed in azimuthally separated windows in order to obtain cleaner information that is less affected by short-range effects. The dependence of correlation coefficients on azimuthal configuration of windows is shown in Fig. 2.

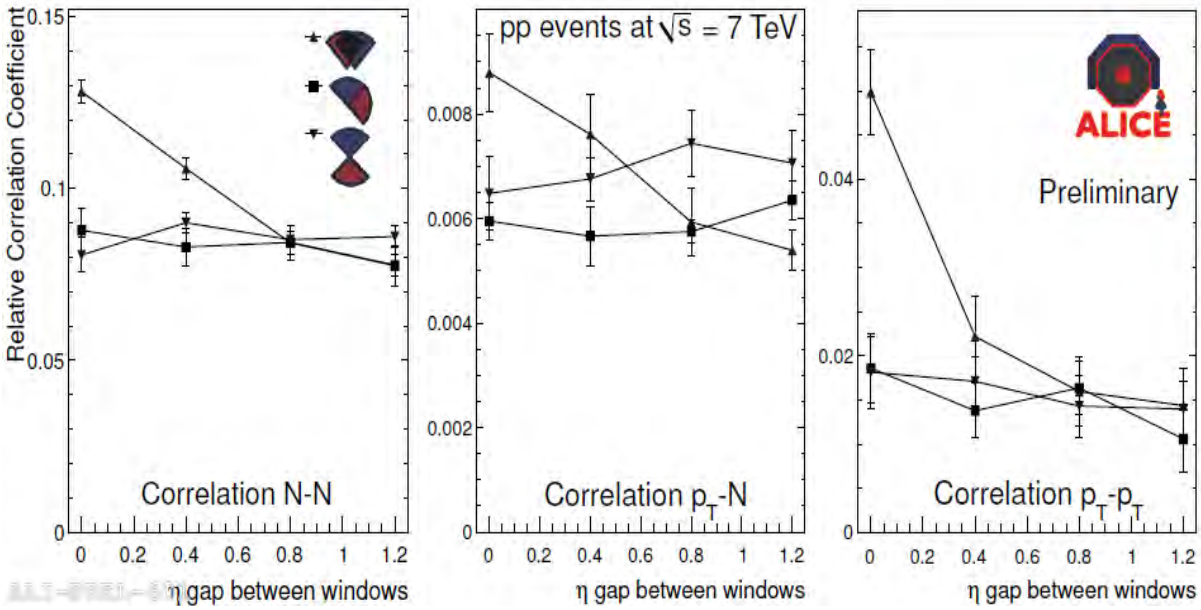


Fig.2: The same as in Fig.1 but for different configurations of backward and forward π/2 azimuth sectors (for the width of the pseudorapidity windows – δη=0.2) [11]. Relative orientations of sectors are marked by color

Monte-Carlo calculation results

The general scheme of the AliAnalysis framework is applied also to Monte Carlo calculations. It implies quick developing, testing and debugging of Monte Carlo algorithms. It permits us to obtain quickly the results for the different Monte Carlo models of heavy ion collisions, try different parameters and assumptions. All this makes it possible to get a better physical understanding of the processes studied.

We implemented the model with color string formation and fusion for pp and AA collisions at the LHC energies and performed event-by-event simulations for pp and PbPb collisions.

In order to perform event-by-event Monte Carlo calculation, three abstraction levels of the algorithms were taken:

1) The core of Monte Carlo generator, which covers the simulation of a collision and corresponding mathematical procedures.
2) AliAnalysis task class, that runs the generator events, collects the simulated data, performs initial statistical processing
3) Final data processing and plotting the results.

In addition, there are several configuration and run scripts.

Monte Carlo calculations for heavy ion collisions with high statistics demand computational power comparable to one of used for data analysis. In order to submit tasks to the Grid (WLCG) or perform PROOF analysis in the AliAnalysis framework some datasets should be provided, but during calculations the actual data are ignored. As a result pure MC simulations are performed in both PROOF and Grid mode.

In Fig. 3 examples of N-N and p_T-N correlation functions are shown for pp collisions at 7 TeV. The plots demonstrate decrease of N-N correlation strength (the slope) and non-zero p_T-N correlation with taking into account the string fusion.

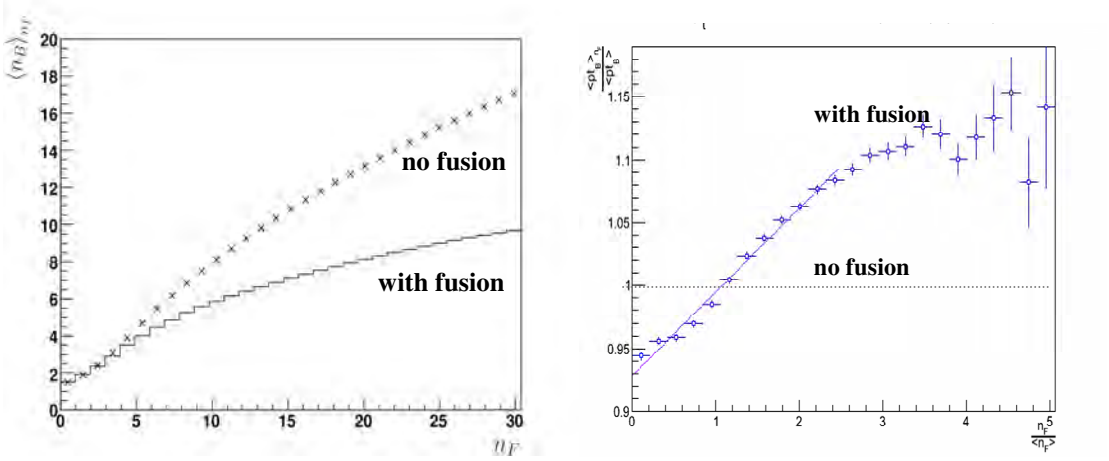


Fig. 3: Examples of N-N (left) and p_T-N (right) correlation functions calculated in Monte-Carlo simulations for pp collisions at 7 TeV. Configuration of pseudorapidity windows is (–0.8, 0) (0, 0.8)

Summary

1) Grid technologies allows us to perform an event-by-event long-range correlations analysis with statistics up to 10^7 events and thereby to get the reliable physical results.

2) Our experience with the ALICE data analysis shows importance of combination of Grid with other interactive High Performance Computing technologies such as PROOF, especially for debugging reasons.

3) Analysis framework can be used also for our own Monte Carlo calculations with high statistics.

References

[1] ALICE Collaboration, "ALICE: Physics Performance Report" - Volume 2, CERN/LHCC 2005-030; ALICE PPR Volume II, 5 December 2005; CERN. (J.Phys. G32 (2006) 1295-2040 [Section: 6.5.15 - Long-range correlations, p.1749-1751])

[2] The Analysis Framework http://aliweb.cern.ch/Offline/Activities/Analysis/AnalysisFramework/index.html

[3] The Parallel ROOT Facility, PROOF http://root.cern.ch/drupal/content/proof

[4] Data Analysis with PROOF, G Ganis, J. Iwaszkiewicz, F. Rademakers, Proceedings of ACAT 2008 Conference

[5] ALICE Environment: Open Source GRID Framework http://alien2.cern.ch/

[6] AliEn: ALICE environment on the GRID, S. Bagnasco, L. Betev, P. Buncic, F. Carminati, C. Cirstoiu, C. Grigoras, A. Hayrapetyan, A. Harutyunyan, A.J. Peters, P. Saiz; CERN. International Conference on Computing in High Energy and Nuclear Physics (CHEP'07); 2008 J. Phys.: Conf. Ser. 119 062012

[7] M. A. Braun and C. Pajares, Phys. Lett. B 287, 154 (1992); Nucl. Phys. B 390, 542, 549 (1993).

[8] ALICE Collaboration, "ALICE: Physics Performance Report" - Volume 2, CERN/LHCC 2005-030; ALICE PPR Volume II, 5 December 2005; CERN. (Journ.Phys.G: Nuclear and Particle Phys., 2006, 733 pages)

[9] SHINE/NA61 proposal: Study of hadron production in hadron nucleus and nucleus nucleus collisions at the CERN SPS. By NA49–future Collaboration (N. Antoniou et al.). CERN–SPS – 2006–034, CERN–SPSC–P–330.

[10] MonALISA: An Agent Based, Dynamic Service System to Monitor, Control and Optimize Grid based Applications, I.C. Legrand, H.B. Newman, R. Voicu, C. Cirstoiu, C. Grigoras, M. Toarta, C. Dobre, CHEP 2004, Interlaken, Switzerland.

[11] G.Feofilov (for the ALICE Collaboration), "Long-Range (Forward-Backward) pt and Multiplicity Correlations in pp collisions at 0.9 and 7 TeV", poster report at QM-2011, 23-28 May 2011, Annecy, France.

GRID IN JINR AND PARTICIPATION IN THE WLCG PROJECT[1]

S.D. Belov, P.V. Dmitrienko, V.V. Galaktionov, N.I. Gromova, I.S. Kadochnikov, V.V. Korenkov, N.A. Kutovskiy, S.V. Mitsyn, V.V. Mitsyn, D.A. Oleynik, A.S. Petrosyan, G.S. Shabratova, T.A. Strizh, E.A. Tikhonenko, V.V. Trofimov, V.E. Zhiltsov, A.V. Uzhinskiy

Joint Institute for Nuclear Research, Dubna

JINR has been participating in the international grid activities since 2001, after starting the EU Data Grid project on creation of grid middleware and testing the initial operational grid infrastructure in Europe [1]. The Joint Institute for Nuclear Research takes an active part in a large-scale worldwide grid project WLCG (Worldwide LHC Computing) in a close cooperation with the CERN Information Technology department since the 2003 year [2]. JINR made a significant contribution to the WLCG, EGEE (Enabling Grids for E-sciencE), and EGI (European Grid Infrastructure) projects. 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 [3]**.** As a result**,** 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 (JINR-LCG2) is fully integrated into the global (world-wide) infrastructure (the name of the JINR grid site in the WLCG/EGI infrastructure is JINR-LCG2). The resources of the JINR grid site are successfully used in the global grid infrastructure, and on indicators of the reliability, the JINR-LCG2 site is one of the best in the WLCG 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. Constantly working to train the grid technologies, JINR created a separate educational grid infrastructure. In the field of grid, JINR actively collaborates with many foreign and Russian research centers. Special attention is paid to cooperation with the JINR Member States.

By June, 2012 the JINR computing cluster has been upgraded from 2064 up to 2560 slots, a total capacity of the Storage Element structured as dCache and Xrootd storage system was extended to 1400 TB. The CICC software includes a number of program packages which form the grid environment. A current version of the WLCG software is mostly gLite 3.2, we have a plan to upgrade it to UMD(Unified Middleware Distribution)/EMI(European Middleware Initiative) latest 2.x version later in this year. A monitoring and accounting system has been developed at JINR and is in use by the entire Russian WLCG segment. The JINR external optical communication channel provides up to 2x10 Gbps data link.

The CICC provides the following services in the WLCG environment:

- Storage Element (SE) services;
- Computing Element (CE) services as grid batch queue enabling access for 13 Virtual Organizations (VO) including ALICE, ATLAS, CMS, LHCb, MPD, HONE, FUSION, BIOMED, BES;
- Information Service (BDII- Berkley DB Information Index);

[1] Partially supported by RFBR (grants 12-07-91501, 12-07-90402, 10-07-00522), by the Ministry of Communications and Mass Media of the Russian Federation (cont. № 0173100007512000020_144316), and by the Federal Target Program "R&D in Priority Fields of the S&T Complex of Russia (2007-2013)" (State contracts 07.524.12.4008 and 07.514.12.4006).

- Proxy service (PX);
- the advanced service for access to the LCG/EGEE resources (MyProxy);
- Workload Management System + Logging&Bookkeeping Service (WMS+LB);
- AMQP (Advanced Message Queuing Protocol) based on APEL node collector and publisher;
- LCG File Catalog (LFC) service and VOboxes special services for ALICE, CMS, CBM and PANDA.

It should be mentioned here that we have the batch queues enabled for PANDA and CBM. Also there is one NFS-server dedicated to VOs. A global file system CVMFS for the access to Virtual Organization's software has been installed, software required for LHC experiments is currently installed (XROOTD, AliROOT, ROOT, GEANT packages for ALICE; CMSSW packages for CMS; LHCb and ATLAS are supported from CVMFS global installation). JINR currently supports and develops the JINR WLCG-segment in the frames of the WLCG infrastructure in accordance with the requirements of the experiments for the LHC running phase.

Current computing activities for ALICE, CMS and ATLAS are carried out in coordination with LHC experiments [4-11]:

- **ATLAS** : Functional Test of the ATLAS DDM (Distributed Data Management); implementation of PD2P (PanDA Dynamic Data Placement); Xrootd and PROOF (Parallel ROOT Facility) for Atlas Tier3 data analysis; development and support of ATLAS DQ2 Deletion Service became a major contribution to the cooperation with ATLAS experiment;
- **CMS**: participation in CMS PhEDEx test data transfers; support of PhEDEx server installed at the CMS VObox at JINR; CMS data replication to the JINR SE; participation in CMS Dashboard data repository maintenance and CMS Dashboard development [12-14], in particular, in improvement of CMS job monitoring and CMS job failures reporting;
- **ALICE:** regular update and testing of ALICE software (AliEn), update and support of VO box operation; installation and support packages required for ALICE production and distributed activities – in 2011-2012: update of Xrootd, CREAM-CE (Computing Resource Execution And Management – Computing Element) installation of p2p torrent application software transport directly to Working Nodes; management of whole RDIG ALICE activity not only at the JINR-WLCG site but also at 7 other ALICE sites in Russia; installation, upgrade and support of PROOF cluster of Alice Analysis Facility – JRAF (JINR Russia Analysis Facility) with 48 workers and 24 TB disk space;
- tests of readiness of the JINR site to store and process data for all the experiments JINR participates in (ALICE, ATLAS, CMS).

Work for development, maintains and improvements of ATLAS DQ2 Deletion Service [15] was provided by JINR LIT specialists. DQ2 Deletion Service serves deletion requests for 130 sites with more than 700 endpoints (space tokens), it is one of critical data management service in ATLAS. A set of improvements aimed to increasing of productivity of service was done. After optimization of algorithms and DB, was achieved deletion rate more than 10 Hz for some sites, and overall deletion performance more than 6 million files per day.

JINR has a large and long-term experience in grid monitoring activities [16]. Currently the main areas of activity are:

- RDIG monitoring and accounting system for the WLCG infrastructure of Russian Tier2 sites (http://rocmon.jinr.ru:8080) and a continuous support is providing for grid site administrators;
- participation in development of global WLCG data transfer monitoring system (https://twiki.cern.ch/twiki/bin/view/LCG/WLCGTransferMonitoring);
- Tier3 monitoring project [17-19] - the overall coordination and development at CERN (https://svnweb.cern.ch/trac/t3mon): software environment and development infrastructure (code repository, build system, software repository, external packages built for dependencies) and, in particular, at JINR:

- VM-based infrastructure for simulating,
- different Tier3 cluster and storage solutions was deployed [20]. For the moment it consists of the following parts: Ganglia server, Torque, Condor, PROOF, OGE-based clusters, two Xrootd and one Lustre-based storage systems.

The JINR local monitoring system (http://litmon.jinr.ru) developed at JINR is an important basis to the global monitoring systems providing actual information on the status of the JINR infrastructure to the higher levels of monitoring.

The dCache monitoring system for the JINR WLCG-segment has been developed using Nagios, MRTG and custom plug-ins. The system provides information on input/output traffic and requested and utilized space for both ATLAS and CMS experiments (http://litmon.jinr.ru/dcache.html)

We continue to take part in the WLCG middleware testing/evaluation. During the last two years the following directions and results have been provided:

- development of gLite MPI (Message Passing Interface) certification tests: MPI patch #3714 certification and evaluation of the current status of MPI enabled CREAM-CE;
- development and modernization of FTS (File Transfer Service) certification tests;
- deployment of few gLite 3.2, EMI, and UMD components was tested;
- in framework of developing of tests for LFC (LCG File Catalog) perl API functions a separate LFC server (gLite 3.2) was installed on gLite testbed at JINR and the corresponding GGUS (Global Grid User Support) tickets were submitted.

Participation in the LCG Monte Carlo database (http://mcdb.cern.ch) results in [21-24]:

- support for CMS users;
- libraries for working with automatic documentation for Monte Carlo simulated events (HepML language) were improved;
- automatic data uploading with unified HepML descriptions was improved.

We support users (conducting courses, lectures, trainings) to stimulate their active usage of the WLCG resources [25-29]. Also a special grid-training infrastructure for JINR and the JINR Member States (Russia, Uzbekistan, Armenia, Bulgaria, Ukraine) has been created. During the 2011-2012 years a number of schools and training events have been held:

- JINR-CERN Schools on JINR/CERN Grid and Advanced Information systems were held on October 24-28, 2011 (http://ais-grid-2011.jinr.ru/) and on May 14-18, 2012 (http://ais-grid-2012.jinr.ru/) (about 100 attended students from JINR, Russian universities, Poland, Ukraine, Georgia and Bulgaria) ;
- 05.09 - 09.09.2011: a training course for system administrators from Bogolyubov Institute for Theoretical Physics - BITP (Kiev, Ukraine) and National Technical University of Ukraine "Kyiv Polytechnic Institute" - KPI (Kiev, Ukraine) was given. That course was focused mostly on gLite 3.2 and AliEn services deployment;
- basic training courses on gLite 3.2 services deployment for system administrators from Mongolia, Kazakhstan and Azerbaijan were held;
- international practice on grid technologies, JINR University Center, 06.09.11-09.09.11;
- 23.07 - 03.08.2012: a training on basic set of EMI2 grid services deployment for system administrators from Egypt, Mongolia and Azerbaijan;
- 24.09 - 28.09.2012: training courses for users and system administrators from Bogolyubov Institute for Theoretical Physics - BITP (Kiev, Ukraine), National Technical University of Ukraine "Kyiv Polytechnic Institute" - KPI (Kiev, Ukraine) and Institute for Scintillation Materials - ISMA (Kharkov, Ukraine) had been held. These courses were focused on several topics: 1) introduction lectures on LHC experiments, NICA, AliEn; 2) practical training on AliRoot; 3) EMI2, XRootD and AliEn services deployment;
- 26.09-12.10.2012: practice on grid basics for student from Republic of South Africa.

Also the trainings for system administrators from Ukraine, Romania and Uzbekistan have been conducted. Two grid sites based on gLite middleware (one at the Bogolyubov Institute for

Theoretical Physics and another at the National Technical University of Ukraine "Kyiv Polytechnic Institute") were set up during one of these trainings. The trainings for Romanian and Uzbekistan administrators were intended for giving practical skills in setting up MPI enabled CREAM Computing Elements. More details about the grid training infrastructure can be found on the portal http://gridedu.jinr.ru.

We provide a continiuos support for the JINR Member States and associated JINR Member States in the WLCG activities working in a close cooperation with partners in Ukraine, Belarus, Azerbaijan, Germany, Czech, Slovakia, Poland, Romania, Moldova, Mongolia, South Africa, Kazakhstan and Bulgaria. Protocols and agreements for cooperation in the field of grid technologies are signed between JINR and Armenia, Belarus, Bulgaria, Moldova, Poland, Czech and Slovak.

Recently at CERN the agreement on construction in Russia of Tier1 center for four LHC experiments was signed by the representatives of Russian official agencies and CERN. This preliminary decision involves the creation of a Tier1 center for ALICE, ATLAS and LHCb experiments at the Kurchatov Institute and a Tier1 center for the CMS experiment - at JINR. Currently the plan of creation of CMS Tier1 center at JINR is developing in detail [30]. Implementation of this plan will require significant investments and also great efforts of JINR specialists, as well as to attract a certain number of new employees who will have to provide a stable operation (24x7) of the future CMS Tier1 at JINR.

The experience accumulated during the participation in the WLCG project helps JINR staff members to accomplish a wide range of works with the usage of grid technologies in different areas, in particular: simulation of off-line data processing for a large JINR project NICA [31], development of grid-services and problem-oriented web-interfaces for Russian grid network [32, 33], development of a special grid commander [34] as a graphical user interface to simplify user's work in grid environment, development of conception of cloud computing resources usage for cosmic data processing [35].

During 2011-2012 the results of JINR grid activities have been presented at ATLAS Software & Computing Workshop (04.04.2011-08.04.2011, CERN), the RDMS conference in Alushta, Ukraine (May 2011) (http://rdms2011.kipt.kharkov.ua/), ATLAS Computing technical interchange meeting (Dubna, JINR, 31.05.2011-02.06.2011), Programme Advisory Committee for Particle Physics, 35th meeting (Dubna, JINR, 21-22.06.2011), conference "Mathematical Modeling and Computational Physics" (MMCP 2011) (Slovakia, July 4 - 8, 2011), ATLAS Software & Computing Workshop (18 July 2011 - 22 July 2011, CERN), International Summer School, ENU (Astana, Kazakhstan, 07.08.2011-13.08.2011), Meeting on cooperation JINR-Mongolia (21.08.2011-25.08.2011, Mongolia, NEC'2011 symposium in Varna, Bulgaria (September, 2011) (http://nec2011.jinr.ru); at ALICE T1/T2 workshop, (Karlsruhe, 24-26 January 2012 (https://indico.cern.ch/conferenceTimeTable.py?confId=157585#20120125), CHEP'2012 conference (New York, US, May 21-25, 2012) and at the conference "Distributed computing and Grid technologies in science and education" (Dubna, July, 2012, http://grid2012.jinr.ru).

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 the Oct. 2011- Oct. 2012 is the most significant one: 47% (see Table 1).

Table 1. Russia Normalised CPU time (HEPSPEC06) by SITE and VO. LHC VOs. October 2011 - October 2012 (data from EGI Accounting portal https://www4.egee.cesga.es)

SITE	alice	atlas	cms	lhcb	Total	%
ITEP	5,325,996	4,069,628	3,619,544	1,604,644	**14,619,812**	**4.45%**
JINR-LCG2	32,895,956	60,666,632	50,242,916	11,625,652	**155,431,156**	**47.36%**
RRC-KI	25,278,904	16,919,004	2,587,708	7,429,316	**52,214,932**	**15.91%**
ru-Moscow-FIAN-LCG2	88	5,137,764	507,336	0	**5,645,188**	**1.72%**
ru-Moscow-SINP-LCG2	324	1,180,168	8,017,396	2,654,124	**11,852,012**	**3.61%**
ru-PNPI	5,838,212	6,658,468	1,970,068	3,884,244	**18,350,992**	**5.59%**

RU-Protvino-IHEP	8,458,160	26,455,612	11,852,592	5,001,700	**51,768,064**	**15.77%**
RU-SPbSU	2,909,584	5,328	2,068	94,136	**3,011,116**	**0.92%**
Ru-Troitsk-INR-LCG2	2,758,408	0	11,042,816	1,478,404	**15,279,628**	**4.66%**

In 2012, the JINR Grid-site is in the top ten of 143 Tier2 sites worldwide (fig.1) and on the III place in Europe (fig.2).

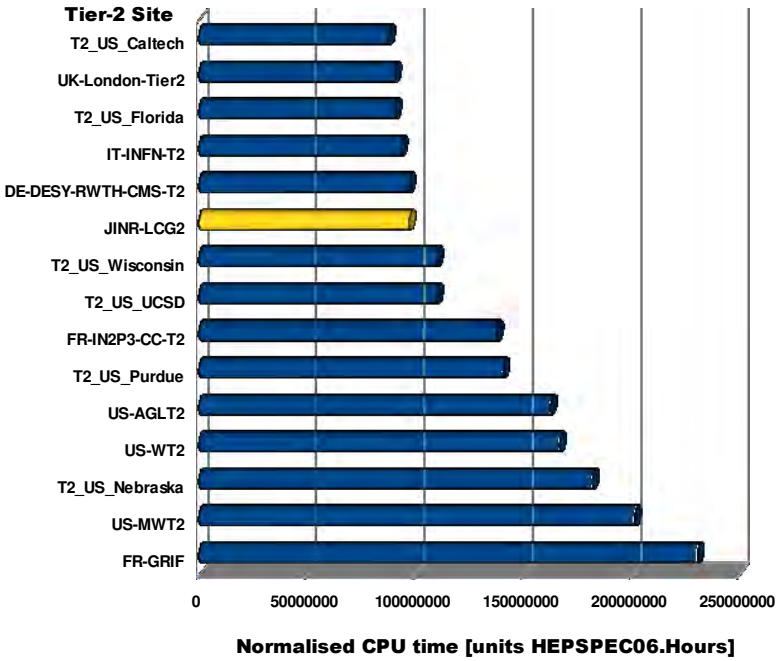


Figure 1: Worldwide rating of WLCG Tier2 sites by productivity

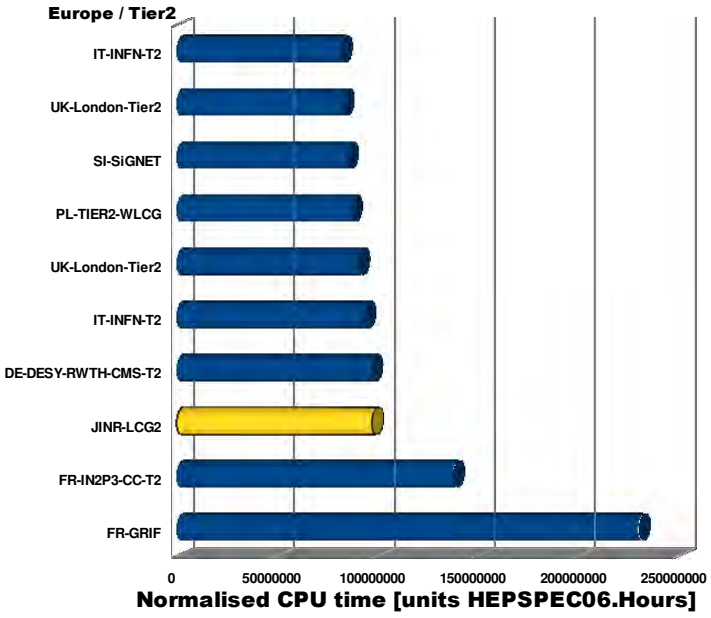


Figure 2: European rating of WLCG Tier2 sites by productivity

Information on JINR activities in the WLCG is currently available on the JINR GRID Portal (http://grid-eng.jinr.ru).

References

[1] S.D. Belov et al., Grid Activities at the Joint Institute for Nuclear Research, in Proc.of NEC'2011, Dubna, 2011, pp.68-73.

[2] V.Korenkov, Grid activities at the Joint Institute for Nuclear Research, Distributed Computing and Grid-technologies in Science and Education IV Int. Conference, Proceedings of the conference, Dubna, 2010, pp.142-147.

[3] Ilyin V.A., Korenkov V.V., Soldatov A.A. RDIG (Russian Data Intensive Grid) e-Infrastructure: status and plans, in the Proc. of NEC'2009, Dubna, JINR, 2010, p.150-153.

[4] G. Shabratova (on behalf of ALICE), The ALICE Grid operation, Distributed Computing and Grid-technologies in Science and Education IV Int. Conference, Proceedings of the conference, Dubna, 2010, pp. 202-214.

[5] Demichev M. et al. Readiness of the JINR grid segment to process the first ATLAS data, Proc. of NEC'2009, Dubna, JINR, 2010, pp.111-112.

[6] E.A. Boger et al., ATLAS Computing at JINR, Distributed Computing and Grid-technologies in Science and Education IV Int.Conference, Proceedings of the conference, Dubna, 2010, pp. 81-82.

[7] V. Gavrilov et al., RDMS CMS data processing and analysis workflow, in Proc. of XXIII Int. Symp. on Nuclear Electronics & Computing (NEC`2011), Dubna, 2011, pp.148-153.

[8] V. Gavrilov et al., RDMS CMS Tier 2 centers at the running phase of LHC, Distributed Computing and Grid-technologies in Science and Education IV Int. Conference, Proceedings of the conference, Dubna, 2010, pp. 103-108.

[9] Bunetsky 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).

[10] V. Gavrilov et al., RDMS CMS Computing, Proc. of the 15th RDMS CMS conference (Alushta, May 2011).

[11] V. Gavrilov et al., CMS experiment data processing at RDMS CMS Tier 2 centers, ibid pp. 133-137.

[12] Andreeva J. et al. Dashboard for the LHC experiments, J. Phys. Conf. Ser.119:062008, 2008.

[13] Sidorova I. Job monitoring for the LHC experiments, Proc. of NEC'2009, Dubna, JINR, 2010, pp. 243-246.

[14] Julia Andreeva, Max Boehm, Sergey Belov, …Irina Sidorova, Jiri Sitera, Elena Tikhonenko et al., Job monitoring on the WLCG scope: Current status and new strategy, *J. Phys.: Conf. Ser.*(2010) 219 062002.

[15] D.Oleynik, A.Petrosyan, V.Garonne, S.Campana on behalf of the ATLAS Collaboration, DDM DQ2 Deletion service. Implementation of central deletion service for ATLAS experiment, ibid pp.189-194.

[16] 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, pp.75-80.

[17] J.Andreeva et al., Tier-3 Monitoring Software Suite (T3MON) proposal, ATL-SOFT-PUB-2011-001, CERN, 2011 (http://cdsweb.cern.ch/record/1336119/files/ATL-COM-SOFT-2011-005.doc).

[18] A.Petrosyan, D. Oleynik, S. Belov, J.Andreeva, I. Kadochnikov, ATLAS off-Grid sites (Tier 3) monitoring. From local fabric monitoring to global overview of the VO computing activities, http://indico.cern.ch/getFile.py/access?contribId=426&sessionId=8&resId =0&materialId = paper&confide = 149557 to be published in the Proceedings of CHEP2012 conference, New York, USA, May 21-25, 2012.

[19] A.Petrosyan, D.Oleynik, S.Belov, J.Andreeva, I. Kadochnikov on behalf of the ATLAS Collaboration, ATLAS off-Grid sites (Tier-3) monitoring, ibid pp. 195-199.

[20] S. Belov et al., VM-based infrastructure for simulating different cluster and storage solutions used on ATLAS Tier-3 sites, https://cdsweb.cern.ch/record/1456851/files/ATL-SOFT-PROC-2012-057.pdf to be published in the Proceedings of CHEP2012 conference, New York, USA, May 21- 25, 2012.
[21] Belov S. et al. LCG MCDB – a Knowledgebase of Monte Carlo Simulated Events, Computer Physics Communications, V. 178, I. 3, 1 February 2008, pp.222-229.
[22] 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.
[23] D. Kekelidze, S. Belov, L. Dudko, A. Sherstnev, On Automation of Monte Carlo Simulation in High Energy Physics, Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 4th Intern. Conf. (Dubna, June28-July 3, 2010) – Dubna: JINR, Д-11-2010-140, 2010. pp.133-136.
[24] S. Belov et al., 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 http://arxiv.org/abs/1001.2576
[25] Korenkov V.V., Kutovskiy N.A. Educational grid infrastructure, Open System, N. 10, 2009 (in Russian).
[26] Belov S.D., Korenkov V.V., Kutovskiy N.A. Educational grid infrastructure: status and plans, Proc. of NEC'2009, Dubna, JINR, 2010, pp.81-83.
[27] N.A. Kutovskiy, Educational, training and testing grid infrastructure // Proceedings of XIV conference of young scientists and specialists (OMUS'2010), Dubna, 2010, pp.70-73.
[28] V.V. Korenkov, N. A. Kutovskiy, Distributed training and testing grid-infrastructure, Distributed Computing and Grid-technologies in Science and Education IV Int.Conference, Proceedings of the conference, Dubna, 2010, pp. 148-152.
[29] Kutovskiy N.A., Distributed training and testing grid infrastructure evolution, ibid pp. 180-185.
[30] Н.С. Астахов, С.Д. Белов, А.Г. Долбилов. В.Е. Жильцов, В.В. Кореньков, В.В. Мицын, Т.А. Стриж, Е.А. Тихоненко, В.В. Трофимов, С.В. Шматов, Создание в ОИЯИ автоматизированной системы обработки данных уровня Tier-1 эксперимента CMS на LHC, ibid pp.254-265.
[31] Кореньков В.В , Нечаевский А.В, Трофимов В.В. Моделирование грид системы off-line обработки данных для эксперимента НИКА, ibid pp.343-348.
[32] S.D. Belov et al., Monitoring, accounting and registration services for Russian Grid Network, ibid pp.30-33.
[33] Kutovskiy N.A, Lensky I.I, Semenov R.N., Problem-oriented web-interfaces for Russian Grid Network , ibid pp. 186-188.
[34] Галактионов В.В., GridCom, Grid Commander: графический интерфейс для работы с задачами и данными в Гриде, ibid pp.270-273.
[35] Кореньков В.В., Котов В.М., Русакович Н.А., Яковлев А.В., Создание облачной платформы уровня Tier3 в грид-инфраструктуре экспериментов на LHC для разработки приложений радиолокационного космического мониторинга, ibid pp.349-354.

MONITORING, ACCOUNTING AND REGISTRATION SERVICES FOR RUSSIAN GRID NETWORK[1]

S.D. Belov[1], T.M. Goloskokova[1], V.V. Korenkov[1], N.A. Kutovskiy[1, 2], D.A. Oleynik[1], A.S. Petrosyan[1], R.N. Semenov[1], A.V. Uzhinskiy[1]

[1]*Laboratory of Information Technologies, Joint Institute for Nuclear Research, Dubna*

[2]*National scientific and educational centre of particle and high energy physics of the Belarusian state university, Minsk, Belarus*

A national project on creation of Russian Grid Network was started in 2009 under support of the Ministry of Communications and Mass Media of the Russian Federation. One of the main tasks of the project is to provide a network infrastructure and connection to it of largest supercomputer centers, enterprises, high-tech industries and research organizations. Then, since 2011 has been in progress work on the development of the basic grid services, infrastructure for the RGN prototype, security systems, and adaptation of software packages to be used in the grid environment.

There are several core grid services to support operation of the Russian Grid Network (RGN). This article discuses three of them: monitoring service, accounting service, service of registration resource and grid services. The monitoring subsystem is used for collecting, storing and providing information on the status of Grid resources and services network as well as information about running in the user's jobs. The accounting subsystem is used for collecting, storing and providing information about grid computing resources consumed by user jobs. The main goal of the resources and grid services registration service is to provide information about resources and grid services within the RGN environment.

1 Introduction

In September 2009 the Commission under the President of the Russian Federation on the modernization and technological development of economy of Russia approved a list of projects in the field of "The development of supercomputers and grid computing" and, in which the project on the development of grid-based networks for high performance computing is approved. In 2010 a pilot zone of the Russian national grid network (RGN) was established. One of the main tasks of the project is to create a network infrastructure and connection to it of largest supercomputer centers, enterprises, high-tech industries and research organizations. Since 2011, the work has been in progress on creating of the basic Grid services infrastructure for the RGN prototype, security systems, adapting software packages for use in the grid environment. The main objectives of the RGN project are:

- creation of a network linking resource centers (supercomputers) and major consumers;
- development of basic Grid services for the construction of RGN infrastructure;
- connection of supercomputer centers using the grid gateways;
- creation of a Web portal to access the system, the creation of user interfaces (preparation and – execution of user jobs, data transfer, license management, etc.);
- development of the security system.

In 2011, the major organizations implementing the project of creation of the RGN grid infrastructure were Research Institute "Voskhod", SINP MSU, JINR, T-Platforms, CC FEB RAS, TESIS Engineering Company.

Joint Institute for Nuclear Research, in cooperation with SINP MSU worked on the development and deployment of basic Grid services, as well as the adaptation of application software packages and development of problem oriented web interfaces to work with such applications.

The main grid services developed in frames of the RGN project are:

- Grid gateway to start jobs in the resource centers (SINP MSU);

[1] Supported by the Ministry of Communications and Mass Media of the Russian Federation, cont. № 0173100007512000020_144316

- service of multi-step jobs execution (Job Management Service, JMS) (SINP MSU);
- Information System (SINP MSU);
- resources' accounting subsystem (collection of data on the use of computing resources in the resource centers) (JINR);
- monitoring subsystem (monitoring of resources, services and jobs in a grid network) (JINR);
- users' and VO accounting subsystem (the consumption of resources by users and virtual organizations) (JINR);
- Service for Registration Resource and Grid services (JINR).

Below we describe the monitoring, accounting and registration services in more details.

2 Registration service

The main task solved by the resources and grid services registration service, is to store and provide information about resources and grid services in RGN environment. All the services' information is available for human user on the web interface and for other services of the Grid Network via RESTful interface. The information stored is sensible and is to be carefully added and changed only manually by administrators of the system using web-interface.

There are several types of registration information in the service: the grid-service level, common grid site information, and administrators' records. Grid service belongs to one and only one grid site (core grid also obey this rule). Each site or single service could be administered by different people, so access rights and roles could be management in a flexible way.

Grid site (resource center) registration information:

- Brief name and full names;
- Site unique ID;
- Link to the web site of the grid site, URL to the page with site usage rules;
- Used network domain;
- URI of site local information service (where the central information service pull data from);
- Contact and emergency e-mail addresses, administrator's work time;
- Postal address, latitude and longitude of physical location.

Main information on the grid-services:

- Grid site (resource center) service belongs to;
- Service type (e.g. computing element, GridFTP service, jobs management service, information index, etc.);
- Entry point service is accessible by;
- Access protocol;
- Current state: running, testing, outage;
- Status of the service: working, new, testing, certified, closed, suspended;
- Distinguished name of grid certificate of service manager;
- Brief service description, contact and emergency contact e-mail addresses (if differs from the site's ones).

In addition to this quasi-static information, it is possible to assign downtime periods to sites and services to change their status automatically for the maintenance time.

Initially there are four roles in the registration service: registration service admin, Grid Network admin, site admin, service admin. New roles with specified access and delegation privileges could be created using web-interface if necessary. To control and manage proper management of registration information, it was developed elaborated and flexible hierarchical structure for access granting and rights delegation. The fifth default role is authenticated but not authorized user (not registered in the service), for it only read access is opened.

Finally, the entered information on grid sites and their services is to be provided to the other

services in a machine readable way. On HTTP request, structured information is returned in JSON format [1].

3 Monitoring

The monitoring subsystem is used for collecting, storing and providing information on the status of Grid resources and services network, as well as information about running in the user's jobs. The current status of resources, tasks and other objects in the system is collected from all Grid services into a single database. Both static and dynamic information is used (e.g., the state of the queue on the computing cluster). The processed information is provided for Grid administrators, managers of virtual organizations (VO) and grid end-users via a web interface.

Most common tasks the monitoring deals 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 for the resources by Virtual Organizations and groups inside them;
- Execution monitoring, tasks and jobs submission, state changes and return codes;
- Resource usage information (especially CPU consumption; it is closely bound with the accounting service);
- Watching the quotas for resource usage by Virtual Organizations.

Main structural components of the monitoring system to provide the declared functionality:

- Module to collect main information on operation of grid network components;
- User job's information collector;
- Special data base with data management module for information processing and storage;
- Web interface modules and parts to prepare web reports.

For the purpose of gathering the initial information about monitored services, monitoring system queries the Registration Service for entry points of the services of grid resource centers and central services of the RGN. Then, in question to collect information on the state of resources and their utilization, main Information Service of the Grid Network is periodically queried. Information on the jobs and tasks are obtained in the same way from the Jobs Management Service.

Along with the monitoring of resource centers and users' jobs, monitoring system performs the functional testing of the core grid services. Such tests are to check if services are available from the Internet and do operate normally.

To store obtained data on the statuses of the resources and user jobs and tasks, the monitoring system addresses to the dedicated database. Just on the insertion, data undergo statistical preparation and aggregation, parts of the information came from different services or consequent events came in different time are being linked with each other. Then the processes data and intermediate results are available for querying via the web interface. Having aggregated and pre-processed information helps to significantly speed up database requests in case of huge amounts of monitoring records (for the Grid Network it was noticeable starting from three months or running).

4 Accounting service

The accounting subsystem is used for collecting, storing and providing information about grid computing resources consumed by user jobs. Information is provided to the billing system, grid administrators, managers of virtual organizations (VO) and grid end-users.

Jobs monitoring information and accounting data are taken mainly from JMS (job management service). JMS 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. 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) are to be taken from local Grid Resource Allocation Managers (GRAMs from the Globus Toolkit [2]) in resource centers via the special accounting service called GACCT. Then the accounting data are to e-linked with job information from JMS servers in the database. At the end, full aggregated accounting information is prepared in the database and is available to end-user via web-interface.

Accounting information is available on the site as several report views with tables and diagrams by resources, users and VOs. Real time jobs monitoring allows displaying on 3D globe how and where jobs are started and finished. 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 [3].

Conclusion

There were developed registration, monitoring and accounting services within the Russian National Grid Network. They are operating with all the project specific components and services of the RGN. Experience of the developing similar services gained in GridNNN [4] project was very helpful and vas heavily used in the RGN project. Developed services are successfully deployed and are in production now. The Russian National Grid Network project is still actively developing; work on the services will be continued.

References

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

[2] I. Foster, Globus Toolkit Version 4: Software for Service-Oriented Systems, IFIP International Conference on Network and Parallel Computing, Springer-Verlag LNCS 3779, pp 2-13, 2005.

[3] S. Mitsyn, S. Belov, Development of Real-time Visualization Service Based on Google Earth for GridNNN Project, Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 4th Intern. Conf. (Dubna, June28-July 3, 2010). Dubna: JINR, 2010.

[4] V.A. Ilyin et al., Design and Development of Grid-infrastructure for National Nanotechnology Network, Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 4th Intern. Conf. (Dubna, June28-July 3, 2010). Dubna: JINR, 2010.

SUPPORT FOR THE CMS EXPERIMENT AT THE TIER-1 CENTER IN GERMANY

J. Berger, C. Böser, T. Chwalek, M. Fischer, O. Oberst, G. Quast, N. Ratnikova[1], S. Röcker, S. Wayand, M. Zeise, M. Zvada

KIT - Karlsruhe Institute of Technology, KIT, Germany

[1] *also at ITEP - Institute of Theoretical and Experimental Physics, Russia*

Natalia.Ratnikova@kit.edu

The Compact Muon Solenoid (CMS) experiment is one of the two general purpose detectors operated at the Large Hadron Collider (LHC) at CERN. The CMS Collaboration includes 191 institute from 42 countries of the world. The Institut für Experimentelle Kernphysik (IEKP) at the Karlsruhe Institute of Technology (KIT) has a strong participation in the CMS programs in physics, statistics, detector study, software and computing. One of the major tasks is the support for the CMS Tier-1 Center at GridKa, KIT, including administration and operations of the experiment specific layer of the distributed computing infrastructure. We discuss the overall organization of support for the CMS activities at GridKa, the infrastructure and the tools used by the site administrators, and coordination with the GridKa and the CMS Central Computing Operations teams. We give an overview of related development projects conducted in this context. Finally, we draw some conclusions based on our experience of operating the Tier-1 for the CMS experiment during the active data taking period.

1. Introduction

The contribution of Germany to the Worldwide LHC Computing Grid (WLCG) [1] consists of a large Tier-1 centre, GridKa at the Karlsruhe Institute of Technology (KIT) and 7 Tier-2 centres at other national institutions or German universities. GridKa provides services to nine high energy physics experiments, among them the four LHC experiments ATLAS, Alice, CMS and LHCb. GridKa operates very closely also with many other Tier2 centres, mostly in eastern Europe. Thus, GridKa is at the heart of a very complex grid structure, with different computing models and workflows adopted by each of the supported Virtual Organisations (VOs). A clear separation line of responsibilities is defined at GridKa, including a formal service level agreement with each experiment. GridKa staff ensures the availability of the resources and is responsible for Grid middleware services. It is impossible for GridKa to provide experts for each of the experimental software set-ups and the specific workflows. Some operations require authorisation, which can only be granted to the members of the experiment. Therefore the responsibility for the experiment-specific tasks belongs to the group of experts within the experiment. Overall experience of GridKa T1 operations and LHC experiments representation are discussed in [2]. This paper highlights the support structures set up in Karlsruhe for the CMS experiment [3].

The Institut für Experimentelle Kernphysik (IEKP) [4] at the Karlsruhe Institute of Technology (KIT) is the fifth largest by membership from 192 institutes in the CMS Collaboration. IEKP members have strong participation in CMS physics data analysis, statistics, detector study, software, computing, and upgrade programs.

Our major task in computing is to provide smooth operation and efficient resource management at the CMS Tier-1 Center at GridKa. This includes configuration and administration of the experiment specific infrastructure at Tier-1, site monitoring, data management, and coordination activities.

Besides Tier-1 support, our institute members play leading roles in several other projects: enabling infrastructure and environment for the local CMS physicists to access and successfully use additional national resources at GridKa (NRG); set up infrastructure for CMS central workload management system based on GlideIn technology (GlideInWMS); development and validation of Physics Experiment Data Export (PhEDEx) project: CMS data consistency tool, validation tool suite,

storage accounting tool; meta-monitoring tool HappyFace. The majority of these tasks are credited as IEKP contribution to CMS central computing services. On the other hand they help our group to build the expertise necessary for operating the complex CMS environment and tools.

In the following sections we present our experience in providing support for CMS Tier-1. Section 7 is devoted entirely to the development projects.

2. Resources

The CMS Offline computing system [5] consists of a large number of geographically distributed centers organised in an hierarchical structure of computing tiers. A single Tier-0 center at CERN accepts data from the CMS Online Data Acquisition System for archival storage, calibration and prompt reconstruction. Tier-0 distributes raw and processed data to a set of large Tier-1 centers for secondary archival storage, organized data processing and data serving to a more numerous set of smaller Tier-2 centers. Tier-2 centers provide resources for analysis, calibration and Monte Carlo simulation activities. The new data produced at Tier-2s are transferred to Tier-1s for custodial storage. Tier-3 centers provide interactive resources for local groups and additional best-effort computing capacity for the collaboration.

The CMS Computing model [6] specifies nominal Tier-1 Resource Requirements in terms of CPU, Disk and Mass Storage capacity, data rate from storage, WAN transfer capacity, CPU node I/O bandwidth. The actual Tier-1 resource contributions and expected service levels are agreed in the context of the WLCG MoU [7]. Accounting of the CMS computing resources is available via REBUS [8], the Resource, Balance and Usage website for the whole of WLCG, including topology information, resource pledges, and installed capacities. Resources pledged by all CMS Tier-1 sites and by the CMS Tier-1 at Gridka in the year 2012 are presented in table 1.

Table 1. CMS Tier-1 resource pledges in 2012

Resource	CMS T1 total	GridKa CMS T1 contribution
CPU power	145 kHS06	3’750.0 kSI2k, 1’456 job slots
Disk space	22 PB	1’950.0 TB
Tape storage	45 PB	5’000 TB
Local Support	217 FTE months of credited service work	31 FTE months of credited service work

In addition GridKa provides to CMS 175,0 TB local and 75,0 TB WAN storage, 10 Gbps national regional network and 10 Gbps OPN international connection speed.

3. Site Performance Metrics

Due to the importance of the Tier-1 services a high level of availability and operability of the site is expected. CMS computing has developed a so-called Site Readiness metric, which takes into account various criteria, such as the level of success of the test jobs sent to the site or the results of data transfer load-tests continuously running between the sites. Unavailability during scheduled down time periods is also properly accounted for. Figure 1 shows CMS Tier-1 Site Readiness status, with a

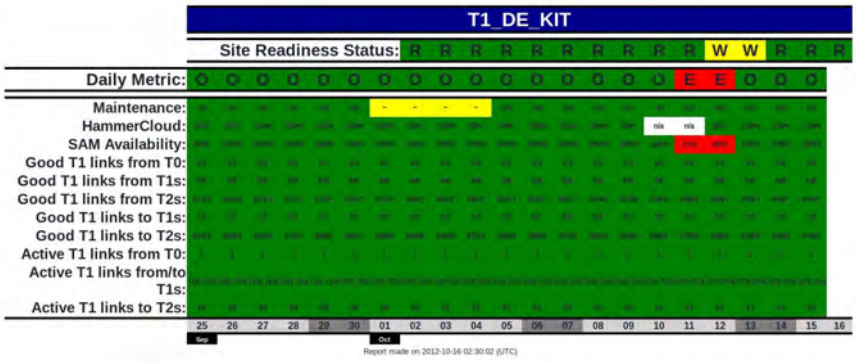


Figure 1: Site Readiness Status

short period of reduced availability below 90 percent which resulted in Warning status. The metric is updated and published daily. A cool-down time is automatically allowed for the site to recover after prolonged failures.

The failure of the automated test job is not necessary indicating the site unavailability. For example, on the multi-VO sites the system may no longer accept new jobs, if the virtual organisation has reached its fair-share limit. Carefully planned scheduling by the central operations helps to avoid this situation. Another example is the time when the system works under heavy load. Test jobs sent to the production system are competing for the resource with the real production jobs. The processing of the test jobs is slowed down, and the system appears to be unresponsive.

The monitoring of the site performance is a steadily ongoing effort [9]. CMS Computing community is constantly working to improve the monitoring and the overall efficiency of the CMS workflows.

Local site administrators and a worldwide distributed crew of central computing shifts are watching the system around the clock and apply monitoring and alarming procedures to the sites, central services, and operators.

4. Local Site Support Operations

Grid sites provide resources and basic Grid services and components, such as security, computing and storage elements, monitoring, accounting, workload management systems, information service and low-level file transfer service. The CMS Central Computing Operations team manages the large-scale distributed workflows over the Grid sites. These are two very different scopes. The ultimate goal of our local CMS support group is to match the resources and the workflows for smooth and efficient operation. This requires expertise in both areas.

Managing data [10] is by far the most labor-intensive task in Tier-1 support. The CMS distributed data transfer system [11] is the key ingredient which enables the optimal usage of all distributed resources. Data transfers from Tier-0 to Tier-1 sites must be done in a timely manner to avoid the overflow of the disk buffers at CERN. Simultaneously, the data are transferred in bursts to Tier- 2 level sites for analysis, and simulated Monte Carlo data produced at Tier-2 centers are moved to Tier-1 sites for archival. Additionally, data may be synchronized between different Tier-1 sites, and served to Tier-3 sites.

CMS uses PhEDEx tool [12] to initiate transfers and to keep track of data placement and transfer status.PhEDEx provides subscriptions and requests mechanisms to handle data operations. Every incoming data transfer or deletion request must be manually approved by the local data manager. Routine data operations include clean-up of obsolete data, maintaining consistency between the storage contents and the central data catalogues, monitoring and debugging transfer issues, attending meetings, providing prompt feedback on requests, managing PhEDEx software installation, configuration and upgrades, restarting PhEDEx agents, inspecting the log files, and providing support to the associated Tier-2 sites.

Another major task is support for data processing at the site. The job submissions to Tier-1 sites are managed by the CMS central operations. However a few actions for proper data handling are required from the site. For compact tape utilisation, sites are asked beforehand to create the tape families for the files that are to be produced and archived on tape. Site may be asked to pre-stage from tape input data required for re-processing or replicate files needed by many jobs. The output of the application jobs and the corresponding log files are first written to the local disk on the worker node. If necessary, the empowered on-site expert may login directly to the worker node, inspect the log files and troubleshoot any site specific problems. After job execution, the application output is staged out into the CMS namespace on the storage element. An additional merging step is applied to the small files before archival to tape in order to reduce the load on the mass storage system. Once the merging step is complete, the original unmerged output files can be removed. Sites are responsible for regular clean-up of those obsolete unmerged files. The CPU efficiency of the jobs is constantly monitored to identify any jobs with a low CPU to wall-clock-time ratio. This condition usually indicates a problem with data access, when jobs stay idle waiting for the required data. Monitoring of the total numbers of

running and queued jobs per experiment helps to distinguish general infrastructure problems from the application specific ones.

Site support services and procedures are part of the Karlsruhe CMS group obligations to the Collaboration formalised in the Maintenance and Operations plans, and are credited with 31 months of service work per a calendar year.

To improve the quality of support and to ensure proper share of the expertise, we have introduced an expert rotation scheme. For the period of six to eight weeks a senior member of the group takes care of the routine operations, *i.e.* responding to service tickets, handling tape families and data transfers, troubleshooting, reporting, etc. Data consistency checking, software upgrades, site configuration, system tuning and optimisation, and other specific tasks are covered by the dedicated experts. The rest of the group, including junior members, provides support by taking local site monitoring shifts, and by participating in various development projects.

5. National Resources at GridKa - NRG

Tier-1 centers provide both regional and global services. The usage by the local community however should not interfere with the ability of the Tier-1 center to fulfil its obligations towards the whole CMS. The German CMS community (DCMS) is able to use a share of 1366 cores of the current KIT Tier 1 resources. CMS users which registered for the dcms group in the CMS VOMS server are able to create a VOMS proxy with a *cms:/cms/dcms* VOMS extension. These users are mapped at KIT to one of the dcms pool accounts which possess a common

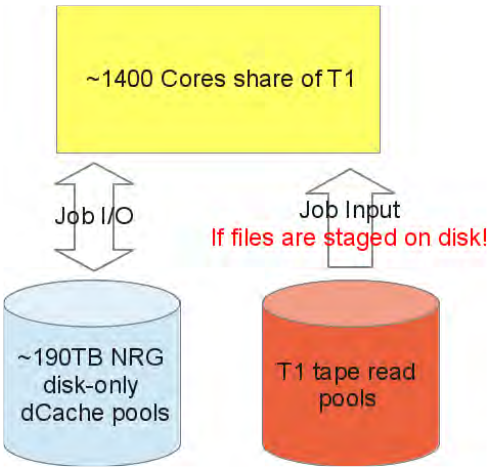


Figure 2: Sketch of the setup providing national resources at GridKa (NRG). Access to CMS T1 datasets is granted, if the required files are staged on disk

DCMS group fair share within the batch system accounting. To further limit the impact of the NRG users on the Tier-1 production operation, the maximum number of concurrently running DCMS pool account jobs is limited to 2000. Unused CPU resources within the NRG budget are available to the WLCG VOs.

Additionally to the computing resources DCMS users have read and write access to 190 TB of disk-only dCache storage via a specific storage element. This storage is foreseen as scratch space for users. Furthermore, access to CMS Tier-1 data is allowed as soon as the required official CMS datasets are staged on disk. A sketch of this setup is given in Fig. 2. Together the DCMS resources at KIT can be seen as a Tier-3 at the Tier-1, named ”National Resources at GridKa” (NRG). As the whole setup is directly integrated within the T1 resources, no additional effort is needed to maintain the CMS software environment which the NRG users might need.

6. Coordination Activities

To insure an efficient use of Tier-1 resources and coherent and friendly computing environment for CMS users, a smooth cooperation of all involved parties is necessary.

The aim of the coordination efforts of our Tier-1 local support group is to maintain close contact both with CMS central computing operations and GridKa support teams, and also to stay alert to individual users' requests.

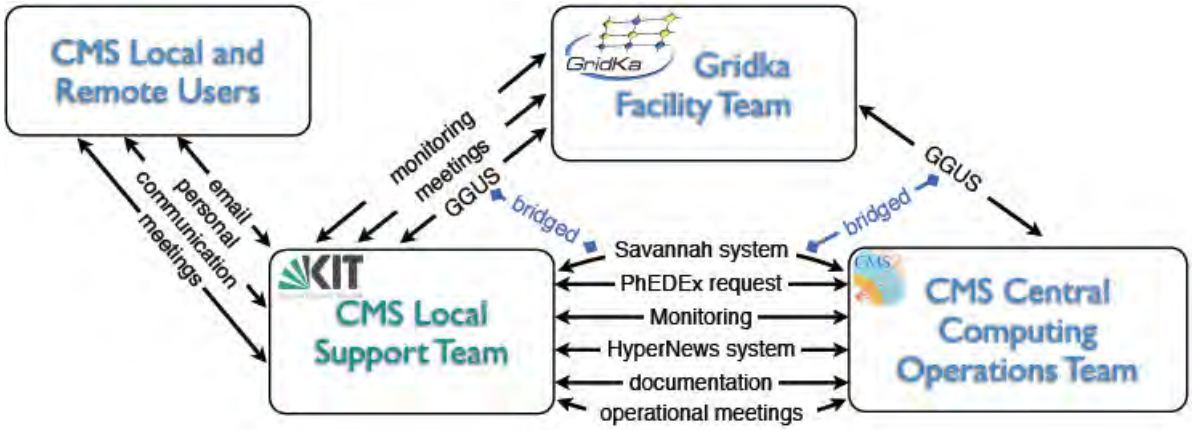


Figure 3: Involved parties and communication channels

Different channels have been established for the exchange of different types of information. Figure 3 illustrates communication channels between the involved parties. As one can see in the chart, CMS users do not communicate directly with the GridKa facility team. CMS Central operations may contact GridKa directly via a bridging mechanism between Savannah [13] issue tracking system used for internal CMS communications, and the WLCG problem tracking system GGUS [14]. According to CMS policies, bridging to GGUS is only used for well identified problems related to WLCG critical services at Tier-1 sites.

7. Development Projects

Senior experts in our group have extensive experience in the development of computer systems and continue to contribute to development of CMS computing infrastructure and tools. Playing a leading role in many CMS computing projects, they also involve students and young scientists in this work. The immediate benefits are the build-up of the group expertise, training, exchange of experiences, and a team-work spirit.

Development work is often conducted in collaboration with other groups, institutes and experiments. The results are regularly presented at various computing conferences and workshops, included in master or diploma and PhD theses, and published in scientific journals.

Most of this work is credited as CMS central computing services. This fits nicely with the CMS intention to extensively involve the youth in the technical aspects of scientific work. There is a general CMS requirement for every new member of the Collaboration to provide six months of service work to become CMS author.

In this section we give an overview of the most prominent development projects conducted at IEKP.

7.1. GlideinWMS

In order to facilitate an efficient management of the organised data reconstruction and Monte Carlo production, CMS uses the grid meta-scheduling system GlideinWMS [15]. The GlideinWMS workload management system is based on a virtual private Condor [16] pool. It is composed of several elements [17], and some of them can be multiplied for improved scalability.

GlideinWMS components and its flow of processes are shown in Figure 4. The operation logic [17], [18] of this system tries to maximise the amount of user jobs, while minimising the amount

of wasted resources. It does this by keeping a steady pressure on the Grid pools; as long as there are jobs in the schedd queues that could potentially run on a Grid pool, a fixed number of pilot jobs is being kept in that pool queue. However, as soon as there are no more suitable jobs waiting in any of the schedds, no more pilot jobs are submitted. If any Glideins start after all the suitable user jobs have started, the Glidein itself will exit within a few minutes.

The pilot jobs are being submitted by the Glidein Factories, but it the job of the VO Frontends to decide how many pilot jobs to keep in each Grid pool. This number is calculated by matching the attributes of the user jobs, provided by the schedds, to the attributes of the Grid pools, provided by the Glidein Factories. If the number of matches is higher than the desired pressure, the Glidein Factories are told to keep the pressure. Else, the pressure is reduced as appropriate.

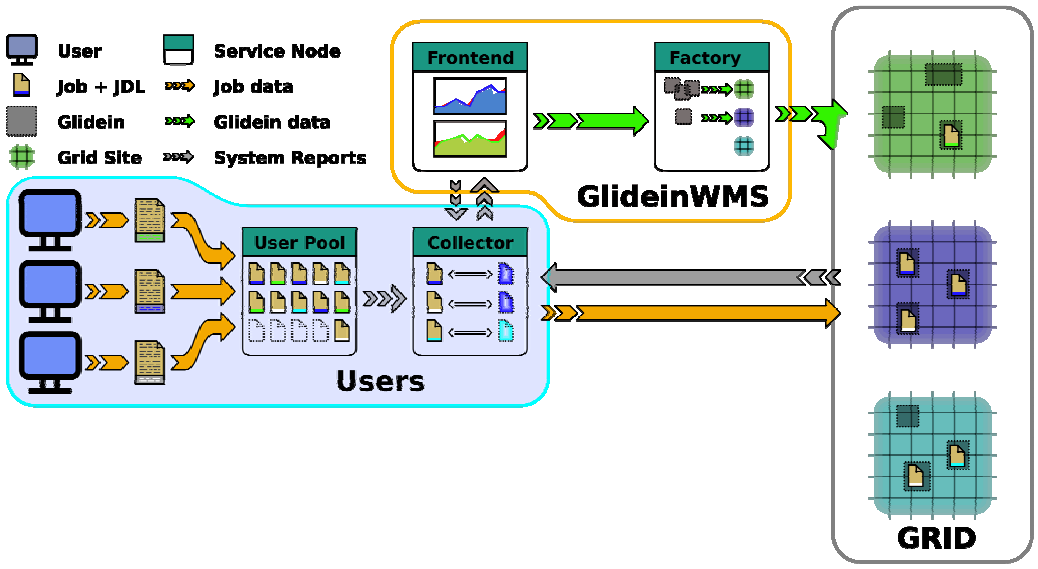


Figure 4: GlideinWMS components and processes flow

Once a pilot job starts on a Grid resource, it first validates it. This includes looking for the appropriate software libraries, ensuring sufficient disk space is available, and so on. Once these tests successfully complete, it configures the Condor daemons (i.e. the startd and supporting code) and starts them. All control is delegated to them, and the pilot job wrapper just waits for their termination to do the final cleanup.

Once the Condor daemons start, they behave like in a dedicated Condor pool. The startd registers back with the Condor central manager and waits to be matched. Once a suitable job is found, the schedd holding the job will contact the startd and the job starts running. The startd can run multiple consecutive user job, to keep the pilot wrapper overhead low for very short jobs. However, the pilot needs to end within the Grid back slot lease time, so the startd will stop accepting new jobs if they cannot complete in time.

The system has been designed to be highly scalable [19]. CMS commissioned this system into production at CERN/FNAL/UCSD as primary workload management system for the production and analysis jobs in HEP.

IEKP contributed in this work by providing expert support for the setup, configuration and tuning of the central components, the most complicated parts of the GlideinWMS deployment.

7.2. GlidInWMS access for users to NRG

The IEKP is currently in the last steps of establishing the usage of a private GlideinWMS system for IEKP physicists. Within this IEKP GlideinWMS factory the resources of the NRG become accessible from local IEKP login machines. The idea is to provide semi-interactive access as provided by GlideinWMS to the NRG Grid resources to IEKP users to help users to overcome the flaws of Grid computing, like the usual overhead in debugging, variety of tools, and the lower efficiency compared

to the use of local resources.

The users will be able to submit the jobs locally via condor to the GlideinWMS factory, which handles all Grid related work. All Grid related problems will be intercepted by the GlideinWMS factory and are therefore directly in the hands of a Grid expert. The user thus can focus on the optimisation of his analysis code and will mostly face self-caused problems related to his executable or data.

7.3. PhEDEx Project

The PhEDEx (Physics Experiment Data Export) project provides the data placement and the file transfer system for the CMS experiment. Created in 2004, PhEDEx has by now become a mature software product. It currently runs on over hundred of CMS distributed computing sites, performing a variety of tasks such as managing data subscriptions and transfers between the sites, staging data from tape, data removal, storage consistency checking, bookkeeping and monitoring all related activities.

Originally implemented as a set of specialised tools, PhEDEx has been substantially refactored. It now adopts a concept of an open framework [20] providing a set of generalised technical solutions, which could also be used standalone. Examples are Core Agent, AgentLite, Namespace, Data Service, Website frameworks, and LifeCycele Agent.

7.4. PhEDEx Namespace Framework

One of our major areas of contribution is Namespace Framework, which provides a communication layer to various types of storage systems and is used for data consistency checking. Figure 5 demonstrates the use of Namespace Framework for the data consistency checks over distributed CMS sites. Communication between the operators and the sites happens via central PhEDEx database. The agents running at the sites check for test requests to be performed on the local data. The results are uploaded back to the database and published to the web site using PhEDEx Data Service.

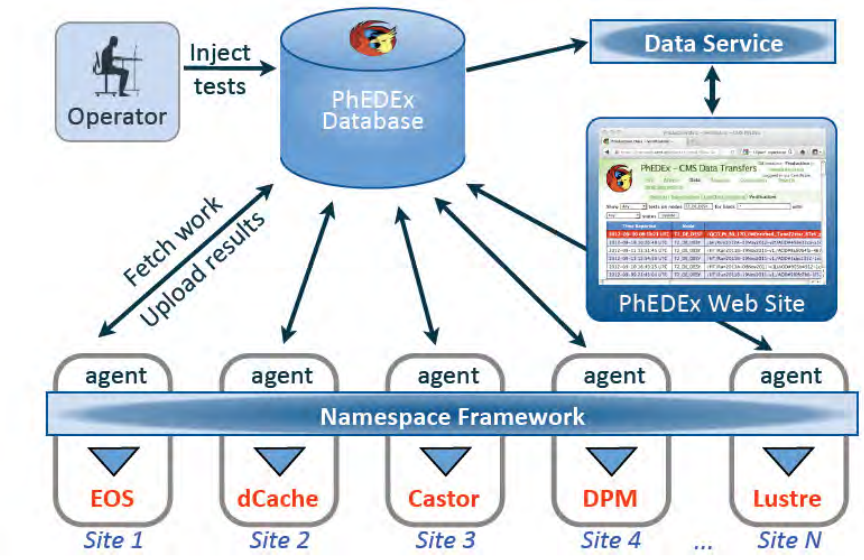


Figure 5: Namespace framework is used by PhEDEx agent for data consistency checking at distributed Grid sites

We have used our Tier-1 site for large scale testing aimed at improving the performance of the consistency checking agents. The time required for a full check of all CMS production data stored at the site have been reduced from several days to less than twelve hours. This allowed to use the system for regular data consistency checks throughout the CMS Tier-1 sites. It was shown that further three-fold improvement of timing can be achieved by using storage dumps instead of accessing the storage directly. These storage dumps are produced by the sites and used for detecting orphaned data not registered in the CMS central file catalogue.

This year, together with CMS data operations team, we have organised a CMS-wide campaign to add support for various storage technologies used at Tier-2 sites. By now automated consistency checks are running at over 50 % of CMS Tier-2s.

7.5. PhEDEx Validation

We keep track of new versions of the three PhEDEx components website, data service and agents, and provide information on the status of the releases, known bugs and bug fixes, and feature requests.

Besides this "bookkeeping" task we used the KIT Tier-1 site as a prototype for large scale testing for new PhEDEx agents releases. Basic test scenarios like testing the subscription and deletion of datasets on the KIT Tier-1 site were performed as well as advanced and specialised test scenarios, like deleting a dataset at Tier-0 and subscribe it to Tier-1, while the deletion is still ongoing.

The major project in the field of PhEDEx validation is the development of a validation tool suite based on the lifecycle agent to test the behaviour of new data service releases in a testbed using different roles (admin, data manager, site manager). Possible scenarios include testing that site admins cannot inject data anywhere, nor approve or disapprove requests and verifying that the data manager for T1 X can do so for T1 X, but not for T1 Y. The final goal is to run the test scenarios in a reliable and automated manner. Another tool is currently developed to test the email notifications of the PhEDEx system prior to the release of a new data service version. Whenever a transfer/deletion request is made via the PhEDEx system, the global PhEDEx admins and the responsible group admins, data managers and site managers of the respective groups and sites should receive an email so that they can further proceed and either approve or disapprove the request. In order to allow for a smooth workflow it is essential that this notification works.

7.6. Storage Accounting project

All major LHC experiments need to measure real storage usage at the Grid sites. This information is equally important for resource management, planning, and operations. To verify the consistency of central catalogs, experiments are asking sites to provide a full list of the files they have on storage, including size, checksum, and other file attributes. Such storage dumps, provided at regular intervals, give a realistic view of the storage resource usage by the experiments. This brought us to the idea of monitoring of storage use based on storage dumps, which resulted in storage accounting project [21]. IEKP contributes in all aspects of this project, including analysis of use cases and requirements, and concrete implementation. This work is conducted in close collaboration with WLCG, CERN IT Experiment Support group, ATLAS and LHCb experiments, CMS Monitoring Task Force [22], CMS PhEDEx development team, CMS central data operations, and a set of CMS pilot sites volunteered to participate in testing.

The Storage Accounting tool is re-using PhEDEx code base and components with some extentions. Particularly, Namespace framework has been extended to provide support for parsing of different types of storage dumps. It is envisioned to use PhEDEx Agents technology for driving the information provider on the local site. The API used to upload and retrieve the disk usage data to and from database is modeled based on PhEDEx Data Service. The data is stored in Oracle database at CERN. However it is using a separate DB instance to keep space usage information separately from PhEDEx data transfer details. Work with the remote sites is ongoing to deploy and test tools for producing the storage dumps. First version of Storage Accounting Data Service has been deployed on the production CMS web server at CERN.

7.7. HappyFace Project for Site Monitoring

The Tier-1 center at GridKa is monitored by a group consisting of local CMS members. The main tool for this job is the HappyFace framework. HappyFace is a meta-monitoring framework which gathers information from different monitoring sources, processes this information and provides an overview of all relevant information, which allows real-time site monitoring for both shifters and experts. HappyFace started as a development project at KIT and is now used by several other German grid sites, both ATLAS and CMS, to monitor their sites. It is also employed centrally by CMS to monitor the batch systems of all CMS Tier-1 and Tier-2 centers. More details about the architecture of the HappyFace framework and the current development can be found in a separate article in these proceedings [23].

8. Summary

The combination of Research Lab and Technical University at KIT allows one to combine Tier-1 operational services with computing development projects and thus provides an excellent environment for training of young scientists in the areas of high-performance and high- throughput computing.

Expertise within the CMS Tier-1 local support group at IEKP spans a wide range of Grid computing topics, including job submission, data management, monitoring, accounting, consistency checking and validation tools. Expert knowledge combined with coordination efforts and an expert rotation scheme help to provide sustained support of CMS-specific Tier-1 services during intensive data taking at LHC.

Members of our team provide strong contribution to CMS core computing development and support, and are open for collaboration with other sites and experiments.

References

[1] Knobloch J *et al.* 2005 ”LHC Computing Grid Technical Design Report” CERN-LHCC-2005-024.

[2] Heiss A, Petzold A, Zvada M, “Overall experience of GridKa T1 operations and LHC experiments representation”, ibid pp. 158-164.

[3] The CMS Collaboration 2008 ”The CMS experiment at the CERN LHC” JINST **3** S08004.

[4] Institut fu¨r Experimentelle Kernphysik, http://www-ekp.physik.uni-karlsruhe.de/

[5] CMS Collaboration 2005 “CMS: The computing project. Technical design report,” CERN-LHCC-2005-023.

[6] Bonacorsi D 2007 ”The CMS computing model” *Nucl. Phys.* B *(Proc. Suppl.)* **172** 53-56.

[7] WLCG Memorandum of Understanding, http://wlcg.web.cern.ch/collaboration/mou

[8] WLCG Resource, Balance and Usage website, http://wlcg.web.cern.ch/rebus

[9] Kreuzer P 2011 “CMS computing performance on the GRID during the second year of LHC collicions”, Proc. of NEC 2011, Varna.

[10] Kaselis R 2012 ”CMS Data Transfer operations after the first years of LHC collisions”, Proc. of CHEP 2012.

[11] N. Magini, N. Ratnikova et al 2011 ”Distributed data transfers in CMS,” J. Phys. Conf. Ser. **331**, 042036.

[12] Rehn J et al. 2006 “PhEDEx high-throughput data transfer management system” Proc. of CHEP06, Mumbai.

[13] Savannah - the LCG software development portal, https://savannah.cern.ch/

[14] T. Antoni et al “WLCG-specific special features in GGUS,” J. Phys. Conf. Ser. **219** (2010) 062032.

[15] GlideinWMS Web site, http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.html

[16] Condor Project: http://research.cs.wisc.edu/condor/

[17] Sfiligoi I 2008 “GlideinWMS - A generic pilot-based Workload Management System” Proc. of CHEP2007.

[18] M. Zvada M , Benjamin D, Sfiligoi I 2009 “CDF GlideinWMS usage in grid computing of High Energy Physics” Proc. of CHEP 2009, Prague.

[19] Bradley D, Sfiligoi I, et al 2009 ’Interoperability and scalability within glideinWMS’, Proc. of CHEP 2009.

[20] Sanchez-Hernandez A, Egeland R, Huang C-H, Ratnikova N, Magini N and Wildish T, 2012 ”From toolkit to framework - the past and future evolution of PhEDEx”, Proc. of CHEP 2012.

[21] Huang C-H, Lanciotti E, Magini N, Ratnikova N, Sanchez-Hernandez A, Serfon C, Wildish T, Zhang X “Data Storage Accounting and Verification at LHC experiments”, Proc. of CHEP 2012.

[22] Bauerdick L and Sciaba A 2012 ”Towards a global monitoring system for CMS computing”, Proc. of CHEP 2012.

[23] S. Röcker et al, ”Meta-monitoring with the HappyFace project”, ibid pp.200-203.

A DISTRIBUTED BRANCH AND BOUND METHOD FOR BOINC DESKTOP GRIDS

Bo Tian, M. Posypkin
Moscow State University, Moscow, Russia
yesyestian@gmail.com, posypkin@isa.ru

The Branch-and-Bound algorithm is fundamental for a variety of applications in Combinatorial Optimization. This paper presents a distributed Branch-and-Bound algorithm based on a popular BOINC platform for distributed computing.

1 Introduction

The Branch-and-Bound method (B&B) is a very efficient and well-known technique to solve combinatorial optimization problems such as Traveling Salesman Problem, Knapsack, Integer Programming, Vertex Covering and many others. This algorithm allows to reduce considerably the computation time required to explore the entire solution space associated with the problem being solved. However, the exploration time remains considerable, using parallel or distributed processing is one of the major and popular ways to reduce it. Many parallel B&B approaches have been proposed so far, such as [1]. However distributed grid-oriented B&B implementations are not well studied.

Normally, B&B algorithm consists of four main parts:

1. The branching rule that consists in a strategy for expanding a parent (sub) problem into child sub-problems. The leaf-nodes of the expanded tree represent all possible solutions;
2. The selection rule that chooses some of the current evaluated sub-problems for expansion using a search heuristic, e.g. Best-First, Depth-First, Breadth-First or others;
3. The bounding rule that consists in reducing the search space by eliminating sub-problems that do not yield to the optimal solution;
4. The termination condition which takes place when all sub-problems are either de-composed or eliminated.

This work discusses the design and the deployment of this algorithm on a computational grid. We parallelize the B&B algorithm in order to make it suitable for the coarse-grained work distribution. This approach was implemented using BOINC - an open source volunteer computing platform. Our implementation works as follows:

1. Master computer (BOINC server) performs server B&B steps and generates the first possible solution from search space then stops calculating immediately and goes to the second step.
2. Master computer creates a collection of work-units based on the terminal nodes of the explored sub-tree and sends work-units to the clients. Load balancing and sending work-units is handled by BOINC.
3. A client receives work-units, extracts and processes sub-problems, performing the given number of steps or less. After that the client returns the result (best obtained solution and the remaining sub-problems which need to be processed by other clients) back to master computer.
4. Master computer collects all results from clients, chooses the best from the collected solutions or re-allocates sub-problems to free clients.
5. This process goes to end unless all sub-problems have been processed. Then master computer chooses the final best solution.

We evaluate and demonstrate the efficiency of the proposed approach on large-scale knapsack instances. In the future we plan to try different strategies to compose work-units and study its impact on

the efficiency of the algorithm. We also plan to use this framework for more practical problems like transport problems, managing network flows, engineering design etc.

2 BOINC Platform

The Berkeley Open Infrastructure for Network Computing (BOINC) is an open source middleware system for volunteer and grid computing. It was originally developed to support the SETI@home project before it became useful as a platform for other distributed applications in areas as diverse as mathematics, medicine, molecular biology, climatology, and astrophysics. The intent of BOINC is to make it possible for researchers to tap into the enormous processing power of personal computers around the world.

BOINC has been developed by a team based at the Space Sciences Laboratory (SSL) at the University of California, Berkeley led by David Anderson, who also leads SETI@home. As a high performance distributed computing platform, BOINC has about 451,260 active computers (hosts) worldwide processing on average 5.655 PetaFLOPS as of February 2012. The framework is supported by various operating systems, including Microsoft Windows and various Unix-like systems including Mac OS X, GNU/Linux and FreeBSD. BOINC is free software released under the terms of the GNU Lesser General Public License (LGPL).

3 Computation model

As previously mentioned, normally Branch-and-Bound (B&B) algorithms have four main parts. Here our model also based on these four parts, but reconstructed to suit the distributed structure.

Step 1: Sorting, master finds the first possible solution

First sort the objects in non-increasing order of their profit/weight ratios. It is needed for Linear relaxation upper bound. Then master finds the first possible solution and gets the incumbent value. In the following example the first incumbent solution is [1, 0, 0, 1, 1, 0, 1] (Fig. 1). This value serves as a lower bound on an optimum and is used to eliminate redundant branches.

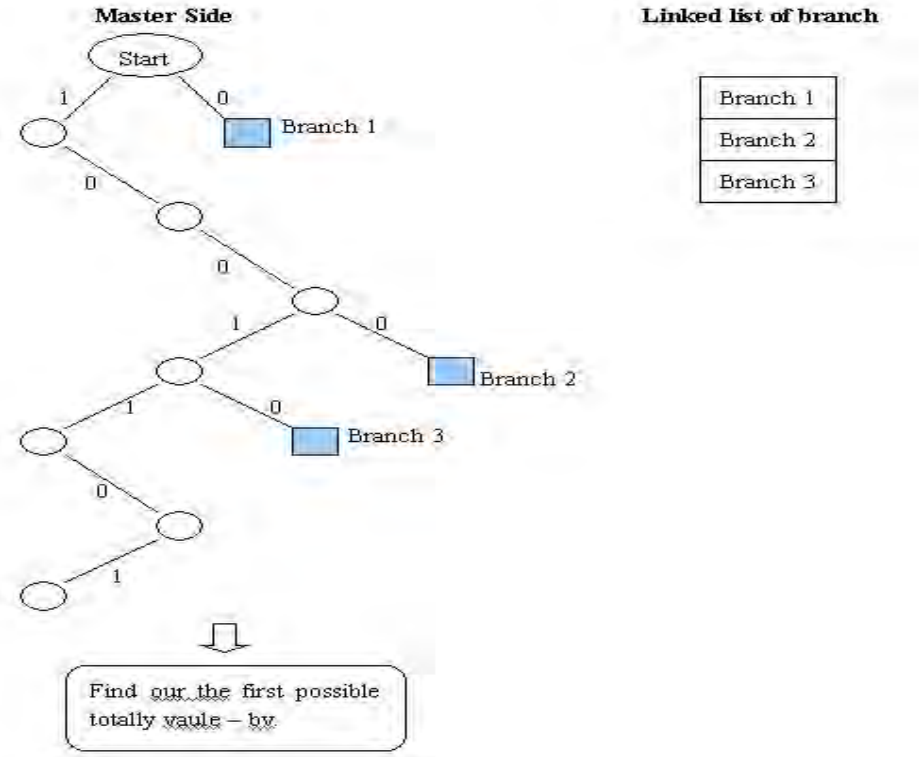


Figure 1: Master finds the first incumbent solution

Step 2: Master creates distributed Work-Unit

Master traverses the search tree and uses bounding function locally to check all possible branches, and delete the infeasible branches (Fig. 2). Master creates work units from the pending nodes of a tree using DC-API, and creates parcel based on linked list and the array of products' weight and price. (Fig. 3), here in the figure, we use "NUM" to represent total number of products, "I[NUM]" to represent the states of each products: "0" means not in bag, "1" means already in bag, and "–1" means still not be considered. "K" to represent how many products be calculated, "CW" to represent current weight in bag, so "CP" is the current price in bag and "BP" represents the best price, it's a global value from master server.

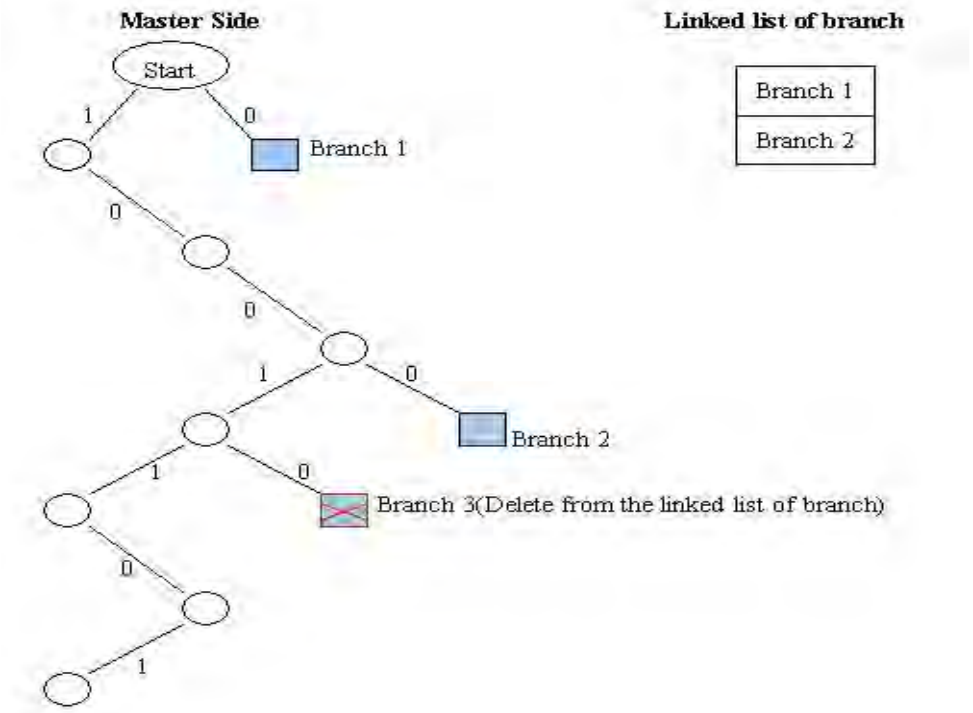


Figure 2: Master side bounding

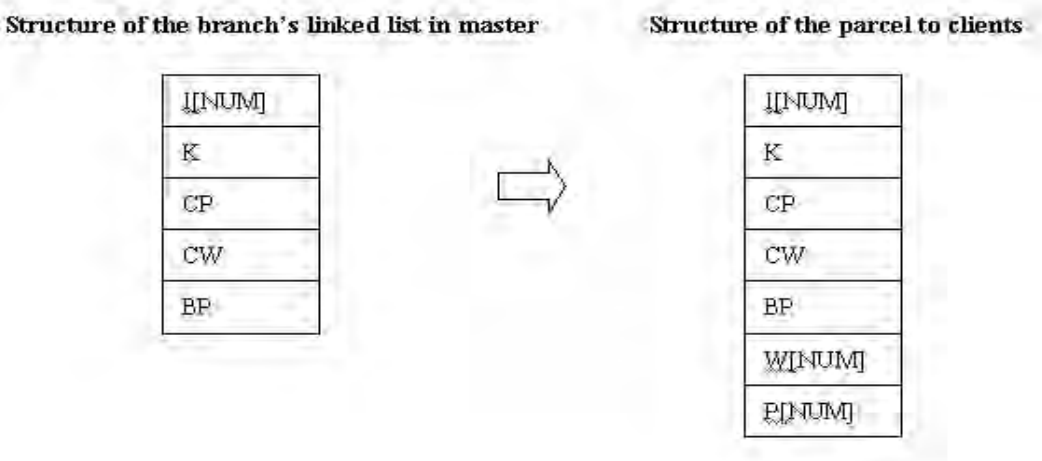


Figure 3: Work-unit format

Step 3: Client received subtask and processing

Client receives sub-task and parcel from master resolving the parcel to linked list. And running

local branch and bounding algorithm continue finding feasible solutions. Here we designed a Value MAX_SETP, in order to limit the maximum steps Client can run. When the calculate step is more than MAX_SETP, client stops running, and returns parcel with the rest of branches and the information of best value (If got a better value) (Fig. 4).

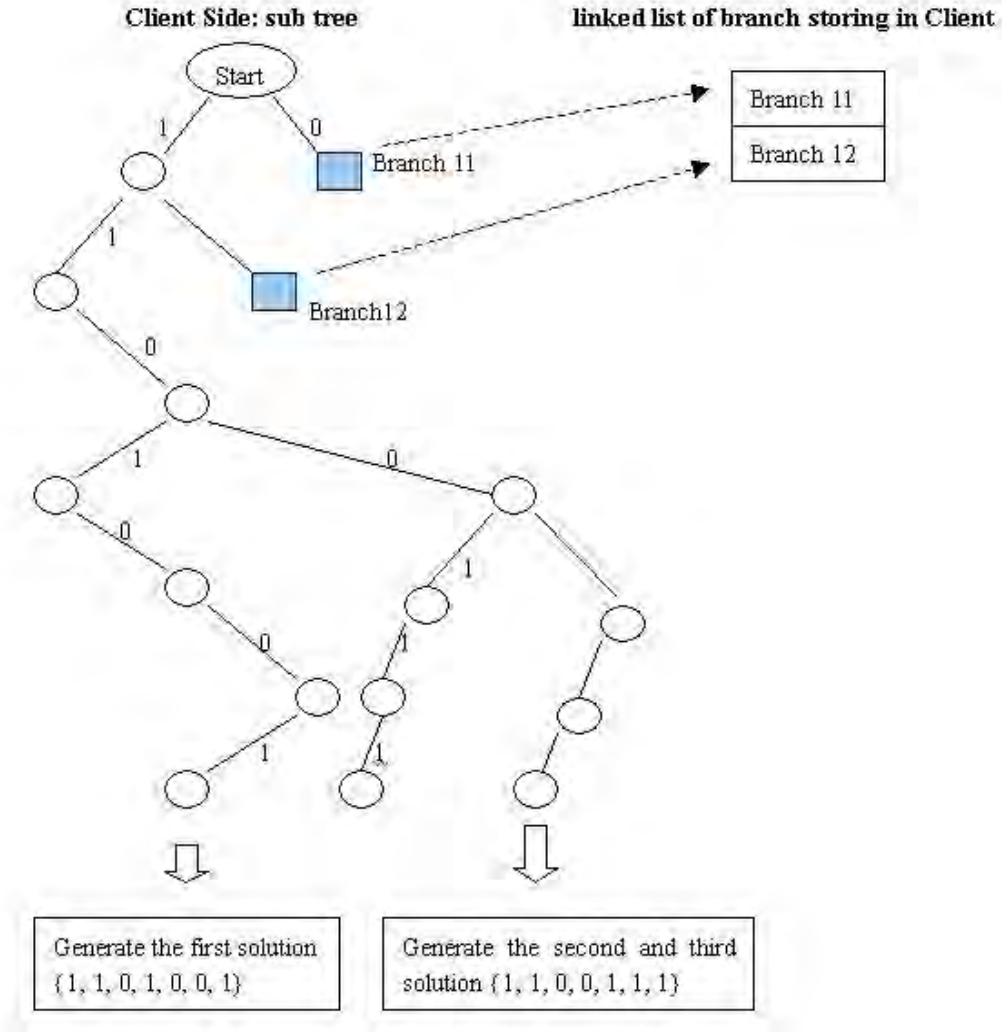


Figure 4: Client processing

Step 4: Master processes the results, then redistributes them

Master computer collects all results from clients, chooses the best from the collected solutions or re-allocates sub-problems to free clients.

4 Deployment in BOINC platform

We created a special BOINC project aimed at solving knapsack problems. The project was created with the help of SZTAKI Desktop Grid package [10] which is a featured BOINC distribution. Both server and client parts of the distributed B&B Algorithms were implemented using DC-API library [11]. The experimental deployed infrastructure for the project comprises one 64-bits dual core nodes acting as BOINC server and another computer as a client, the machines are connected to each other through a direct 100Mb link. (Fig. 5)

To be added into a BOINC project, applications must incorporate some interaction with the BOINC client: they must notify the client about start and finish, and they must allow for renaming of any

associated data files [8], so that the client can relocate them in the appropriate part of the guest operating system and avoid conflicts with workunits from other projects.

Once the data has been collected back in the server, some processing must be done to validate and assimilate it. In our case, we need to get the result from each workunit, and then check if there are still branches need be processed, or there is only result. The branches should recreate new workunits and resend to clients.

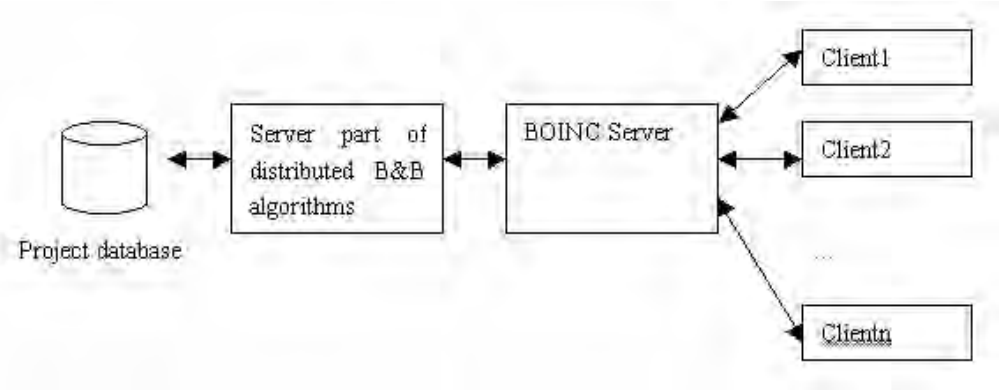


Figure 5: Physical structure of the system

Conclusion and perspectives

In this paper, we have presented the distributed Branch-and-Bound algorithm implemented on the BOINC platform. We have proved that the algorithm finds the best solution and that it terminates only after finding it. Next steps will be to create and use more clients to check if this approach will work for large-scale distributed system with large-scale data.

References

[1] B. Gendron and T.G. Crainic. Parallel Branch and Bound Algorithms: Survey and Synthesis. Operations Research, 42:1042–1066, 1994.

[2] Gendron B., Crainic T.G. Parallel Branch-and-Bound Algorithms: Survey and Synthesis. Rapport technique, Montreal (Canada), Centre de recherché sur les transports, May 1993.

[3] Anderson D. P.: BOINC: A system for public-resource computing and storage. [in:] Fifth IEEE/ACM International Workshop on Grid Computing, 2004, pp. 4–10.

[4] J.Peknyand D.Miller, A parallel branch and bound algorithm for solving large asymmetric traveling salesman problems, Mathematical Programming, 55(1992), pp.17-33.

[5] Lai (T.) et Sahni(R.). – Anomalies in parallel branch-and-bound algorithms. Communication of the ACM, vol. 27, June 1984, pp. 594-602.

[6] Roucairol (C.). – Parallel computing and combinatorial optimization. In : Proc. Of Tutorials Conf. Of the EURO XIII, Operational Research Designing Practical Solutions. July 1994.

[7] V. D. Cung, S. Dowaji, B. Le Cun, T. Mautor, and C. Roucairol. Parallel and distributed branchand-bound/A* algorithms. Technical Report 94/31, Laboratoire PRISM, Universit´e de Versailles, 1994.

[8] D. P. Anderson. BOINC: A System for Public-Resource Computing and Storage. 5th IEEE/ACM International Workshop on Grid Computing. November 8, 2004, Pittsburgh, USA

[9] P.Laursen, Simple approaches to parallel branch and bound, Parallel Computing, 19(1993), pp.143-152.

[10] Kacsuk P., Kovacs J., Farkas Z., Marosi A. C., Gombas G., Balaton Z.: SZTAKI Desktop Grid (SZDG): A Flexible and Scalable Desktop Grid System. Journal of Grid Computing, vol. 7, No. 4, 2009, pp. 439–461.

[11] Balaton Z., Gombas G., Kacsuk P., Kornafeld A., Kovacs J., Marosi A. C., Vida G., Podhorszki N., Kiss T.: Sztaki desktop grid: a modular and scalable way of building large computing grids. [in:] Proc. of the 21th Int. Parallel and Distributed Processing Symposium, Long Beach, California, USA, 2007, pp. 1–8.

PRACTICAL EFFICIENCY OF OPTIMIZING COMPILERS IN PARALLEL SCIENTIFIC APPLICATIONS

A.V. Bogdanov[1], I.G. Gankevich[2]
[1]*Saint-Petersburg State University, Russia*
bogdanov@csa.ru
[2]*Saint-Petersburg State University, Russia*
gig.spb@gmail.com

Optimizing compilers are essential for building any scientific application, however they are not general purpose tools. Although, many compilers offer similar functionality, different optimization strategies as well as code structure can lead to different performance results [1]. Additionally, modern scientific applications often solve large-scale problems concurrently on a set of processors thus demanding not only serial but parallel code optimizations. So, choosing the right compiler for a particular problem is a topical question of today.
In the paper a variety of commercial and open source optimizing compilers are compared in terms of their functionality. Then their relative performance is measured benchmarking different sets of algorithms and scientific applications classified by their problem domains. The final results are presented as a cumulative compiler rating scored in a particular problem domain. Based on this rating conclusions are made.

1 Introduction

History of optimizing compilers implementing parallelization can be divided into three major milestones showing emergence of different parallel technologies. The first standardized parallel technology introduced in 1994 was MPI (Message Passing Interface) exposing coarse-grained parallelism of a homogeneous cluster. The second one standardized in 1995 was Pthreads (POSIX Threads) exposing fine-grained parallelism of a shared memory multi-processor or multi-core homogeneous system. The most recent standard released in 2008 was OpenCL (Open Computing Language) that can be used to exploit parallelism of a heterogeneous hybrid system consisting of multiple CPUs and GPUs. These standards are bundled into compilers either as libraries, as a set of directives or as means of automatic parallelization.

Analysis of compiler output was traditionally performed using sophisticated instrument tools, however better understanding of compiler's work can be obtained using optimization reporting options. These options make compiler produce valuable information on succeeded and failed optimizations showing corresponding lines of code. Optimization reports work in harmony with conventional instrumentation tools (e.g. Valgrind, Oprofile) providing justification of compiler decisions and optimization hints and are especially useful when translating parallel code.

Compilers have different level of support for introduced parallel technologies and current top level of each technology can be outlined as follows (Figure 1):

1. MPI library support,
2. Pthreads auto-parallelization support,
3. directive-based GPU parallelization support.

Library-level support for open standards (unlike the proprietary CUDA) is the default level for any compiler and at this level there is not much compiler can optimize. The efficiency of low-level system parallel library is dependent on its implementation and not the compiler used to link it to a program, not to mention that MPI library is distributed also by “non-compiler” vendors (Platform MPI, OpenMPI). Therefore, it leaves discussion of library-level support for MPI out of topic.

Directive-based parallelization and auto-parallelization support are the levels where compiler has the most control over parallelization process and they are the most useful levels for compiler benchmarking purposes. Directive-based approach may include optimizations involving reduction of

redundant data movement, elimination of unneeded synchronization points and others. Auto-parallelization level adds an heuristic algorithm to determine the most suitable program part to perform parallelization. All in all, benchmarking directive-based and automatic parallelization of compilers may give insight into their overall efficiency.

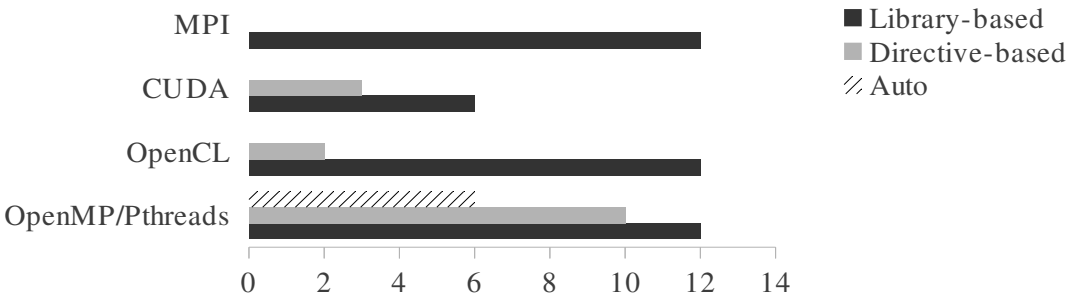


Fig. 1: Different levels of support for parallel standards (both open and proprietary).
The numbers show amount of compilers supporting technology at a specified level.

To summarize, from the parallel applications developer point of view the most useful compiler facilities aiding in parallelization process are support for compiler-based parallelization and optimization reporting. In the first case, parallelization reveals compiler's ability to optimize program in multi-threaded environment and for benchmarking purposes both CPU and GPU parallelization should be considered (Section 2 and 3). Finally, optimization reports are useful to reveal compiler's decisions in problematic lines of a source code (Section 4). So, compiler's ability to parallelize on par with optimization reports can enhance program performance.

2 Compiler-based parallelization efficiency

Compiler-based parallelization is the simplest approach to optimize and boost resource-intensive programs, and other than that, it is also a good way to show efficiency of compiler's optimizations with a view to parallel code. One of the most useful optimizations for scientific applications are those involving loops such as loop tiling, loop unrolling and loop interchange, as they maximize cache usage when processing large arrays of data [1]. They often work in harmony with other kinds of optimizations such as vectorization and invariant code motion that further improve resulting program performance. Thus, auto-parallelization can reveal the level of compiler's “skills” with a view to these optimizations.

Benchmark was carried out on the basis of two BLAS (Basic Linear Algebra Subprograms) library implementations and in a similar manner both for CPU and GPU parallelization. The first implementation is non-optimized reference BLAS implementation [2] compiled with a maximum optimization level and auto-parallelization options. The second one is hand-tuned GotoBLAS library [3, 4] compiled with self-chosen set of options. In case of GPU OpenACC [4] directives were added before each outermost *for* loop inside BLAS routines so that it can be parallelized by the compiler. So, each compiler built auto-parallelized and hand-tuned versions of the library and PGI also built GPU version of the library.

BLAS library was chosen because it is de facto standard in scientific software development and it is also contains the simplest algorithms to parallelize. Algorithms consist of vector-vector, vector-matrix and matrix-matrix operations that correspond to the Level 1, 2 and 3 of library routines respectively. Each algorithm includes no more than a triple of nested loops assembled exclusively for different argument variations (in case of Level 3 – matrix transpositions) [2]. The two inner loops can be used to expose both coarse-grained parallelism of multiple processor cores and fine-grained parallelism of vector registers. So, the choice of BLAS library was based on a desire to test compilers on the algorithms that are easy to parallelize.

Benchmarking strategy consisted of running a subset of Level 3 routines with a variable set of

actual arguments' values and using different number of threads. Arguments' variations included different matrix sizes, transposed/non-transposed cases, upper/lower triangular matrices and left/right sided equations. Thread number varied from 2 to 12 threads that correspond to the total of 12 machine cores. Benchmark results were summarized in one table containing approximately 2500 rows for subsequent analysis.

Average performance of auto-parallelized BLAS code is lesser than of hand-tuned BLAS library (Table 2), and further investigation shows significant variations in efficiency when using transposed and non-transposed matrices (Table 1). In case of *SGEMM* the source code for *SGEMM(A^T, B)* and *SGEMM(A^T, B^T)* differs only in one index (Figure 2), and their relative performance differs by an order of magnitude. Naturally, this index prevents loop vectorization and leads to ineffective cache utilization due to non-unit stride memory access pattern. As matrix operations such as *SGEMM* are not arithmetically intensive involving no more than floating point additions and multiplications, it is essential to optimize CPU cache usage and GPU loads/stores coalescing to achieve scalable performance. However, it is hard for compilers to do so and matrix transpositions lead to performance degradation.

SGEMM(A^T, B)

```
DO J = 1,N
  DO I = 1,M
    TEMP = ZERO
    DO L = 1,K
      TEMP = TEMP + A(L,I)*B(L,J)
    CONTINUE
    IF (BETA.EQ.ZERO) THEN
      C(I,J) = ALPHA*TEMP
    ELSE
      C(I,J) = ALPHA*TEMP +
BETA*C(I,J)
    END IF
  CONTINUE
CONTINUE
```

SGEMM(A^T, B^T)

```
DO J = 1,N
  DO I = 1,M
    TEMP = ZERO
    DO L = 1,K
      TEMP = TEMP + A(L,I)*B(J,L)
    CONTINUE
    IF (BETA.EQ.ZERO) THEN
      C(I,J) = ALPHA*TEMP
    ELSE
      C(I,J) = ALPHA*TEMP +
BETA*C(I,J)
    END IF
  CONTINUE
CONTINUE
```

Fig. 2: Source code for *SGEMM(A^T, B)* and *SGEMM(A^T, B^T)* differs only in one index and performance differs by an order of magnitude (see Table 1)

GCC inferior performance compared to other compilers seems to be attributed to the restriction of the compiler to parallelize only innermost loops [5]. Since benchmarked Level 3 routines contain loops nested up to the third level it is inefficient to parallelize only innermost loop as it may involve synchronization overheads and also as it constitutes only a part of a whole problem. As GCC does not produce any optimization report during compilation there is no easy way to give a reliable explanation of low performance.

Table 1. The effect of matrix transposition on the performance of auto-parallelized BLAS library.

		Performance, Mflops				PGI SGEMM cache statistics $\times 10^9$			
op(A)	*op(B)*	Intel	PGI	GCC	PGI CUDA	Reads	Writes	Misses	Instructions
A^T	B	19108	10193	1683	1326	137.4	0.016	0.008	412
A	B^T	17935	9738	2982	1341	137.5	68	0.039	481
A	B	16824	9453	3110	2335	137.5	68	0.008	481
A^T	B^T	3424	946	199	2407	137.4	0.016	0.104	549

GPU parallelization which was carried out on the basis of directive-based PGI CUDA compiler shows inferior performance compared to other compilers, however, matrix transpositions

have an opposite effect on the efficiency (Table 1). The effect. which consists of GPU code having the best performance in cases where CPU code having the worst, is attributed to CUDA storing matrices in a row-major order in contrast to Fortran storing them in a column-major order. Differences of matrix storage scheme affects memory access pattern which in turn affects CPU cache utilization and GPU memory loads/stores coalescing and finally defines efficiency of arithmetically non-intensive code. Although, GPU is perfectly suitable for matrix operations it is inherently parallel device in contrast to CPU and the best performance is generally achieved by using specific algorithm and implementation. So, to show comparable performance GPU compiler should rewrite a whole routine and not just parallelize existing CPU code. All in all, the best results are achieved when parallelizing code on CPU.

Table 2: Compiler parallelization efficiency of Intel, PGI and GCC compilers in Mflops. Target platform: HP SL390s G7, 2x Intel X5650 2.67 Ghz (12 cores total), 96 Gb RAM, 3x NVIDIA Tesla M2050.

Routine	Hand-tuned (CPU)			Parallelized (CPU)			Parallelized (GPU)
	Intel	PGI	GCC	Intel	PGI	GCC	PGI
SGEMM	111974	**130197**	57816	**16173**	7333	1824	1939
SSYMM	108611	**130466**	60525	9089	**12123**	2390	1632
SSYR2K	103813	**112828**	60230	**18157**	13728	3136	1917
SSYRK	95721	**106000**	43239	**14562**	10027	1811	1218
STRMM	99073	**115354**	43204	**14423**	7148	1990	2102
STRSM	97371	**115009**	41857	**11381**	6295	1745	1998

To summarize, benchmarks show that average performance of hand-tuned optimized BLAS library is an order of magnitude higher than of automatically parallelized library. The largest performance variations appear when performing operations on transposed matrices that affect efficient CPU cache usage and GPU memory operations coalescing. Finally, as GPU and CPU vastly differ in their architectures and programming style, the best GPU performance is achieved when implementing algorithm from a scratch and not adapting CPU source code. All in all, compiler auto-parallelization facilities can be seen as a means of making the first parallel program prototype which is incrementally optimized by hand to reach its best performance.

3 Compiler-based parallelization overhead

Overheads of compiler-based parallelization are defined by a corresponding runtime library implementation and are imposed by usage of directives. These include thread scheduling algorithms overhead, parallel region entering overhead and synchronization constructs overhead. Analysis of those overheads can be performed using micro benchmarks specifically designed to measure efficiency of directives.

Benchmark, which was carried out via EPCC benchmark suite [6] developed to analyze multi-processor machine efficiency, showed that it can also be used to compare compiler's OpenMP runtime library performance. In the original paper this suite is used to show variations in directive overheads running benchmarks on different hardware platforms, however, if the source code is translated by different compilers and run on a single machine then it shows representable performance of a particular OpenMP runtime library. So, EPCC benchmark suite was used to measure overhead of OpenMP directives using a set of supporting compilers.

Results of the benchmarks revealed most directives having similar overheads in case of PGI and Intel compilers and larger overheads in case of GCC, however, there are also some directives with

different overhead pattern. These are *omp reduction* and *omp schedule(dynamic)* showing inferior performance of PGI and *omp schedule(guided)* showing inferior performance of both PGI and Intel compilers (Figure 4). In addition, these directives also account for no more than 5% of a total count of *omp for* directives used in examined scientific applications (Figure 5). All in all, commercial compilers have more efficient OpenMP runtime library implementations than open source GCC compiler does.

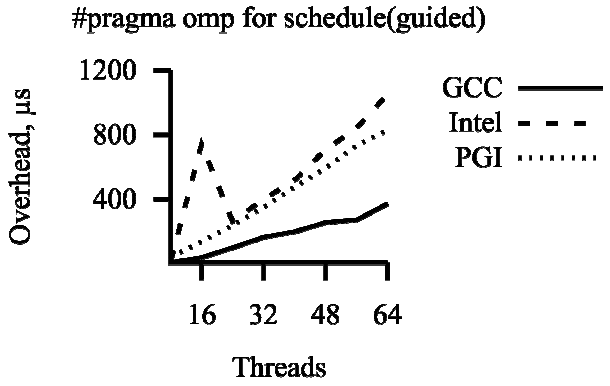


Fig. 4: Overhead of *omp schedule(guided)* directive showing inferior performance of commercial PGI and Intel compilers compared to open source GCC. Target platform: HP Proliant DL980, 8x Intel X7560 2.2 Ghz (64 cores), 512 Gb RAM.

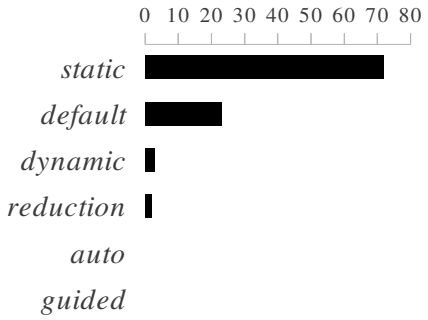


Fig. 5: Distribution of scheduling and reduction directives among all *omp for* directives in scientific software packages WRF (Weather Research Forecasting) and WaveWatch3.

4 Optimization reports

Optimization report is a special type of compiler's output telling developer about compiler's decisions in optimization process. Although not standardized, this output often includes decisions on loop vectorization, loop tiling, data distribution during loop parallelization and also decisions on an alternative loop code generation. The information is presented in a form of hints about why particular optimization technique was or was not employed, thus providing developer with a way of optimizing a source code for a given compiler.

Optimization hints may be useful, however, it is hard to measure their impact on an application performance. On one hand, they provide user with a knowledge of compiler's inner workings relieving him from tedious machine code analysis. On the other hand, following optimization hints of one compiler not always results in an optimal code for another compiler. So, this limits usefulness of optimization reports only to the most simple cases of inefficient code structure considering target's machine architecture.

Conclusion

In conclusion, practical efficiency of optimizing compilers is dependent on many factors. Intel compiler lets developer achieve easy parallelization of a program prototype, but reduced efficiency of a hand-tuned source code version. In contrast to this, PGI compiler offers quite an opposite: a prototype is slow but a final version is highly optimized. In either case, usage of non-commercial GCC compiler degrades performance of both a prototype and a final application. From a different point of view, one can use the most efficient compiler on a corresponding development stage and also for a particular target platform. All in all, final program performance is dependent on many factors and compiler's impact shall not be considered in isolation.

Acknowledgements

The research was carried out using computational resources of Resource Center Computational Center of Saint-Petersburg State University (T-EDGE96 HPC-0011828-001) that provided commercial compiler's licenses and access to high-performance machines for the scientific work.

References

[1] Bacon D.F., Graham, S.L., Sharp O.J. Compiler transformations for high-performance computing. ACM Computing Surveys (CSUR). Vol. 26:4, pp. 345-420, 1994.

[2] J. J. Dongarra, J. Du Croz, I. S. Duff, and S. Hammarling, A set of Level 3 Basic Linear Algebra Subprograms, ACM Trans. Math. Soft., 16 (1990), pp. 1-17.

[3] Kazushige Goto and Robert A. van de Geijn. "Anatomy of High-Performance Matrix Multiplication," ACM Transactions on Mathematical Software 34(3): Article 12, 25 pages, May 2008.

[4] Kazushige Goto and Robert van de Geijn. "High-Performance Implementation of the Level-3 BLAS." ACM Transactions on Mathematical Software 35(1): Article 4, 14 pages, July 2008.

[5] http://gcc.gnu.org/wiki/Graphite/Parallelization. GCC Graphite Parallelizer official WIKI page.

[6] J. M. Bull. Measuring Synchronisation and Scheduling Overheads in OpenMP. 1999.

VIRTUALIZATION WITH ORACLE SOLARIS 10

A.V. Bogdanov[1], Pyae Sone Ko Ko[2]
[1] *Institute for High-performance computing and the integrated systems, Saint-Petersburg, Russia*
bogdanov@csa.ru
[2] *St.Petersburg State Marine TechnicalUniversity, Saint-Petersburg, Russia*
pyaesonekoko@gmail.com

In this paper we review our experience in virtualization with Oracle Solaris. The information and communications technology (ICT) infrastructure has expanded the scale of systems so that processing can be conducted faster and more efficiently. But as the number of servers increases, it becomes more important to reduce their operating and administration costs and power consumption.

The Oracle Solaris Operating System (Solaris OS) installed in SPARC (scalable processor architecture) Enterprise UNIX server features virtualization functions that become more powerful with every update of release. Among the virtualization functions provided as standard in SPARC Enterprise, in this paper we focus on the Oracle VM Server for SPARC (VM: virtual machine) and Solaris Containers used for server virtualization and the Solaris ZFS (Solaris Zettabyte File System) functions for storage virtualization.

Virtualization functions of UNIX Server SPARC Enterprise

SPARC Enterprise provides three server virtualization functions. They are 1) Hardware partition, 2) Oracle VM Server for SPARC, 3) Solaris Containers.

1) **Hardware Partition**. A partition running an independent Solaris OS can be configured by logically partitioning the physical system board. The CPU and memory can be dynamically modified in response to requests for business expansion, the addition of new business, and so on without the system operation being halted.

2) **Oracle VM Server for SPARC**. Oracle VM Server for SPARC (previously called Sun Logical Domains) can configure logical domains running separate instances of Solaris OS by using the SPARC hypervisor in the firmware layer to partition the physical server into virtual servers. The CPU, memory, and input/output (I/O) devices are flexibly allocated by the Domain Manager.

3) **Solaris Containers**. Solaris Containers allows the Solaris OS to be virtually partitioned into zones that constitute independent virtual OS environments. CPUs and memory are flexibly allocated according to the zone's operating conditions. I/O devices are allocated when the zone is configured. Solaris OS also provides Solaris ZFS for storage virtualization as a standard function. The ZFS file system manages multiple physical disks as a storage pool. Virtualized volumes can be created by allocating the necessary space from the storage pool. The ZFS file system is not only durable and scalable but also easy to administer. The virtualization functions of Solaris OS are described below.

Oracle VM

Oracle VM is a family of products that work together to facilitate virtual environment creation and management. Consisting of Oracle VM Server and the integrated Oracle VM Manager browser-based management console, Oracle VM makes it easy to create and manage virtual server pools running on systems across the enterprise. Organizations can create multiple virtual machines on a physical x86, x64, or SPARC processor-based server, yet have each environment behave independently with its own virtual CPUs, network interfaces, storage, and operating system.

• **Oracle VM Server for x86.** Free to download, Oracle VM Server for x86 provides an easy-to-use graphical interface for creating and managing virtual server pools running on x86 and x64 systems. This server virtualization software fully supports Oracle and non-Oracle applications, as well as Oracle Solaris, Linux, and Windows guests. Backed by Oracle's world-class support organization, Oracle VM Server for x86 provides customers with a single point of enterprise-class support for virtualization environments and delivers more efficient performance. A wide range of Oracle products

including Oracle Database, Fusion Middleware, Oracle applications, and Oracle Enterprise Linux are certified with Oracle VM Server for x86.

• **Oracle VM Server for SPARC.** Purpose-built for Oracle servers with chip multithreading (CMT) technology, Oracle VM Server for SPARC (previously called Sun Logical Domains) provides a full virtual machine that runs an independent operating system instance and contains a wide range of virtualized devices. A hypervisor that largely resides in a chip on the server is tightly integrated with the hardware, enabling virtual machines to take advantage of underlying system advancements and reduce the overhead typically associated with software-based solutions [4]. Unlike solutions from other vendors that do not permit add-on networking or cryptographic devices to be partitioned, shared, or abstracted, Oracle VM Server for SPARC supports virtualized CPU, memory, storage, I/O, console, and cryptographic devices, and redundant I/O paths, to make maximum use of platform resources.

• **Oracle VM Manager.** Oracle VM Manager provides an easy-to-use, feature-rich graphical interface for creating and managing Oracle VM environments. With Oracle VM Manager, administrators can enable advanced functionality to load balance across resource pools and automatically reduce or eliminate outages associated with server downtime [4].

Oracle Solaris Containers

Solaris Containers consists of a Solaris Zone function for virtually partitioning a single OS space to make it appear as if multiple Oss are running and a Solaris Resource Manager that flexibly allocates hardware resources such as CPUs and memory. A Solaris Zone is a virtualized OS environment that implements a safe isolated environment suitable for running applications. Processes running in each zone are isolated and unable to affect other zones.

• **Solaris Zones.** A Solaris system has just one global zone, which is responsible for managing the entire system. Tasks such as the creation and administration of non-global zones and the allocation of physical I/O devices can be performed only in the global zone.

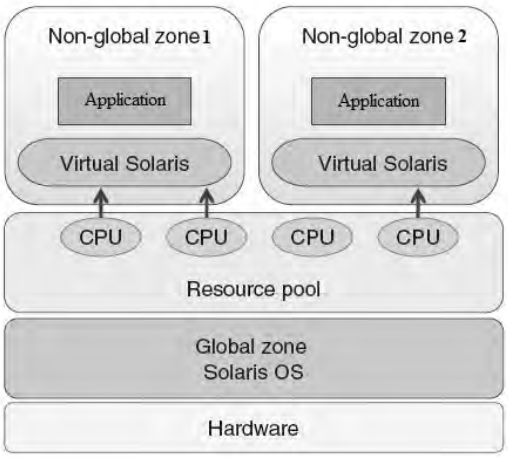


Figure 1: Solaris Containers configuration

A non-global zone is a software partition of a virtual Solaris environment, in which applications can run without affecting other zones. Up to 8191 zones can be created, each of which can use only its permitted file system and permitted physical I/O devices. The constituent system files of a non-global zone are copied from the global zone when the zone is created. When the global zone is patched to modify these files, all the non-global zone files are also synchronously updated.

Solaris ZFS (Solaris Zettabyte File System)

Solaris ZFS is a 128-bit file system that can manage a practically unlimited data capacity. The metadata used for the administration of a ZFS file system is dynamically allocated as required, so there is no limit to the number of file systems or the number of files. In a conventional file system, the file system size is limited to the physical device size. However, Solaris ZFS is not limited to specific physical devices because the physical devices are hidden by the ZFS storage pool. The ZFS file

system can create file system hierarchies easily without initialization, and it automatically expands within the range of the disk capacity allocated to the ZFS storage pool.

Performance Results

The results were obtained on a Sun SPARC Enterprise T5120 server from Oracle. The system had a single UltraSPARC T2 processor with 8 cores and 8 hardware threads per core.

In Figure 2, the elapsed times in seconds for the Automatically Parallelized and OpenMP implementations are plotted as a function of the number of threads used.

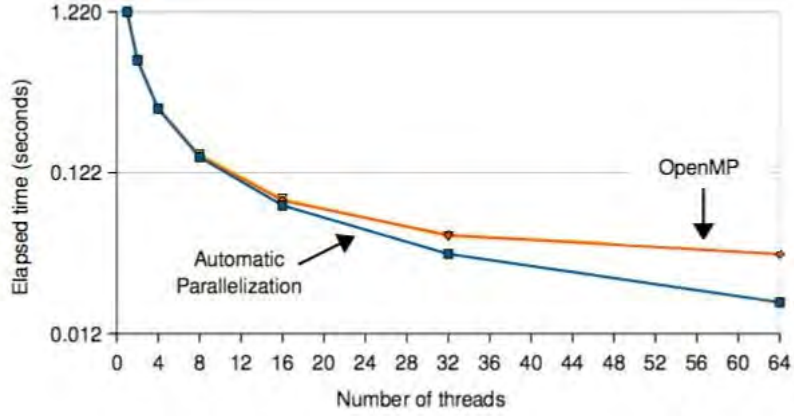


Figure 2: Performance of the Automatically Parallelized and OpenMP implementations

For up to 8 threads, both versions perform equal. For 16 threads the Automatically Parallelized version performs about 9 percent faster than the OpenMP version. Both versions scale very well for up to 8 threads. When using 32 threads, the Automatically Parallelized version is about 30% faster than OpenMP version. For 64 threads, the elapsed time is about twice as high. This difference is caused by the parallel overheads increasing as more threads are used. If more computational work was performed, this overhead would not be as dominant.

Conclusion

This paper reviews the virtualization functions in Solaris OS: server virtualization implemented in Oracle VM Server for SPARC and Solaris Containers and Solaris ZFS storage virtualization. Oracle Solaris 10 is reliable and predictable and has what we need in an operating system [5]. The Solaris 10 OS allows multiple applications to be consolidated onto a single system through virtualization, which makes it an ideal solution for standardization. The experiments we provided show high efficiency of parallelization with Solaris standard tools. We intend to continue enhancing the virtualization capabilities of Solaris OS.

References

[1] Oracle Solaris virtualization products: http://www.oracle.com/technetwork/serverstorage/solaris/overview/virtualization-163570.html

[2] Oracle Solaris Containers: http://www.oracle.com/technetwork/server-storage/solaris/containers-169727.html

[3] Oracle Solaris 10:

[4] http://www.oracle.com/us/products/servers-storage/solaris/index.html

[5] System Administration Guide: Oracle Solaris Containers-Resource Management and Oracle Solaris Zones [see“Chapter 24 Oracle Solaris 10 9/10: Migrating a Physical Oracle Solaris System Into a Zone(Tasks)”]: http://docs.sun.com/app/docs/doc/817-1592/

[6] Get hands on experience with Solaris OS using "Solaris 10 " http://developers.sun.com/solaris/whitepapers/index.jsp

[7] Participate in the OpenSolaris community www.opensolaris.org

[8] OpenMP 3.0 for Parallel Programming http://docs.oracle.com/cd/E18659_01/html/821-2763/gjggm.html

PRIVATE CLOUD VS PERSONAL SUPERCOMPUTER

A.V. Bogdanov
Saint-Petersburg State University, Russia
bogdanov@csa.ru

We discuss the possibility to use private cloud environment to organize the joint resources for complex problem solutions. First of all, we propose eight basic principles for clouds, which can be used for such purposes. Then, we argue that proper UNIX–like PSE should be used for coordination of resources to make load balancing realistic. We did some experiments to show, that the necessary speed-up is achieved only with the use of virtual memory and virtual common file system. One should also take care of data consolidation, and for such purpose we propose the so called "Principle of limited resource consolidation". To realize all those principles we propose middle-ware toolkit. Finally, we discuss couple of important applications of such systems.

1. Introduction

To have all necessary computing power at your desk was a dream for many generations of computer scientists. But although hardware is being developed at such a speed, that researcher have scarcely enough time to adopt new technology before emergence of a new one, this dream is far from being realized on site. From time to time it appeared that new distributed technology would make the idea realistic. But metacomputing was too demanding for resources to work for single user, and Grid was so carefully cut from the users, that you could not really operate distributed system. Cloud is a new token in the hands of computer scientist to give a new try to old hopes [1].

Cloud Computing is a rapid developing area of today's computer science. The idea of Cloud Computing is the transmission of the organization of data computing and processing mainly from personal computers to the servers of the World Wide Web. Today there can be distinguished several main technologies (models) of this area: Infrastructure as a Service (IaaS); Platform as a Service (PaaS); Software as a Service (SaaS); Workplace as a Service (WaaS).

Within the models of IaaS, PaaS, SaaS and WaaS the clients do not pay for the ownership of a software product as itself, but they rent it and therefore pay for the usage of the product through the Web-interface. Thus, as opposed to a classical scheme of purchase of a licensed software product, the client does not have to invest a lot of money in purchasing a product and a hardware platform for its delivery and maintenance of the system. The client only pays comparatively low periodic expenses, i. e. user charge, and can discontinue, suspend or resume it whenever the software product is required. Incidentally, a similar model has been used in the past.

Thus, it seems that we are very near to original idea and private supercomputer at your desk is nearby. But if you try to organize such virtual infrastructure via one of cloud providers you will have so many problems, that sometimes one may think of waiting for still new technology. Nevertheless we will argue that idea of personal supercomputer is realistic even on current stage of development, but you should very carefully choose and assemble the components.

2. Basics

A cloud, to wide extent, is an API. It is a certain contract model meeting the principles of the cloud computing. Five principles can be singled out: scalability, load balancing within the scalability, high availability up to the disaster-proofness, easy access to the resources from almost every place in the world and from any device, and payment on demand on the rental basis, i. e. on the most favorable terms for the client. When you have to use the service of a cloud-computing provider, you get an API and some guarantees. An API can be an implementation based on REST/WADL, SOAP/WSDL, XML-RPC, CORBA and other technologies, as well as a conventional client's web-interface (web GUI). Most commonly, these two ways are employed in parallel. But the more complex is your task, the more documentation and technologies you have to apply. Excess resources can be conveniently presented as an API or a web-interface. Another option represents the idea of co-usage of the resources

on demand and only in the needed amount. And "virtual organization" is a real organization. This requires consolidation and rearrangement of the resources within the organization itself: a private cloud creation. But as long as there are always not enough resources, it's necessary to cooperate with cloud service providers such as Amazon. This means that it is necessary to support common standards between the clouds, therefore being able to create hybrid clouds, which combine private and public computational resources. What helps to create hybrid clouds? One should be interested in products that support the standards and provide open source licenses [2]. Eucalyptus and OpenNebula seem to be quite suitable: their API is back compatible with Amazon [3, 4].

You can trust only your own private cloud, where you control the servers, channels and electricity. You still need to know how to use public clouds when necessary, choosing the most suitable cloud service provider. For scientific tasks, for numerical experiments, it is reasonable to use only the environment that can be most controlled by the researchers. In the educational process it is necessary to create up-to-date private clouds with modern business solutions, using the support of the government, other investors and free software products. Our researches need to have no vendor lock and be based on the open standards. The components have to be convenient, inexpensive and simple. Resource suppliers have to be interchangeable. Therefore, the result of the work will be universal and test-open for the other teams of scientists (program transportability, repeated use of components). This is the service-oriented approach, when the components interact with the help of API and the clients get the service that meets the contract guarantees.

If your problem needs a supercomputer, it should be a complex one. To work with such you need to organize proper environment (PSE). Our experiments show, that proper PSE should have many standard UNIX features; otherwise it is impossible to achieve load balancing. And we found that it is much more effective to adopt for Cloud PVM load balancing tools. To use middleware with single image PSE was also of a great help and not only to make it easy to launch the problems.

To process large data is a real challenge even to supercomputer. On a distributed system it can be a real bottleneck and we found, that even proper data consolidation is not enough to do it properly. Experiments show, that only proper choice of virtual shared memory and distributed file system solves the problem.

And of course in hybrid cloud security problems are essential. We found two examples, how Grid security tools can be incorporated into hybrid cloud without loss of control over submitted jobs.

3. Hybrid Cloud Approach

So, we propose to use a cloud approach based on open standards and utilizing several up-today technologies, that make it very effective for large scale problems.

Our main principles are

1. **Cloud is determined completely by its API**. And it is obvious from the user point of view, but the same is true from the point of view of different clouds interaction, as explained in item 2.
2. **Operational environment must be UNIX – like**. One of the main problems of computational Grid's is load balancing and it is very difficult task since user is cut off from the resources. Partly this problem is solved by PSE, but to make it really active many standard UNIX tools must be introduced into API.
3. **Cloud uses protocols, compatible with popular public clouds**. Public clouds are not very useful for complex problems and the reason of this is clear – the more difficult is the problem you are solving, the more robust tools you must use. The universal tools cannot be used for complex problems. That is why specialized private clouds must be built for complex problems, but if its resources are not enough, some additional resources can be added from public cloud.
4. **Cloud processes the data on the base of distributed file systems.** The main problem with public cloud for data processing comes from the fact that on each computer in the cloud its own file system is used. That prevents both: to process large data sets and to scale out the problem solution. To overcome this obstacle the distributed file system should be used in

private cloud, the type of which is determined by the nature of problem to be solved. If we add here three ways of providing data consistency (Brewer's theorem) we can see that there are a lot of possibilities of organization the processing of data out of which only a few are in use.

5. **The consolidation of data is achieved by distributed Federal DB**. There are three levels of consolidation – servers, data and resources. It is more or less clear how server consolidation is done. Consolidation of data is more difficult and consolidation of resources is a real challenge to the cloud provider. We assume that most natural way to do this is to use Federal DB tools. Up to now we managed to do this by utilizing IBM's DB2 tools, but we believe that possibilities of latest PostgreSQL release will make it possible to work out freeware tool for such purpose [5].
6. **Load balancing is achieved by the use of virtual processors with controlled rate**. New high-throughput processors make it possible to organize virtual processors with different speed of computation. This opens natural possibility of making distributed virtual computational system with architecture adapted to computational algorithm and instead of mapping the algorithm onto the computer architecture we will match the architecture with the computational code.
7. **Processing of large data sets is done via shared virtual memory.** Actually all previous experience shows, that the only way to comfortably process large data sets is to use SMP system. Now we cam effectively use shared memory tools (OpenCL) in heterogeneous environment and so make virtual SMP. The same tool is used for parallelization. The possibilities of single image operational environment are also very effective.
8. **Cloud uses complex grid–like security mechanisms.** One of the cloud problems is security issues [6] but we feel that proper combination of Grid security tools with Cloud access technologies is possible.

Conclusions

Recent experience with public clouds shows that principle problems of resources consolidation are far from being solved. We argue that this comes from the fact, that out of about 15 ways of cloud organization only one, most simple, is realized as yet.

We insist, that different problems need different ways of cloud organization, and, more than that, only hybrid cloud can be such solution. And there are the ways to organize such a cloud that makes it possible to solve your complex problem, or, can we say, to provide you with personal supercomputer?

References

[1] Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy H. Katz, Andrew Konwinski, Gunho Lee, David A. Patterson, Ariel Rabkin, Ion Stoica and Matei Zaharia. Above the Clouds: A Berkeley View of Cloud Computing.

[2] Peter Sempolinski, Douglas Thain. A Comparison and Critique of Eucalyptus, OpenNebula and Nimbus.

[3] Amazon Web Services and Eucalyptus. http://www.eucalyptus.com/learn/amazon-aws-compatibility

[4] Extend private cloud into Amazon Web Services with OpenNebula. http://cloudbestpractices.net/2011/11/07/extend-private-cloud-into-amazon-web-services-with-opennebula-2/

[5] The PostgreSQL Global Development Group. PostgreSQL 9.1.5 Documentation.

[6] Kevin Hamlen, Murat Kantarcioglu, Latifur Khan, Bhavani Thuraisingham. Security Issues for Cloud Computing.

VIRTUAL WORKSPACE AS BASIS OF SUPERCOMPUTER CENTER

A.V. Bogdanov[1], A.B. Degtyarev[1], V.Yu. Gaiduchok[2], I.G. Gankevich[1], V.I. Zolotarev[1]

[1]*Saint-Petersburg State University, Russia*

[2]*Saint Petersburg Electrotechnical University "LETI", Russia*

Virtual workspace has recently become one of the ways to perform routine tasks in cloud environment, however, its use in scientific experiments is novel and offers a number of advantages compared to traditional grid-based approach. These are: universal access to all computational and storage resources from within single private virtual machine, easy workspace customization considering user's own needs and an ability to create virtual private cluster with desirable configuration. In such an environment network storage devices are connected both to computational nodes and to your virtual machine so that all the experiment input data and output results are automatically saved in one place and can be easily accessed from the workspace. Implementation of described approach is presented on the example of Resource Center Computational Center of Saint-Petersburg State University.

1 Introduction

Virtual workspace as the name suggests can be thought of as a desk of a scientist where all papers, referential materials and document's drafts are stored and developed, but it offers some advantages over such conventional desk. First of all, as more and more scientists rely in their research on software and hardware equipment they want universal access to storage, computational resources and software licenses. Moreover, as their research varies in scale and time constraints scientists want to dynamically extend their resource pool to the desired capacity. Last but not the least, scientists want to customize and convert their workspace into the problem solution environment by installing additional software and adapting architecture to the problem solved. Eventually, this is how advantages offered by a virtual workspace are characterized.

Virtual workspace alone is a powerful concept, however it is beneficial to review its implementation on the basis of university resource center. Such resource center should not only administer and support hardware and software equipment but also provide scientists with high performance computing services, offer distributed software licenses and also service unconventional large-scale scientific projects. In addition to this, HPC services should be readily accessible to users with a wide range of technical skills and easily customizable for advanced users. Resource Center Computational Center (RCCC) of Saint-Petersburg State University will be provided as an example of virtual workspace approach [1].

Conventional resource center often services many faculties and should be able to resolve conflicts between user's own preferences and problem requirements. An incomplete list of common problems includes human-machine interaction, cooperative usage of shared computational and storage resources, prioritized provisioning of software licenses, security of experiment's data. Using such resource center as an example it is easy to show advantages and disadvantages of virtual workspace approach and also demonstrate its practical implications. Resource center infrastructure is discussed in Section 2, implementation of virtual workspace is discussed in Section 3 and advantages and disadvantages as well as performance considerations are discussed in Section 4 and 5 respectively.

2 Resource center infrastructure

Typical resource center maintains a range of clusters with possibly different topologies, a range of special purpose computing machines (hybrid or SMP architecture) and some storage devices all using high-speed interconnect. In case of RCCC these are: conventional T-Platform cluster, 3 SMP machines, hybrid cluster with GPU accelerators (Table 1) and HP X9300 storage system. Cluster

nodes are managed by CentOS 5.6 and resources are exposed by PBS (Torque with Maui job scheduler) [2].

Table 1. RCCC infrastructure

	T-Platfrom cluster T-EDGE96 HPC-0011828-001	**SMP cluster, HP Proliant DL980**	**Hybrid cluster, HP SL390s G7**
CPU	2x Intel E5335 2.0 GHz	8x Intel X7560 2.2 GHz	2x Intel X5650 2.67 GHz
GPU	–	–	3x (8x) NVIDIA Tesla M2050
RAM (Gb)	16	512-1024	96
HDD (Gb)	160	2000	120
Commutator	Infiniband 20 GB/s		
Total characteristics	768 TB RAM, 48 nodes, 384 cores	3 TB RAM, 3 nodes, 192 cores	2.3 TB RAM, 24 nodes, 288 cores, 112 GPUs
Peak performance (TFLOPS)	3.07	1.7	59.6

Although traditional infrastructure has an advantage of being simple to configure and maintain, it also has the following disadvantages:

(1) it offers no data consolidation, that is, user who obtained experiment results using special purpose machine should explicitly copy it to cluster access node for further analysis;
(2) private resources are restricted only to special purpose machines and are accessed in a non-unified way;
(3) there is no easy way to adapt machine architecture to problem solved.

The aim of virtual workspace is to solve these problems by borrowing some well-established principles of building private clouds. The key points are:

(1) usage of single storage to improve data consolidation;
(2) usage of single UNIX-based software repository;
(3) usage of virtual machine as a basis of virtual workspace;
(4) usage of private virtual clusters, conventional clusters and dedicated machines to extend virtual workspace resource capacity in a unified way.

The configuration of virtual workspace is not as easy as of traditional resource center infrastructure, however it is also superior to conventional setup in terms of flexibility and ease of customization.

3 Virtual workspace approach

Virtual machine is the main building block of a virtual workspace. In its simplest form a workspace consists of a single virtual machine connected to storage and licensed software repository. If desired, resource capacity can be extended naturally by replicating virtual machine to form a virtual cluster. Cluster can be owned exclusively by a single user or shared by members of a whole research group. Moreover, considering large scale problem one can acquire resources of dedicated high performance machine (SMP or hybrid) or conventional cluster. Resource capacity extension occurs dynamically and acquired resources can be accessed from within single virtual machine.

Virtual machine is customized by changing its hardware characteristics and by selecting desired operating system and software packages. Hardware characteristics include CPU cores, RAM and virtual storage capacity and operating system can be UNIX-like or Windows (Table 2). User is provided with restricted administrative access to update and configure operating system. Furthermore, user can choose between command line (SSH) and GUI-based (VNC, FreeNX or RDP) access and also select OpenVPN or announced IP option. Possibility of such flexible configuration improves

usability without interfering with virtual machine management.
Intended virtual machine usage is summarized as follows:

- solve scientific problems that fit into single virtual machine resources;
- access to computational resources, both clusters and dedicated machines;
- store experiment's data;
- develop applications (programming using commercial and open source compilers);
- perform other routine tasks.

Unified access to the workspace resources is essential to achieve usability and data coherence. Although computational resources are distributed, experiment's data remains in one place due to automatic mounting of file system from virtual machine. Applications are launched in a unified way by using package specific scripts taking cluster host name and job queue as arguments. In addition to this LDAP single sing-on [3] is used to simplify user login to multiple machines and to restrict cluster access. Organizing services in that way provides users with easy access to both research data and computational resources.

Backup process is simplified with data residing in one place and user have an option to backup only selected directories or the whole virtual machine. The first option is sufficient for experienced user processing arrays of data on a regular basis and has a short backup period. The second option is suitable for normal user who accidentally put system out of order and it also has a long backup period. Finally, using any backup approach increases overall system ability to recover from unexpected failure.

Virtual machine is a lively entity with constantly varying resource consumption, however it does not implies wasting resources of low-loaded virtual machine. In such case hypervisor (VMware in case of RCCC) dynamically migrate active virtual machines to available low-loaded physical machines to balance overall workload [4]. Moreover, in most cases virtual machine is acquired on a finite period of time until the research is complete, so unused virtual machines are eventually destroyed and resources are reclaimed by hypervisor. All in all, efficiency of using virtual machines is totally defined by operating hypervisor.

Table 2. Virtual machine characteristics

	Default	**Maximum**
CPU	Intel Xeon X5760 2.93Ghz	
CPU cores	4	12
RAM	4 Gb	24 Gb
HDD	50 Gb	~1 Tb
OS	CentOS 5.6	

To sum up, virtual workspace allows dynamic control over resource consumption adapting to the problem solved and provides coherent interface to the extended resources. If the problem is small enough to fit resources of a single virtual machine there is no need to use clusters and problem can be solved «in-place». If the problem is solved on the daily basis and does not fit into single virtual machine, then private virtual cluster is the best option. Finally, if the problem is so large that it consumes significant part of a cluster resource pool, then dedicated high performance machine is the right way to go. This approach leaves medium-sized problems to conventional clusters.

4 Advantages and disadvantages of virtual workspace

Virtual workspace advantages and disadvantages are summarized and validated as follows.
Advantages:

(1) Universal access to high performance resources over internet (SSH, VNC, RDP)
(2) Unified access to both computational and storage resources from within single virtual machine (PBS, file system auto mounting)

(3) Customizability with restricted administrative rights
(4) Improved security: all data is accessed only from a private virtual machine
(5) Easy selective backup of meaningful data
(6) Resilience to unexpected failures: virtual machine can be restored from regular backup snapshots
(7) Easy configuration of virtual machine characteristics by means of hypervisor
(8) Natural extension of resources by creating virtual private cluster adapted to problem being solved

Disadvantages:

(1) Load balancing efficiency and resource consumption control are totally dependent on a hypervisor
(2) Complex virtual machine configuration: creating virtual machine templates and performing initial machine configuration consumes time when done manually and should be automated
(3) Virtual cluster interconnect performance is degraded when multiple virtual nodes reside on a single physical host thus limiting flexibility of cluster topology configuration (Table 3)

Table 3. Crystal09 SrTiO3 test case wall clock time in minutes showing virtual cluster performance degradation. Virtual cluster characteristics: 16 virtual machines on 4x BL460c G7, 2x Intel X5675 CPUs and 96 Gb RAM on one node, 64 cores total.

Total cores	T-Platform cluster	SMP machine	Virtual cluster
32	84	42	35
64	66	36	48

5 Scientific application examples

Each method of resource capacity extension found efficient application area, but with different kinds of software. First, private virtual cluster approach proved to be beneficial when using interactive resource-hungry software like Materials Studio [5] or ADF [6]. In that case computational resources of a single host exposed as a virtual machine are not enough for application to run smoothly, and virtual cluster boosts its performance. Second, dedicated high performance machine approach is known to be successful in solving large-scale problems in Crystal09 [7] (Figure 1).

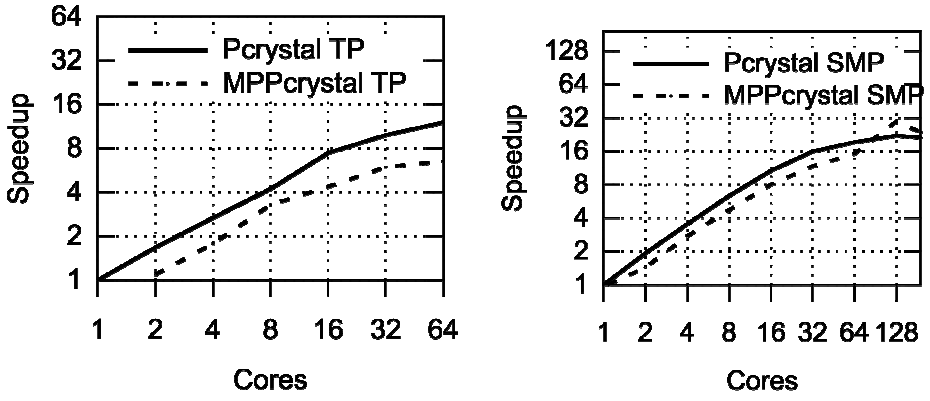


Fig. 1: Speedup of Crystal09 SrTiO3 test case running on conventional public cluster (left) and dedicated SMP machine (right). SMP machine has better speedup (see 16 cores mark).

In that case a single application run takes more than 1 month to complete even on multi-core SMP system with OpenMPI [8] shared memory interconnect, so running application on conventional cluster takes even more time due to slower link or will put heavy burden on network throughput in case of a virtual cluster. So private virtual cluster can be recommended for interactive applications used on the daily basis and dedicated machine is the most suitable to large-scale problems and long-

term simulations.

Performance degradation may occur when using single storage and mounted directories, however only a simple fix is required to restore it. Degradation occurs when multiple parallel processes of a single job simultaneously write to a mounted directory thus inevitably creating performance bottleneck. One demonstrative example of such application is OpenFOAM [9] running simple cavity case in parallel having each process writing multiple files into separate directories of mounted file system (Figure 2). As was mentioned before, this bottleneck is easy to fix by modifying PBS job script to use temporary directory of each node as a working directory and copying results back from each node to mounted file system after computation is over.

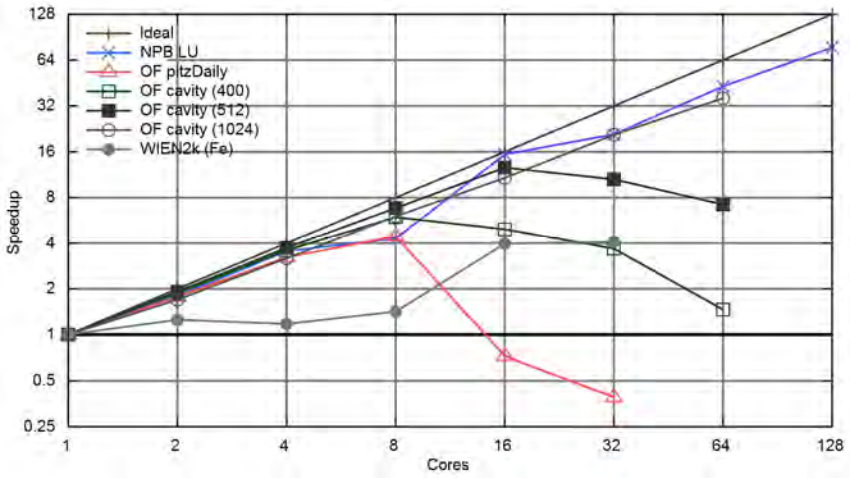


Fig. 2: Speedup of different applications on hybrid cluster. Performance degradation occurs when multiple processes write simultaneously to single mounted directory.

6 Conclusion

Virtual workspace hides intricacies of distributed computing behind a virtual machine to streamline and boost scientific research work flow. It provides a convenient way of accessing hardware and software resources using unified tools, consolidates experiment's data and offers options to dynamically extend available resources using either private virtual cluster or high performance dedicated machines and also public university cluster. Resources are accessed universally and in a unified way.

Virtual workspace is a novel approach of conducting scientific experiments which harmonically combines ideas of building private clouds with scientific software requirements and specifics of resource center operation. Although, virtual workspace configuration requires more human resources than configuration of conventional cluster do, it gives convenience of flexible, easily extensible and coherent system. Finally, the problems mentioned in the paper that occurred during workspace integration were solved thus showing RCCC as a demonstrative example of exploiting virtualization benefits for scientific needs.

Acknowledgements

The research was carried out using computational resources of Resource Center Computational Center of Saint-Petersburg State University (T-EDGE96 HPC-0011828-001).

References

[1] http://ptc.spbu.ru Resource Center Computational Center Website.

[2] Jackson D., Snell Q., Clement M. Core algorithms of the Maui scheduler. Job Scheduling Strategies for Parallel Processing, pp. 87-102, 2001.

[3] Zeilenga K., Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map, IETF, 2006.

[4] VMware Infrastructure: Resource management with VMware DRS. VMware Whitepaper, 2006.
[5] Segall M., Linda P., Probert M., Pickard C., Hasnip P., Clark S., Payne M. Materials Studio CASTEP, 2002.
[6] Guerra C.F., Visser O., Snijders JG, te Velde G., Baerends EJ. Parallelisation of the Amsterdam density functional program. Methods and Techniques in Computational Chemistry, 1995.
[7] R. Dovesi, V.R. Saunders, C. Roetti, R. Orlando, C. M. Zicovich-Wilson, F. Pascale, B. Civalleri, K. Doll, N.M. Harrison, I.J. Bush, Ph. D'Arco , M. Llunell. CRYSTAL09 User's Manual.
[8] Gabriel E., Fagg G., Bosilca G., Angskun T., Dongarra J., Squyres J., Sahay V., Kambadur P., Barrett B., Lumsdaine A. Open MPI: Goals, concept, and design of a next generation MPI implementation. Recent Advances in Parallel Virtual Machine and Message Passing Interface, pp. 353-377, 2004.
[9] Jasak H., Jemcov A., Tukovic Z. Openfoam: A c++ library for complex physics simulations. International Workshop on Coupled Methods in Numerical Dynamics, IUC, Dubrovnik, Croatia, 2007.

CPU AND GPU CONSOLIDATION BASED ON OPENCL

A.V. Bogdanov[1], I.G. Gankevich[1], V.Yu. Gaiduchok[2], Pyae Sone Ko Ko[3]
[1] *Saint Petersburg State University, Russia,*
[2] *Saint Petersburg Electrotechnical University "LETI", Russia,*
[3] *State Marine Technical University of St. Petersburg, Russia*
bogdanov@csa.ru

The use of GPU for general-purpose computing is a relatively new and promising direction. The use of GPUs as vector accelerators becomes more and more popular, but different vendors offer their own API and languages. As a result, transition from one platform to another sometimes requires enormous code changes. The answer to this problem is OpenCL [1]. All major CPU and GPU manufacturers created their own implementations for their devices. OpenCL is the open standard for parallel programming of heterogeneous computations.

This report discusses questions on organization of computing with OpenCL. First, we will talk about advantages and disadvantages of CPU and GPU computing. Then test results will be shown, we will represent algorithms that have significant speedup using GPGPU computations. At last, some examples of applications and libraries that use OpenCL will be discussed. Special attention is paid to generation of a proper operational environment for such heterogeneous computations. We argue that virtualization is crucial for proper load balancing. All calculations were performed on a hybrid cluster of SPbSU computing center. Its nodes contain a NVIDIA Tesla M2050 system that was developed specifically as a GPGPU unit. Such devices provide substantial speedups for scientific calculations [2].

1. Introduction

However much CPUs' performance grows, it pales beside performance of GPUs that aren't encumbered with control tasks. This difference can be explained by the definition of a GPU: a specialized device that was designed for compute-intensive tasks. Until recently GPUs were used only for intended purpose, i.e. for graphics. But their capabilities led to the advent of a brand-new technique – GPGPU. This technique implies the usage of GPUs for general-purpose computations.

Graphics processing units show excellent results [2] on computation-intensive problems. So, GPUs now are considered as vector accelerators [3]. Such usage is spreading far and wide today. One can observe a tendency of the last years when GPU manufacturers regard GPGPU as a separate branch (a good example is NVIDIA Tesla that was designed specifically for GPGPU). New APIs and SDKs are emerging in order to use all GPU potential. There is also a separate standard (OpenCL). In fact, this "inappropriate" use of GPUs was evolved in one of the most promising directions.

GPGPU is not only used for scientific calculations. There are many fields where CPGPU is applied: software developers report about interesting, sometimes unexpected projects that use the described approach.

There are many papers that concern GPGPU researches. These works usually contain one similar idea: computations are carried out on GPUs, a CPU only starts a task and receives results (actually, only one core of a CPU is used). Even papers on OpenCL. These papers usually imply computations on a single GPU. At the same time one forgets about that fact that OpenCL was developed as a standard for computations on different platforms. And one doesn't take into account capabilities of up-to-date multicore CPUs in this case. That's why CPU and GPU consolidation within the same task is an interesting and important problem. The solution of this problem will lead to the maximum usage of all system powers. This task was substantially simplified with the advent of the OpenCL standard.

2. Capabilities of OpenCL

It is known that code for GPGPU was always platform dependent. Different vendors offer their own APIs and programming languages. So, developers usually face many problems when they want to replace their old platform with a new one, more powerful. OpenCL is here to solve these problems.

OpenCL is an open standard of parallel programming for heterogeneous systems [4]. It defines C-like programming language for writing OpenCL kernels and API. It allows utilizing all computational powers of a system: CPU, GPU and, even more, some other specialized accelerators.

OpenCL is maintained by Khronos committee. It is a relatively new standard. Its history begins in 2008 when it was proposed by Apple.

So, what can OpenCL offer to a programmer? It provides developers with a toolkit that makes it possible to harness all system powers. OpenCL program finds available resources, compiles OpenCL kernels (kernel compilation is usually carried out at runtime), prepares all necessary input data for kernels, starts calculations and receives results from devices that run kernels [4]. One should write two programs: an OpenCL kernel and a host program. "Host" is a device that initiates all calculations (usually CPU). The host program detects OpenCL devices, starts calculations and retrieves results [1]. The OpenCL kernel is a code that will be executed on an OpenCL device (CPU, GPU, accelerators). The kernel should contain a compute-intensive part of an algorithm.

3. Equipment for calculations

We used the hybrid cluster of Resource Center "Computational Center" of Saint-Petersburg State University as testbed. This cluster is comprised of 24 nodes, each of these nodes contains 2 CPU Intel Xeon X5650 (6 cores), 96 Gb RAM, 8 or 3 GPU NVIDIA Tesla M2050. Peak performance of the cluster is 59,6 TFLOPS. All tests were carried out on one node of this cluster (that node was running CentOS 6).

The key feature of the cluster is powerful GPUs that can achieve a high percentage of the peak performance. The peak performance of the GPUs that was used is 0,5 TFLOPS [5] while the CPU performance is only 0,075 TFLOPS. The mentioned GPU contains 448 cores that exclusively dedicated to calculations [6]. Unlike the CPU that used MIMD technique, GPU utilizes SIMD technique: all cores execute the same code with different data. That's why GPU doesn't require complex control elements.

4. Tests

There are many software products that use GPGPU technique. In most cases these programs are written using CUDA programming model. So, these programs can be run only on NVIDIA GPUs with CUDA architecture.

CUDA is widely used today due to that reason that CUDA appeared earlier than OpenCL, OpenCL implementations are relatively immature and they are less effective in comparison with CUDA. However OpenCL in perspective seems much more advantageous variant: portability allows programmers to avoid "vendor lock-in", while the ability to use essentially different platforms opens completely new horizons.

One can note: software packages that use OpenCL are still rare. Programs that harness both CPU and GPU for active computations are very few. That's why we chose this theme for our research. Our main questions during this research were: is consolidation of GPU and CPU possible? Is OpenCL effective? Which algorithms suits best for described variant? The main purpose is an investigation of OpenCL potentials. We also examine some other aspects of OpenCL programming.

"CPU and GPU consolidation" means computations on CPU and GPU within the same task (when CPU participates in calculations as OpenCL device). In terms of OpenCL, we want to launch on a CPU not only the host program but also some OpenCL kernels.

And one of the first problems was load balancing. We raised the question about its efficiency. That's why there were two types of test programs: with load balancing and without it.

We started from a case when our task is divided between OpenCL devices in the beginning of programs. We assess devices performance and then split the task into subtasks and assign them to the devices accordingly. Subtask size is proportional to the number of device's computing units. The obvious advantage of this method is simplicity. Moreover, we call function for data transmitting only once for each OpenCL device. But we can estimate performance only indirectly. So, there can be a situation when fast devices have already calculated their subtasks and idle while the slowest device is still calculating. The dynamic balancing was proposed in order to eliminate the situation described.

The tests showed that load balancing overheads are quite small, while load balancing provides effective resources usage. Callback functions were used for load balancing tasks (using setEventCallback function, new function that appeared in OpenCL 1.1).

Program execution, in general case, includes next steps: retrieving list with available devices, determining first subtask for each device, preparing environment (e.g. creating contexts and memory objects), starting calculations on the devices, receiving results. But buffer reading invokes asynchronously: we just put this task in a queue and don't wait for it. We specify callback-function instead of waiting. This function will be invoked when buffer reading is finished. It will happen only when the subtask is computed, so buffer will store the results (tasks in a queue are executed in order of appearance). Callback-function checks are there any other subtasks. If so, device continues working with a new subtask. And we specify the same callback-function for the buffer reading operation. If there are no subtasks, callback-function informs main function and the latter saves results.

Several tests were carried out. We implemented some algorithms of linear algebra. But let's look at two characteristic examples. In case when we calculate on GPU too simple and small task, we can get strange results. Figure 1 corresponds to matrix multiplication. One can see that usual CPU computes this task faster than powerful GPU (note: two 6-core processors were seen by OpenCL program as one 12-core).

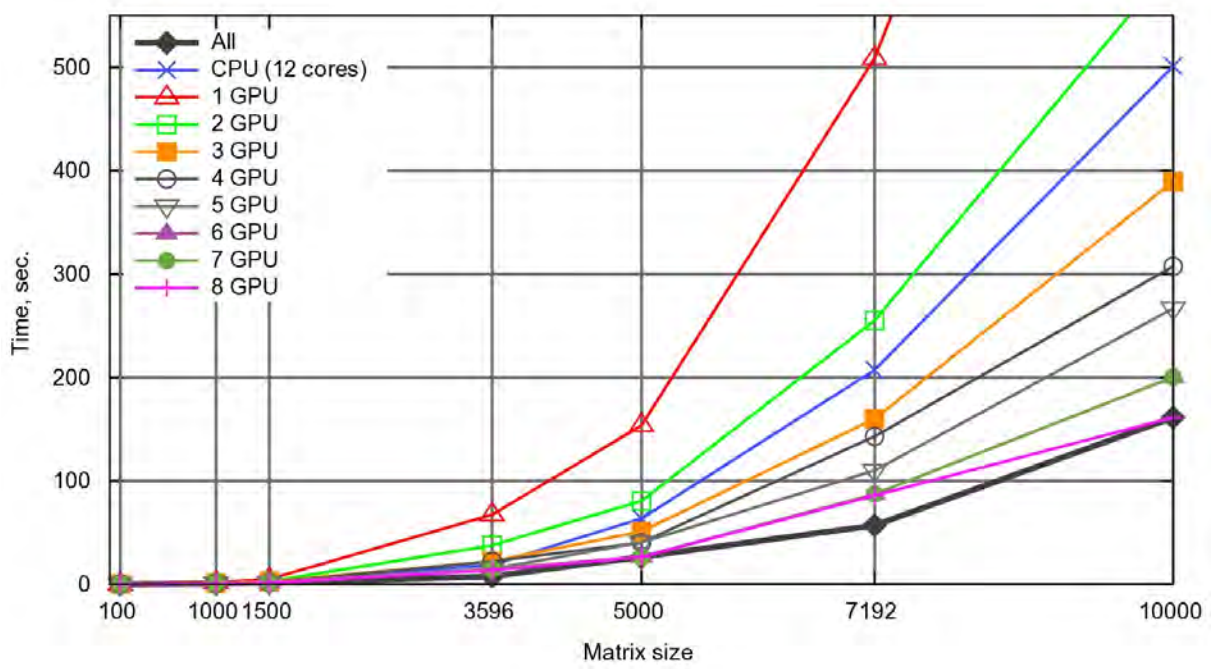


Fig. 1: "Matrix multiplication" test results

What is the reason for such strange results? First of all, one should pay attention to the OpenCL kernel. In this example we used only slow global memory (we couldn't use fast local memory due to its size). The other reason is well-known problem of data transferring. In our example we had to transfer hundred megabytes of input data to the GPU memory. This transfer consumes substantial

time. But we have a different situation in case of CPU: the device memory is an operating memory that is also a host memory, so there is no need for data transfer.

And the main reason is our task. The described test isn't a good one for GPU. Matrix multiplication is a simple task for GPU. And this is the root cause of the mentioned results. CPU can compute this task with ease.

But let's look at more difficult task. Next program will have the same OpenCL kernel with one exception: we'll calculate trigonometric expressions instead of multiplication in the cycle. The results of time measurement for different device sets are on Figure 2.

The results speak for themselves. Now the GPU computes faster than the CPU, while 8 GPUs computing together leave no chance to the CPU (speedup is more than 40 in case of 10000 x 10000 matrix). However, speedup achieved is not the limit. GPU can show substantially better results on compute-intensive tasks.

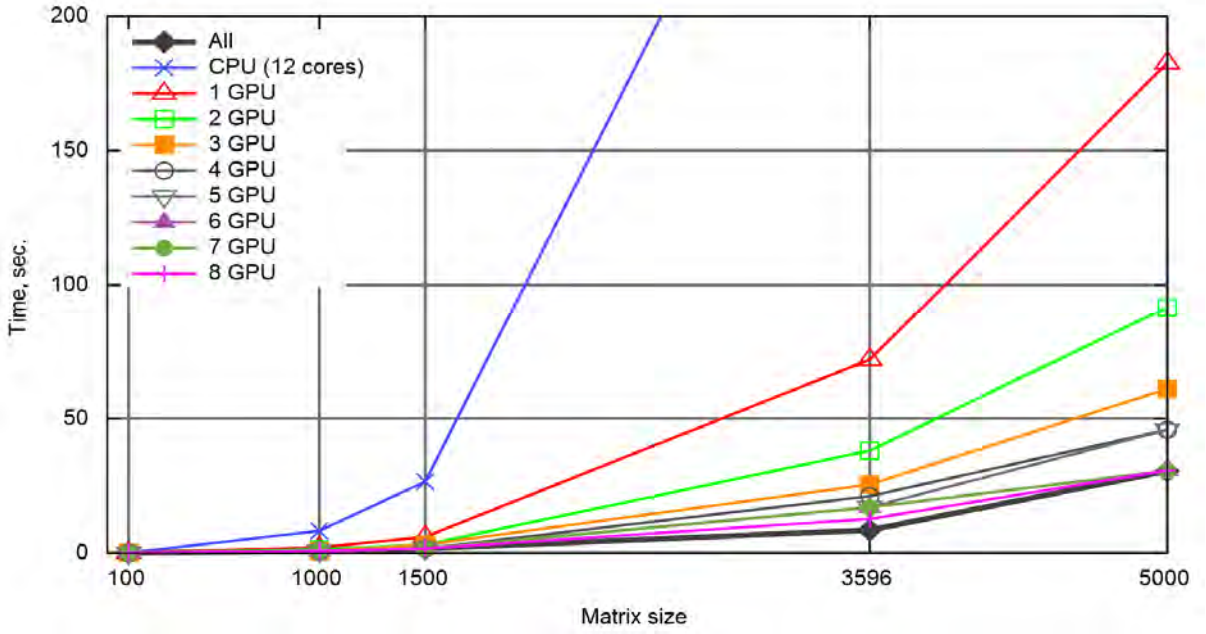


Fig. 2: "Kernel with trigonometry" test results

We can draw a conclusion based on the test results: CPU can compete with GPU at some tasks. GPU is preferred for algorithms that imply coarse-grained parallelization, compute-intensive, complex calculations with rare data transfers between the host and the device, while CPU should calculate computationally simple subtasks that imply big data sets processing. If one can split the initial task in the specified groups, then CPU and GPU consolidation is advantageous.

We should also note that our tests were carried out not only on the cluster. We also ran them on personal computers (Intel CPU – NVIDIA GPU, AMD CPU – ATI GPU) without any problems.

5. Recommendations for programmers

As a result some recommendations for OpenCL programmer is given.

1) The usage of OpenCL local and private memory. Global memory is too slow when compared with local memory. The fastest memory is private memory. One should use as much as it possible local and private memory. But local memory has one substantial drawback – its size. Small local memory can't store big data sets, so one should always remember about it when writing OpenCL kernels.
2) Different GPU and CPU kernels. GPU kernels imply compute-intensive, complex calculations, while CPU kernels can work with intensive data transferring.

3) Dynamic load balancing. This recommendation seems to be excess in case of identical devices. But when one uses different devices, this approach will lead to substantial speedup.
4) The usage of callback-functions. This convenient novelty was introduced in OpenCL 1.1.
5) Asynchronous function invocations. It should be used in order to eliminate the idle time of devices.
6) Deep insight of the architecture of the device. This primarily concerns GPUs.
7) OpenCL should be used only for big tasks. Due to that fact that OpenCL program usually includes kernel compilation, much preparation work, data transfers to devices, even simple task launching will consume much time. So, one will gain only in case of a big task.

6. Conclusion

In the course of this work, it was shown that CPUs and GPUs can successfully work within the same job, the main task for programmer is to split initial job into groups: GPU should work with complex, intensive computations with rare data transferring; CPU can carry out simple operations on big data sets. One should also remember about load balancing. It can be very effective in such systems.

Tests were carried out on the hybrid cluster of Resource Center Computational Center (SPbSU). Such clusters are representatives of the new generation of computer systems that use heterogeneous computations. We can suppose that these clusters will soon become a de facto standard for scientific calculations.

Acknowledgements

The research was carried out using computational resources of Resource Center Computational Center of Saint-Petersburg State University [7].

References

[1] Ryoji Tsuchiyama, Takashi Nakamura, Takuro Iizuka, Akihiro Asahara, Satoshi Miki. The OpenCL Programming Book.

[2] Degtyarev A., Gankevich I. Efficiency Comparison of Wave Surface Generation Using OpenCL, OpenMP and MPI // Proceedings of 8th International Conference «Computer Science & Information Technologies» — Yerevan, Armenia, — 2011. — P. 248-251.

[3] Volkov V., Demmel J. W. Benchmarking GPUs to Tune Dense Linear Algebra // SC08.

[4] OpenCL standard. http://www.khronos.org/opencl/

[5] NVIDIA. Tesla M-Class GPU Computing Modules, 2011.

[6] NVIDIA. OpenCL Programming Guide for the CUDA Architecture, 2011.

[7] Resource Center Computational Center website. http://cc.spbu.ru/

SCALING THE SPEEDUP OF MULTI-CORE CHIPS BASED ON AMDAHL'S LAW

A.V. Bogdanov[1], Kyaw Zaya[2]
[1] *Institute for High-performance computing and integrated systems, Saint-Petersburg, Russia*
e-mail: bogdanov@csa.ru
[2]*Saint-Petersburg State University, Saint-Petersburg, Russia*
e-mail: kyawzaya4436@gmail.com,

This paper discusses the speedup of parallel processing in multi-core chips based on Amdahl's law and gives a theoretical analysis of multi-core scalability. By fact, speedup also depends upon various factors including the inherent parallelism in the system itself, the hardware architectures of the machines, and an OS with flexible facilities to allocate and assign processors and memory resources [6]. And, thus this paper investigates the multi-core scalability. For asymmetric multi-core chips, architecture using one large amount of cores and base core, that is assumed originally for simplicity, is proved to be the optimal architecture in the sense of speedup. The potential maximum speedup's obtained by using the architecture of symmetric, asymmetric or dynamic multi-core are determined. The parallel fraction, performance index and the number of base core, precise quantitative conditions, how to obtain optimal multi-core performance, are derived.

Introduction

The scalabity problems were the majority issues of the recent years and even nowaday still play as great rolls in information technology and research goals .In the multicore era, the scalability problem is still as an important roll. The estimation of potential speedup to be gained from parallel processing, since then, large investments have been made in different hardware architectures. Estimating speedup for time-critical applications is a difficult problem involving many factors and viewpoints [3]. One view is that of computer manufacturers looking at potential markets and corresponding software applications that affect sales, and the other is dealing with scientific applications involving large simulations and extreme speed constraints, are also well represented. Transaction processing applications are already parallelized, distributed over large numbers of client-server platforms. However, these applications are "embarrassingly parallel," running on separate platforms, each with their own operating system (OS) [4]. That make us more necessary to estimate the speedup issues in information technology today.

1. The Speedup Concept

Speedup S(p) – is defined as the ratio of the executing time of the best possible sequential algorithm on a single processor to the executing time of the selected algorithm on a processor parallel system under the assumption that both algorithms solve the same problem

$$S_P = \frac{T1}{TP},$$

where, T1 = algorithm execution time on a single processor, TP = the executing time of the algorithm in a system of p processors.

2. Amdahl's Law

Consider the case where you want to solve some computational problem. Let α - fraction of the algorithm, which can be calculated only in series, respectively, (1 - α) - that portion of the algorithm, which can be successfully parallelized. Then the acceleration, which can be obtained on a system of N processors, compared with a uniprocessor system does not exceed [1]:

$$S_N = \frac{1}{\alpha + \frac{(1-\alpha)}{N}},$$

where, α - fraction of the algorithm, which can be calculated only in series, respectively, (1 - α) - the portion of the algorithm, which can be successfully parallelized.

3. Parallelism, architecture and decomposition

Parallelism implies that a software architecture can be produced that decomposes the system into modules such that modules can be designated as independent, implying that they may run concurrently with other modules in the system when they are invoked [2]. The decision to invoke a module at run-time is based upon the application requirements and software design. We note that the software architecture for a single processor may be different from that for a parallel processor [4].

4. Amdahl's law in Multi-core Chips

4.1 Symmetric Multi-core Chip

A symmetric multi-core chip (fig.1) involves a multiprocessor computer hardware architecture where two or more identical processors are connected to a single shared main memory and are controlled by a single OS instance [5-7].

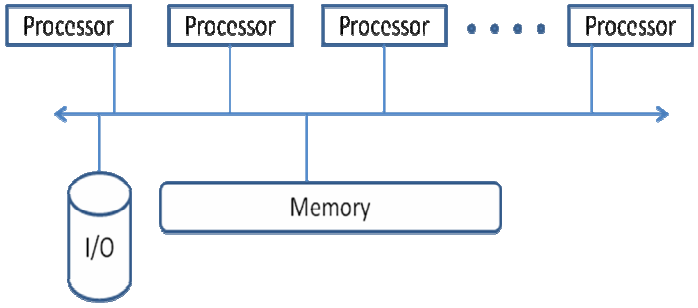


Fig. 1: Structure of symmetric multi-core chip

Under Amdahl's Law, the speedup of a symmetric multi-core chip (relative to using one single BCE (base core equivalents) core) depends on the software fraction that is parallelizable (*f*), total chip resources in BCEs (*n*), and the BCE resources (*r*) devoted to increase the performance of each core. The chip uses one core to execute sequentially at performance *perf(r)*. It uses all *n/r* cores to execute in parallel at performance *perf(r)*n/r*. Overall, we get:

$$Sp_{symmetric} = \frac{1}{\frac{1-f}{perf(r)} + \frac{f*r}{perf(r)*n}},$$

where, (*f*)= software fraction that is parallelizable, *n*= the total number of BCE, r= number of BCE on a kernel, *perf(r)*= performance of ***r*** number ***cores.***

4.2 Asymmetric Multi-core Chip

An alternative to a symmetric multi-core chip (fig.2) is an *asymmetric multi-core chip,* where one or more cores are more powerful than the others. With the simplistic assumptions of Amdahl's Law, it makes most sense to devote extra resources to increase the capability of only one core [5-7].

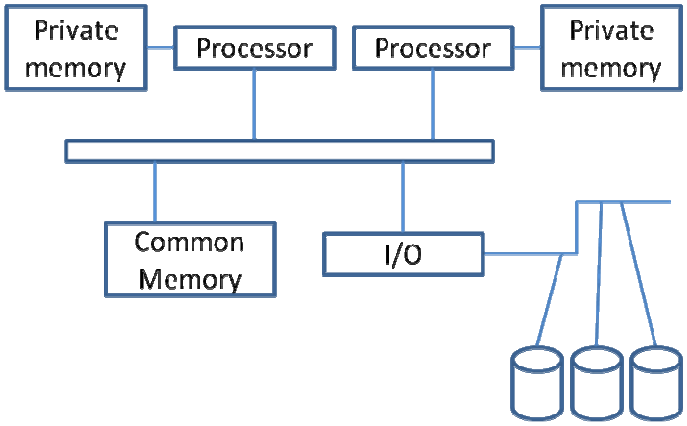


Fig. 2: Structure of asymmetric multi-core chip

Amdahl's Law has a different effect on an asymmetric multi-core chip. This chip uses the one core with more resources to execute sequentially at performance *perf(r)*. In the parallel fraction, however, it gets performance *perf(r)* from the large core and performance 1 from each of the *n-r* base cores. Overall, we get:

$$Sp_{Asymmetric} = \frac{1}{\frac{1-f}{perf\ (r)} + \frac{f * r}{perf\ (r) + n - r}}$$

4.3 Dynamic Multi-core Chip

There we can increase both the parallel fraction and the number of base core equivalents *n* to enhance the speedup of dynamic multi-core chip continuously [5-7]. In sequential mode, this dynamic multi-core chip can execute with performance *perf(r)* when the dynamic techniques can use ***r*** BCEs. In parallel mode, a dynamic multi-core gets performance *n* using all base cores in parallel. Overall, we get:

$$Sp_{Dynamic} = \frac{1}{\frac{1-f}{perf\ (r)} + \frac{f}{n}}$$

5. Analyzing speedup of T-Platforms Cluster

We analyzed the speedup of multi-core chip based distributed computing environment, T-Platform Cluster of the department of applied mathematics, control process , St. Petersburg State University (tabl.1). By fact, speedup is depends upon various factors, including the inherent parallelism in the system itself, the hardware architectures of the machines, and an OS with flexible facilities to allocate and assign processors and memory resources [6]. We tested speedup by using the application "Crystal", which is good parallelizable on each cores of the computing environment (fig.3).

Table1. Characteristics of T-Platforms Cluster

	T-Platforms Cluster T-EDGE96, HPC – 0011828-001
CPU	2x Intel E 5335 (2.0 GHz)
Communicator	Infiniband 20 Gb/s

Disk Memory(per node)	160 Gb
GPU	-
RAM(per Node)	16 Gb
Total Ram Amount	786Gb
Total	48 nodes , 384 cores
Peak Efficiency	3,07 Tflops

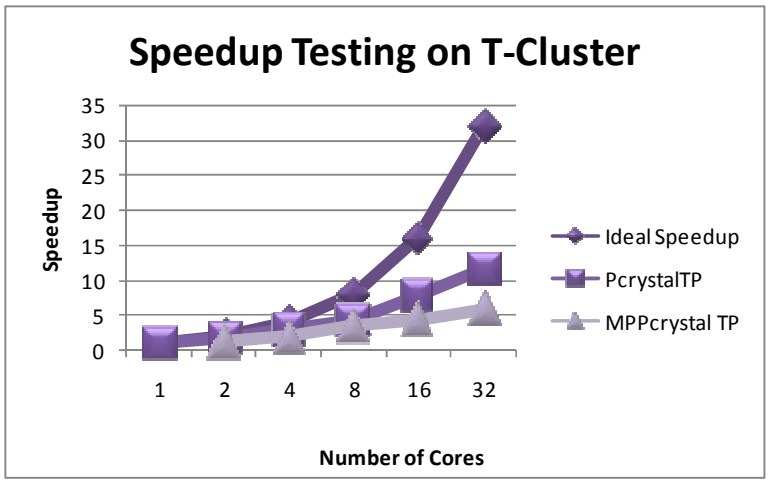


Fig. 3: Speedup testing on T-Platforms Cluster

6. Benchmarking the T-Platforms Cluster

We analyzed not only the speedup of the T-Platforms cluster, but also made testing the performances by using the NPB 3.3 (fig. 4). The NAS Parallel Benchmarks (NPB) are a small set of programs designed to help evaluate the performance of parallel supercomputers. The benchmarks are derived from computational fluid dynamics (CFD) applications and consist of five kernels and three pseudo-applications in the original "pencil-and-paper" specification (NPB 1). The benchmark suite has been extended to include new benchmarks for unstructured adaptive mesh, parallel I/O, multi-zone applications, and computational grids. Problem sizes in NPB are predefined and indicated as different classes. By the problem sizes we can categorize like, as shown below [8]:

- Class S: small for quick test purposes,
- Class W: workstation size (for workstations),
- Classes A, B, C: standard test problems; ~4X size increase going from one class to the next,
- Classes D, E, F: large test problems; ~16X size increase from each of the previous classes.

6.1 Benchmark Specifications

The original eight benchmarks specified in NPB 1 mimic the computation and data movement in CFD applications are:

- five kernels
  - IS - Integer Sort, random memory access
  - EP - Embarrassingly Parallel
  - CG - Conjugate Gradient, irregular memory access and communication
  - MG - Multi-Grid on a sequence of meshes, long- and short-distance communication, memory intensive

- FT - discrete 3D fast Fourier Transform, all-to-all communication
- three pseudo applications
  - BT - Block Tri-diagonal solver
  - SP - Scalar Penta diagonal solver
  - LU - Lower-Upper Gauss-Seidel solver.

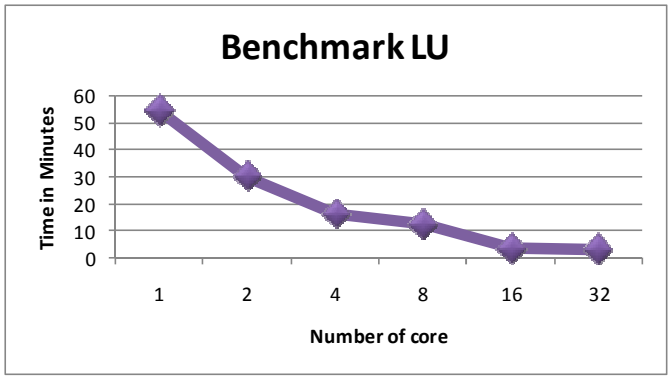


Fig. 4: Test result of NPB LU Class C

Conclusion

The speed multipliers to be gained from using a parallel processor depend upon a number of factors that may be evaluated prior to making an implementation investment. First is the inherent parallelism of the system. This determines the potential for useful processing to occur concurrently on a parallel processor. Second, the hardware architecture must support the ability to access memory local to each processor concurrently as well as minimize the time for transfers between processors. Third, the OS supporting the hardware must provide facilities to allocate and assign processor resources by a run-time system that has been optimized to use knowledge of the software architecture. This requires a software development environment that makes it easy to develop architectures of independent modules with minimized communication between them, reducing overhead and idle time. It also requires that a run-time environment be generated with knowledge of the module independence (the software architecture), so that modules are allocated to processors, and threads assigned within those independent modules, in a way that takes maximum advantage of parallel processor resources [5].

References

[1] G. M. Amdahl. Validity of the Single-Processor Approach to Achieving Large Scale Computing Capabilities. In *AFIPS Conference Proceedings*, 1967.

[2] Gustafson, J.L., Reevaluating Amdahl's Law, Communications of the ACM, Vol. 31 Issue 5, p. 532 – 533, 1988.

[3] Shi, Y., Reevaluating Amdahl's Law and Gustafson's Law, Temple Un., Oct 1996.

[4] Erlin Yao, Yungang Bao, Guangming Tan, Mingyu Chen, Extending Amdahl's law in the multicore era, ACM SIGMETRICS Performance Evaluation Review, v.37 n.2, September 2009.

[5] Mark D. Hill and Michael R. Marty, Amdahl's law in multicore era, Computer, Vol. 41, Issue: 7, Page(s): 33 - 38, 2008.

[6] Xian-He Sun, Yong Chen, Reevaluating Amdahl's law in the multicore era, Journal of Parallel and Distributed Computing, v.70 n.2, p.183-188, February, 2010.

[7] Hwang, K., Advanced computer architecture, parallelism, scalability, programmability(McGraw-Hill, NewYork) chapter 3, 1993.

[8] NAS Parallel Benchmark , official documents center http://www.nas.nasa.gov/publications/npb.html

DATABASE CONSOLIDATION USED FOR PRIVATE CLOUD

A.V. Bogdanov[1], Thurein Kyaw Lwin[2], Ye Myint Naing[2]
[1] *Institute for High-performance computing and integrated systems, Saint-Petersburg, Russia*
bogdanov@csa.ru
[2]*St.Petersburg State Marine Technical University, Saint-Petersburg, Russia*
trkl.mm@mail.ru, yemyintnaing@gmail.com

This paper describes the benefits of cloud computing, virtualization, database integration and how Database can be successfully consolidated onto a private cloud through several deployment models. Consolidating databases onto a private cloud is typically done in one of two ways: infrastructure cloud (server consolidation) or database cloud (operating system consolidation). Consolidation can be achieved through server, operating system, and database consolidation, and the higher the consolidation density achieved, the greater the efficiency. And then we evaluate two database architectures - shared-disk and shared-nothing for their compatibility with cloud computing. Technological advances have put shared-disk performance on par with shared-nothing, while cloud computing and virtualization strongly favor the shared-disk architecture.

Introduction

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources or shared services (e.g., networks, servers, storage, applications, and IT services). The key benefits of cloud computing are reduced costs, reduced complexity, improved quality of service, and increased flexibility when responding to changes in workload. We can choose either public or private clouds to meet these needs. However, driven by concerns over security, regulatory compliance, control over quality of service, and long-term costs, many choose internal private clouds. Private clouds provide the same cost and flexibility benefits as public clouds, but they also enable to control the quality of service delivered to users. In addition, private clouds allow better to secure data. This cloud model promotes availability and is composed of three service models, and four deployment models.

Cloud Software as a Service (SaaS) - The capability provided to the consumer is to use the provider's applications running on a cloud infrastructure. The applications are accessible from various client devices through a thin client interface such as a web browser (e.g., web-based email). The user does not manage or control the underlying cloud infrastructure including network, servers, operating systems, storage, or even individual application capabilities, with the possible exception of limited user-specific application configuration settings.

Cloud Platform as a Service (PaaS) - The capability provided to the user is to deploy onto the cloud infrastructure user-created or acquired applications created using programming languages and tools supported by the provider. The user does not manage or control the underlying cloud infrastructure including network, servers, operating systems, or storage, but has control over the deployed applications and possibly application hosting environment configurations.

Cloud Infrastructure as a Service (IaaS) -The capability provided to the user is to provide processing, storage, networks, and other fundamental computing resources where the user is able to deploy and run arbitrary software, which can include operating systems and applications. The user does not manage or control the underlying cloud infrastructure but has control over operating systems, storage, deployed applications, and possibly limited control of select networking components (e.g., host firewalls).

Private cloud - The cloud infrastructure is operated solely for an organization. It may be managed by the organization or a third party and may exist on premise or off premise.

Community cloud - The cloud infrastructure is shared by several organizations and supports a specific community that has shared concerns (e.g., mission, security requirements, policy, etc.). It may be managed by the organizations or a third party and may exist on premises or off premises.

Public cloud - The cloud infrastructure is made available to the general public or a large industry group and is owned by an organization selling cloud services.

Hybrid cloud - The cloud infrastructure is a composition of two or more clouds (private, community, or public) that remain unique entities but are bound together by standardized or proprietary technology that enables data and application portability (e.g., cloud bursting for load-balancing between clouds) [1].

Private Cloud

Using virtualization, some companies are building private cloud computing environments intended to be used only by their employees or designated partners. Also referred to as internal clouds, private clouds can offer the benefits of public cloud computing, while still enabling the organization to retain greater control over the data and process. Virtualization offers a means to consolidate applications and servers, and changes the traditional relationship between software and hardware. The advantage of virtualization is the realization of further consolidation and more utilization of the systems. Virtualization enables more than one application to run on a server and the capability to distribute multiple applications over multiple servers for resiliency. The primary driver of cloud computing's cost advantages is virtualization. Virtualization is the ability to create, operate and manage computing instances independent of the underlying hardware. Virtualization delivers many advantages but from the perspective of the cloud company, the greatest advantage is cost savings. In these clouds, users are given access to virtual machines on which they can install and run arbitrary software, including database systems. Users can also deploy database appliances on these clouds, which are virtual machines with pre-installed pre-configured database systems [2-4].

Database management system (DBMS) architectures and the Cloud

Whether an application resides on a desktop or is virtualized in a cloud somewhere, when data is used or stored, it often requires the use of a database. A database is a structured collection of records or data that is stored in a computer system. A database relies on software known as a database management system (DBMS) to organize, store, and retrieve data [5]. Designing a database management system (DBMS) is an exercise in identifying and minimizing the various performance constraints imposed by the current computing technologies. The two primary DBMS architectures are shared-nothing and shared-disk (fig.1). Shared-nothing is the most common database architecture. Shared-nothing databases split or partition the data so that each database server exclusively processes and maintains its own piece of the database. Shared-disk is analogous to a single large trough of data, where any number of database nodes can process any portion of that data. Based upon traditional computing constraints, the shared-nothing architecture has been the price-performance leader [3]. If your database resides in a single database, and high-availability is not a concern, then the shared-nothing architecture is ideal. Shared-nothing can scale-up moving to a more powerful server but scaling-out is a much more significant challenge.

The shared-disk architecture also scales-up to larger machines, but the primary advantage of shared-disk is the ability to scale-out. Scale-out refers to running your database on multiple servers or virtual machines. IBM's mainframe databases IMS and DB2 use shared-disk architecture. The Oracle Parallel Server (OPS) started as shared-disk, but they later added a shared-cache called Cache Fusion, renaming the product Oracle Real Application Clusters (RAC). By using a shared cache, the data sharing between nodes is much more efficient than sharing data via the disk [5].

The one fly in the ointment is that shared-nothing databases don't work in virtualized environments. When using a shared-nothing architecture, applications must be hardwired to specific database servers. Those databases are then hardwired to their specific data partitions, you can't

virtualize them. So, while web servers, applications, middleware and storage can all exploit virtualization, databases require a dedicated server. Since that database server is static, you must size that server for your peak load. As a result, you'll get the cost advantages of cloud computing at all levels except the database level.

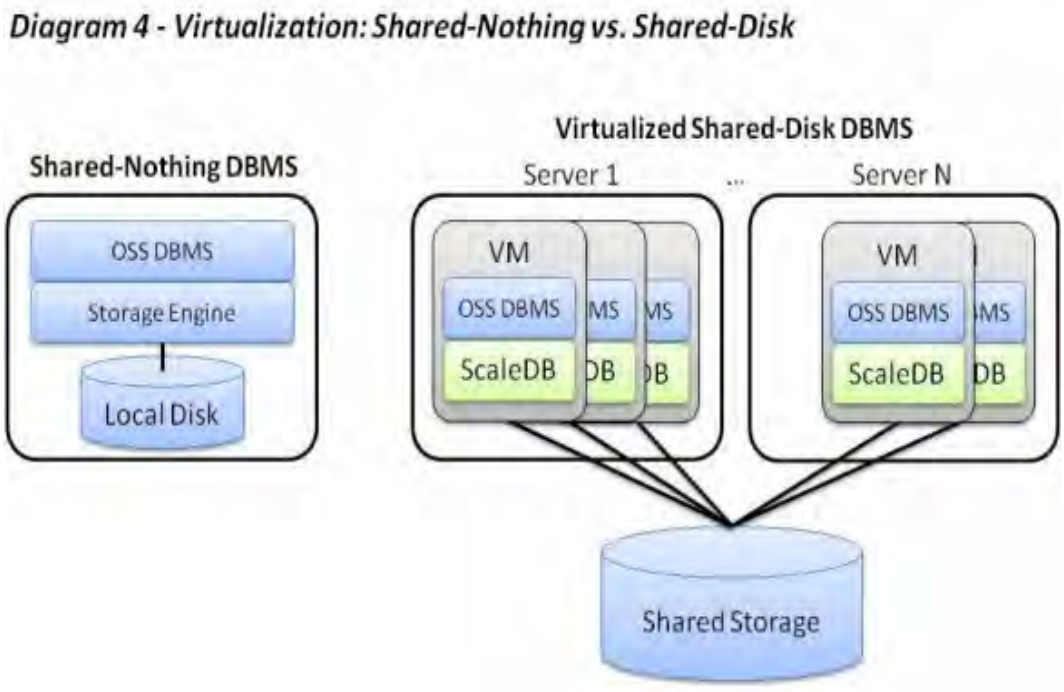


Figure 1: Virtualization: Shared-Nothing vs. Shared-Disk

The shared-disk DBMS relies on a cluster of identical database nodes processing a single trough of data. These identical database compute nodes are ideal for virtualization. As a result, you can scale your database computing elastically on demand. It works effectively with dedicated and underutilized servers. [3].

The shared-disk database architecture is ideally suited to cloud computing. The shared-disk architecture requires fewer and lower-cost servers, it provides high-availability, it reduces maintenance costs by eliminating partitioning, and it delivers dynamic scalability.[7] Shared-disk databases allow clusters of low-cost servers to use a single collection of data, typically served up by a Storage Area Network (SAN) or Network Attached Storage (NAS). All of the data is available to all of the servers; there is no partitioning of the data. The shared-disk DBMS architecture has other important advantages in addition to elastic scalability that make it very appealing for deployment in the cloud [3].

Database Services on a Private Cloud

For database environments, the PaaS cloud model provides better IT services than the IaaS model. The PaaS model provides enough resources in the cloud that databases can quickly get up and running and still have enough latitude for users to create the applications they need. Additionally, central IT management, security, and efficiency are greatly enhanced through consistency and economies of scale. Conversely, with the IaaS model, each tenant must build most of the stack on their own, lengthening time to deployment and resulting in inconsistent stacks that are harder to manage. A private cloud is an efficient way to deliver database services because it enables IT departments to consolidate servers, storage, and database workloads onto a shared hardware and software infrastructure. Databases deployed on a private cloud offer compelling advantages in cost, quality of service, and agility by providing on-demand access to database services in a self-service, elastically scalable, and metered manner[5]. Private clouds are a better option than public clouds for many reasons. Public clouds typically provide little or no availability or performance service-level

agreements, and there are potential data security risks. In contrast, private clouds enable IT departments to have complete control over the performance and availability service levels they provide [6].

Steps to Database Consolidation onto a Private Cloud

Building a private cloud requires the transformation and optimization of the IT infrastructure, and that is typically executed in three steps: rationalization, architecture optimization, and implementation of shared services.

Rationalization

IT rationalization determines the best use of IT services and reduces nonproductive redundancy throughout the enterprise. IT departments should rationalize their technology architecture by standardizing their service portfolio and technology stack. Through standardization, the IT environment becomes much more homogenous, which makes it easier to manage. It also reduces costs and complexity and increases agility [6].

Architecture Optimization

All layers of the technology stack must support service-level objectives and growth requirements. Scalability, availability, data security, and datacenter management are only as strong as the weakest link. Balanced technology architecture employs virtualization, consolidation, and management automation to meet business requirements. Virtualization, for example, transforms the typical server-to-application silo model to a multi tenancy model. The key to virtualization is not necessarily the underlying technology, but rather the capability to abstract resources requested from resources provided [6].

Shared Services

One can leverage shared services to reduce costs and meet the demands of their users, but there are many operational, securities, organizational, and financial aspects of shared services that must be managed to ensure effective adoption. Consolidation is vital to shared services, as it allows IT to restructure resources by combining multiple applications into a cohesive environment. Consolidation goes beyond hard cost savings; it simplifies management, improves resource utilization, and streamlines conformity to security and compliance standards. Therefore, the next item to consider is the level of consolidation that can be achieved in private cloud architecture [6].

First need to consider is the level of consolidation that can be achieved in private cloud architecture.

- Server consolidation. Reduce the number of physical servers and consolidate databases onto a smaller server footprint.
- Storage consolidation. Unify the storage pool through improved use of free space in a virtual storage pool.
- Operating system consolidation. Reduce the number of operating system installations. Reducing server footprint does not always provide the best ROI, but reducing the number of operating systems will improve overall manageability.
- Database consolidation. Reduce the number of database instances through schema consolidation. Separate databases are consolidated as schemas in a single database, reducing the number of databases to manage and maintain.
- Workload consolidation. Merge the redundant databases that support business intelligence or operational data store systems. By consolidating into a single data store, these workloads benefit from the additional resources and scalability provided by the private cloud infrastructure [6], [7].

Conclusion

The paper summarizes the database requirements for cloud databases and compares the suitability of different database architectures to cloud computing. Whether you are assembling, managing or developing a cloud computing platform, you need a cloud-compatible database. The shared-nothing DBMS architecture gained widespread adoption on the basis of performance and cost advantages that no longer exist. Shared storage, with the help of extremely fast interconnects, now delivers data to the CPU several times faster than a local disk. Consolidating databases onto a private cloud is a new model for the delivery of database services. Private clouds consolidate servers, storage, operating systems, databases, and mixed workloads onto a shared hardware and software infrastructure. Deploying databases on a consolidated private cloud enables IT departments to improve quality of service levels as measured in terms of database performance, availability, and data security and reduce capital and operating costs. Consolidation can be achieved through server, operating system, and database consolidation, and the higher the consolidation density achieved, the more efficient is the data processing.

References

[1] Peter Mell and Tim Grance. National Institute of Standards and Technology, Information Technology Laboratory. The NIST Definition of Cloud Computing.

[2] Ronald L.Krutz and Russell Dean Vines. Cloud Security: A Comprehensive Guide to Secure Cloud Computing.

[3] Mike Hongan, CEO, ScaleDB Inc. Database Virtualization and the Cloud: How database Virtualization, Cloud computing and other advances will reshape the database landscape. December 10, 2009.

[4] Ashraf Aboulnaga, Kenneth Salem, Ahmed A. Soror, Umar Farooq Minhas, Peter Kokosielis, Sunil Kamath. Deploying Database Appliances in the Cloud.

[5] Moshe Shadmon, CTO, ScaleDB Inc. Clustered Database Storage Engine: Scaling MySQL in the Cloud. November 12, 2010.

[6] Nitin Vengurlekar, Contributing Authors: Michael Timpanaro-Perrota, Mark Macdonald, Philip Newlan. Database Consolidation onto Private Clouds.

[7] Mike Hogan, CEO ScaleDB Inc. Cloud Computing & Databases How databases can meet the demands of cloud computing. November 2008.

IMPROVING THE EFFICIENCY OF DISTRIBUTED INTELLIGENT SYSTEMS

A.B. Degtyarev[1], V.P. Guskov[2], A.V. Eroshkin[2]
[1] *Saint-Petersburg State University , Russia*
deg@csa.ru
[2] *Saint-Petersburg State Electrotechnical University "LETI", Russia*
dev90@mail.ru eroshkin.a@pochta.ru

The modern complex dynamic objects and systems are often difficult to control. Analysis of a significant amount of information about their condition is required to select the correct control strategy. For this analysis the decision maker should have an expert level of knowledge and produce a series of calculations. When adding real-time requirements, there is a need to ensure additionally the timeliness of decision making. In that case intelligent decision support system (DSS) comes to the aid. It removes part of the work from the decision-makers: provide information about the object in readable form and suggest ready-made decisions options [1].

In the intelligent systems design within a single software package appears some disadvantages. Such systems should have a complicated internal structure, because their effective work requires processing of a large amount of polytypic information. It is necessary to introduce into the system a large amount of expert knowledge, eliminating the contradictions that arise in this case [1, 2].

The single system has a number of major drawbacks, namely:

1. A single system will generate a large amount of data, most of which is not required constantly. For optimization of this data producing a complex control subsystem is required, which determines modules that need to run on the current step.
2. A single system is centralized, therefore has a low fault tolerance.
3. A single system does not scale well. When adding new features balancing problems and conflict resolution are faced again.
4. To deploy a single system requires powerful and expensive computer, since it is necessary to ensure high performance.

An alternative to a single software system can be a set of smaller systems that interact with each other. These expert subsystems can be developed separately, which simplifies the design. Subsystems are independent, so the probability of any conflicts is reduced.

The work of each subsystem starts after request of the user or other subsystem. Services are provided to customer "on-demand", therefore special scheduling mechanism is not required. Addition of new subsystems in the complex does not cause difficulties.

Let us define efficiency as increased productivity (improvement in one or more metrics required for the project) at a fixed resource cost or reducing resources costs to achieve a fixed performance.

Speaking about distributed systems, particular attention should be paid to the metric indicator "The cost of network sharing." Precisely the necessity of the cost for slow exchanges between nodes is a major difficulty in distributed computing. This factor becomes especially important when the nodes are distributed geographically. Downtimes in anticipation of input data can many times exceed the duration of the actual computation.

Needs to move in several directions to improve efficiency in the metric "The cost of the network exchange":

- improve the quality of communications between the nodes;
- optimize procedure of initial data preparation for nodes in accordance with the application logic (outpacing data distribution to the nodes);

- reduce the amount of data transferred between nodes (try to divide functionality across the nodes such a way that the largest possible amount of required data is available locally).

The first direction has no influence on the application architecture. In such a way it is necessary to improve any distributed system. The second, by contrast, is closely linked to the specific logic of application and the services of system.

The third direction also depends on the specific logic, but within the subject area. Within this direction it is convenient to use the SOA concept. [4]. Service-oriented architecture provides system construction using loosely-coupled components and this matches the requirements of a developed approach (Fig. 1). Also, one of the main advantages of SOA is that it is cross-platform, which allows systems to operate in a distributed heterogeneous environment. Another advantage is high scalability, making it possible to easily add new functionality to the system [3].

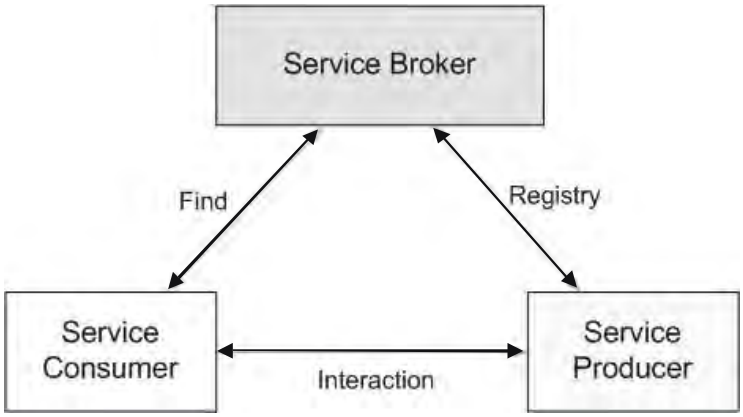


Fig.1: Structural diagram of SOA

Within service-oriented architecture intelligent decision support system consists of the following units (Fig. 2):

- expert components;
- broker component;
- archive.

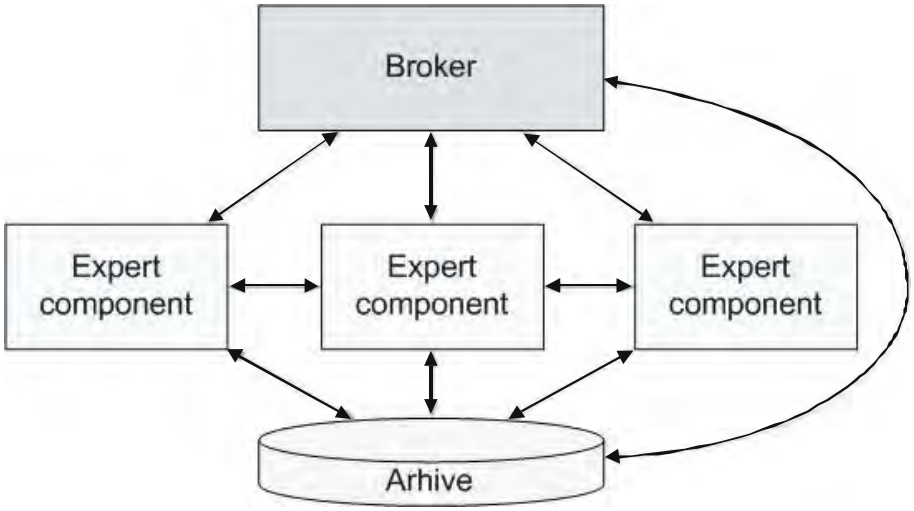


Fig. 2: Decision support system in the SOA

Expert components receive data from external sources, process them and send to the archive for storage and potential processing by other components. They should be designed as independently

of each other as it is possible. Overall productivity will be higher if the volume of transmitted data is low.

Broker records information about the data that you can get within the system. If any of the components require additional data, it sends a request to the broker and receives a link to the source. When registering data sources incorrect description can appear and it makes the system work more complex. The broker should be able to distinguish between the data sources to identify duplication and also identify and fix incorrect description.

Archive is used for long-term storage of data about the state of the object. In normal operation the components require a data for the recent short period of time. However, in some cases, archived information may be needed. Also external customers can use the service to retrieve data from the archive.

Using a single archive also decreases the system performance because of constant synchronization with expert components. It makes sense to use a distributed archive (Fig. 3). In this case, each component archives the data which coming to its input, and registers data access service at the broker.

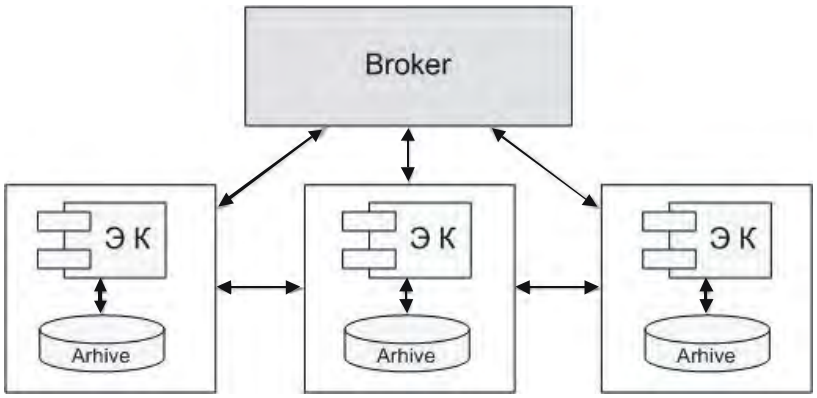


Fig. 3: Distributed archive

The presented architecture is good, if it a possible to design subsystems so that they have no common input. This is difficult to actually existing DSS. Components that are stored in the archive copies of the same data appear in the system. For fixing such duplication you can create a single archive for such common data and organize a coherent cache for the consumers. If the data sets are used only by one component, they are stored in its archive.

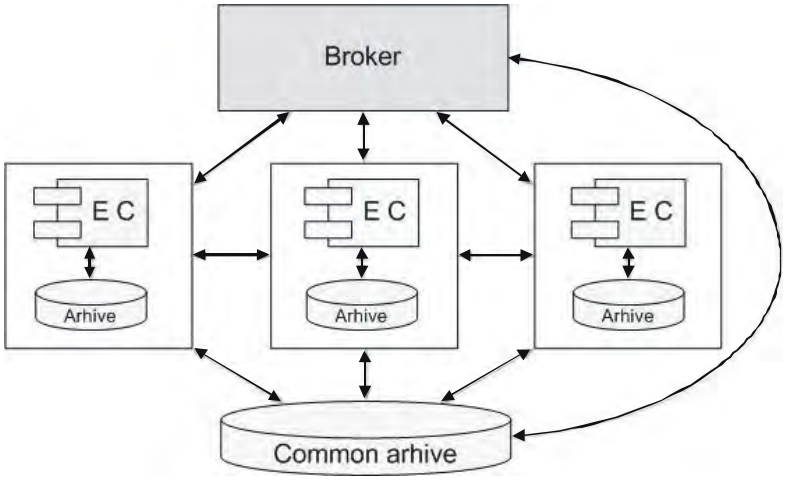


Fig. 4: Common archive for shared data

We should remember about providing high availability of DSS. Consequently data duplication and additional network exchanges will appear in the most important places where the system can break down. The mission of this work is to minimize redundancy.

In the frameworks of the project, a prototype of the described system is developed. The system is a set of expert components that can be deployed on different hardware systems. Broker provides the interaction between users and services. Depending on the input data service component can either run the procedure provided by the system or transfer of other components. The broker may be distributed, that is to be deployed on different machines; the load is distributed evenly.

At this moment the system consists of three components: operation with a distributed database, made on the basis of database PostgreSQL, searching by ID and declaration of two identical objects in the database, searching juveniles and giving warning of their support - and the broker.

Components of the project have been developed on Java in a view of the convenience and cross-platform development of the language. JAX-WS used to publish services (Java API for XML Web Services).

Services are described using WSDL 2.0 – web services description language based on XML. WSDL description contains the minimum information required to run the service. It supports SOAP, and REST services [5]. In addition, WSDL 2.0 is a recommendation of W3C and is used by many service providers and also it is easy to integrate external sources of services in the system. Further work is planned for the implementation of the components necessary for the subject area. The currently working prototype was installed on the server in St.Petersburg Electrotechnical University "LETI", in the future we will map the system on cloud.

References

[1] Aleksandrov V.L., Matlah A.P., Nechaev Yu.I., Polyakov V.I., Rostovtsev D.M. Intelligent systems for marine research and technology – Saint-Petersburg: Publishing Center of State Marine Technical University of St.Petersburg, 2001.

[2] Russel S., Norvig P. - Artificial Intelligence. A Modern Approach, Second Edition, 2006.

[3] Intersoft Lab. Service-oriented architecture, 2005. URL: http://citforum.ru/internet/webservice/soa/

[4] K. Channabasavaiah, K. Holley, E.M. Tuggle Jr. Migrating to a service-oriented architecture. – NY: Business on demand, 2004.

[5] Lawrence Mande. Describe REST Web services with WSDL 2.0, 2008. URL: http://www.ibm.com/developerworks/webservices/library/ws-restwsdl/

BES-III DISTRIBUTED COMPUTING

Z.Y. Deng[1], W.D. Li[1], L. Lin[2], C. Nicholson[3], X.M. Zhang[1], A. Zhemchugov[4]
1) Institute of High Energy Physics, 100049 Beijing, China
2) Soochow University, 215000 Suzhou, China
3) Graduate University of Chinese Academy of Sciences, 100049 Beijing, China
4) Joint Institute for Nuclear Research, 141980, Dubna, Russia

Introduction

The BES-III experiment [1] started to take data in 2009 after a major upgrade of the electron-positron collider BEPC-II at the Institute of High Energy Physics (Beijing, China). The experiment is run by an international collaboration of more than 400 active members from 52 institutes in 12 countries from around the world. The main physics goals of the experiment are precision measurements in the τ-charm domain in the energy range of 2 – 4.6 GeV. The BES-III experiment has already taken the world's largest data samples of J/ψ ($1.2x10^9$ events) and ψ' decays ($0.3x10^9$ events), as well as a large amount of ψ(3770) data and a unique sample of ψ(4040) data. The total volume of experimental data is about 0.5 PB, of which about 120 TB is event summary data for physics analysis (DSTs). This amount of data is rather large to be processed in a single computing center. Use of distributed computing looks like an attractive option to increase the computing power of the experiment and speed up data analysis.

The BES-III computing model

Experimental data are taken from the BES-III detector and stored as raw to the tape storage managed by CASTOR. The maximum data rate is about 40 MB/s. After reconstruction DSTs are produced and used in further physics analysis. DSTs are stored in a disk pool managed by Lustre and can be accessed only from internal IHEP network. The total amount of DSTs currently is about 100 TB. Both inclusive and exclusive Monte-Carlo simulation (MC) is made for each data sample as well. Experimental data taken with random trigger are used in the simulation to reproduce noise and machine background individually for each run. The total amount of MC DSTs is more than 20 TB now. The BES-III offline software is based on the Gaudi framework and runs on Scientific Linux CERN operating system. Almost all data processing and user analysis are carried out at IHEP local computing farm so far.

The BES-III distributed computing system

After successful deployment in the LHC experiments, Grid computing became a routine tool for data processing in high energy physics. However, the main difficulty for widespread use of the Grid tools developed in the WLCG project is their large scale and complexity. It is not easy to adapt the distributed computing software which was designed for LHC experiments for use in a medium scale experiment, and because of limited manpower, it is even more difficult to maintain. For BES-III the situation is even worse, because very few participating sites are members of WLCG; there are therefore few experienced Grid users and developers and little corresponding computing infrastructure already installed. Another problem is that network connectivity between institutes participating in the BES-III experiment is typically low. All these considerations motivate the following approach to the BES-III distributed computing model.

It is assumed that remote sites participate only in MC production and physics analysis, while all reconstruction of experimental and simulated data is done at IHEP as before. If this is the case, three operation models are considered, depending on the capabilities and priorities of each site:

a) MC simulation runs at remote sites. The resulting data are copied back to IHEP and then MC reconstruction runs there. (This model is convenient for sites with no SE or with only a small one);
b) MC simulation and reconstruction runs at remote sites. The resulting data are copied back to IHEP;
c) DSTs are copied from IHEP and other sites and analyzed using local resources. For the moment there are no plans to develop a distributed analysis system.

Several components are necessary to implement these models in BES-III: authentication and authorization system, production job management system, data management system and information and monitoring system. Authentication and authorization is based on use of X.509 certificates and on membership in the virtual organization 'bes' which is managed by VOMS from the gLite software stack [2]. For the information and monitoring system, custom tools will be developed, reusing components of the CERN Dashboard [3]. The most challenging parts of the BES-III distributed computing system are the job management and data management systems.

Job management system

The DIRAC (Distributed Infrastructure with Remote Agent Control) project is the most advanced and complete Grid solution for medium-scale high energy physics experiments today [4]. This solution was designed originally for the LHCb experiment, but was later developed as a generic product which could be used to access distributed computing resources in various communities of users. The key point of DIRAC is its workload management system, based on generic pilot jobs.

DIRAC is adopted as a central part of the BES-III job management system. A prototype installation of DIRAC has already been set up for BES-III, with five remote sites and the DIRAC server running at the IHEP central site (Beijing). Users of DIRAC can also benefit from use of the GANGA tool to manage production and analysis jobs [5]. The main problem is that not all LRMS used at BES-III remote sites, like Condor, are supported by DIRAC yet, so new DIRAC plugins need to be developed.

CVMFS (CERN VM File System) [6] is deployed to centrally manage the experiment software BOSS and distribute it to the target sites.

Data management system

The data management system is a key issue when building the BES-III distributed computing system. Of course, DIRAC has certain data management functionality, but it is not sufficient for BES-III, taking into account that network is not stable between most of the BES-III remote sites. There are several issues to be solved before BES-III data management becomes operational.

The first one concerns reliability of data transfer. A number of services to provide reliable file transfer between Grid sites already exist. FTS from the EMI/gLite software stack is adopted as a data transfer service for BES-III. The BES-III FTS server is installed at JINR, providing reliable data transfer between IHEP and remote sites via both SRM and GridFTP protocols. Certain modifications are made, though, to avoid using BDII and to optimize performance of data transfer.

The second issue is related to the file and metadata catalog. There are two solutions – one is AMGA from the gLite software [7] and the other is DFC from DIRAC. Of course, the latter fits better because integration with other pieces of the BES-III Grid is easy. Several tests were made to assure that the performance of DFC meets BES-III requirements. The BES-III metadata schema was implemented in both AMGA and DFC catalogs, using current data (~$2x10^5$ files) and a MySQL backend. Configuration was optimized both for AMGA and DFC (8 DFC instances, max. 50 threads / instance; for AMGA maximum 140 processes allowed). The tests have shown that with a low number of clients AMGA queries are ~10x faster than DFC, but with a high number of clients query times become approximately equal (see Fig.1). At the same time, DFC CPU usage rises more slowly with number of concurrent clients (see Fig.2). As a result, both AMGA and DFC give acceptable performance, but DFC meets more of BES-III requirements.

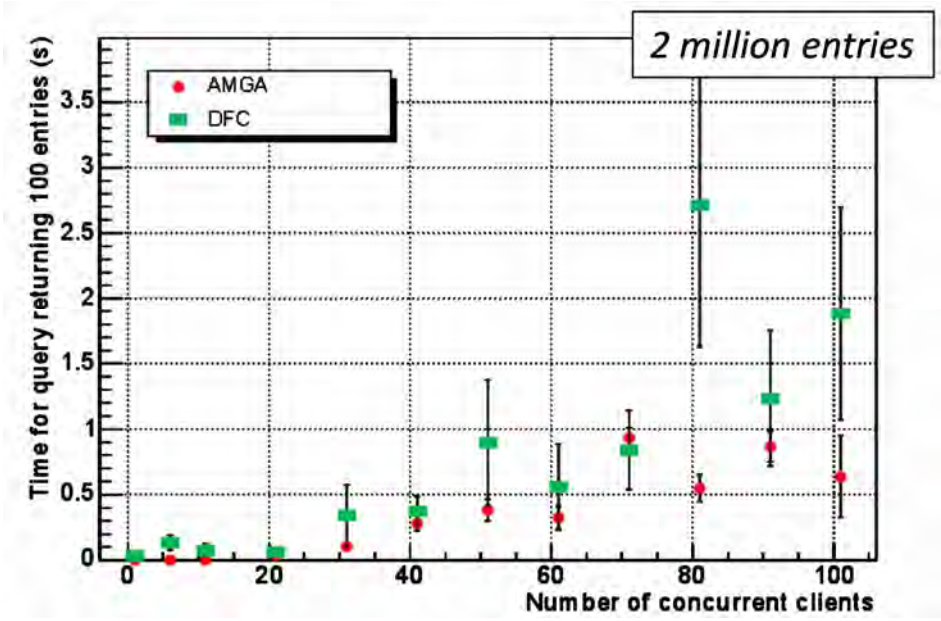


Fig. 1: Performance of AMGA and DFC versus the number of concurrent clients

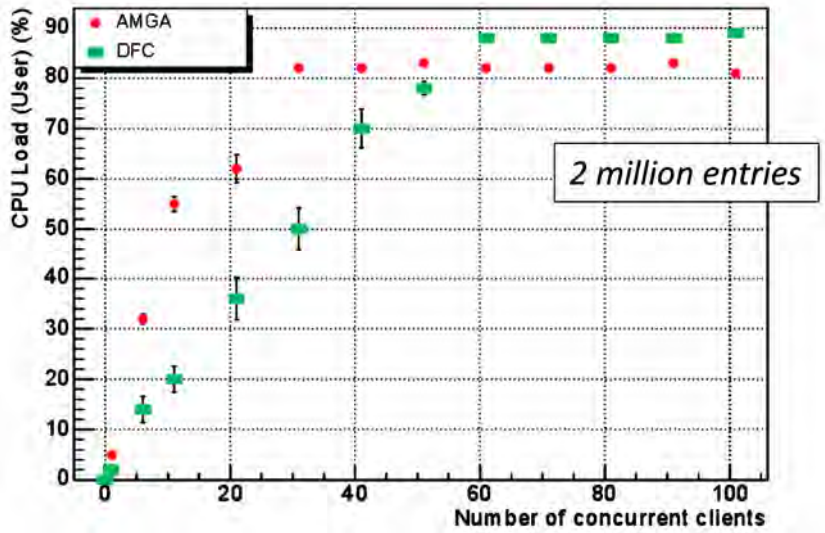


Fig. 2: CPU load created by AMGA and DFC versus the number of concurrent clients

The last issue concerns management of datasets. The BES-III experiment collects different types of data, so using datasets as containers of files and metadata is very convenient. DIRAC allows DFC queries or 'meta-sets' to be used, which can be considered as dynamically changing datasets. At the same time, most of the physics tasks at BES-III require reproducibility of results of these queries, because the total number of events is important in the data analysis. Datasets at BES-III should therefore be static in the sense that their content is always the same. For the moment it is assumed that dynamic datasets from DIRAC can be used provided extra instruments to assure the dataset constancy are implemented.

Of course, to glue all these pieces into a working system one has to develop other tools which take into account BES-III specific issues, provide a user interface and API for the job management system etc. These tools are under development as the BADGER (BES-III Advanced Data manaGER) project.

Summary

The BES-III experiment has been running since 2008 and is currently the best source of data in the τ-charm domain. The amount of data is increasingly high so using distributed computing is an attractive option to go beyond the limits of the computing power available at BES-III now. A BES-III Grid is being constructed, based on the DIRAC infrastructure combined with experiment-specific data management. A working prototype has already been set up which unites the computing resources of IHEP CAS, GUCAS, Peking University, USTC, the University of Minnesota and JINR, with several more sites planning to join. In conclusion, it is worth mentioning that Grid computing is becoming widely used not only in big projects like the LHC experiments, but also in many medium-scale experiments in high energy physics and even beyond. Experience gained in the BES-III Grid is valuable to better design a yet missing universal Grid solution for medium scale projects like these.

References

[1] M.Ablikim et al., Design and construction of the BESIII detector, Nucl. Instrum. Meth A614 (2010) 345.
[2] http://glite.cern.ch/
[3] http://dashboard.cern.ch/
[4] http://diracgrid.org/
[5] J.T.Moscicki et al., Ganga: a tool for computational-task management and easy access to Grid resources; Comp. Phys. Comm. Vol 180, Issue 11, (2009) ;arXiv:0902.2685; see also http://cern.ch/ganga
[6] http://cernvm.cern.ch/portal/filesystem
[7] http://amga.web.cern.ch/amga/

CERTREQ: A STANDALONE TOOL FOR CERTIFICATE REQUESTS GENERATION AND CERTIFICATES RETRIEVING IN GRIDNNN[1]

Yu.Yu. Dubenskaya, A.P. Kryukov, L.V. Shamardin
Scobeltsyn Institute of Nuclear Physics Lomonosov Moscow State University
jdubenskaya@gmail.com

Certreq is a standalone tool for management of certificates that are intended for authentication and authorization purposes in the National Nanotechnology Network (GridNNN). The security concept of GridNNN is fully based on a public key infrastructure (PKI). To be able to send any request to services every GridNNN user has to obtain his/her personal X.509 certificate. Nowadays certificate obtaining procedure is only partly automated. A user has to understand the certificate issuance workflow and perform all the steps in the right order. Some of the steps are manual and thus error-prone while the others require use of special cryptographic libraries. The main goal of the certreq development is to provide users with an easy-to-use tool for managing their certificates and certificate requests. Certreq tool allows one to perform the following tasks:

- create new certificate requests;
- retrieve issued certificates from the Certification Authority (CA);
- save downloaded certificates with the proper names in the proper places;
- archive previously used private keys and certificates.

Using of the certificates in GridNNN should not complicate access to the resources for the legitimate users. Thus the key points for the certreq tool are:

- certificate obtaining does not require special skills in PKI and its specific implementations in Windows and/or Unix operating systems,
- certificate obtaining is as automated as possible.

Certreq is available in two implementations: a command-line tool that could be useful for the Unix users and a tool with a graphical interface for Windows users.

1. Introduction

The main target of the Russian grid project (GRID for the National Nanotechnology Network GridNNN [1, 2]) is to provide scientists with a comfortable and secure access to supercomputer resources. The security concept of GridNNN is fully based on a public key infrastructure (PKI) [3]. A personal X.509 certificate [4] must be created for every user. This certificate must be present whenever a user wants to request any data or to execute a computational task.

Use of PKI as a base for authentication and authorization along with the incontestable benefits has also some disadvantages. First of all it is about the usability issues that have been a concern for users and administrators of GRID computing for the past several years. Ideally GRID environment should provide its users with seamless and easy access to every resource that they are authorized to use. In practice an end user may come across a formidable obstacle trying to work with certificates. The fact is that obtaining and management of the X.509 certificates requires deep understanding of the basic concepts of the PKI: a private key, a certificate request, a certificate, a registration authority (RA) and a certification authority (CA), a digital signature and so on.

The certificate obtaining procedure is described in the international standards and is a common one. To get the certificate, a user has to:

- generate a certificate request and send it to the CA;
- pass the registration procedure (directly in CA or through the RA): confirm his/her identity, and prove to have possession of the private key associated with the public key requested for a certificate;
- download certificate on his/her personal computer (PC).

[1] Partially supported by RFBR (Grant No 11-07-00434-a).

There are some additional requirements and restrictions that are specific for the current implementation of PKI in GridNNN:

- a private key and a certificate request must be generated on the user's PC;
- a printing form of the request must be prepared and brought to the RA;
- a printing form of the request must contain key modulus;
- a user must put his handwritten signature on the printing form of the request.

The situation is aggravated as the certificate obtaining procedure in GridNNN is only partly automated. Gathering of the user data and request generation are performed in several steps. At first a user is asked to fill the special form on the CA web server. As a result a partly filled printing form of the certificate request and a request generation script are prepared. Next a user has to run the script on a Unix computer. After that a prepared request should be sent to the CA, and the key modulus should be written into the printing form. Thereby filling of the printing form of the request with the key modulus is manual and thus error-prone.

Moreover everyone who wants to get a certificate is expected to be an experienced PKI user:

- To prepare a certificate request correctly a user has to understand the certificate issuance workflow and perform all the steps in the right order. Otherwise there is a good chance to visit RA twice or to accidentally delete a private key. As for certificate it's possible to download it again from the CA web server but the private key is kept on the local machine only and can not be restored in such an easy way.
- Standard request generation step requires use of the OpenSSL [5, 6] cryptographic library on Unix computer. Many users use Windows operation system (OS) and have no special cryptographic software (like OpenSSL) installed on their PCs by default. So a user is either expected to have access to the computer with Unix-like operating system or to have enough skills to generate a certificate request on Windows by some non-typical means.

All these tends to complication of the access to the GridNNN resources for the legitimate users (especially for the newcomers). So there is a need to develop a tool that will facilitate management of certificates and certificate requests for GridNNN users and thus will help to improve overall system usability.

2. Certificates management for GridNNN: points to improve

Summarizing the above mentioned GridNNN features, we can distinguish two main points in the certificate obtaining procedure that can be improved:

a. Automation of the certificates management.

b. Cryptography hiding.

Automation includes the following ideas:

- The certificate request handling procedures: gathering of user data, key and request generation, preparing and filling of the printing form, request uploading to the CA should be done automatically at one step.
- Similar, the certificate handling procedures: downloading of the issued certificate, replacement of the obsolete certificate and private key with the new ones, archiving of the previous credentials (if needed) should be done automatically at one step as well.

The overall idea of the cryptography hiding is that the certificate obtaining should not require special skills in PKI and its specific implementations in Windows and/or Unix operating systems. All the cryptographic operations should be unnoticeable for the user. It means that the user should neither install any cryptographic library nor execute directly any cryptographic command.

Surely the balance of security and usability is a main point for every complex system. Introducing of the new tool should not reduce the overall security of the Grid network. Another important aspect to be kept in mind is that the proposed solution should not require modifications of the existing infrastructure. Total change often tends to total inability to work for a long time. So it is preferable to perform the minimal changes.

3. A tool for certificates management: certreq

3.1. Description

In this paper we present a certreq - a standalone tool that is intended to facilitate management of certificates and certificate requests for GridNNN users. Also certreq is fully compatible with the existing infrastructure: the CA web-server is not affected, and all the changes only concern a certificate handling on the user's PC.

One of the benefits of the certreq design is that no installation and/or configuration are needed. The program is distributed as an archive. To get the program working, one only needs to:

- download a program archive;
- unpack it on the local computer;
- run the program file: certreq.py (on Unix) or certreq.exe (on Windows).

3.2. Functionality

Certreq functions could be divided into three separate blocks:

a. certificate requests management functions;
b. certificates management functions;
c. export/import functions.

Certificate requests management functions are executed at one step and include:

- generation of the new certificate request;
- preparation and filling of the printing form of the request;
- certificate request uploading to the CA.

Certificate management functions are executed at one step as well and include:

- searching for the issued certificate on the CA web site;
- retrieving of the issued certificate from the CA;
- archiving previously used private key and certificate;
- saving of the downloaded certificate to the proper location under the proper name.

Export/import functions:

- export of the private key and certificate in PKCS12 file;
- import of the private key and certificate from PKCS12 file.

The export/import functions are needed to copy/move a private key and a certificate from one PC to another. One the first PC a user should run "export" function of certreq to prepare a file in PKCS12 format. Then this file should be copied/moved to the second PC where "import" function of certreq is needed to be executed.

3.3. Modes of operation

Certreq tool is available in two implementations: a command-line tool that could be useful for the Unix users and a tool with a graphical user interface (GUI) for Windows users. On Windows command-line tool is also available but is not frequently asked-for. The GUI is arguably simpler and much more user-friendly for end users than a complex command-line interface. Every user action is accompanied with the extended commentaries. All the messages in GUI are localized. A graphical interface for Unix in not implemented yet but this task does not seem to be insuperable and could be done if there would be any demands. An example of a certreq GUI is shown in Fig.1.

Command-line interface is intended for administrators and advanced users. As opposed to the graphical interface, a command line interface is not localized and uses the English language for all the messages. This is done especially for those Unix users who prefers C or en locale. Command line interface is available in two modes:

- interactive mode;
- non-interactive mode (all parameters are set via options).

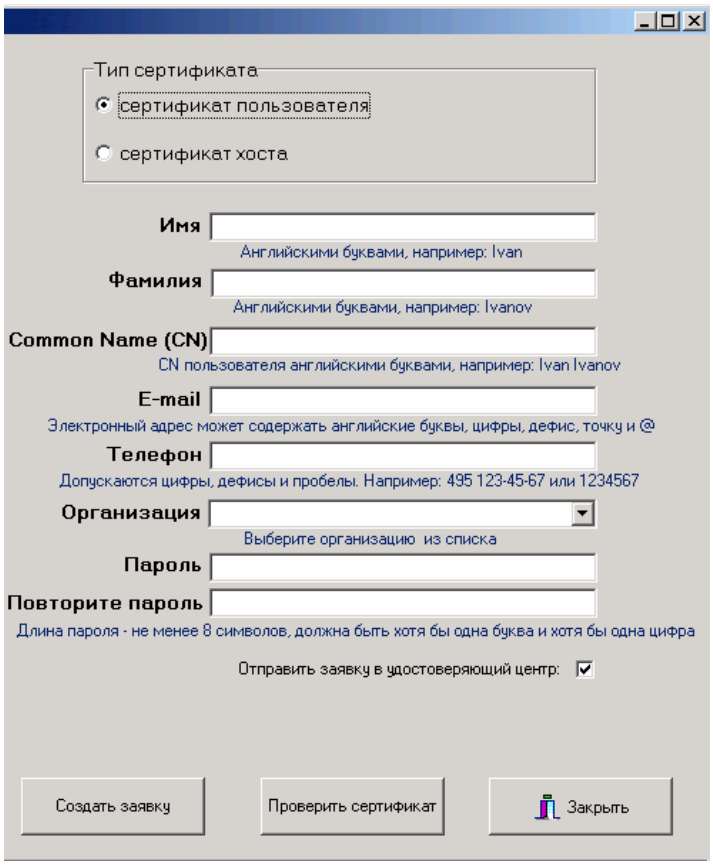


Fig.1: Graphical user interface of certreq

In the interactive mode command line interface has the same functionality as the GUI but is less user-friendly. On the other hand it does not need any GUI support in the OS and could be run on Unix computers without any windows manager installed.

The non-interactive mode could arouse interest among those administrators and advanced users who prefer to write batch scripts for certificates management.

3.4. Command line options

All the command line options could be divided into three blocks according to the appropriate certreq functions: options for request generation, options for certificate retrieving and export/import options. For each block there is at least one main option used to indicate that the appropriate function should be executed.

Option	Description	Comments
Options for request generation		
--genr	Create a new certificate request.	The main option. Required for request generation.
--host	Create a host request.	Optional. By default a user request is created.
--upload	Upload request to the CA.	Optional. By default a request is created, stored on the user's PC but not uploaded to the CA.
--cn=*common_name*	CN of a user or of a host.	Required if "--genr" option is used.
--username=*user_name*	User name of a user or of a host administartor.	Required if "--genr" option is used.
--ou=*organization_unit*	Organization unit.	Required if "--genr" option is used.
--email=*user_email*	E-mail of a user or of a host administartor.	Required if "--genr" option is used.
--phone=*user_phone*	Phone number of a user or of a host administartor.	Required if "--genr" option is used.
--password=*user_password*	Password.	Required if "--genr" option is used.
Options for certificate retrieving		
--checkcert	Retrieve issued certificates from the CA.	The main option. Required for certificate retrieving.
--rewrite	Save downloaded certificate to the default location.	Optional. By default a certificate is saved on the local computer to the temporary location and the existing

		credentials are not overwritten.
--reqname=*file_name*	Check if certificate for the specified request has already been issued and if so download that certificate.	Optional. By default certreq with "--checkcert" option will look for certificates for every request found in the user's home directory.
Export/import options		
--export=*file_name*	Export certificate and private key to the pkcs12 file.	The main option. Required to perform export of the user credentials to the file.
--import=*file_name*	Import certificate and private key from the pkcs12 file.	The main option. Required to perform import of the user credentials from the file.

3.5. Implementation

Command-line part of the certreq tool is written in Python language [7, 8]. Python is a scripting language and some of its advantages go very well with the task concerned. On Unix the tool is distributed as a non-compiled script. Python is installed by default on most Unix systems. For Windows we provide already compiled executable file. So neither Unix nor Windows users have no need to perform any compilation. Moreover even the configuration is done in advance and is hidden from the user.

The base cryptographic functionality is provided by OpenSSL [5, 6] - the open source tool that is generally used to manipulate X.509 certificates as part of Grid certificate management practices. Certreq uses M2Crypto library [9] - a Python interface to OpenSSL's cryptographic functions.

Py2exe [10] was used to prepare certreq.exe file for Windows. The result of the py2exe work is a specially prepared archive that contains the program itself, and the needed parts of the M2Crypto library and the OpenSSL tool. A user will treat this archive as a regular executable file.

The graphical user interface (GUI) was developed using C++.

4. Discussion

Alternative to the proposed solution is a CA with highly functional web-interface. As an example of such a CA, OpenCA [11, 12] project can be considered.

Advantages of the alternative solution:

- Web-interface has a wide functionality to create certificate requests and download issued certificates in the interactive mode.
- The only program a user should install on his/her PC is a web browser.
- Server-side checks of the user data ensure compliance with the certificate policy.

Disadvantages and restrictions of the alternative solution:

- By security reasons there is no possibility to automatically put an issued certificate to the default location on user's PC.
- Some servers do not provide any interface for non-interactive requests. Others does not support working with the command-line interface via network (only from the local computer).
- In GridNNN the solution would imply modification of the existing CA implementation and thus is beyond the scope of this work.

The latest restriction is the main reason why the alternative solution could not be applied in our case as the initial task was to increase comfort and usability of the GridNNN within the bounds of existing infrastructure.

5. Conclusion

Certreq is a standalone tool for management of certificates and certificate requests for GridNNN users that is intended to facilitate management of the certificate requests and certificates:

- After request generation the printing form is prepared and filled while the request is uploaded to the CA.

- Certreq automatically checks all the actual user requests and downloads issued certificates.
- There is no need to install any cryptographic library and/or other programs. Certreq is distributed as an archive that already contains all the needed tools and libraries.

Usability improving is achieved by automating some manual operations and the total system infrastructure is not modified.

This tool was initially developed for GridNNN project. Besides common PKI operations, the tool takes into account some specific for GridNNN features and rules (like default file locations and certification policy). With minor configuration changes it could be used in other GRID projects, e.g. in WLCG/EGEE/RDIG [13, 14]. With a little bit more changes (with switching off all specific to GRID functions) it can be applied to any system where X.509 certificates are used.

References

[1] The GridNNN project on development of grid infrastructure for the National Nanotechnology Network. URL: http://ngrid.ru

[2] Kryukov A., Demichev A., Ilyin V., Shamardin L. Architecture of grid for national nanotechnology network (GridNNN). In Distributed Computing and Grid-Technologies in Science and Education: Proceedings of the 4th Intern. Conf., pp.352-356. Dubna. 2010.

[3] Vacca, Jhn R. Public key infrastructure: building trusted applications and Web services. Taylor & Francis. 2004.

[4] Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List Profile. URL: http://www.ietf.org/rfc/rfc5280.txt

[5] OpenSSL: The Open Source toolkit for SSL/TLS. URL: http://www.openssl.org/

[6] Viega J., Messier M., Chandra P. Network Security with OpenSSL. O'Reilly Media. 2002

[7] Python Programming Language. URL: http://www.python.org/

[8] Lutz M. Programming Python, 4th Edition. O'Reilly Media. 2010.

[9] M2Crypto: a Python interface to OpenSSL's crypto, SSL and S/MIME functionality. URL: http://chandlerproject.org/Projects/MeTooCrypto

[10] Py2exe: a distutils extension to convert Python scripts into executable Windows programs. URL: http://www.py2exe.org/

[11] The OpenCA PKI Research Labs. URL: http://www.openca.org/

[12] Masse N. Open source PKI with OpenCA. Deutsche Telekom, AG. 2006.

[13] Worldwide LHC Computing Grid. URL: http://wlcg.web.cern.ch/

[14] Ilyin V., Korenkov V., Nikonov E., Oleynik D., Maksimiva Yu., Prikhodko A., Semenov R., Isayev R. Web-portal www.egee-rdig.ru: unified information environment of RDIG participants. In Distributed Computing and GRID-technologies in Science and Education: Proceedings of the 1st Intern. Conf., pp.116-123. Dubna. 2004.

GRID AND HPC SUPPORT FOR NATIONAL PARTICIPATION IN LARGE-SCALE COLLABORATIONS[1]

M. Dulea, Ş. Constantinescu, M. Ciubăncan, T. Ivănoaica, C. Plăcintă, I.T. Vasile, D. Ciobanu-Zabet

Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), RO-077125, Măgurele, Romania

The Romanian scientific computing infrastructure is currently evolving from the implementation phase towards increasing the efficiency, the quality and diversity of the services it offers, and adapting to the support of new, multi-disciplinary applications. Most of the grid and HPC activity is related to the topics investigated by research consortia engaged in European and international collaborations. In this framework, the department of Computational Physics and Information Technologies (DFCTI) at IFIN-HH provides resources, monitoring services and coordination for the distributed computing system that sustains these cooperative activities.

This article presents recent advances in improving the grid and HPC services that DFCTI delivers to the national and international scientific community, in connection with the projects it takes part in.

Relevant examples are drawn from the contribution to the WLCG collaboration and the related bilateral cooperation with IRFU/CEA [1] in the framework of the computational support for the LHC experiments, or the participation in the implementation of the *High-Performance Computing Infrastructure for South East Europe's Research Communities* (HP-SEE project [2]).

Important steps were made towards developing the computing environment for data analysis in high-energy physics and biophysical simulations. Scalable solutions were designed and implemented for improving the transfer, processing and storage of large datasets at the Tier2 centers that participate in the computing support of the ATLAS experiment at LHC-CERN.

Also, an integrated system for modeling, production runs, and data analysis of complex biomolecules was developed for the molecular dynamics simulations performed in the framework of the HP-SEE infrastructure.

A significant contribution to the performance management of the grid system came from the NGI-independent set of tools implemented for monitoring the data transfer and storage efficacy in the resource centres.

Finally, the strategy for adapting the infrastructure of the National Grid for Physics and Related Areas - GriNFiC [3] to the requirements of new scientific collaborations is shortly reviewed.

1. Introduction

During more than two years since the end of the EGEE projects [4] and the foundation of the EGI [5], the national computing infrastructure for science has continuously evolved in terms of resources and service quality, towards supporting an increasing number of research communities and the reaching of its sustainability. Currently, the grid infrastructure contributes to the computational support of several international experiments in high-energy physics, regional collaborations, and national R&D projects. Also, many research activities developed in major institutes and universities are sustained by medium-size HPC centres, some of them being organized in a consortium that takes

[1] This work was partly funded by ANCS under the contracts no. 8/2012– PNII-Capacities-M3-CERN, C1-06/2010-IFA, and from the Hulubei -Meshcheryakov collaboration, JINR Order 82/06.02.2012.

part in the implementation of the network of advanced computing resources for the research communities in the region of South Eastern Europe (SEE) [2].

After shortly presenting an overview of the national infrastructure for scientific computing and of the partnerships it supports, this paper focuses on the grid and HPC activities carried out within DFCTI/IFIN-HH for ensuring the operation and monitoring of its resource centres, together with the contribution it provides to the coordination and development of consortia that participate in regional and international collaborations.

2. National infrastructure for advanced scientific computing

The grid infrastructure was built between 2004-2009 through various independent projects funded by the National Authority for Scientific Research (ANCS) [6], and with support from the EGEE [4], and SEE-GRID [7] FP6 projects. It was subsequently upgraded between 2010-2011 by means of the EU structural funds under the Sectoral Operational Programme for Increasing the Economic Competitiveness, which is co-funded by the European Fund for Regional Development.

The infrastructure for grid production currently counts 10 certified and active sites, which are connected to the 10 Gbps backbone of the national research and education network – RoEduNet [7] and provide more than 6200 cores and a total disk capacity of 1.8 Petabytes to various research communities in physics, environment and earth sciences. All the production sites use gLite 3.2 or EMI middleware and can be monitored centrally by EGI. Two sites involved in the ALICE collaboration use the AliEn middleware [8] as well. Besides the production grid, testbed prototypes using different middleware were developed, such as the Globus-based system for real-time management of distributed databases presented in [9].

Due to the important role played by the community of high-energy physicists since the early days of the implementation of the national grid, most of the activity is dedicated to the international collaborations in elementary particle physics that are related to the LHC – CERN experiments [10]. The virtual organizations (VOs), the collaborations they support, the number of supporting sites, and the percentage of the total grid activity realized between 07.2011-06.2012, as recorded by the EGI portal [11], are listed in Table 1 below:

Table 1: Virtual organizations actively supported by the national grid infrastructure

VO	Collaboration supported	No. of sites	% of total
alice	ALICE experiment [12] - LHC	3	54.94%
atlas	ATLAS experiment [13] - LHC	4	39.64%
lhcb	LHCb experiment [14] - LHC	3	3.87%
envirogrids.vo.eu-egee.org	FP7 enviroGRIDS project [15]	1	0.55%
gridifin	RO physics & related areas [3]	1	0.37%
see	Multidisciplinary, regional (six countries from SEE) [16]	2	0.26%
seegrid	SEE-GRID project community	1	0.19%
hone	H1 experiment [17] - DESY	1	0.17%

During the 12-month interval specified above the production centres have run more than 117 million HEPSpec06-hours [18] for processing more than 15 million jobs.

The second component of the Romanian advanced computing infrastructure consists of medium-size HPC centres which are distributed in the main academic and research institutions and connected to RoEduNet. Many of these centres, which were developed through funding from national and/or European sources, are located in the same data centres as the grid sites and are operated by the same technical staffs.

The main contribution to the HPC infrastructure comes from 10 parallel computing systems (one BlueGene/P supercomputer at the Western University of Timisoara (UVT) [19] and 9 medium-size clusters), which are hosted by three R&D institutes and five universities. In total, more than 8,000 cores and approx. 50 Tflops Rpeak are made available in these centers for specific applications of the scientific computing in various research fields, such as Computational Physics, Computational Chemistry, Astronomy & Astrophysics, Life Sciences, Meteorology and Environmental Sciences.

3. Computational support for research in DFCTI

DFCTI represents a key component of the research infrastructure of IFIN-HH, providing computing resources and services for the scientific research in the fields of numerical modeling and simulation of physical phenomena. Its mission of is twofold: a) to manage, operate and develop the computing infrastructure of the institute, providing reliable technical support for national and international scientific collaborations; b) to perform scientific R&D in areas that require the application of numerical methods of investigation.

The department conducts studies on modeling and simulations of complex systems, algorithms development, computing optimization, and the investigation of topics of current interest in condensed matter physics, computational biology, and subnuclear physics. Due to the interdisciplinary character of its research activities, the team involves specialists with different professional backgrounds, such as physicists, computer scientists, engineers in various specialties, and programmers. The team participates in various national and European R&D and infrastructure projects on Grid and HPC, and collaborates with research groups from EU, USA, and JINR-Dubna.

DFCTI hosts, develops and operates one of the most important IT infrastructures in the country, which is dedicated to the support of the scientific research and of the large scale international collaborations in which the institute takes part. Its technical staff administrates two EGI-registered grid sites (RO-07-NIPNE – that provides resources to the `alice`, `atlas`, and `lhcb` VOs, and RO-11-NIPNE – which serves the LHCb experimental group), the main site of the National Grid for Physics and Related Areas (GriNFiC) – GRIDIFIN [3], and four HPC clusters. All these resources are hosted in two main and one secondary data centers, as depicted in Figs. 1 and 2 below.

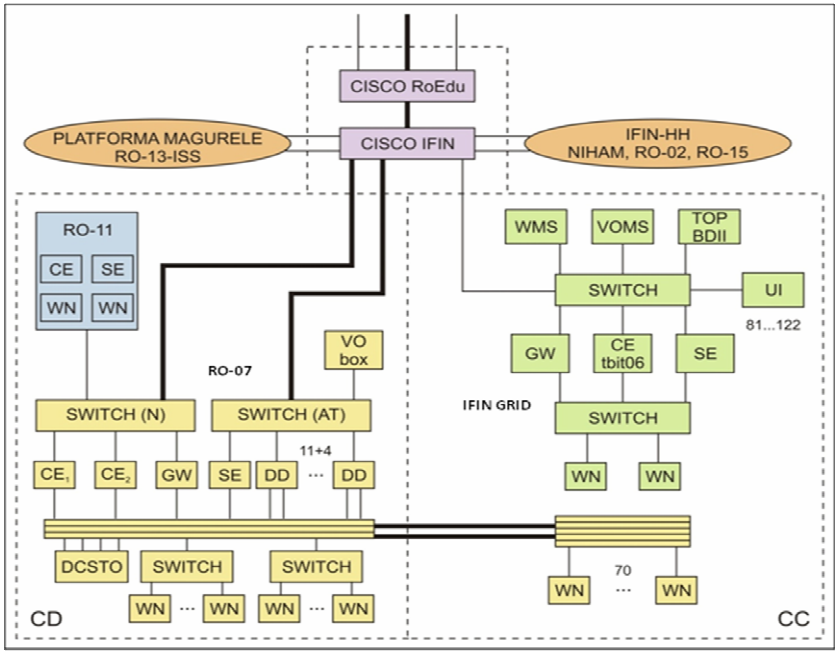


Fig. 1: Topology of the grid clusters hosted at DFCTI

The grid sites are connected to RoEduNet through a dark fiber link currently providing a bandwidth of 10 Gbps, and two backup connections of 1 Gbps each. The workernodes of the main site, RO-07-NIPNE, are distributed in two locations, being connected to two separate switch stacks that intercommunicate through 2x10 Gbps links. The site currently contributes to the WLCG collaboration with more than 1450 processing cores and 400 TB storage capacity. It uses gLite 3.2 middleware, two CREAM-CE Computing Element servers (of which one is dedicated to `alice`), PBS/TORQUE queuing system with MAUI job scheduler, and a DPM/SRM Storage Element (SE) with 15 DPM disk servers.

The development of the HPC infrastructure started in 2006 and benefited of the cooperation with LIT-JINR within the Hulubei-Meshcheriakov programme (2005-2013), project *Optimization Investigations of the GRID and Parallel Computing Facilities at LIT-JINR and Magurele Campus* [20]. Today this infrastructure encompasses two major clusters, IFIN_BC and IFIN_Bio, in Infiniband and Myrinet 2G technology, respectively, and two smaller test clusters, Myr (Myrinet 2G) and Teo (Gigabit Ethernet), which are connected to the grid network as depicted in Fig. 2 below.

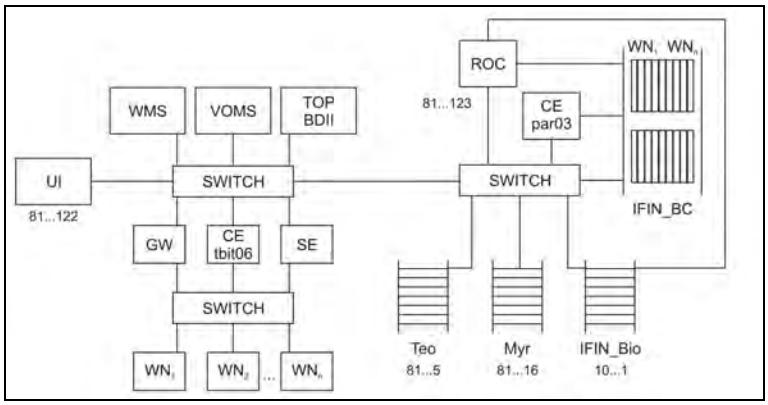


Fig. 2: Schematic representation of GRIDIFIN and the four parallel computing clusters

The main parallel cluster, IFIN_BC, is a heterogeneous IBM Blade Center system dedicated to molecular dynamics simulations and testing of parallel software applications within the computational biology collaborations. The system supports a few MPI implementations (MVAPICH, MVAPICH2, OpenMPI), and is endowed with specific software tools, such as CHARMM [21], NAMD [22], Gaussian [23], VMD [24], etc. Its hardware configuration is described in Fig. 3.

Server type:	IBM QS22	IBM LS22	IBM HS22	Total
CPU	IBM PowerXCell 8i	AMD Opteron 2376	Intel Xeon X5650	
Clock frequency	3,2 GHz	2,3 GHz	2,67 GHz	
Core no. / cpu	1x PPE + 8x SPE	4	6	
Cache level2/cpu	512 KB	512 KB	6x256 KB	
FSB frequency	1066	1000	3200 MHz	
HDD / node	8 GB SSD	76 / 146 GB SAS	500 GB SAS	
RAM / node	32 GB	8 GB	24 / 36 / 48	
Total RAM	512 GB	80 GB	21x24+23x36+12x48 GB	1908 GB
Nodes total	16	10	56	82
CPUs total	32	20	112	164
Cores total	32x PPE + 256x SPE	80	672	1040
Interconnects	Infiniband 4x QDR 40 Gbps			

Fig. 3: Structure of the IFIN_BC cluster

The infrastructure of advanced computing is used by the DFCTI researchers for:

- Simulation and modeling of large biomolecular systems (like, for instance, G Protein-Coupled Receptors) by means of molecular dynamics codes (e.g. NAMD), in collaboration with the Faculty of Biology at the University of Bucharest;

- Modeling drug - efflux pump inhibitors interaction dynamics, in the framework of the project *Activity modeling and simulation of efflux pump inhibitors based on advanced laser methods* (collaboration with the National Institute for Laser, Plasma and Radiation Physics);
- Studies on the dynamics of Bose-Einstein condensates, in collaboration with the Institute of Physics Belgrade (IPB);
- Ab-initio investigation of charge transport in nanostructures, in collaboration with the Physics Faculty at the University of Bucharest.

4. Participation in large-scale collaborations

DFCTI currently coordinates two national consortia:

- the *Romanian Tier-2 Federation RO-LCG* - consortium that participates, since 2006, to the WLCG collaboration with CERN, and
- the *Joint Research Unit for High Performance Computing and Supercomputing* – a consortium of four institutions which was founded in 2009 in preparation of the HP-SEE partnership.

IFIN-HH and three of its RO-LCG partners, the Institute of Space Science (ISS), the National Institute for R&D in Isotopic and Molecular Technologies from Cluj-Napoca, and the 'Alexandru Ioan Cuza' University of Iasi, provide more than 4,800 cores and 1.8 PetaBytes storage capacity for thousands of particle physicists within the ALICE, ATLAS and LHCb collaborations. Since 2010, more than 24 million LCG jobs were processed within RO-LCG, running 200 million HEPSPEC06-hours, a result that ranks it in 10th place among the 33 national Tier-2 contributions to WLCG.

DFCTI and its partners in the HPC consortium (UVT, ISS, and 'Politehnica' University of Bucharest) have concluded a memorandum of understanding regarding the development of the national infrastructure for HPC and supercomputing, on the basis of which they participate to the HP-SEE [2] project (FP7-RI-261499, 2010-2013).

The goal of the project, which is coordinated by GRNET – Greece, is the implementation of a common, integrated HPC infrastructure for 13 countries that lie in a wide region between Hungary and Caucasus. Its main objectives are:

- Linking the existing and upcoming HPC facilities of the partners in the integrated infrastructure, and providing operational solutions for it
- Opening the HPC infrastructure to a wide range of user communities in the region, providing advanced capabilities to researchers, with an emphasis on strategic groups in Computational Physics, Computational Chemistry and Life Sciences
- Ensuring that all participating countries in the region have access to latest HPC facilities in Europe (including PRACE [25]), if necessary
- Ensuring the long-term sustainability of the infrastructure.

At present, there are 11 HPC centres integrated in the regional infrastructure. They provide resources for 26 software applications that are supported within the project and run in the HP-SEE infrastructure, which was recently open to access by new research communities through a pilot call. Grid middleware is used in in order to hide the various ways of accessing HPC sites provided by the local resource management systems.

Within HP-SEE, DFCTI coordinates the *Computational Physics Virtual Research Community* and the *Virtual Research Communities support* activity. Also, the department contributes with two parallel clusters (IFIN_BC and IFIN_Bio) to the regional HPC infrastructure, and participates to the software development in the field of the simulation and modeling of large biomolecular systems.

5. Grid monitoring in RO-LCG

Within the RO-LCG consortium, the monitoring is provided at 4 levels: 1) utility and support equipment (electric power, UPSes, cooling, etc); 2) data traffic on main switches/routers; 3) service availability; 4) grid activity: jobs, consumed cpu time, data traffic on the main servers of the grid

cluster. Various proprietary, open-source, and/or in-house developed software solutions are used in the RO-LCG centres for performing the local monitoring at the levels 1, 2 and 4 above. Most of the tools can alert the technical staff in case of malfunction, for fast intervention.

In the case of RO-07-NIPNE, the monitoring and accounting of the grid activity is presented on a common web interface (http://www.nipne.ro/RO-07-NIPNE.html), which displays the number of jobs running or pending for different VOs and job types (production, analysis, etc.), the accounted grid production on each job type, and the traffic recorded on the interfaces of the main servers (storage element, grid gateway, etc.).

In order to increase the availability of the of the grid services provided by RO-LCG, a more reliable availability monitoring service than the one provided by the Romanian NGI became necessary. The service availability monitoring (SAM) system was duplicated at the GriNFiC's scale [3]. Within the new system SAM tests are provided by GRIDIFIN through the `ifops` VO (which is an equivalent of the EGI's `ops` VO, adapted to the monitoring of all the GriNFiC's sites, including those of RO-LCG). The results of the tests are transferred to WMS and published by Nagios.

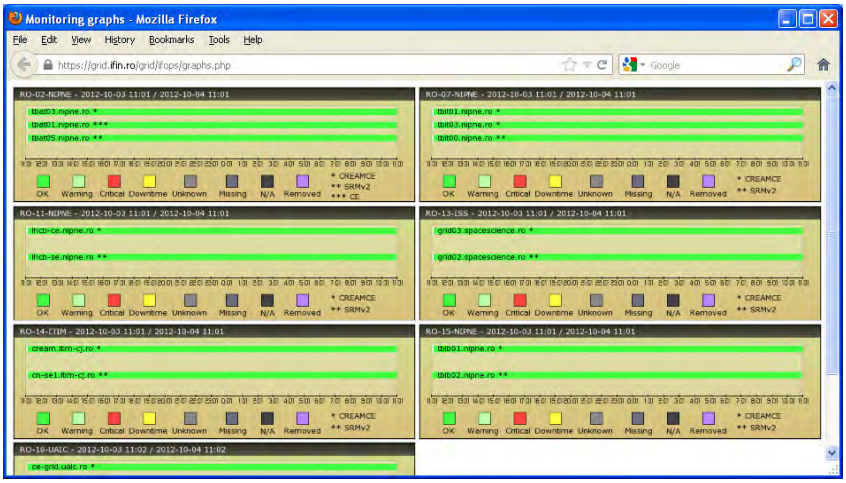


Fig. 4: RO-LCG monitoring desk

Web interfaces for individual and global display of the results of the SAM tests for RO-LCG sites were developed. The individual web interface allows to track of the availability history of a given site for arbitrary periods of time, taking the results from a database which is fed from the Nagios' log. The global display represents a monitoring desk of the status of the services provided by all the RO-LCG sites during the last 24 hours, and is updated every 5 minutes (Fig. 4).

The ifops – SAM system allows performing the tests frequently enough to take corrective measures in good time in case of incidents. Also, a better understanding of the reasons and magnitude of some grid incidents is achieved by comparing the results of the ifops tests with those of the ops tests performed by NGI and published on the EGI monitoring portal.

6. Large data management in RO-LCG

The grid sites that run atlas analysis jobs can experience specific problems during the concurrent transfers of large files in the external or internal network. The external connectivity can be affected by the transfers of input files for data analysis, the sending of large job logs and result files from SE to the users, and the sending of the simulation results from SE to the Tier-1 centres. Also, the bandwidth of the internal network is considerably consumed during the transfer of large input files for analysis from SE to the workernodes. In a fixed bandwidth configuration these problems are expected to grow with the size of the site.

Solutions for improving the efficiency of the analysis sites were proposed within the bilateral collaboration with IRFU/CEA on *Efficient Handling and Processing Petabite Scale Data for the Computing Centres within the French Cloud* (HaPPSDaG project) [1].

The running of an increasing number of atlas analysis jobs required a scalable configuration for handling the concurrent data transfers between the SE and the workernodes. A financially convenient solution was the stacking of switches, as shown in Fig. 1, which can provide the required minimum bandwidth when the number of simultaneous file transfers grows. This allows to preserve the scalability of the cluster, by increasing the bandwidth available for data transfer at a constant rate whenever the storage capacity is upgraded. Moreover, the method offers advantages over a fixed large-bandwidth network (e.g. 10 Gbps), because it allows to exceed this threshold without significant expenses.

7. Towards an integrated HPC infrastructure in SEE

One of the important objectives of the HP-SEE project is to make the multiple HPC resources in the region accessible to the scientific communities in a convenient way for the users.

The application ISyMAB [26], developed at DFCTI, achieves this goal in what regards the modeling and data analysis of complex biomolecules. The application provides a remote access framework on molecular dynamics clusters which offers the users an integrated interface with analysis tools.

The user with access rights can launch jobs through PBS and execute shell scripts on various parallel computing systems in the HP-SEE infrastructure.

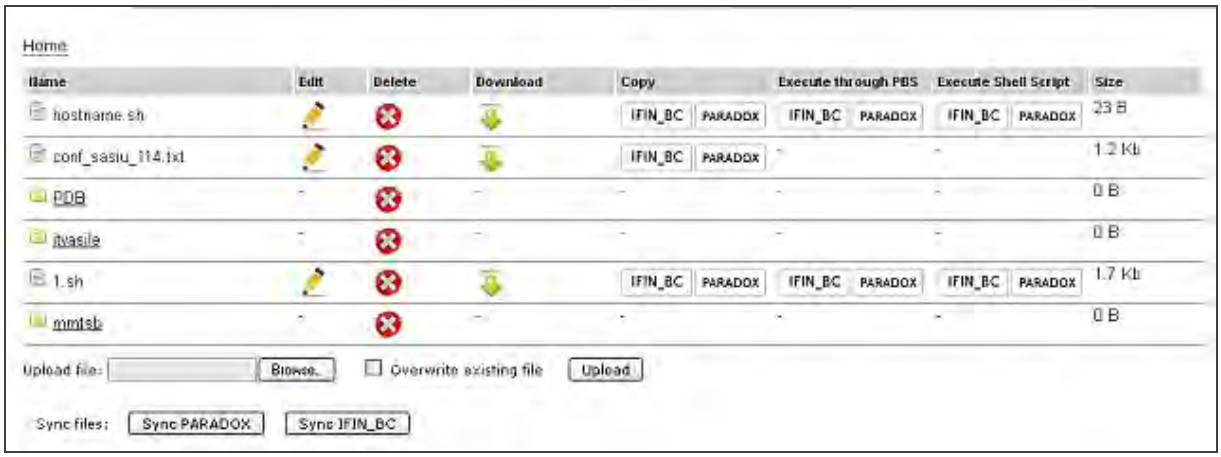


Fig. 6: ISyMAB user interface offering the choice of working on IFIN_BC or PARADOX cluster

Fig. 6 above presents the interface through which the user can launch jobs on two different systems, IFIN_BC and the PARADOX cluster at the Institute for Physics in Belgrade. The working directory of the user on any cluster can be synchronized with the ISyMAB directory using the Sync buttons.

Facilitating the communication between the user and the remote computing system, ISyMAB is particularly useful for the repeated checks of the setup process (e.g. with CHARMM), the physical interpretation of the results (e.g. with VMD) at the end of the heating phase (simulated e.g. with NAMD), and the repeated checks of the system stability in the production stage.

8. Prospects

The long-term grid strategy will be focused on increasing the efficiency of the management and operation at the central and site level. The time of adoption of new procedures and new computer architectures will be considerable shortened, and redundant activities will be avoided, preserving the

costs and manpower within reasonable limits. The monitoring and accounting system provided by GRIDIFIN will further be maintained and developed, ensuring the independence on the funding from external Grid projects.

A greater resilience in solving the RO-LCG and GriNFiC problems is expected to be obtained through the adoption of virtualization in the management of the Grid sites and the access to new interfaces towards public and/or private clouds. The technical staff will closely follow the development and implementation of the Helix Nebula initiative [27], due to the potential it has for the particle physics community (ATLAS - distributed analysis), and also for other collaborations that are of interest for RO-LCG partners, such as ESA (ISS), and ELIXIR [28] (to which Romania intends to participate with a consortium that includes IFIN-HH).

Providing HPC support for large scale, long-term international collaborations to which Romania will participate, such as ELI-NP [29], ITER/EURATOM [30], and FAIR-GSI [31], will be a priority, together with the connection to the European partnerships regarding the high-performance scientific computing infrastructure, such as PRACE.

9. Conclusions

As coordinator of the RO-LCG Federation, of the national consortium that participates in the South Eastern European HPC collaboration (HP-SEE), and of the National Grid for Physics Research and Related Fields (GriNFiC), DFCTI significantly contributes to the support of the computational science at national and international levels.

Running long-term infrastructure projects with a considerable number of partners necessarily requires appropriate technical measures for handling the ever increasing resource needs of the user communities, while preserving the high quality of the provided services. These problems can successfully be solved through the implementation of scalable hardware solutions and of software tools for increasing the reliability and decreasing the incident response time.

The positive results obtained in this respect at DFCTI and the experience acquired during the last few years allow the successful continuation of the existing long-term projects and the participation to new large scale collaborations in computational science.

References

[1] Efficient Handling and Processing of PetaByte-Scale Data for the Grid Centers within the FR Cloud, http://happsdag.nipne.ro/

[2] The HP-SEE project, http://www.hp-see.eu/

[3] T. Ivănoaica, M. Ciubăncan, Ş. Constantinescu, M. Dulea: *GriNFiC - Romanian Computing Grid for Physics and Related Areas*, in Proceedings of the XXIII International Symposium Nuclear Electronics & Computing (NEC'2011), Varna, Bulgaria, Sept. 12-19, 2011, E 10,11-2011-133, JINR, Dubna, 2011, pp. 163-168.

[4] Enabling Grids for E-science, http://www.eu-egee.org/

[5] European Grid Infrastructure, http://www.egi.eu/

[6] National Authority for Scientific Research, http://www.ancs.ro

[7] South Eastern European Grid-enabled eInfrastructure Development, http://www.see-grid.org/

[7] Romanian Research and Education Network - RoEduNet, http://www.roedu.net/en/node/43

[8] AliEn, http://alien2.cern.ch/

[9] C. Placintă, I. Vasile, M. Dulea: *Grid system for rel-time management of distributed databases, with applications to the coordination of complex projects*, in Proceedings of the 4th International Conference "Distributed Computing and Grid Technologies in Science and Education (GRID'2010)", JINR, Dubna, Russia, 28 June – 03 July 2010, pp. 183-190, ISBN 978-5-9530-0269-1.

[10] The Large Hadron Collider – CERN, http://lhc.web.cern.ch/lhc/

[11] The European Grid Initiative accounting portal, http://accounting.egi.eu/egi.php

[12] ALICE experiment, http://aliceinfo.cern.ch
[13] ATLAS experiment, http://atlas.ch/
[14] LHCb experiment, http://lhcb-public.web.cern.ch
[15] The enviroGRIDS project, http://www.envirogrids.net/
[16] http://operations-portal.egi.eu/vo
[17] The H1 experiment, http://h1.desy.de/
[18] Worldwide LHC Computing Grid collaboration,
[19] BlueGene/P @ UVT, http://hpc.uvt.ro/infrastructure/bluegenep/
[20] Gh. Adam, S. Adam, A. Ayriyan, V. Korenkov, V. Mitsyn, M. Dulea, I. Vasile, *Consistent performance assessment of multicore multiprocessor computer system architectures*, Third National Conference on Theoretical Physics, Busteni, 10–13 June 2008, Romanian J. Phys. 53, No.9-10, 985-991 (2008); Gh. Adam, S. Adam, A. Ayriyan, E. Dushanov, E. Hayryan, V. Korenkov, A. Lutsenko, V. Mitsyn, T. Sapozhnikova, A. Sapozhnikov, O. Streltsova, F. Buzatu, M. Dulea, I. Vasile, A. Sima, C. Visan, J. Busa, I. Pokorny, *Performance assessment of the SIMFAP parallel cluster at IFIN-HH Bucharest*, Romanian J. Phys. 53, No.5-6, 665-677 (2008).
[21] CHARMM, http://www.charmm.org/
[22] NAMD, http://www.ks.uiuc.edu/Research/namd/
[23] Gaussian, http://www.gaussian.com/
[24] VMD, http://www.ks.uiuc.edu/Research/vmd/
[25] Partnership for Advanced Computing in Europe (PRACE), http://www.prace-project.eu/
[26] Integrated System for Modeling and data Analysis of complex Biomolecules – ISyMAB, http://wiki.hp-see.eu/index.php/ISyMAB
[27] Helix Nebula, the Science Cloud, http://helix-nebula.eu/
[28] European Life Sciences Infrastructure for Biological Information, http://www.elixir-europe.org/
[29] Extreme Light Infrastructure – Nuclear Physics, http://www.eli-np.ro
[30] International Thermonuclear Experimental Reactor (ITER/EURATOM), http://www.iter.org/
[31] Facility for Antiproton and Ion Research (FAIR-GSI), http://www.gsi.de/

CONFIGURATION MANAGEMENT FOR IT INFRASTRUCTURE

O. Dulov
Steinbuch Computing Center (SCC)
Karlsruhe Institute of Technology (KIT)
Karlsruhe, Germany
oleg.dulov@kit.edu

Every IT system should be configurable. Working station, personal notebook, high performance, grid, cloud or other cluster need to have state which was design for this system. There are different software products to provide a possibility to construct the IT infrastructure, update and change their state. CFEngine, Puppet, Chef, and some other open source projects are available for usage. Special interest for IT system configuration platforms is coming when organization tries to realize ITIL (Information Technology Infrastructure Library) recommendations. Automatisms, system administration, policy-based management, flexibility and scalability – these terms are on top of the idea configuration management for IT systems. At the same time, there are some additional efforts for the system administration to switch from direct management of their systems into indirect management by using configuration management tools. Open source project cfeditor has the idea to provide IDE for the system programming based on the configuration management tools. A current version includes syntax for the CFEngine version 3, but other platforms are also in plan.

Motivation

This paper is devoted to software configuration, not to hardware. Before speaking about configuration, let's refresh activities to create software product and try to analyze what can be improved. Software development cycle can be seen as three groups of activities:

1. Development
2. Maintenance
3. Operations

A first group is about to create a software product according to defined release cycle procedures. These activities are area of responsibility system or application programmers and normally do not associated within system administration.

A second group is about how to bring created software into the working environment. This is clear one from the system administration activities. It has connections with software development release cycle and includes modifications (changes) to correct software system attributes. Some of the attributes are: faults, reliability, availability or downtime for software system.

The way software is operated varies from system to system. However there are two activities that occur during most software operations: user support and reporting problems. Where under user can be end user or operator and may require direct assistance from experts in the development or maintenance teams, or by help desks. Users should document problems in for of Software Problem Report.

Some of the software maintenance and operations can be done without direct influence from the system administration team, by using other software tools. One group called configuration management tools, provides platform to manage IT systems for system administration team.

Normally, system administrators (sysadmins) are doing scripting, and from such a perspective can be considered as a system programmer. Tools to help organize working process for software developers can be also included into the arsenal of system administration.

Policy-based management

There are several different types or styles of how the systems can be managed. One is called a policy-based management. This is an administrative approach that is used to simplify the management of a given endeavor by establishing policies to deal with situations that are likely to occur.

Policies are operating rules governing the choices in behavior of a system. That can be referred to as a way to maintain order, security, consistency, or otherwise reach goals or mission. For example:

- security policy can describe the access rules for users into the system;
- Quality of Service (QoS) management policy declare the different priorities inside job queue, some business rules can define the discount, according to the order;
- Service Level Agreement (SLA) declare which availability level should be provided and what will be if not.

There are some software packages available to automate elements of policy-based management. In general, the way these works is follows:

- Business policies are input to the system which are mapped into the tool syntax, which will manage configurations;
- Software communicates to network hardware how to support those policies;
- Monitoring is getting the data about system state;
- Policies are set and all changes will get the same data flow: from mapping business rules to syntax till maintenance into the system.

Direct vs. indirect control

Let's consider how the system administration work is organized, by grouping according to controlling procedure: direct and indirect system administrator access into the system.

Direct access is still the most often used way how to implement changes within the system and includes the following list of system administrator activities:

1. Log in into the system.
2. Change a state or configuration by using operating system or application specific commands.
3. Log out from the system.

There are some potential problems with such a way of doing things. First of all, the sysadmin must log into the system and get stable connection between the system and his communication client. Sometimes communication channel will be not secure or does not provide enough quality of service. If service is into distributed environment, it can be necessary to login into different parts of the system.

Making changes "by hand" may cause other problems, like: cannot undo changed configuration, not documented change, change in the parallel with someone from administration team, and so on. Scripts for the system normally can be located anywhere into operating system or even under different hardware and can have unstructured locations.

At the same time, sysadmin is communicating with the system by using supporting software (for example Linux shell and ssh clients/server) and why do not use the software, which will do what we want to have, without knowing different "how to" for such a specific operating system? System administration team can use the common Domain Specific Language (DSL) of such a platform and provide directives to implement.

Additional abstraction layer between operating system commands, editing tools and system administration team can implement policy-based management within the system, independent of the operating system itself. If so, the system administrator is describing policy, based on the DSL for configuration management platform. Such a platforms called configuration management software, where under configuration management assumes activities (planning, organizing, leading, controlling, and so on) to bring the system into defined state.

Configuration is a complete description of the current situation (of the current general state) of our machine. System configuration defines system state in time. Movement from one configuration to another determined by the execution of an actual instruction (elementary operation).

Configuration management platform implements policy, care of the system state and is responsible for:

- install, deploy, maintain system, security features management;
- procedures for handling of all changes to control a system:
  - software and documentation;
  - firmware and documentation;
  - test and documentation;
- status monitoring, reporting;
- build, process, environment management.

Systems themselves do not stay along, but they are connected into the different software repositories, monitoring tools, and some other systems (which can be called support for our target system). Hence, we can consider the following control flow of activities for indirect system control (see fig. 1).

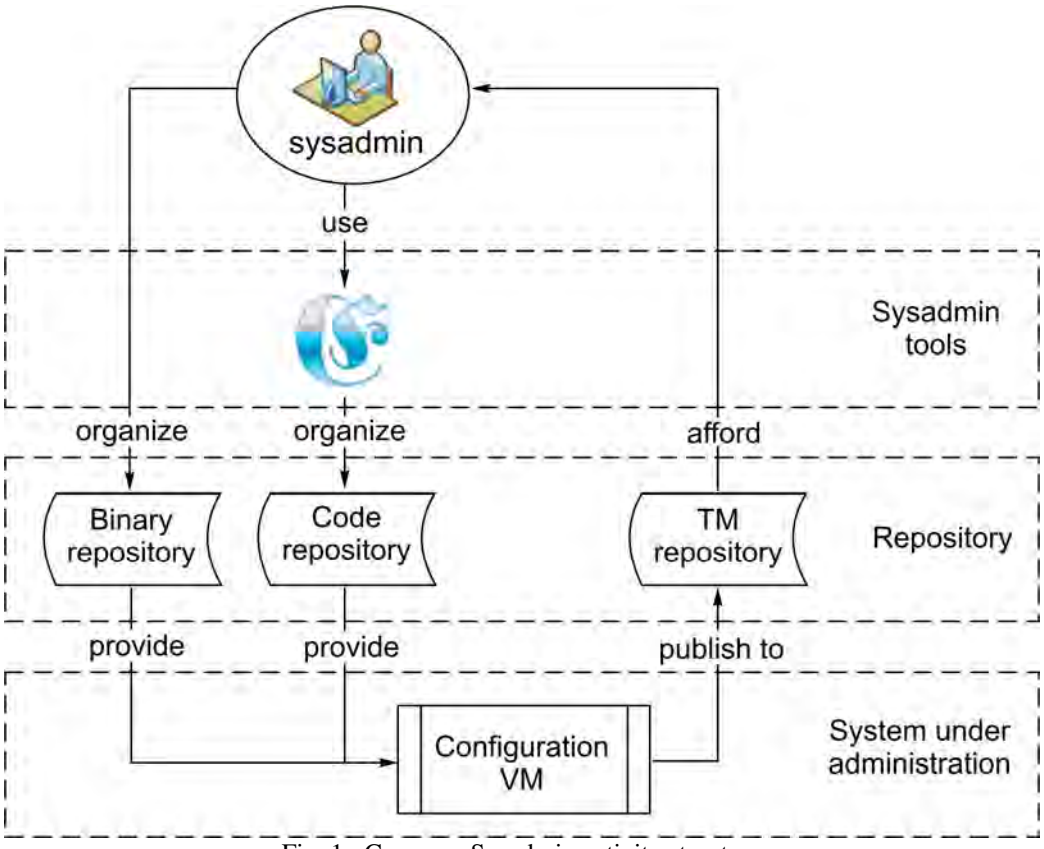


Fig. 1: Common Sysadmin activity structure
(http://cfeditor.blogspot.com/2011/01/common-sysadmin-activity-structure.html)

Team of system administrators develops scripts and brings them into the software code repository by using Integrated Development Environment (IDE) or editor with code repository client. We can speak about two types of scripts: configuration management specific (scripts on platform DSL) and other language scripts (e.g. bash, python, java and others). Additionally, code repository can consist of the configuration files for applications or other repositories and so on.

Binary repository stores software packages which are ready to be maintained into system. In terms of IT Infrastructure Library (ITIL) terminology this can be called Definitive Media Library (DML).

Code and binary repositories data can be seen as input into the configuration management platform (or virtual machine) to bring target system into defined in policy state.

In order to get information (or knowledge) about target system, configuration management platform can monitor and publish information about the system state. There is a standard for knowledge interchange, called Topic Map, let call the repository Topic Map (TM) repository. This repository can be seen as feedback into control circle.

Hence, what we get is the simple structure:

- the target system, which is represented by a configuration management platform;
- three different types of repositories: code, binary and Topic Map or knowledge;
- system administration team to control the target system by manipulating code and binary repositories, and is informed by the system knowledge repository.

Such a structure for organizing the target system represents the simple control loop and can be extended by the sysadmin specific tools, which we put into the layer "sysadmin tools". One tool, called cfeditor we developed to support our system administration needs and shortly will present later in this article.

Software configuration management tools

There are some goals for software configuration management, which should be supported by software configuration tools. Some of such goals are: environment management, configuration control, identification, accounting of status, auditing, teamwork, build management and some others.

Environment management is about to manage the software and hardware that host the system, consider IT system together with other supported systems. Identifying configurations, configuration items and baselines is about identification. By implementing a controlled change process, software configuration control can be achieved. Recording and reporting all the valuable information on the status of the development process to account status.

Managing in correct way different types of documentation, specifications can be used to provide the audit for configurations. Managing the configuration process and tools used for builds is call to provide build management for the system.

Currently, there are many efforts to implement software configuration management tools, but some tools well known during the last years. The first software package to implement software management is called CFEngine. We can be not really wrong, saying that all other products try implementing the main CFEngine ideas in different matters.

CFEngine

The primary function for CFEngine is to provide automated configuration and maintenance of large-scale computer systems, including the unified management of servers, desktops, embedded networked devices, mobile smartphones, and tablet computers.

From 1993 CFEngine was implemented as a set of scripts to support some of software building processes, control for the maintenance. After a couple of years, the next version CFEngine 2 gets improved possibility to manage configurations, based on the policies descriptions.

The original idea was changed after the creation of the so called promise theory. This theory is about model of distributed cooperation for self-healing automation. Consider different aspects of voluntary communication for autonomous actors or agents who publish their intentions to one another in the form of promises.

CFEngine version 3 constructed, based on the promise theory and includes implementation for some of theoretically constructed communicational and self-repairing models. For a couple of years, CFEngine 3 also integrated knowledge management and discovery mechanisms—allowing configuration management to scale to automate enterprise-class infrastructure.

In comparison to CFEngine 2, the new version is not only open source community product, but it also has a commercial version with an extended functionality to manage systems in different scale.

Cfeditor

The ideas of how to manage IT systems indirectly and avoid problems of direct control, as well as how to organize the sysadmin teams, pushed us to create editor for configuration management platform which we are using (cfengine) with integrated support of code repository clients.

Cfeditor (configuration editor) is one of the open source sysadmin tools, which is developed by the Karlsruhe Institute of Technology (fig.2). It is Eclipse-based plugin to support configuration management platform (currently for cfengine version 3). There are some basement technologies for cfeditor: Xtext - language framework to model Domain Specific Languages, Google Guice and Eclipse Modelling Framework EMF.

The current idea is to implement support for DSL CFEngine language, but the functionality can be extended to other configuration management platforms. Even IT service description can be done, independently from the DSL languages, based on the powerful Eclipse functionality.

The Eclipse platform has rich functionality, including connectors to different code repositories and can be easily used as a tool for common work by not only one administrator, but administration team. Working with graphical objects and mapping between text and graph inside Eclipse can provide the extended functionality for system administration needs in the future.

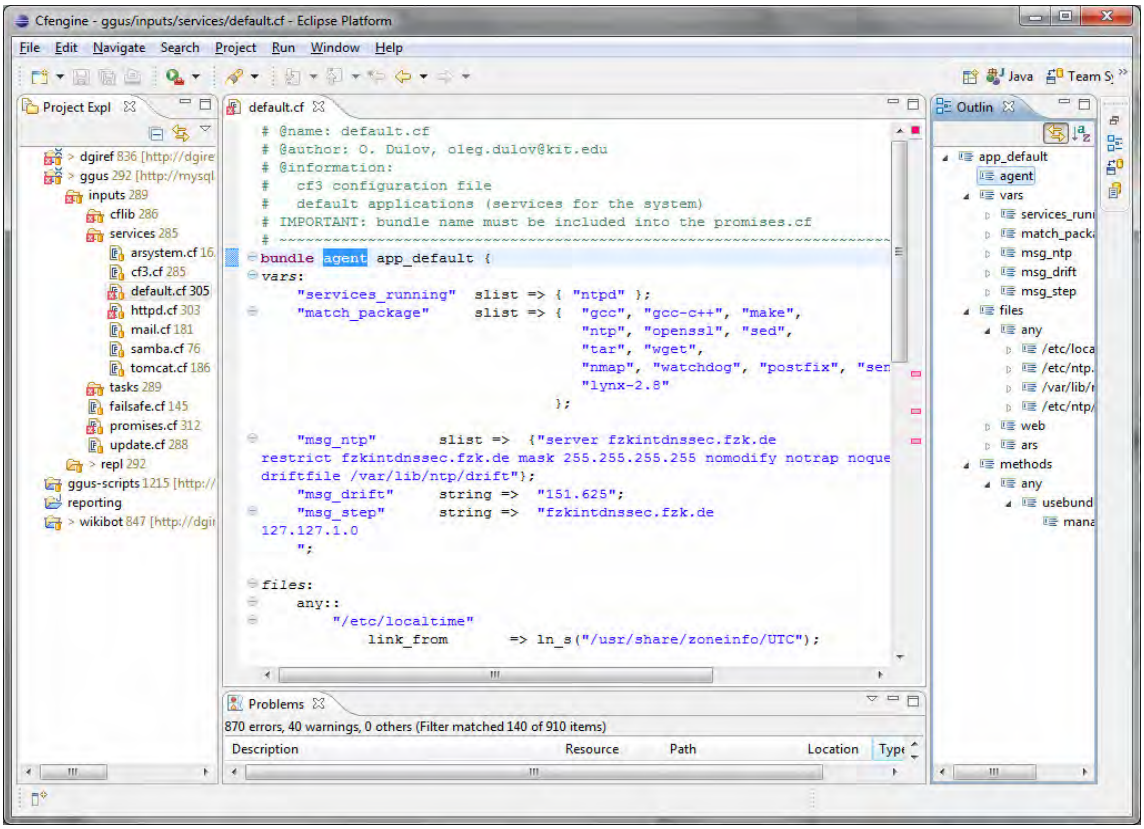


Fig.2: Example of cfeditor project view

Eclipse perspective for cfeditor is typical for any other eclipse plugins and consists of the project structure, frame for cfeditor files editing and outline. User can use “standard” Eclipse features as syntax highlighting or call support for method into current position, as well as syntax validation.

Talking about the cfeditor project, we mean the structure with two parts: one is about CFEngine configuration files (stored into input folder), another about non CFEngine files (stored into replication folder). Additionally both parts are including separation between tasks and service scripts or configurations. Under service considered specifically grouped tasks to provided main service, which is provided by IT system.

Example: GGUS administration

Global Grid User Support (GGUS) is a well-known user support platform for Grid community. The service is provided by a small development team with collaboration between major stakeholders. One requirement is availability. For this, fault safe environment should be improved by the organizational procedures to avoid (or drop) time when system is not available or in state, do not sufficient to get service to the end user.

There is a defined release cycle for development team. Release maintenance is done during scheduled downtime. But, according to the availability requirements, time when system is down should be as small as possible.

There are three separated GGUS environments: development, testing and production. Every environment has some machines, which provides the environment platform (fig.3). Release is started from development, follows to testing and in scheduled downtime is maintained into production environment.

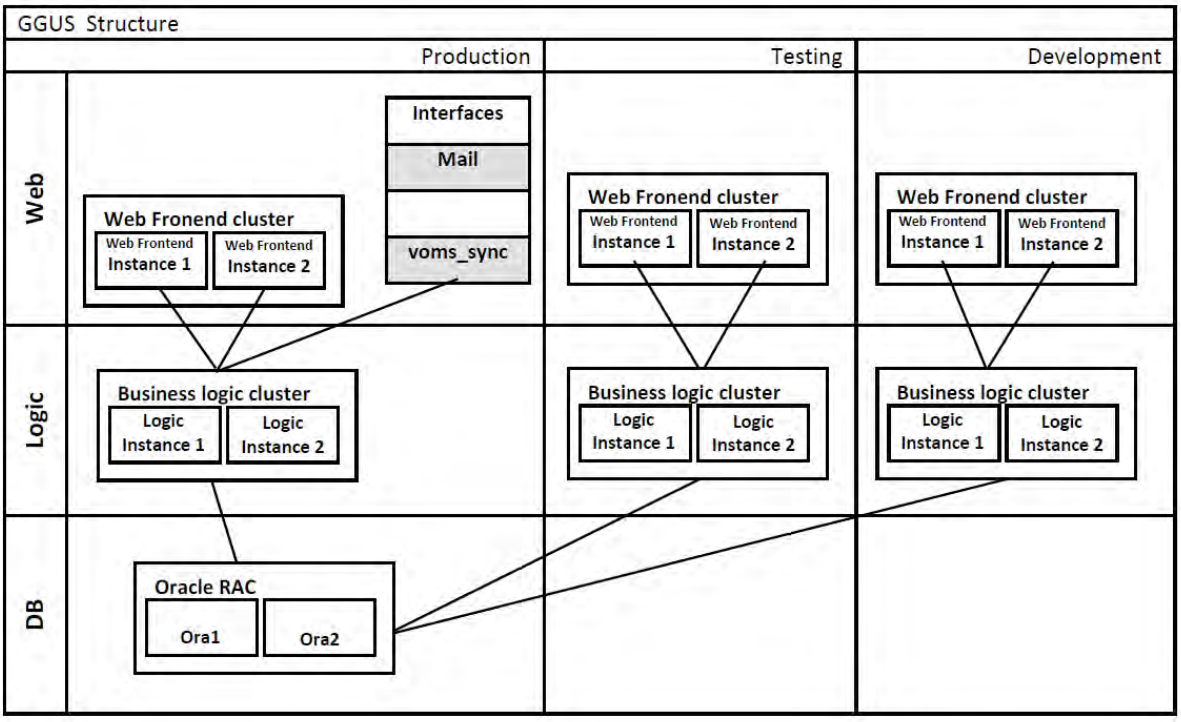


Fig.3: Three environments for GGUS

Currently, all machines for web frontends and logic parts are managed by the CFEngine version 3, which controls and executes all necessary commands and configurations into the system.

To organize the system administration activities for GGUS, a code repository is used. This is a based on the open source subversion server to store all scripts which will be maintained by CFEngine. Plus also CFEngine scripts itself. Configuration files, cronjobs and bash scripts will be located under operating system, based on rules, which are described in CFEngine policies. GGUS administration is using cfeditor as editor for any code repository input.

As the GGUS system is based on Red Hat Enterprise Server operating system, the binary repository is Red Hat Package Management (or rpm) repository. All binaries: operating system patches, own constricted software, additional binaries and libraries are stored into binary repository.

Configurations from code repository manage the procedures when binaries will come to the system, but direct access from the GGUS sysadmin is not closed, and sometimes is necessary to react directly from the Linux shell environment.

Knowledge about the system configurations currently stored into repository, based on open source OCS inventory project. Hence, the groups of packages with installed version, state of the host certificates or configuration files can be easily found pro host or pro service.

Summary

Let's group together some aspects of using configuration management software:

- Management of IT resources to provide IT services can be done in different ways. One way is a policy-based management with idea to bring systems into described in policy state.
- Part of system administration activities can be organized based on tools from development teams. With a purpose to simplify processes of scripting, storing, releasing, patching, documenting and project management needs.
- System Configuration is about system state and changes for this state. And is an activity for different groups of interests (Service management, administration, changes implementation, maintenance, etc.), and in the same time is a core part of IT Infrastructure.
- Configuration management can be seen as system programming on top of configuration management platform. This view is not really against to "old-school" direct access to the system, but improvements, considering teams of administration, documenting activities and complexity grow for IT systems.
- There is a list of open source configuration management projects to provide indirect access for administrator into the system. Examples: Cfengine, Puppet, etc.
- Cfeditor is an editor for configuration management platform syntax and tries to be as a part of IDE for System administration team.

References

[1] CFEngine: http://cfengine.com/
[2] GGUS: https://ggus.eu/
[3] Cfeditor http://code.google.com/a/eclipselabs.org/p/cfeditor/
[4] Eclipse: http://www.eclipse.org/
[5] Eclipse Modelling Framework EMF: http://www.eclipse.org/modeling/emf/
[6] Xtext: http://www.eclipse.org/Xtext/
[7] Google Guice: http://code.google.com/p/google-guice/

ATLAS TIER 3 IN GEORGIA

A. Elizbarashvili
Ivane Javakhishvili Tbilisi State University, Georgia
1, Chavchavdze Ave., 0218, Tbilisi; Tel.: +995 32 2225107

I. PC FARM FOR ATLAS TIER 3 ANALYSIS

Arrival of ATLAS data is imminent. If experience from earlier experiments is any guide, it's very likely that many of us will want to run analysis programs over a set of data many times. This is particularly true in the early period of data taking, where many things need to be understood. It's also likely that many of us will want to look at rather detailed information in the first data – which means large data sizes. Couple this with the large number of events we would like to look at, and the data analysis challenge appears daunting.

Of course, Grid Tier 2 analysis queues are the primary resources to be used for user analyses. On the other hand, it's the usual experience from previous experiments that analyses progress much more rapidly once the data can be accessed under local control without the overhead of a large infrastructure serving hundreds of people. However, even as recently as five years ago, it was prohibitively expensive (both in terms of money and people), for most institutes not already associated with a large computing infrastructure, to set up a system to process a significant amount of ATLAS data locally. This has changed in recent years. It's now possible to build a PC farm with significant ATLAS data processing capability for as little as $5-10k, and a minor commitment for set up and maintenance. This has to do with the recent availability of relatively cheap large disks and multi-core processors.

Let's do some math. 10 TB of data corresponds roughly to 70 million Analysis Object Data (AOD) events or 15 million Event Summary Data (ESD) events. To set the scale, 70 million events correspond approximately to a 10 fb^{-1} sample of jets above 400-500 GeV in PT *and* a Monte Carlo sample which is 2.5 times as large as the data. Now a relatively inexpensive processor such as Xeon E5405 can run a typical analysis Athena job over AOD's at about 10 Hz per core. Since the E5405 has 8 cores per processor, 10 processors will be able to handle 10 TB of AODs in a day. Ten PCs is affordable. The I/O rate, on the other hand, is a problem. We need to process something like 0.5 TB of data every hour. This means we need to ship ~1 Gbits of data per second. Most local networks have a theoretical upper limit of 1 Gbps, with actual performance being quite a bit below that.

Today, however, we have another choice, due to the fact that we can now purchase multi-TB size disks routinely for our PCs. If we distribute the data among the local disks of the PCs, we reduce the bandwidth requirement by the number of PCs. If we have 10 PCs (10 processors with 8 cores each), the requirement becomes 0.1 Gbps. Since the typical access speed for a local disk is > 1 Gbps, our needs are safely under the limit.

II. FIRST ACTIVITIES ON THE WAY TO TIER3S CENTER IN ATLAS GEORGIAN GROUP OMPUTING

The local computing cluster (14 CPU, 800 GB HDD, 8-16GB RAM, One Workstation and 7 Personal Computers) have been constructed by Mr. E. Magradze and Mr. D. Chkhaberidze at High Energy Physics Institute of Ivane Javakhishvili Tbilisi State University (HEPI TSU). The creation of local computing cluster from computing facilities in HEPI TSU was with the aim of enhancement of computational power (resources). The scheme of the cluster network is following on Figure 1:

Figure 1: Scheme of cluster at High Energy Physics Institute of TSU

There are number of research activities in particle physics at HEPI TSU [2].

The first project was the search for and study of rare processes within and beyond standard model at ATLAS experiment of large hadron collider at CERN with INTERNATIONAL SCIENCE & TECHNOLOGY CENTER (ISTC). It was the Grant G-1458 (2007-2010) where the leading institution was the Institute of High Energy Physics of Javakhishvili Tbilisi State University (HEPI TSU), Georgia and the participant institution was Joint Institute for Nuclear Research (JINR), Dubna, Russia. Participants from IHEPI TSU are L. Chikovani (IOP), G. Devidze (Project Manager), T. Djobava, A.Liparteliani, E. Magradze, Z. Modebadze, M.Mosidze, V.Tsiskaridze. Participants from JINR are G. Arabidze, V. Bednyakov, J. Budagov (Project Scientific Leader), E. Khramov, J. Khubua, Y. Kulchitski, I.Minashvili, P. Tsiareshka. Foreign Collaborators are Dr. Lawrence Price, (Senior Physicist and former Director of the High Energy Physics Division, Argonne National Laboratory, USA), Dr. Ana Maria Henriques Correia (Senior Scientific Staff of CERN, Switzerland).

The second was G-1458 Project Scientific Program:

1. Participation in the development and implementation of the Tile Calorimeter Detector Control System (DCS) of ATLAS and further preparation for phase II and III commissioning.
2. Test beam data processing and analysis of the combined electromagnetic liquid argon and the hadronic Tile Calorimeter set-up exposed by the electron and pion beams of 1 ÷ 350 GeV energy from the SPS accelerator of CERN.
3. Measurements of the top quark mass in the dilepton and lepton+jet channels using the transverse momentum of the leptons with the ATLAS detector at LHC/CERN.
4. Search for and study of FCNC top quark rare decays $t \rightarrow Zq$ and $t \rightarrow Hq$ (where q= u, c; H is a Standard Model Higgs boson) at ATLAS experiment (LHC).
5. Theoretical studies of the prospects of the search for large extra dimensions trace at the ATLAS experiment in the FCNC-processes.
6. Study of the possibility of a Supersymmetry observation at ATLAS in the mSUGRA predicted process $gg \rightarrow \tilde{g}\tilde{g}$ for EGRET point.

Another project was Grant 185 - ATLAS Experiment Sensitivity to New Physics with Georgian National Scientific Foundation (GNSF). Leading Institution was Institute of High Energy Physics of I. Javakhishvili Tbilisi State University, Georgia and Participant Institution was E. Andronikashvili Institute of Physics (IOP). Participants from IHEPI TSU are G. Devidze (Project Manager), T. Djobava (Scientific Leader), J.Khubua, A.Liparteliani, Z. Modebadze, M.Mosidze, G.Mchedlidze, N.Kvezereli. Participants from IOP are L.Chikovani. V.Tsiskaridze, M.Devsurashvili, D.Berikashvili, L.Tepnadze, G. Tsilikashvili, N.Kakhniashvili.

The cluster was constructed on the basis of PBS (Portable Batch System) software on Linux platform and for monitoring was used “Ganglia” software. All nodes were interconnected using gigabit Ethernet interfaces.

The required ATLAS software was installed at the working nodes in SLC 4 environment. The cluster have been tested with number of simple tests and tasks studying various processes of top quarks rare decays via Flavor Changing Neutral Currents $t \rightarrow Zq$ (q= u,c quarks), $t \rightarrow Hq \rightarrow bbar,q$, $t \rightarrow Hq \rightarrow WW^*q$ (in top-antitop pair production) have been run on the cluster. Signal and background processes generation, fast and full simulation, reconstruction and analysis have been done in the framework of ATLAS experiment software ATHENA. (L.Chikovani, T.Djobava, M.Mosidze, G.Mchedlidze)

III. ACTIVITIES AT THE INSTITUTE OF HIGH ENERGY PHYSICS OF TSU (HEPI TSU)

The installed cluster system is working with PBS. PBS consist of four major components (working model is shown on the Figure 2):

Commands: PBS supplies both command line commands and a graphical interface. These are used to submit, monitor, modify, and delete jobs. The commands can be installed on any system type supported by PBS and do not require the local presence of any of the other components of PBS. There are three classifications of commands:

Job Server: The Job Server is the central focus for PBS. Within this document, it is generally referred to as the Server or by the execution name pbs_server. All commands and the other daemons communicate with the Server via an IP network. The Server's main function is to provide the basic batch services such as receiving/creating a batch job, modifying the job, protecting the job against system crashes, and running the job (placing it into execution).

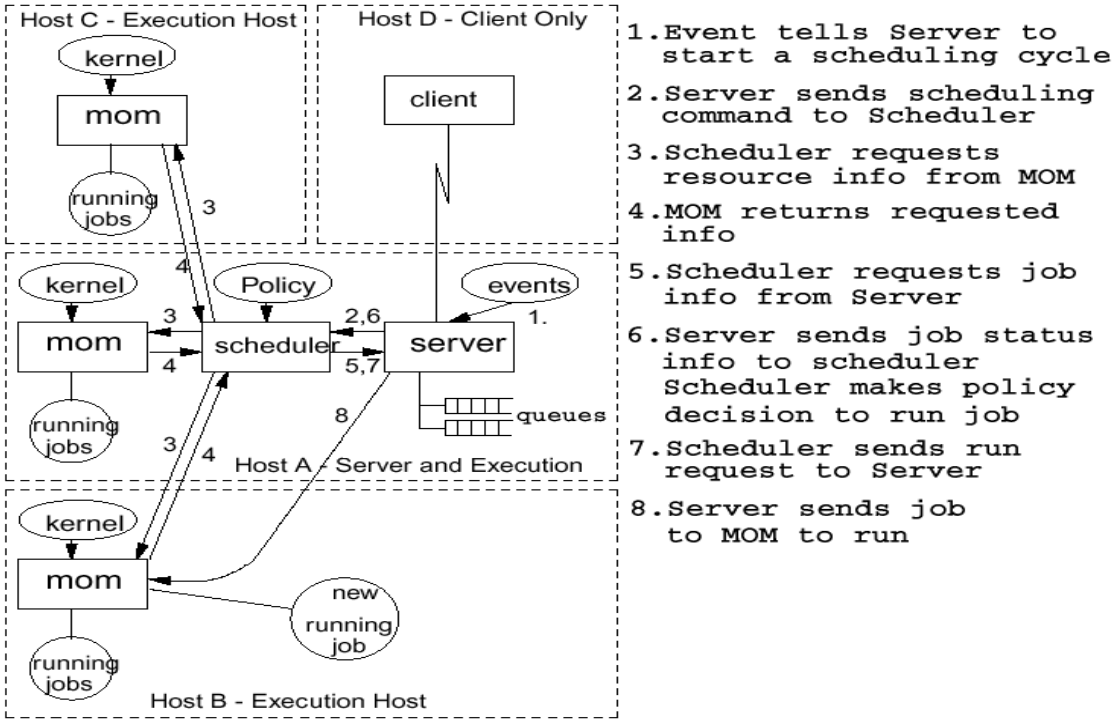


Figure 2: PBS working schema

Job executor: The job executor is the daemon which actually places the job into execution. This daemon, pbs_mom, is informally called Mom as it is the mother of all executing jobs.

Job Scheduler: The Job Scheduler is another daemon which contains the site's policy controlling which job is run and where and when it is run. Because each site has its own ideas about what is a good or effective policy, PBS allows each site to create its own Scheduler.

On that Batch cluster had installed Athena software 14.1.0 and 14.2.21. The system was configured for running the software in batch mode and the cluster had been used on some stages of the mentioned ISTC project. Also the system used to be file storage.

IV. PLANS TO MODERNIZE THE NETWORK INFRASTRUCTURE

It is planned to rearrange the created the existing computing cluster into ATLAS Tier 3 cluster. But first of all TSU must have the corresponding network infrastructure. Nowadays the computer network of TSU comprises 2 regions (Vake and Saburtalo). Each of these two regions is composed of several buildings (the first, second, third, fourth, fifth, sixth and eighth in Vake, and Uptown building (tenth), Institute of applied mathematics, TSU Library and Biology building (eleventh) in Saburtalo). Each of these buildings is separated from each other by 100 MB optical network. The telecommunication between the two regions is established through Internet provider the speed of which is 1 000 MB (see Fig. 3).

Servers and controllable network facilities are predominantly located in Vake region network. Electronic mail, domain systems, webhosting, database, distance learning and other services are presented at TSU. Students, administrative staff members and academic staff members, research and scientific units at TSU are the users of these servers. There are 4 (four) Internet resource centers and

several learning computer laboratories at TSU. The scientific research is also supported by network programs. Total number of users is 2500 PCs. The diversity of users is determined by the diversity of network protocols, and asks for maximum speed, security and manageability of the network.

Initially, the TSU network consisted only from dozens of computers that were scattered throughout different faculties and administrative units. Besides, there was no unified administrative system, mechanisms for further development, design and implementation. This has resulted in flat deployment of the TSU network.

This type of network does not allow setting up of sub-networks and Broadcast Domains are hard to control. Formation of Access Lists of various user groups is complicated. It is hard to identify and eliminate damages to each separate network. It is almost impossible to prioritize the traffic and the quality of service (QOS).

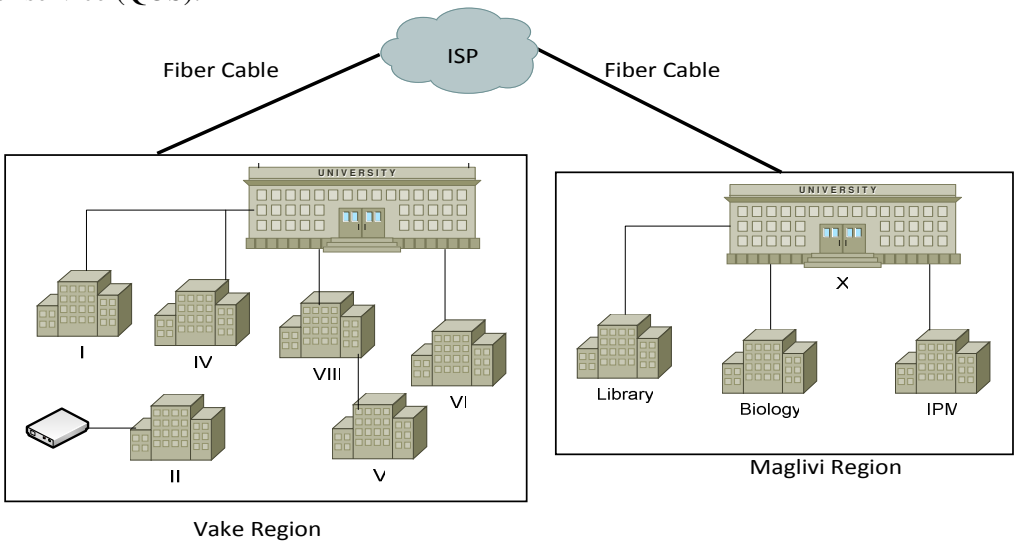


Figure 3: TSU existing network

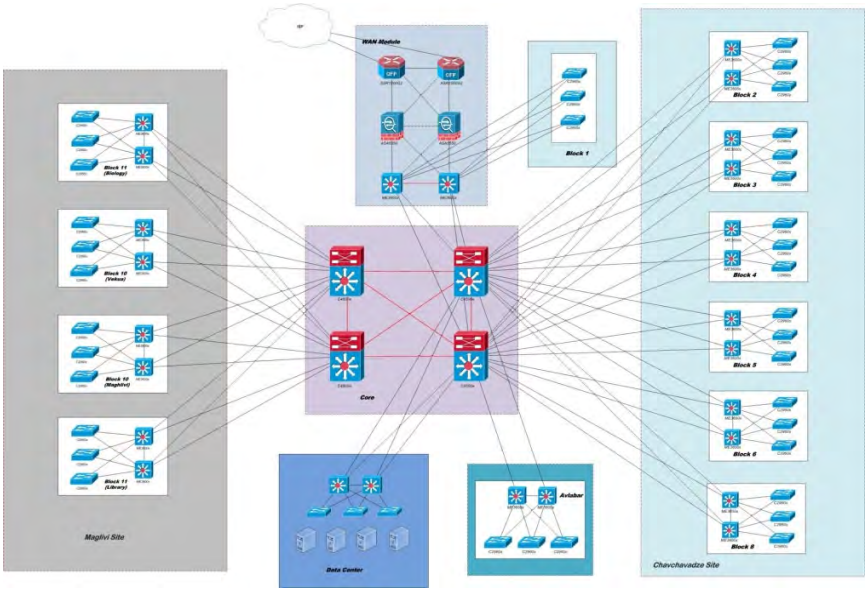


Figure 4: TSU planned network topology

Because there is no direct connection between the two above-mentioned regions it is impossible to set up an Intranet at TSU. In the existing conditions it would have been possible to set

up an Intranet by using VPN technologies. However, its realization required relevant tools equipped with special accelerators in order to establish the 200 MB speed connection. This is the equipment that TSU does not possess. The reforms in learning and scientific processes demands for the mobility and scalability of the computer network. It is possible to accomplish by using VLAN technologies, however in this case too absence of relevant switches hinders the process of implementation.

With all above-said, through implementing all of the devices we will have a centralized, high speed, secured and optimized network system:

- *Improving TSU informatics networks security* - traffic between the local and global networks will be controlled through network firewalls. The communications between sub-networks will be established through Access Lists.
- *Improving communication among TSU buildings* - main connections among the ten TSU buildings are established through Fiber Optic Cables and Gigabit Interface Converters (GBIC). This facilities increase the speed of the bandwidth up to 1 GB.
- *Improving internal communication at every TSU building* - internal communications will be established through third-level multiport switches that will allow to maximally reducing the so-called Broadcasts by configuring local networks (VLAN). The Bandwidth will increase up to 1GB.
- *Providing the network mobility and management* - In administrative terms, it will be possible to monitor the general network performance as well as provide the prioritization analysis for each sub-network, post or server and installing the tier 3g/s system at TSU.

V. ATLAS Tier-3s

The minimal requirement is on local installations, which should be configured with a Tier-3 functionality [3]. A Computing Element known to the Grid, in order to benefit from the automatic distribution of ATLAS software releases

- Needs >250 GB of NFS disk space mounted on all WNs for ATLAS software
- Minimum number of cores to be worth the effort is under discussion (~40?)

A SRM-based Storage Element, in order to be able to transfer data automatically from the Grid to the local storage, and vice versa:

- Minimum storage dedicated to ATLAS depends on local user community (20-40 TB?)
- Space tokens need to be installed:
  LOCALGROUPDISK (>2-3 TB), SCRATCHDISK (>2-3 TB), HOTDISK (2 TB)
- Additional non-Grid storage needs to be provided for local tasks (ROOT/PROOF)

The local cluster should have the installation of:

- A Grid User Interface suite, to allow job submission to the Grid
- ATLAS DDM client tools, to permit access to the DDM data catalogues and data transfer utilities
- The Ganga/pAthena client, to allow the submission of analysis jobs to all ATLAS computing resources

Tier 3g [1] work model is shown on the fig. 5.

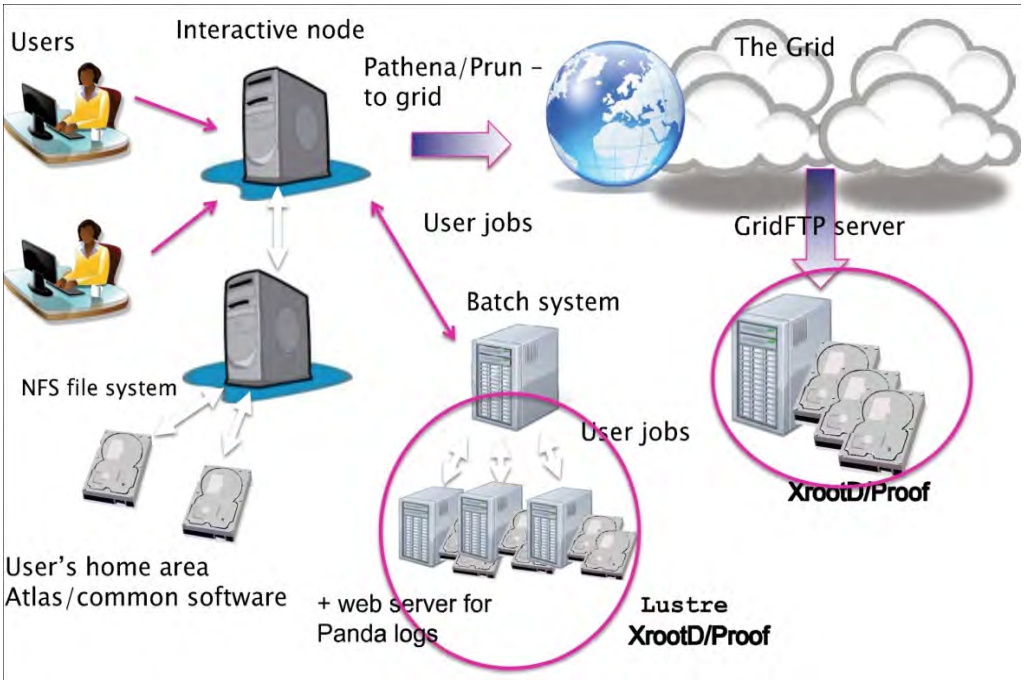


Figure 5: Atlas Tier 3g work model

REFERENCES

[1] http://indico.ific.uv.es/indico/materialDisplay.py?contribId=7&materialId=slides&confId=386
[2] http://ebookbrowse.com/djobava-kekvisit-pdf-d281316569
[3] A. Elizbarashvili, ATLAS TIER 3 IN GEORGIA, in Proc. of XXIII Int. Symp. on Nuclear Electronics & Computing (NEC`2011), Dubna, 2011, pp.122-131.

TECHNOLOGY OF SEMANTIC STRUCTURING OF THE DIGITAL LIBRARY CONTENT

I.A. Filozova
Laboratory of Information Technologies,
Joint Institute for Nuclear Research, 141980, Dubna, Russia
fia@jinr.ru

Introduction

The current situation illustrates the some following trends. First, shifting from a traditional publishing paradigm to a digital archive-based approach. Second, accumulation of the expansive content volume in a special information fund. Third, a growth number of institutional repositories in the open access form [1]. It leads to the content integration on a metadata level and appearance of the common Data and Information Spaces. Currently, there are about 2 900 repositories with a total number of records of about 40 million according to statistics from the Register of Open Access Repositories (ROAR - http://roar.eprints.org). And it's growing permanently. It's required a lot of time and human resources to research it by the traditional search technologies. However, if the user has an effective tool to study the content of the information system, then he has a chance to achieve the desired result faster and better.

1. Problematic situation

The main purpose of the information search for the user — a satisfaction of his information need. It's possible by two ways: 1) Information Request: Establish an information request within the specified information search language. 2) Question: Ask the question in the subject domain language (truncated natural language).

The search results depend of how well the user has formulated his request/question. Often the user doesn't understand well if the received response is the pertinent to his question. This problematic situation is presented on fig.1.

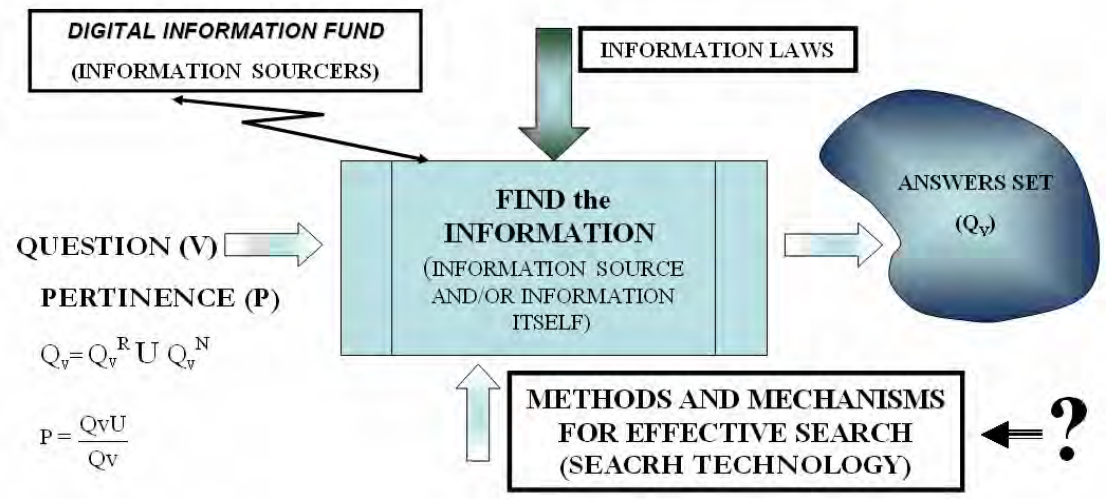


Fig.1: Problem of the search in the digital information funds

Herein:

Q_v^R — set of relevant answers;
Q_v^N — set of answers representing the information noise;
Q_v^U — set of useful answers;
P — pertinence indicator,

i.e. correspondence found documents to the information user need, regardless of how fully and exactly this information need is expressed in the information request. Pertinence is a pragmatic metrics. The pragmatic level of the information search requires additional knowledge about the subject domain. So, the creation of effective mechanisms to search the answers to the questions in the digital information collections is the actual problem. However, semantic structuring of the content is required.

Also the following should be noted. Research results, scientific and engineering efforts represented in publications have semantic relations between them via a citation mechanism. The linkages between the member staff — the author of publications, their affiliation, participation in the collaborations, experiments, projects, etc. The description of these relationships and their properties opens up new possibilities for studying a documents corpus of digital libraries. Relations, reflecting the presentation logic of the author's thoughts in this publication, topic, subject area will be covered. The discussed approach is based on the technology of structuring scientific texts by the logic-semantic network (LSN) Question-Answer-Reaction for the organization of semantic search in digital libraries [2].

2. Research lines and realization ideas

Proposed are: *catalog service* creation and support for the funds-corpuses. Question-Answer Navigator creation that provides such features as: 1) the ability of the refinement and deepening of the understanding the question meaning; 2) the ability of refining, deepening, expansion of the knowledge or the obtaining a new knowledge during the answer to the question on the search process.

So, research lines are:

(1) Development of the method and mechanism for an effective search of the set of the relevant answers to the questions.
(2) Technology development for the creation and support of the *catalog service* of the information fund for providing an efficient search of the answers to the questions.
(3) Software development – cataloguer workstation for the structuring of the information fund.

The realization ideas are the following. The method basis is a way to describe the scientific and technical information by a set of logic-semantic networks *Question-Answer-Reaction (LSN QAR).* The bases for the search engine are:

- motion way along *LSN*, controlled by the user;
- choice of *LSN* nodes (questions or answers) based on an ontological model of user question.

The basis of the technology is a way of the description of the subject domain by *LSN QAR* set. Mechanism of technology is a workstation of the cataloguer (LSN QAR developer)

3. Logic-Semantic Network *Question-Answer-Reaction*

3.1. Cognitive function of the question

Communication of specialists in some subject area is effective when it occurs in the question – answer form. The question as a tool of information search is very high. No question – no new knowledge. It is a bridge between the known and the unknown.

Question – a thought query expressed in the interrogative sentence and realized in the answering form, directed at the development, refinement or supplement of the knowledge.

Answer – a realization of the cognitive function of the question in the form of the new obtained judgment.

The cognitive function of the question is aimed at the supplement, refinement and development of the previously obtained general representations of objects and phenomena of reality (fig.2). The process of asking question and the answer search is a complex iterative process. The question is based on an already-known knowledge that acts as a datum question always. The answer search assumes to address to a specific area of theoretical or empirical knowledge that is called the answer search scope. Setting process of the adequacy question and answer is aimed at the detection of the possible inconsistencies in the answer. Based on that, answer search scope or datum question or subject research is expanded.

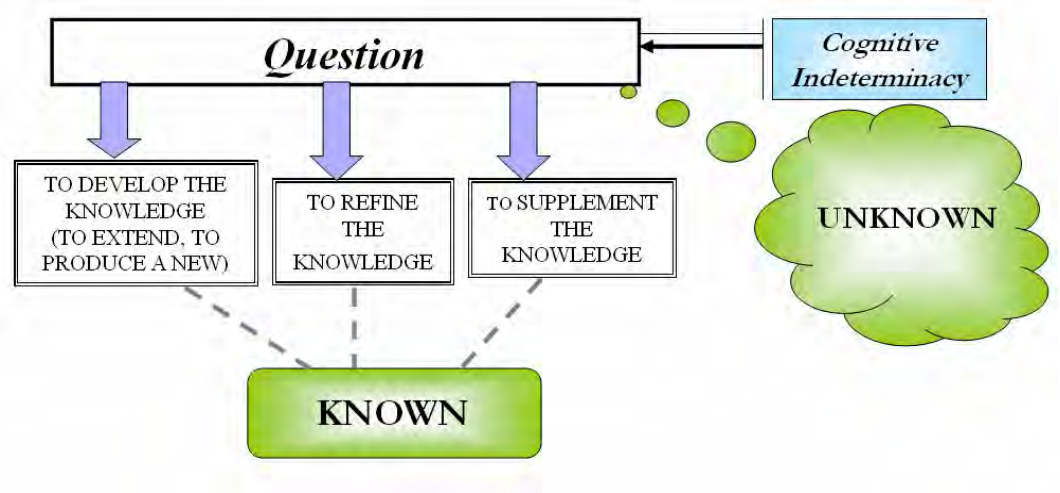


Fig.2: Cognitive function of the question

3.2. Basic Statements of the LSN Question-Answer-Reaction

Logic-semantic network Question-Answer-Reaction – a set of the questions, answers and relationships between them forming a uniform system.

Question – query expressed in the interrogative sentence aimed at the development, refinement or supplement of the knowledge.

Answer – a realization of the cognitive function of the question in the form of the new obtained judgment. *Answer must be built in accordance with the content and structure of the asked question. Only in this case, the answer is regarded as relevant.*

Reaction – a semantic description of the question and answer.

Types of reactions:

1. Question Reaction – a description of the datum question *(to understand the environment and causes of the question and to establish the semantic adequacy with the answer scope).*
2. Answer Reaction – a description of the answer scope *(to understand the question semantics and relationship with answer).*

Thus, Question-Answer-Reaction model may be presented by directed graph, where nodes are questions and answers (fig.3.). Questions are placed on the odd level, answer — on the even level. Edges — the relations between them. Navigation is a motion way along *LSN*, controlled by the user.

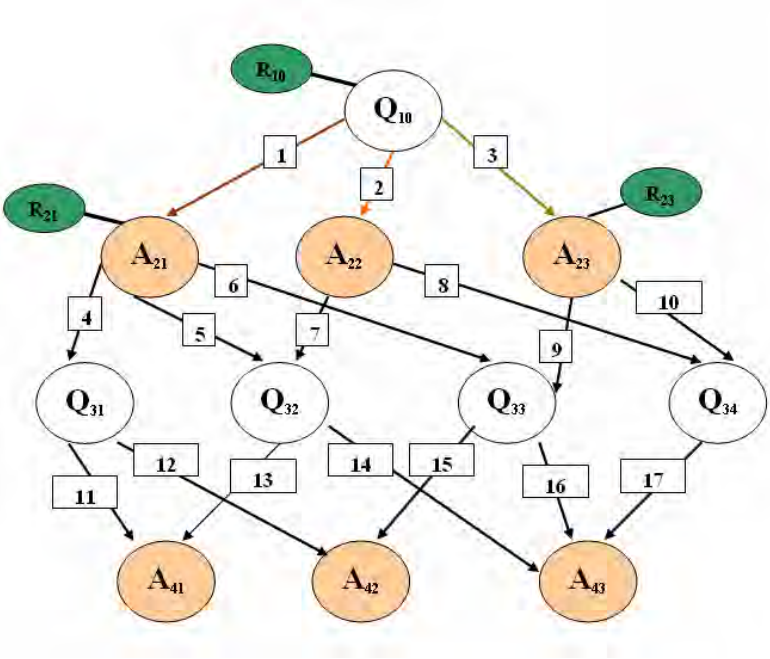
3.3. Reactions

The role of the reaction in this unit *Question-Answer-Reaction* is very important. It helps the user understand where the question (Datum Question) and answer (Search Scope) appeared. Reaction can be text information, with links to primary sources, illustrative material (drawings, graphs, tables, slide shows, videos, etc.) and / or a combination thereof. Reactions help the user understand the semantic field questions and received answers to it, thus can improve pragmatic metrics of information search — pertinence. Let's illustrate the reaction example and consider the following logical sequence of the units *Question-Answer-Reaction*.

Question 1 (Q1). What is a JAVA?

Question 1 Reaction 1 (QR11). With respect pronunciation formed two different standards - borrowed from the English / dʒɑ:və / and traditional «Ява» (on russian), corresponding to the traditional pronunciation of the Java name island.

Question 1 Reaction 2 (QR12). Java (Indonesian: *Jawa*) is an island of Indonesia with a population of 135 million. Square – 132 000 k2… .



№ edge	Way
1	1,4,11
2	1,4,12
3	1,5,13
5	1,5,14
6	1,6,15
7	1,6,16
8	2,7,13
9	2,7,14
10	2,8,17
11	3,9,15
12	3,9,16
13	3,10,17

Fig.3: LSN Graph Question-Answer-Reaction

Question 1 Reaction 3 (QR13). Slide show, photo-collage with the views of Java island:

Answer 1 to Question 1 (A11). Java – an object-oriented programming language developed by Sun Microsystems.

Reaction 1 of the Answer 1 to the Question 1 (RA11). Why the language is called JAVA?
There is a version that language got its name from coffee grown on the same island. As you know, this drink is hot like some programmers. Therefore, a cup of steaming coffee is displayed on logo.

Reaction 2 of the Answer 1 to the Question 1 (R2A11). Sun Microsystems, Inc (now part of Oracle Corporation) — U.S. company that produces software and hardware…
Answer 2 to Question 1. Java — not only the language itself, but also a platform for development and execution of the applications based on this language (fig.4.)

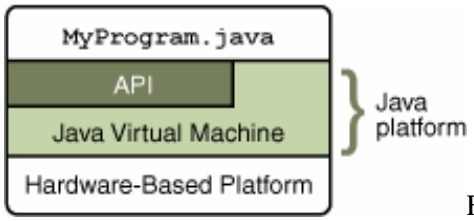


Fig.4: Java Platform

3.4. Formal View of Subject Domain

The accumulated knowledge in the subject domain is expressed in scientific reports, monographs, articles, educational materials, information collections, books, dictionaries, etc. It is possible to present the whole volume of information as a set of the ordered thematic sections, each of that reflects a certain aspect of the subject domain. Each topic can be associated with LSN Question-Answer-Reaction. Integrated semantic structuring of fund digital libraries on the LSN basis leads to the creation of multilevel network structure that can be the basis for the navigation mechanism. User has a possibility to navigate in the horizontal and vertical directions (fig.5). Motion from i-th to the i + k-th level of network deepens on the knowledge. Horizontal motion network expands the knowledge. Motion up the network summarizes knowledge.

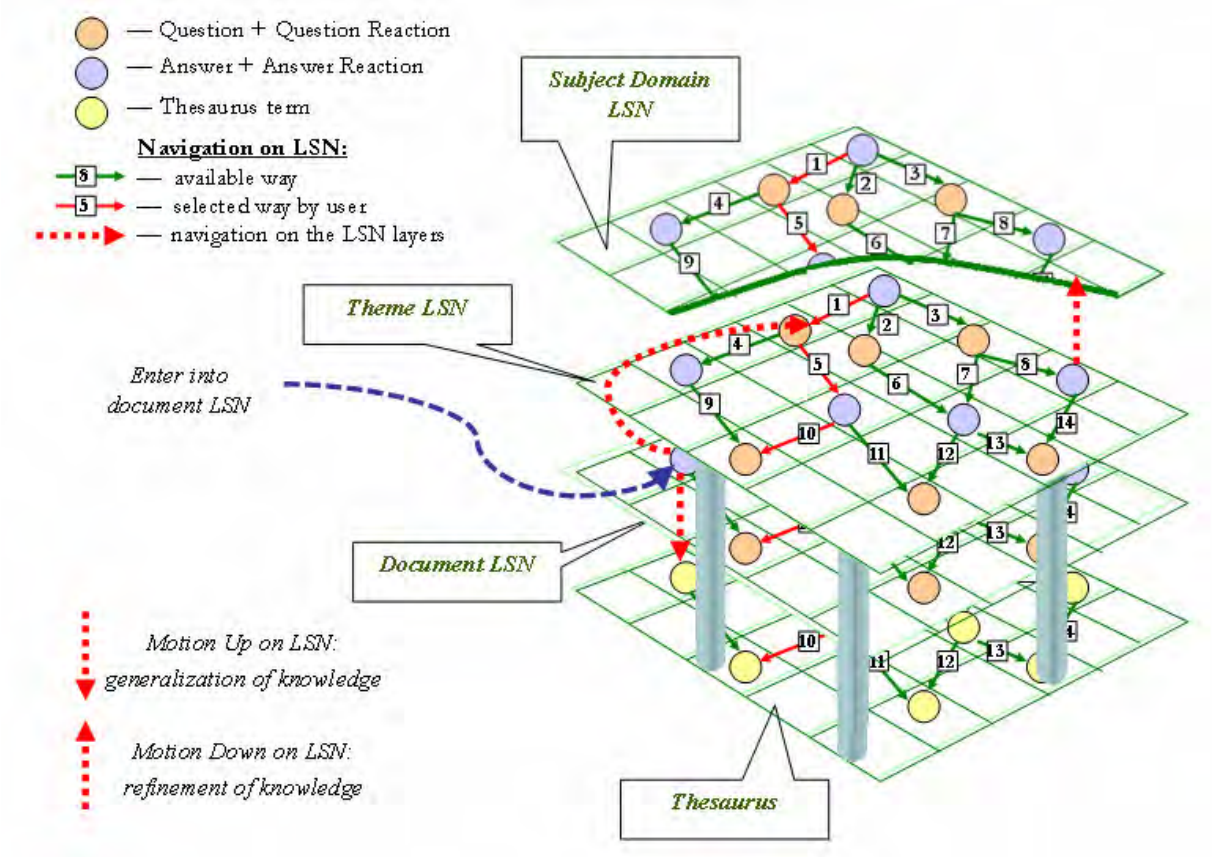


Fig.5: Multilayer Related Set of Graphs

3.5. Analysis Method of Scientific Texts

To build document LSN, one needs to analyze it. The document is studied by the expert in terms of:

1) semantic matching title and content; 2) set of filters:

Filter 1 (F1) - General Part. F1 includes an analysis of the problem, its history, overview, topicality.

Filter 2 (F2) - Author concept. F2 includes new terms introduced by the authors, traditional terms with the author's interpretation, the narrowing semantics.
Filter 3 (F3) - Examples and illustrations. To clarify difficult places in the text, reduce the text size under stringent restrictions on the volume.
Filter 4 (F4) - The idea of the author. Describes and explains the author's main idea.
Further, the basic questions that correspond to the text are formed.
As an optional service, the user may be offered some visualization for navigation.

Conclusion

Discussed approach proposes:

- *Catalog Service* creation and support for the funds-corpuses,
- Question-Answer Navigator creation that provides such features as 1) the ability of the refinement and deepening of understanding the question meaning; 2) the ability of refining, deepening, expansion of the knowledge or obtaining a new knowledge during the answer to question search process.

Realization of such *Catalog Service* and Navigator will allow to study the content of the digital information funds by the natural mode for the human: *refinement, generalization and obtaining a new knowledge* — question-answer mode. The main problem of the question- answer system is a maximal automation of the process of the creation and support of the fund *service catalog*.

References

[1] I.A. Filozova, V.V. Korenkov, G. Musulmanbekov. Towards Open Access Publishing at JINR. – Proceedings of XXII International Symposium on Nuclear Electronics and Computing (NEC`2009), Varna, Bulgaria, Sept. 7-14, 2009. – Dubna: JINR, 2010. – p.124-128 - (JINR ; E10,11-2010-22).

[2] V.N. Dobrynin, I.A. Filozova. The search based on the logical semantic network «Question-Answer-Reaction». – Proceedings of XII Russian Conference RCDL'2010, Kazan, Russia, Oct.13-17, 2010. – Kazan: Kazan State University, 2010. – p. 301-308. (on russian).

APPLICATION OF DESKTOP GRID TECHNOLOGY IN MATERIAL SCIENCE[1]

O. Gatsenko*, O. Baskova, B. Bandalak, V. Tatarenko, Yu. Gordienko
G.V.Kurdyumov Institute for Metal Physics, National Academy of Sciences of Ukraine,
36 Academician Vernadsky Blvd., 03680 Kiev, Ukraine
**gats@imp.kiev.ua*

The global distributed computing infrastructure (DCI) on the basis of BOINC and Desktop Grids (DGs) technologies for high-performance distributed computing was used for porting the sequential molecular dynamics (MD) application to its parallel version for DCI with DGs and Service Grids (SGs) connected by EDGeS-bridge. It is shown that the mechanical characteristics evaluated on the basis of MD simulations using LAMMPS package in the DG-SG DCI are in satisfactory agreement with the experimental data and allowed to discover the new aspects of deformation and fracture mechanisms in nanomaterials. Porting MD-applications to DG-SG DCI is easy and efficient, if BOINC SZTAKI DC-API and SG-DG EDGeS Bridge are used; parameter decomposition and sweeping parallelism are possible; message passing is localized at worker side.

1.Introduction

Simulation of structure and mechanical properties of materials is extremely important in materials science to quantify their deformation and strength characteristics. Among variety of new materials a special place is occupied by materials of nanoscale structure (nanomaterials), such as metal nanocrystals and nanoscale non-metallic materials with unique properties (nanotubes, graphene). Molecular dynamics (MD) simulations of nanoscale processes with physical parameter decomposition for parameter sweeping in a brute force manner are very perspective. The recent advances in computing algorithms and infrastructures, especially in development of distributed computing infrastructures, allow us to use the efficient methods for solving these tasks without expensive scaling-up. DCIs on the basis of the BOINC SZDG [1], XtremWeb-HEP [2], OurGrid [3], EDGeS [4], WS-PGRADE [5] platforms for high-performance distributed computing are very promising to use the donated computing resources of idle PCs and integration with global computing grid. In this context, the main objective was to demonstrate the capabilities of the proposed DCI for simulation of some physical processes: tension of metal nanocrystals under different conditions, tension of ensemble of metal nanocrystals under the same conditions, and simulation of graphene nanosamples.

2.Distributed computing infrastructure

Some sequential applications, which allow for physical parameter decomposition, by slight modifications in its code could to be ported to the parallel version for worker nodes of a distributed computing environment (DCI) as Desktop Grid (DG) by means of the BOINC software platform and availability of simple and intuitive Distributed Computing Application Programming Interface (DC-API) [1]. In this work the very popular non-commercial open-source package LAMMPS by Sandia Labs [6] was selected for such porting to DG DCI as the DG-enabled application *LAMMPSoverDCI* and the details of such porting were given recently in [7,8]. The typical simulation of the metal nanocrystal with 10^7 atoms for 1-10 picoseconds of the simulated physical process takes approximately 1-7 days on a single modern CPU. But the massive MD simulations of plastic deformation processes for the large quantity of Al nanocrystals (~10^2-10^3) can be carried out

[1] The work was partially funded by EU FP7 DEGISCO (Desktop Grids for International Scientific Collaboration) project, No. RI-261561, EU FP7 SCI-BUS (SCIentific gateway Based User Support) project, No. RI-283481, and partially supported in the framework of the research theme "Introduction and Use of Grid Technology in Scientific Research of IMP NASU" under the State Targeted Scientific and Technical Program to Implement Grid Technology in 2009-2013.

independently in DG-SG DCI *SLinCA@Home* (http://dg.imp.kiev.ua) (Fig. 1, 2) connected to the computing resources of the European Grid Initiative (EGI) by EDGeS-Bridge technology [4].

3.Results

From a physical point of view the work was motivated by the previous results [9-11] that single crystal aluminum foil under the influence of compressed cyclic stretch revealed macro- and micro- scale evolution of crystalline defects in bulk and on surface. Such defect evolution demonstrates several signs of self-similar geometry and self-organized behavior [12-14], that was analyzed by several idealized models [15-19]. These rough models cannot take into account the details of interatomic interactions in real crystal lattices, and that is why MD simulations have been performed on the basis of embedded atom method (EAM) for Al nanocrystals with 10^5-10^7 atoms.

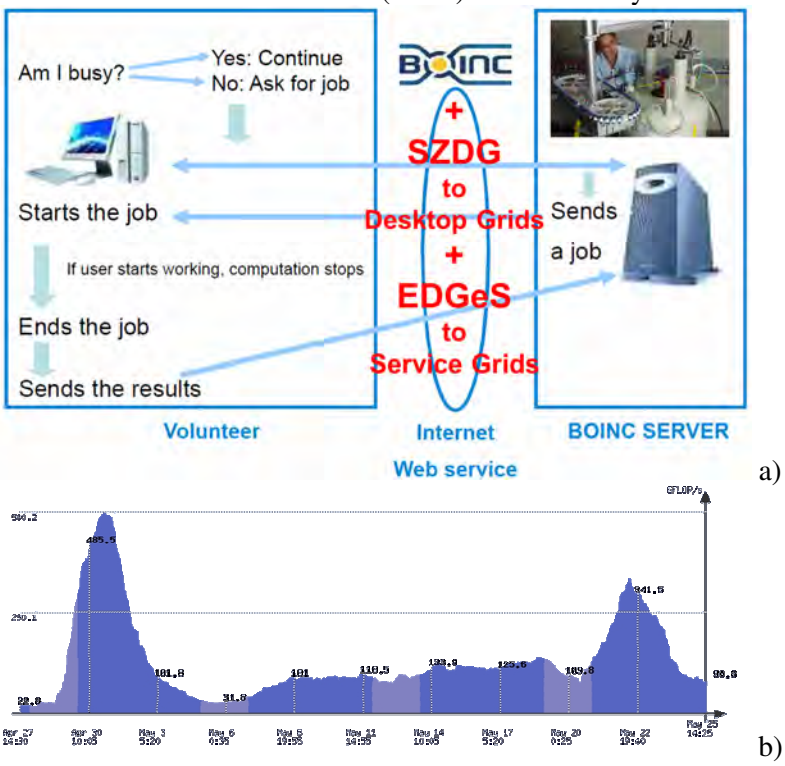


Figure 1: The schematic workflow in DG-SG DCI *SlinCA@Home* on the basis of BOINC, SZDG, and EDGeS technologies (the adapted figure by courtesy of Fermin Serrano) (a);
and the typical variable performance of DG-SG DCI *SLinCA@Home* (b)

3.1.Unixaial tension of Al single crystals

MD simulations were performed for two orientations: <011>{011} and <010>{010}, where <011> and <010> — directions of the load, and {011} and {010} — planes of test machine grips. The figures show atomic arrangements (with visualization of defects only), with exclusion of atoms in fcc lattice and atoms at the ends, which were inside the test machine grips and for which the tensile was performed. They are snapshots of the atom positions near crystalline defects (atoms displaced from the position of the ideal fcc lattice), that were calculated by the defect determination method based on the common neighbor analysis (CNA) [20]. Direction of tensile is from the center to the lower left corner and from the center to the upper right corner, the rate of loading — 80 m/s, the number of atoms — 0.25 million. The dislocation cores are presented in the form of gray atoms at the edge of the plane of atoms denoted by dark gray (red in the electronic version) colors. (*Note:* This and other anaglyph stereo figures give 3D visual representations of defect substructures, if anaglyph red-green glasses are used for viewing the color electronic version of the paper.)

Some perturbations in the form of point-like defects (like atom-vacancy states) can be observed in groups of 6 neighboring atoms (grey color). For high strains the nanocrystals of both

orientations deform by correlated restructures and displacements of large clusters of point-like defects in areas with large concentrations of defects. In general, the general monotonous softening can be observed up to "neck" formation and fracture (Fig. 2), because of plastic flow localization in the neck, i.e. appearance of collective (hydrodynamic) modes of deformation.

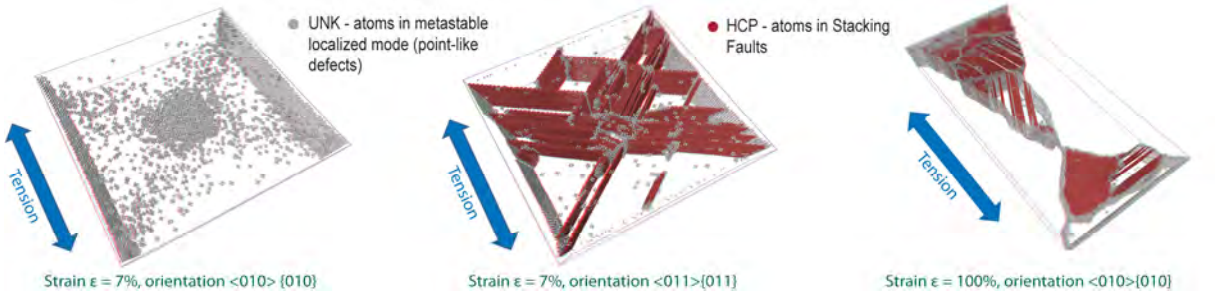
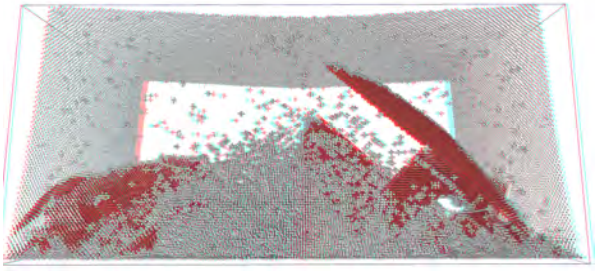


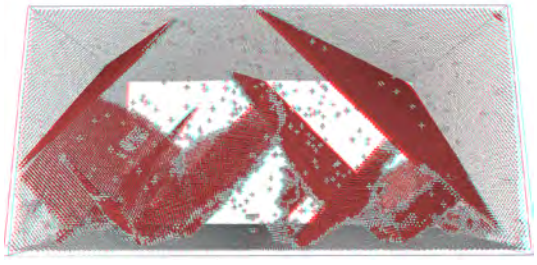
Figure 2: Tension of Al nanocrystal, strain $\varepsilon = 100\%$, orientation $<010>\{010\}$. (This and next figures are anaglyph stereo figures; please, see explanations in the text.)

3.2.Cyclic constrained tension of Al single crystals

For the low loading rate (40 m/s) (Fig. 3,*a*) defects have time to relax in dense complexes with the formation of dislocation sources distributed in the planes of easy slip (111). For the high loading rate (400 m/s) (Fig. 4,*a*) defects have *no* time to relax in dense systems, and form large spatial clusters of defects with non-compact (fractal) morphology [12-13]. It is more favorable for the higher concentration of point defects those do *not* relax even after return to the initial state (Fig. 4,*b*). These results confirm the experimental results, i.e. formation of non-crystallographic defect substructures, which were observed in Al single-crystal foils with $<010>\{010\}$ orientations under constrained tension [8-10], is the behavior of point defects and their ensembles, while other types of defects are revealed only at the stage of relaxation and unloading. It confirms the previous propositions to monitor the defect substructures *in situ* (for example, by their surface manifestations or using modern non-destructive methods), because the majority of defect substructures are observed *post factum*, i.e. after unloading and relaxation, that does not give a complete picture [21-24].

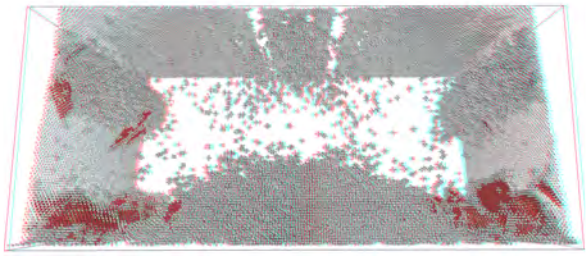


a)

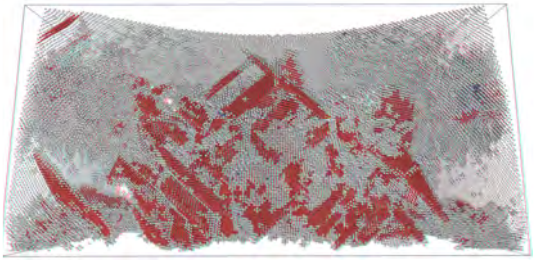


b)

Figure 3: Cyclic tension (strain rate 40 m/s, $<010>\{010\}$) for strains: a) 10%; b) 0%



a)



b)

Figure 4: Cyclic tension (strain rate 400 m/s, $<010>\{010\}$) for strains: a) 10%; b) 0%.

3.3.Unixaial tension of the ensemble of Al single crystals — statistical analysis

The results of MD simulations for plastic deformation of the big number of statistical realizations (i.e. many samples with identical conditions of strain, but different initial random values of atomic velocities) were analyzed. Distributions of some variables (stresses, concentrations of defects) were evaluated by the theory of extreme values [25]. It corresponds to the view on the localized plastic deformation, as a critical process with correlated behavior of some parts of the deformed crystal [26]. That is, the localized plastic deformation can take place in the areas linked by deformation events ("links" of the "chain" of localized strain), according to the model of "a chain with a weak link". [27]. The aim of the original approach was to carry out MD simulations of plastic deformation under the same conditions (except for the initial configuration of atomic velocities) for the very large number of samples (~10^3 different initial velocity distributions), and to analyze parameter distributions over the ensemble of statistical realizations. In the context of the current state of the MD simulation of plastic deformation, this approach is essentially new and has no analogues, and, moreover, it requires very large computing resources. In Fig. 5 an example of the statistical distribution over the statistical ensemble of >600 nanocrystals (Al single crystals with 1.5×10^5 atoms deformed with strain rate 200 m/s in $<010>\{010\}$ orientation) is shown for the concentration of atoms (in %) in atom-vacancy states with the unknown lattice type (UNK). Due to these massive character of MD simulations reliability of distribution fitting was possible to estimate by Kolmogorov-Smirnov test, moment and bootstrapping analysis on the Pearson diagram. They confirm the qualitative results about change of the behavior of the defect substructure from the unrelated state (from the normal distribution) to the correlated and linked one (to the zone of Weibull distributions), and about the change of the deformation mode: from uncorrelated motion of defects to correlated plastic flow [27].

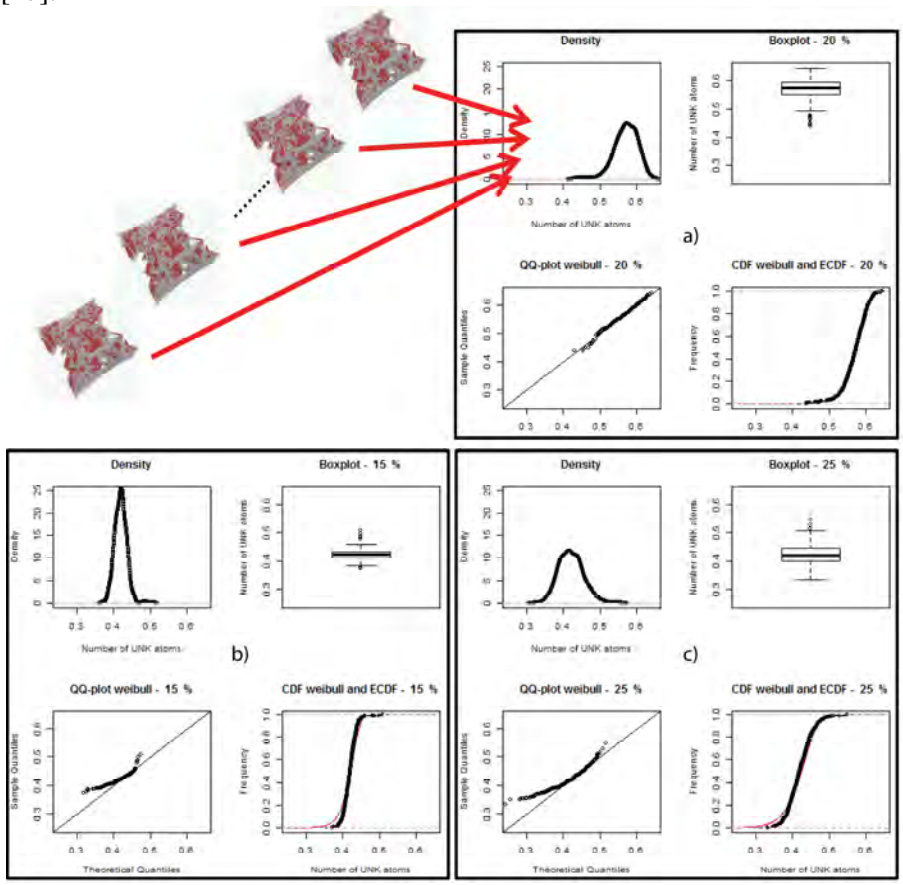


Figure 5: Probability density distribution plots ("Density"), boxplots, experimental cumulative density distribution plots ("ECDF") fitted to Weibull distributions in QQ-plots and CDF plots for defect concentrations in atom-vacancy states (UNK): a) $\varepsilon = 20\%$, b) $\varepsilon = 15\%$, c) $\varepsilon = 25\%$

3.4.Unixaial tension of graphene nanosamples

Simulation of various graphene nanosamples was performed in DCI *SLinCA@Home* by LAMMPSoverDCI application for several interatomic interaction potentials (including "Tersoff" and "Airebo", those are widely accepted for MD simulations in LAMMPS package [6]) (Fig. 6, left). The qualitatively different deformation scenarios were found, which are explained by the intrinsic differences in the potentials. Qualitative analysis shows that fragile fracture scenario occurs without formation of a stable defect substructure during deformation. The size effect was investigated for graphene plates from 2×2 to 2×32 nm (widht×length), and the size effect disappears for graphene plates with size >2×8 nm. For example, the first defect (vacancy) appears at the same level of strain; and fracture occurs at the same strain and stress amplitude (Fig. 6, right). The simulations of graphene nanocrystals are seem to be the most promising in DG-SG DCI due to their low demands to the computing resources, because they are predominantly 2D objects with the relatively low number of atoms for the big samples (>10^2×10^2 nm) even in comparison to the metal nanocrystals.

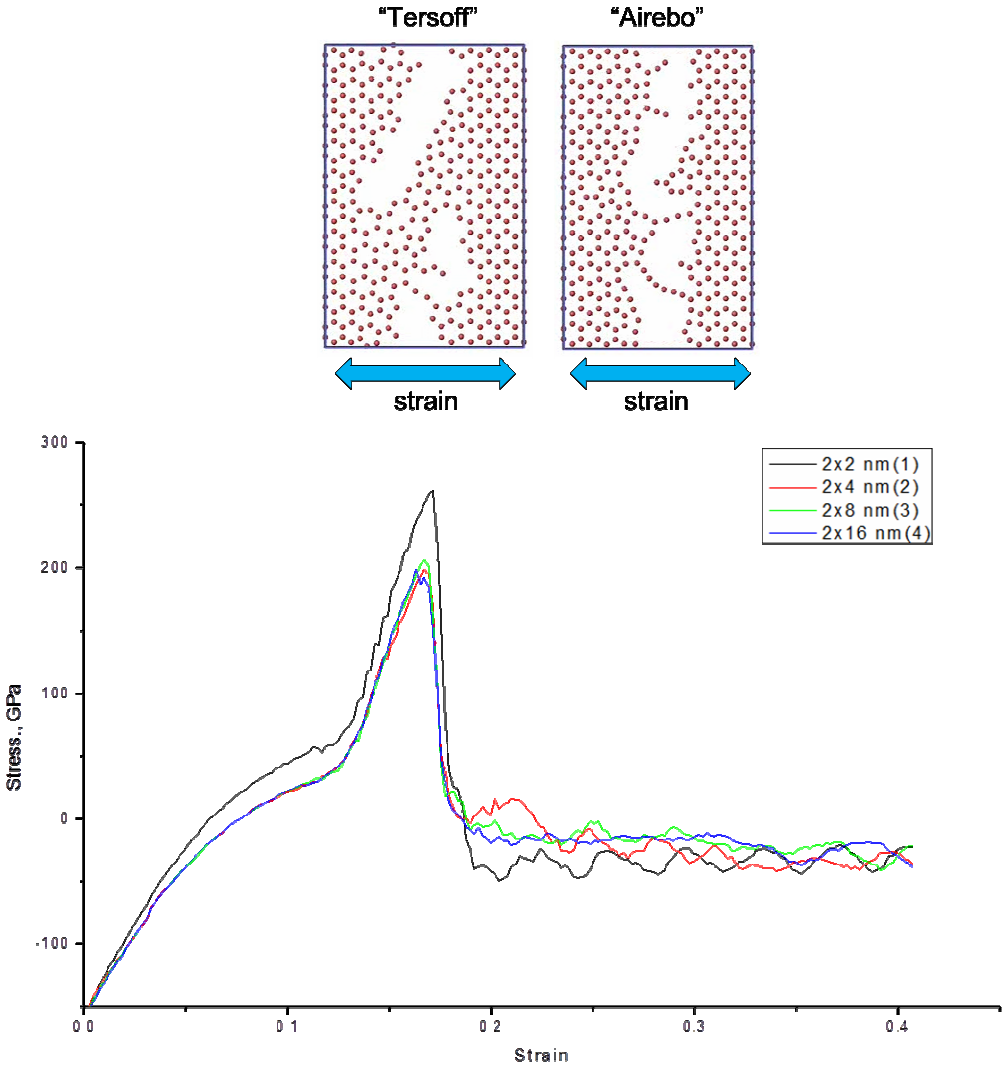


Figure 6: Fracture of graphene plates for two potentials (left); the stress-strain plot — the size effects disappears for graphene plates with size >8×2 nm (right)

4.Conclusions

The global DCI on the basis of Desktop Grids technologies for high-performance distributed computing can be effectively used for MD application of various objects: from metal nanocrystals to graphene nanoplates. For example, it allowed us to show that the mechanical characteristics evaluated

on the basis of MD simulations using LAMMPS package in the DG-SG DCI are in satisfactory agreement with the experimental data and allowed to discover the new aspects of deformation and fracture mechanisms in nanomaterials. The big perspectives can be foreseen for the usage of the DCI in combination with PGRADE platform that can hugely increase efficiency of scientific computations and simplify complexities of high-performance computing workflows for ordinary material scientists without a special background in computer science.

References

[1] Kacsuk P., Kovacs J., Farkas Z., Marosi A., Gombas G., Balaton Z., Sztaki Desktop Grid (SZDG): a flexible and scalable desktop grid system. *Journal of Grid Computing*, 7(4):439–461, 2009.

[2] Cappello F., Djilali S., Fedak G., Herault T., Magniette F., Neri V., Lodygensky O., Computing on large-scale distributed systems: XtremWeb architecture, programming models, security, tests and convergence with grid. *Future Generation Computer Systems*, 21(3):417–437, 2005.

[3] Cirne W., Brasileiro F., Andrade N., Costa L., Andrade A., Novaes R., Mowbray M., Labs of the world, unite!!! *Journal of Grid Computing*, 4(3):225–246, 2006.

[4] Urbah E., Kacsuk P., Farkas Z., Fedak G., Kecskemeti G., Lodygensky O., Marosi A., Balaton Z., Caillat G., Gombas G., others., EDGeS: Bridging EGEE to BOINC and XtremWeb. *Journal of Grid Computing*, 7(3):335–354, 2009.

[5] Kacsuk P. and Sipos G..Multi-Grid, Multi-User Workflows in the P-GRADE Grid Portal. *Journal of Grid Computing*, 3:3-4, 2005.

[6] Plimpton S., Fast parallel algorithms for short-range molecular dynamics. *Journal of Computational Physics*, 117(1):1–19, 1995.

[7] Baskova O., Gatsenko O., Gordienko Yu. Enabling high-performance distributed computing to e-science by integration of 4th generation language environments with desktop grid architecture and convergence with global computing grid. *Proc. of Cracow Grid Workshop* (CGW'10), 234--243, Cracow, Poland, 2011.

[8] Gatsenko O., Bekenev L., Pavlov E., Gordienko Yu., From Quantity to Quality: Massive Molecular Dynamics Simulation of Nanostructures under Plastic Deformation in Desktop and Service Grid Distributed Computing Infrastructure. *Computer Science Journal*, 2013 (accepted).

[9] Gordienko Yu., Kuznetsov P., Zasimchuk E., Gontareva R., Schreiber J., Karbovsky V., Multiscale 2D Rectangular and 3D Rhombic Gratings Created by Self-Organization of Crystal Structure Defects under Constrained Cyclic Deformation and Fracture, *Materials Science Forum*, 567, 421-424, 2008.

[10] Zasimchuck E., Gordienko Yu., Schreiber J., Sensing System For Determining The Fatigue on Metal Components. WO Patent WO/2011/098,079. 2011.

[11] Gordienko Yu., Zasimchuk E., Single-crystal indicators of fatigue and plastic deformation damage. *Proceedings of SPIE*, 2361:312, 1994.

[12] Gordienko Yu., Gontareva R., Zasimchuk E., Zasimchuk I., Fractal properties of the surface pattern of single crystals of aluminum under the conditions of their simultaneous loading with specimens of high alloys. *Metallofizika i Noveishie Tekhnologii*, 24:1561–1571, 2002.

[13] Zasimchuk E., Gordienko Yu., Gontareva R., Zasimchuk I., Equidimensional fractal maps for indirect estimation of deformation damage in nonuniform aircraft alloys. *J. Mater. Eng. Perf.*, 12(1):68–76, 2003.

[14] Kuznetsov P., Petrakova I., Gordienko Yu., Zasimchuk E., Karbovskii V., Formation of self-similar structures on {100}00l aluminum single-crystal foils under cyclic tension. *Physical Mesomechanics*, 12(1-2):85–93, 2009.

[15] Gordienko Yu., Zasimchuk E., Synergetic model of structure formation during plastic deformation of crystals. *Philosophical Magazine A*, 70(1):99–107, 1994.

[16] Gordienko Yu., Zasimchuk E., Simulation of building one-and two-dimensional structures on many scales in metals under load. *Systems Analysis Modelling Simulation*, 18:837–840, 1995.

[17] Gordienko Yu., Zasimchuk E., Dynamical phase transition in a lattice gas model with aggregation and self-organization. *Physica A*, 229(3-4):540–551, 1996.

[18] Gordienko Yu., Zasimchuk E., Metastable fractal aggregates as a result of competition between diffusion-limited aggregation and dissociation. In *Proc. of the 8th Joint EPS-APS Int. Conf. on Physics Computing: PC'96: September 17-21, 1996, Kraków, Poland*, page 293, 1996.

[19] Gordienko Yu., Generalized model of migration-driven aggregate growth-asymptotic distributions, power laws and apparent fractalitYu. *Int. J. Mod. Phys. B*, 26(1), 2012.

[20] Tsuzuki H., Branicio P., Rino J., Structural characterization of deformed crystals by analysis of common atomic neighborhood. *Computer physics communications*, 177(6):518–523, 2007.

[21] Gordienko Yu., Karuskevich M., Zasimchuk E., Forecasting the critical state of deformed crystal by analysis of smart defect structure: Fractal characteristics and percolation critical indexes. In *Proc. of Seventh Conf. on Sensors and Their Applications, Dublin, Ireland*, pages 387–392, 1995.

[22] Gordienko Yu., Zasimchuk E., Karuskevich M., Smart sensors for monitoring of fatigue damage and exhaustion of exploitation resource in intelligent transportation systems. *Progress in Technology*, 105:635–642, 2003.

[23] Gordienko Yu., E.E.Zasimchuk., R.G.Gontareva., V.Alexandrov., Extra dimensions by gif-animation: Industrial opportunities for online monitoring fatigue tests of metals in intranet and the web. *Int. J. Eng. Simulation*, 1(3):1–16, 2000.

[24] Zasimchuk E., Gordienko Yu., Gontareva R., Zasimchuk I., Sensor analysis of deformation damage in heterogeneous aircraft alloys. *Physical Mesomechanics*, 5(2):81–88, 2002.

[25] Cramer H., *Mathematical methods of statistics*, volume 9. Princeton Univ Pr, 1999.

[26] Rinaldi A., Peralta P., Friesen C., Sieradzki K., Sample-size effects in the yield behavior of nanocrystalline nickel. *Acta Materialia*, 56(3):511–517, 2008.

[27] Gordienko Yu., Molecular dynamics simulation of defect substructure evolution and mechanisms of plastic deformation in aluminum nanocrystals. *Metallofizika i Noveishie Tekhnologii (in Ukrainian)*, 33(9):1217–1247, 2011.

DEVELOPMENT OF THE VO-SPECIFIC dCache DATA BROWSER[1]

M. Gavrilenko, I. Gorbunov, V. Korenkov, D. Oleynik, A. Petrosyan, S. Shmatov

Joint Institute for Nuclear Research, Dubna

The Worldwide LHC Computing Grid (WLCG) [1] is a global collaboration of more than 140 computing centres in 35 countries, the 4 LHC experiments, and several national and international grid projects. The mission of the WLCG project is to build and maintain a data storage and analysis infrastructure for the entire high energy physics community that will use the Large Hadron Collider at CERN [2]. The LHC was built to help scientists to answer key unresolved questions in particle physics. E.g. : What is the origin of mass? Why do tiny particles weigh the amount they do? Why do some particles have no mass at all?

One of the 4 major experiments at LHC is CMS [3]. The CMS experiment uses a general-purpose detector to investigate a wide range of physics, including the search for the Higgs boson, extra dimensions, and particles that could make up dark matter. To have a good chance of producing a rare particle, such as a Higgs boson, a very large number of collisions is required. Most collision events in the detector are "soft" and do not produce interesting effects. The amount of raw data from each crossing is approximately 1 megabytes, which at the 40 MHz crossing rate would result in 40 terabytes of data a second, an amount that the experiment cannot hope to store or even process properly. The trigger system reduces the rate of interesting events down to a manageable 100 per second or 100 megabytes of data respectively. Nevertheless that's a huge amount of data to manage. To provides the data placement and the file transfer of the CMS experiment data, the PhEDEx project was established [4].

In CMS experiment for event data monitoring the CMS Dataset Bookkeeping System (DBS) is used [5]. DBS is a database and user API that indexes event-data data for the CMS Collaboration. The primary functionality is to provide cataloging by production and analysis operations and allow for data discovery by CMS physicists. Nevertheless, there is one major gap in data monitoring system of the CMS experiment at the level of storage elements (SE). There is an urgent need in tool for monitoring and managing of data on them. One of the most common SE types used in WLCG is dCache [6].

The dCache provides storing and retrieving huge amounts of data, distributed among a large number of heterogenous server nodes, under a single virtual filesystem tree with a variety of standard access methods. Depending on the Persistency Model, dCache provides methods for exchanging data with backend (tertiary) Storage Systems as well as space management, pool attraction, dataset replication, hot spot determination and recovery from disk or node failures. Connected to a tertiary storage system, the cache simulates unlimited direct access storage space. Data exchanges to and from the underlying HSM are performed automatically and invisibly to the user. Beside HEP specific protocols, data in dCache can be accessed via NFSv4.1 (pNFS) as well as through WebDav. dCache system is used by more than one third of the sites in WLCG. While dCache consists of various subsystems such as for example Location Manager [7], Name server (PNFS or Chimera) [8], gPlazma [9] authentication manager etc., the most important component for development of our project is the Chimera name server database.

The dCache is a distributed storage system, nevertheless it provides a single-rooted file system view. While dCache supports multiple namespace providers, Chimera is the recommended provider and is used by default. The inner dCache components talk to the namespace via a module called PnfsManager, which in turn communicates with the Chimera database using a thin Java layer, which in turn communicates directly with the Chimera database. Practically all important for monitoring

[1] The work is supported by the grant RFBR 10-07-00522-a and the Federal Project №07.524.12.4008.

information can be taken from Chimera database. For this work three component monitoring system were developed:

- database backup system (dumps original the Chimera database to a separate server),
- initial data processing (compute directory sizes, convert adjacency tree table structures to nested set etc.),
- Web interface access monitoring information.

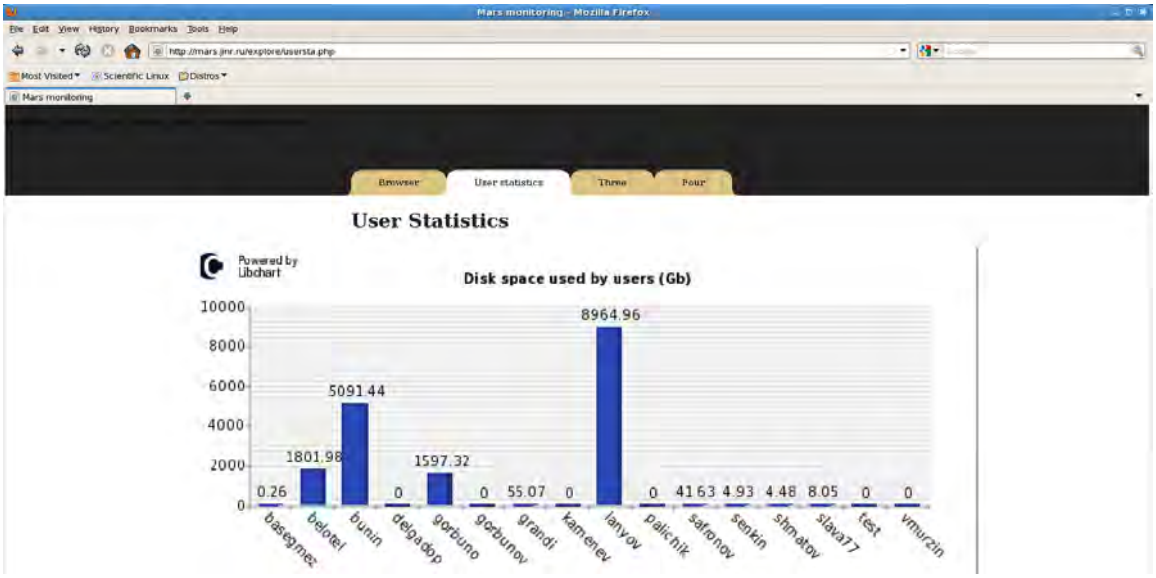
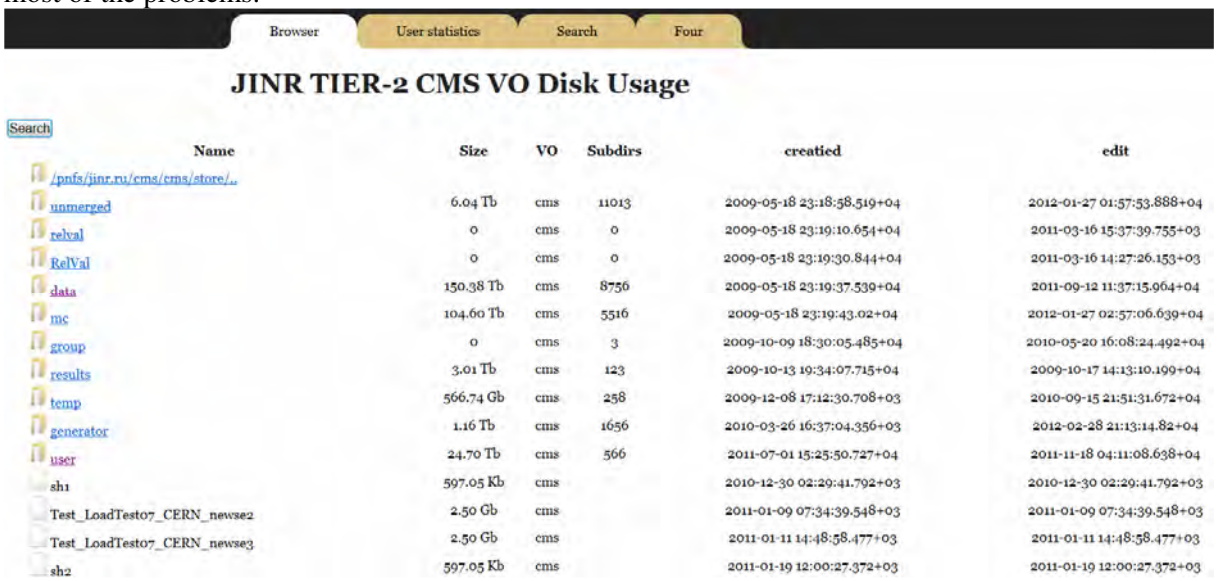


Fig. 1: Page with user statistics

It is worth mentioning that Chimera database uses an adjacency tree structure [10] which does not allow fast enough reading and requires big computational resources for a relatively simple procedures such as generating a list of child of a node and this situation is highly inappropriate for online monitoring. So first of all we convert the adjacency list model to nested sets [11] which solves most of the problems.



Name	Size	VO	Subdirs	creatied	edit
/pnfs/jinr.ru/cms/cms/store/..					
unmerged	6.04 Tb	cms	11013	2009-05-18 23:18:58.519+04	2012-01-27 01:57:53.888+04
relval	0	cms	0	2009-05-18 23:19:10.654+04	2011-03-16 15:37:39.755+03
RelVal	0	cms	0	2009-05-18 23:19:30.844+04	2011-03-16 14:27:26.153+03
data	150.38 Tb	cms	8756	2009-05-18 23:19:37.539+04	2011-09-12 11:37:15.964+04
mc	104.60 Tb	cms	5516	2009-05-18 23:19:43.02+04	2012-01-27 02:57:06.639+04
group	0	cms	3	2009-10-09 18:30:05.485+04	2010-05-20 16:08:24.492+04
results	3.01 Tb	cms	123	2009-10-13 19:34:07.715+04	2009-10-17 14:13:10.199+04
temp	566.74 Gb	cms	258	2009-12-08 17:12:30.708+03	2010-09-15 21:51:31.672+04
generator	1.16 Tb	cms	1656	2010-03-26 16:37:04.356+03	2012-02-28 21:13:14.82+04
user	24.70 Tb	cms	566	2011-07-01 15:25:50.727+04	2011-11-18 04:11:08.638+04
sh1	597.05 Kb	cms		2010-12-30 02:29:41.792+03	2010-12-30 02:29:41.792+03
Test_LoadTest07_CERN_newse2	2.50 Gb	cms		2011-01-09 07:34:39.548+03	2011-01-09 07:34:39.548+03
Test_LoadTest07_CERN_newse3	2.50 Gb	cms		2011-01-11 14:48:58.477+03	2011-01-11 14:48:58.477+03
sh2	597.05 Kb	cms		2011-01-19 12:00:27.372+03	2011-01-19 12:00:27.372+03

Fig. 2: Directory tree browser

The Web interface at the moment provides functionality to monitor disk space usage by users or by directory (Fig. 1) and allows browsing tree structure. While browsing the tree, there is information about directory or file size, grid virtual organization (VO) directory or file belongs to, and a number of subdirs in the directory available (Fig. 2). Also there is a built in search by files and directories available.

As soon as monitoring system and name server are physically separated, usage of the system does not effect dCache. Also the system dumps Chimera database so it can not lead to any mistakes in dCache functioning. Database dumps its information ones per day but this can be changed by request.

Data transfers between web server and client machine can be encrypted via enabling Transport Layer Security (TLS) at the web server' side. Besides, Apache web server can be configured to restrict access for users basing on user certificates. To enable security data transfers and authorization a host certificate was requested and installed on the server and TLS configuration was performed. Current configuration implies full prohibition of unsecured connect to the server without user certificate, confirmed by Russian Data Intensive Grid Certification Authority. The configuration enables to extract of user certificate credentials such as DN and passes them to web server's environment, which allows one to manage access to data through the UI.

The future plans for development include integration links to VO-specific outside data base and enabling file management from the web interface for VO-specific roles.

References

[1] http://lcg.web.cern.ch/lcg/
[2] http://public.web.cern.ch/public/
[3] http://press.web.cern.ch/public/en/LHC/CMS-en.html
[4] https://cmsweb.cern.ch/phedex/about.html
[5] http://cmsdbs.cern.ch/
[6] http://www.dcache.org/
[7] http://www.dcache.org/manuals/cells/docs/api/dmg/cells/services/LocationManager.html
[8] http://trac.dcache.org/projects/dcache/wiki/Chimera
[9] http://trac.dcache.org/projects/dcache/wiki/gPlasma
[10] http://xlinux.nist.gov/dads/HTML/adjacencyListRep.html
[11] http://explainextended.com/2009/09/24/adjacency-list-vs-nested-sets-postgresql/

CMS EXPERIMENT DATA PROCESSING AT RDMS CMS TIER 2 CENTERS[1]

V. Gavrilov[1], I. Golutvin[2], V. Korenkov[2], E. Tikhonenko[2], S. Shmatov[2], V. Zhiltsov[2], V. Ilyin[3,4], O. Kodolova[3], L. Levchuk[5]

[1]*Institute of Theoretical and Experimental Physics, Moscow, Russia*

[2]*Joint Institute for Nuclear Research, Dubna*

[3]*Skobeltsyn Institute of Nuclear Physics, Moscow State University, Moscow, Russia*

[4]*National Research Centre "Kurchatov Institute"*

[5]*National Science Center"Kharkov Institute of Physics and Technology", Kharkov, Ukraine*

1. Introduction

Russia and Dubna Member States (RDMS) CMS collaboration was founded in the year 1994 [1]. The RDMS CMS takes an active part in the Compact Muon Solenoid (CMS) Collaboration [2] at the Large Hadron Collider (LHC) [3] at CERN [4]. RDMS CMS Collaboration joins more than twenty institutes from Russia and Joint Institute for Nuclear Research (JINR) member states. RDMS scientists, engineers and technicians were actively participating in design, construction and commissioning of all CMS sub-detectors in forward regions. RDMS CMS physics program has been developed taking into account the essential role of these sub-detectors for the corresponding physical channels. RDMS scientists made large contribution for preparation of study QCD, Electroweak, Exotics, Heavy Ion and other physics at CMS. The overview of RDMS CMS physics tasks and RDMS CMS computing activities are presented in [5-11]. RDMS CMS computing support should satisfy the LHC data processing and analysis requirements at the running phase of the CMS experiment [12].

2. Current RDMS CMS Activities

During the last decade, a proper grid-infrastructure for CMS tasks has been created at the RDMS CMS institutes, in particular, at Institute for High Energy Physics (IHEP) in Protvino, Joint Institute for Nuclear Research (JINR) in Dubna, Institute for Theoretical and Experimental Physics (ITEP) in Moscow, Institute for Nuclear Research (INR) of the Russian Academy of Sciences (RAS) in Moscow, Skobetsyn Institute for Nuclear Physics (SINP) in Moscow, Petersburg Nuclear Physics Institute (PNPI) of RAS in Gatchina, P.N.Lebedev Physical Institute (LPI) in Moscow and National Scientific Center "Kharkov Institute of Physics and Technology" (NSC KIPT) in Kharkov. In the CMS global grid-infrastructure these RDMS CMS sites operate as CMS centers of the 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.

Since 2012 year, the CMS basic requirements to the CMS "nominal" Tier2 grid-site are:

- persons responsible for site operation at each CMS Tier-2 site;
- site visibility in the WLCG global grid-infrastructure (BDII);
- availability of recent actual versions of CMS Collaboration software (CMSSW);
- high efficiency of regular file transfer tests;
- certified links with CMS Tier-1 and Tier-2 grid-sites;
- regular CMS Hammer Cloud (HC) tests;
- 9.8 kHS06 of Processing Resources;
- disk space of 620 TB for: 30 TB of stage-out space; 200 TB of group space (100 TB per

[1] These activities are partially supported by a grant of the Russian Foundation for Basic Research (RFBR) and the Ukrainian Academy of Science (12-07-90402-Ukr_a)

group), 150 TB of central space, 120 TB of regional space and 120 TB of user space (~40 users of 3 TB each).

The integrated RDMS Computing infrastructure as a whole completely satisfies these requirements.

The RDMS CMS computing model provides a valuable participation of RDMS physicists in processing and analysis of CMS data. Since 2008, the RDMS Tier-2 centers have been associated with CMS Exotics Physics Analysis Group and CMS Muon Physics Object Group (both groups hosted at the JINR site), CMS Heavy Ion Physics Analysis Group (hosted at the MSU site) and JetMet/HCAL Physics Object Group (hosted at the ITEP). Some later the KIPT site was associated with CMS Electroweak Analysis Group. The special tests shown that the RDMS Tier-2 sites are satisfied all requirements for such 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 processing and analysis of experimental data provided by the LHC and for simulation.

By the spring of the year 2011 CMS Tier-2 sites (computing centers) were considered in the context of the CMS computing requirements as "ready" for the data-taking 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 HC efficiency > 90%;
- commissioned links TO Tier-1 sites ≥ 2;
- commissioned links FROM Tier-1 sites ≥ 4.

The status of readiness of RDMS CMS Tier-2 sites in October, 2012 is shown on Fig.1.

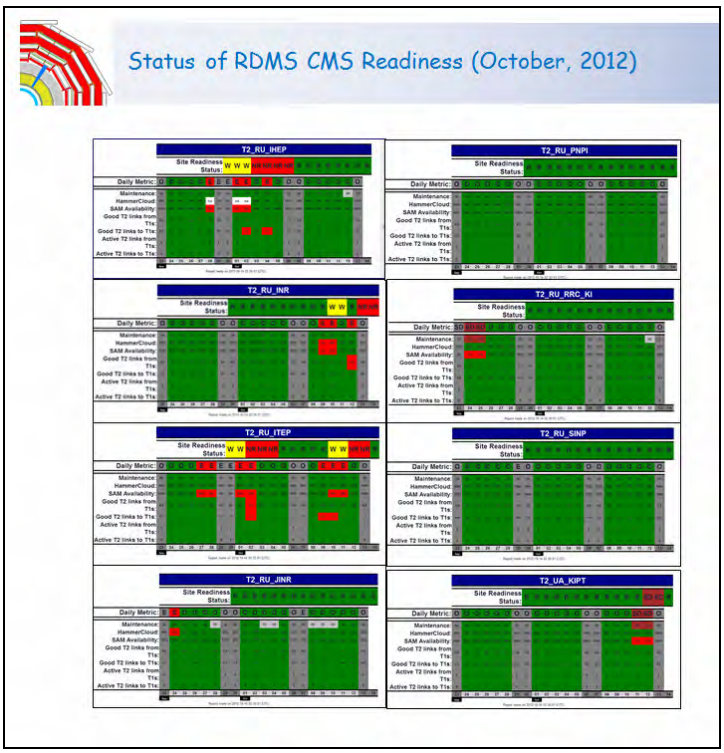


Fig.1: Readiness of RDMS CMS Tier-2 sites. October, 2012 (the permanent updated link see here *http://lhcweb.pic.es/cms/SiteReadinessReports/SiteReadinessReport.htm*)

More than 560 TB were transferred to the RDMS Tier-2's from November 2011 to October 2012 (Fig.2). The maximum transfer rate to RDMS Tier-2 was more than 100 MB/s (Fig.3).

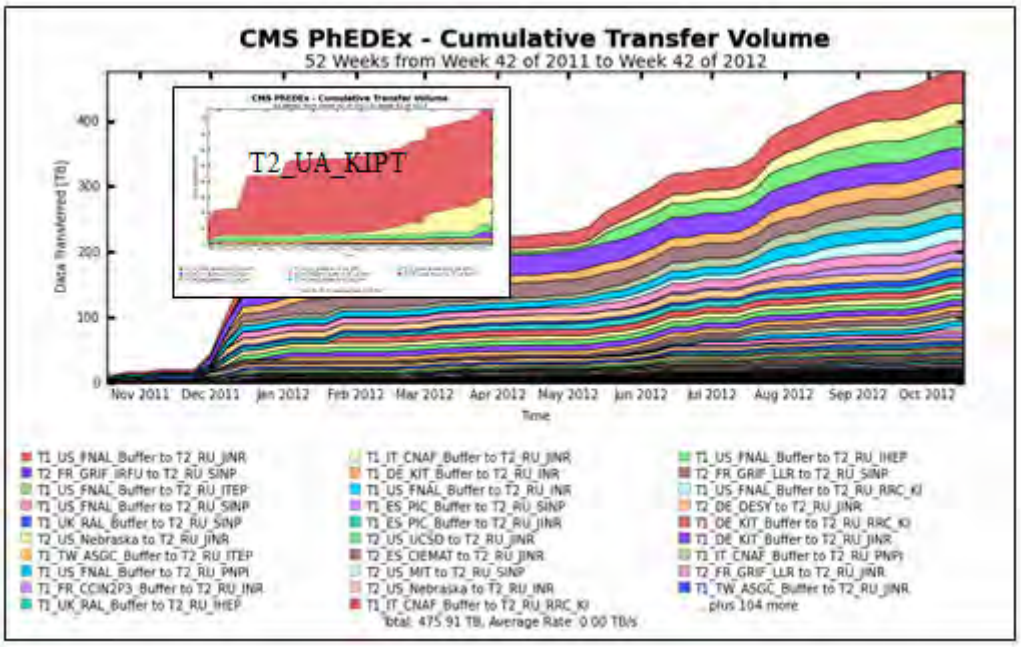


Fig.2: The cumulative transfer volume for the RDMS T2-sites from November, 2011 to October, 2012

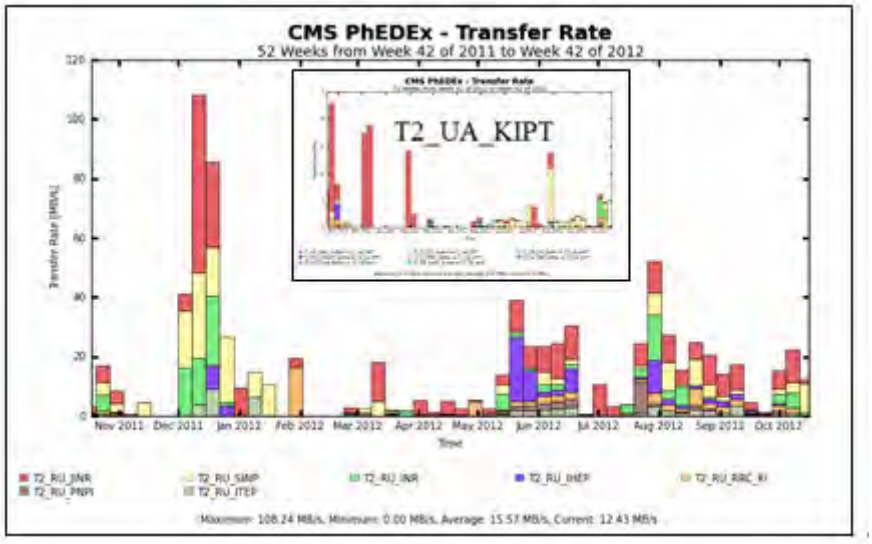


Fig.3: Transfer rates (more than 100 MB/s) at the RDMS Tier-2 sites during November, 2011 – October, 2012

The RDMS CMS Tier-2 sites are actively used by the CMS collaboration: 2,178,260 jobs (more than 7×10^9 events, see. Fig.4) of the CMS virtual organization were processed on the RDMS CMS Tier-2 sites in 2012.

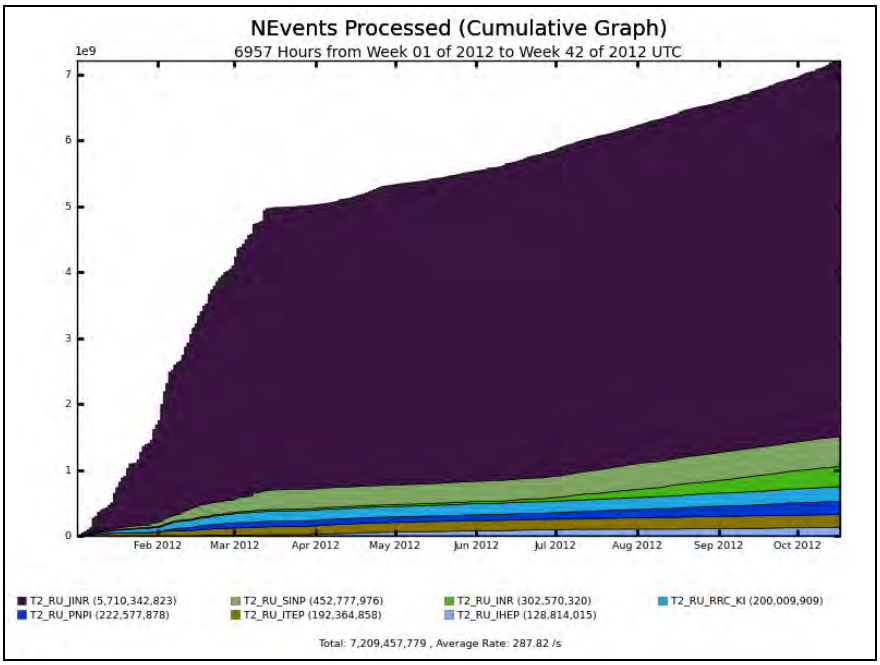


Fig.4: The number of processed events in 2012 with the RDMS CMS Tier-2 sites

In line with the CMS computing requirements for the data-taking phase of the experiment, now the RDMS CMS grid-sites provide:

- the computing and data storage resources in full;
- centralized deployment of actual versions of CMS specialized software (CMSSW);
- data transfers between the CMS grid-sites with the usage of the FTS grid-service on basis of VOBOX grid-services for CMS with the Phedex Server;
- 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 Hammer Cloud 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 the 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).

The dedicated CMS remote worldwide-distributed centers (ROC) were built in different scientific organization [13]. 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 JINR CMS ROC is designed as 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 also at the SINP MSU to provide similar functions for CMS participants in Moscow.

RDMS CMS physicists work in the WLCG environment, and now we are having more than 30 members of the CMS Virtual Organization.

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 the last two years a significant modernization of the RDMS CMS grid-sites has been accomplished. As result, 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. Regular testing of the RDMS CMS computing centers functionality as grid-sites is provided.

All the necessary conditions for 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] V. Matveev, I. Golutvin, "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; http://rdms-cms.jinr.ru

[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] V. Gavrilov et al.,"RDMS CMS Computing Model", in Proc. of the Int.Conference "Distributed Computing and Grid-Technologies in Science and Education", Dubna, 2004, pp.240-247.

[6] V. Gavrilov et al.,"RDMS CMS Computing", in Proc. of the 2nd Int.Conference "Distributed Computing and Grid-Technologies in Science and Education", Dubna, 2006, pp.61-66.

[7] D.A Oleinik et al., RDMS - CMS Data Bases: Current Status, Development and Plans, in Proc.of the XXth Int. Symposium on Nuclear Electronics and Computing, JINR, Dubna, 2006, pp.216-221.

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

[9] D.A. Oleinik at al.,Development of the CMS Databases and interfaces for CMS experiment, in Proc. of XXI Int. Symp. on Nuclear Electronics & Computing (NEC`2007), Dubna, 2008, pp.376-381.

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

[11] V. Gavrilov et al., RDMS CMS data processing and analysis workflow, in Proc. of XXIII Int. Symp. on Nuclear Electronics & Computing (NEC`2011), Dubna, 2011, pp.148-153.

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

[13] A.O. Golunov et al, "The JINR CMS Remote Operation Centre", Distributed Computing and Grid-technologies in Science and Education IV Int.Conference, Proceedings of the conference, Dubna, 2010, pp.109-113.

ARCHITECTURE OF A SOA-BASED BPM PLATFORM FOR THE EGI

R.D. Goranova
Faculty of Mathematics and Informatics, University of Sofia "St. Kliment Ohridski",
5 James Baucher, 1164 Sofia, Bulgaria
radoslava@fmi.uni-sofia.bg

The SOA-based BPM platforms provide capabilities for business process modeling, execution, monitoring and optimization through the support of the Web services standards. Features of the platforms are process flexibility, easy integration and reuse of the assets from the platform. The European Grid Infrastructure (EGI) uses partially service-oriented grid middleware for grid computing (g-Lite). In this context applying the SOA-based model of the BPM platforms for the EGI will improve the development of flexible service-oriented solutions and will provide a framework for business process management in the grid infrastructure. In this article we propose architecture of platform that supports the management of business processes in the EGI. The main modules and components of the platform are presented.

1. Introduction

The Service-oriented architecture (SOA) [1] is an architectural style for developing systems and applications. Basic feature of the SOA is the service - well-defined logical entities, which can be independently used. Realization of the SOA model is Web Service Architecture (WSA) [2]. The WSA specifies and defines a set of protocols and standards for Web services' transport (HTTP), messaging (SOAP), description (WSDL), service discovery (UDDI) and composition (BPEL).

The Business process management (BPM) [3] provides methods, techniques, and software for design, enacts, control, and analyze of business processes. Combining SOA with BPM will benefit the software platforms with features like process flexibility, easy integration and reuse of existing assets. Something more, SOA-based BPM platforms provide capabilities for business process modeling, execution, monitoring and optimization through the support of the Web services standards. These features are important, because they make the software platform extendable and interoperable. The basic characteristics of the SOA-based BPM platforms can be summarized as follows:

- Support of the standard WSDL - for Web service description;
- Support of the standard BPEL - for Web service composition;
- Support of the standard UDDI – for Web service reuse and discovery;
- Support of adapters - for access to other systems;
- Support of the Enterprise Service Bus (ESB) - infrastructure for messages exchange;
- Support of Human Tasks (HT) – for interactions between human and the business process;
- Support of Business Rules (BR) management – for management of the basic rules for the business processes;
- Support of Business process monitoring (BM) – for monitoring and optimization of the existing business processes.

Example realization of SOA-based BPM platforms are IBM SOA Foundations [4], Oracle SOA Suite [5] and JBoss Enterprise SOA Platform [6].

The European Grid Infrastructure [7] (EGI) uses partially service-oriented grid middleware for grid computing (g-Lite) [8]. In the context of SOA, the business process management, the definition, monitoring and optimization of a business process in the infrastructure are still not supported. In [9] the author discusses the important aspects of service-orientated grids and underlines the lack of widely accepted mechanisms for business process orchestration, mediation and monitoring in it. G-Lite is not an exception. Basic indicators for that are:

- Lack of service registry or service registry support (UDDI is not supported);
- Lack of discovery services;

- Lack of service for composition (BPEL is not supported);
- Lack of well-defined Web services' descriptions (WSDL is not fully supported).

In this context applying the SOA-based BPM platforms for the EGI will improve the development of flexible service-oriented solutions and will provide a framework for business process management in the grid infrastructure. Something more, well known fact is that the service-oriented approach is the preferred approach for building cloud systems – the next grid generation. In [10] the authors proposed such service-oriented generic resource framework for cloud systems, which represents datacenter resources in a uniform way, allowing generic administration without knowledge of the underlying resource access protocol. Therefore to use SOA approach in the EGI will be an advantage.

The grid middleware g-Lite is improved constantly. Some of the improvements are in the reliability of the provided services and other in the quality of the services. Example improvements in this direction are presented in [11], where the authors present architecture of service for job tracking into the grid environment. In [12] the authors present a grid application. Interesting fact is that the presented application is based on the message-oriented middleware, which is an infrastructure for message exchange. The equivalent of such infrastructure in the SOA is the Enterprise service bus.

In [13] we overview some tools which can be used for building and executing service compositions in the EGI. Our choise of the tools is influenced and from [14], where the same tools are described as grid tools for monitoring and control of the workflow execution. We compare these tools according to their service-orientation. The conclusion is that, all of the presented tools use their one mechanism for service composition, independently from g-Lite and none of them cover the characteristics of the SOA-based BPM platforms, mentioned above.

All this observations motivated us for the current research. In this article we propose architecture of a SOA-based platform which supports the management of business processes in the EGI. This includes business process definition, business process design and business process execution in the EGI.

2. Component model of a SOA-based BPM platform for the EGI

In [15] we presented an idea for a framework for service-composition in the EGI. The presented SOA-based BPM platform in this article is software implementation of the framework. The component model of the platform is presented on the Figure 1.

The presented SOA-based BPM platform for the EGI consists of five layers. On the first layer of the model are all the legacy EGI services and applications. This layer covers already built it grid infrastructure that includes all the EGI services and applications, which are available. Because of the legacy the services can not be changed. Examples for legacy EGI services are the Storage Element (SE, DPM, d-Cache, etc.), Computing Element (lcg-CE, CREAM, etc.) of the grid environment. Example for EGI application is ROOT [16].

On the second layer are all of the developed Web services for access to the EGI services and application. Example Web services are the Web services for job submission, the Web services for grid proxy creation, etc. The Web services are important part form the framework. They participate into compositions in the higher layers of the framework.

On the third layer are the registry services. They provide features for publishing and discovery of the developed Web services from the second layer. In [17] the authors present comparison of grid resource discovery approaches. Based on the functional requirements that they defined, we can conclude that, the UDDI approach is the most appropriate approach for grid resource discovery with respect to the defined criterions. As we mentioned earlier, one of the basics characteristics of the SOA-based BPM system is exactly support of the UDDI standard. Therefore the support of the UDDI standard in the platform is desirable.

On the last two layers are modules for business process design and execution. In the development module the business process is modeled, by using the existing Web services from the second layer. The designed business process is deployed into the runtime environment module, where

the business process is executed. As the Web services represent functionality from the grid infrastructure, the business process represents the execution of the service composition in the EGI.

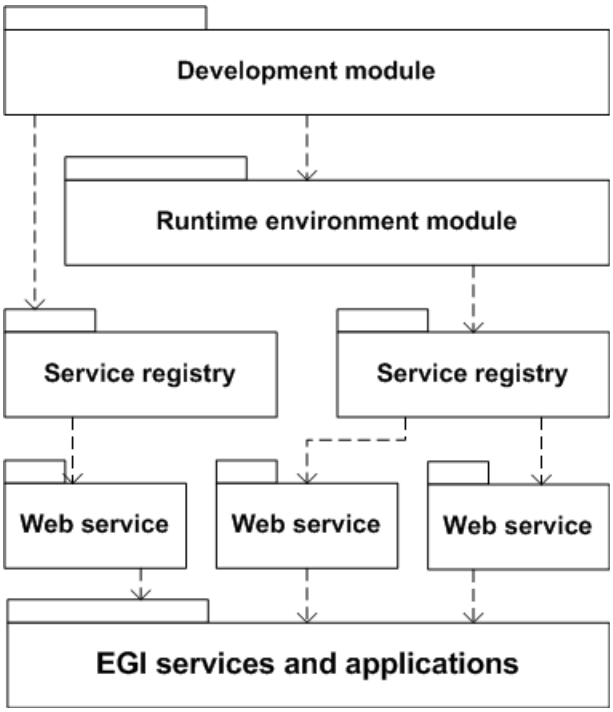


Figure 1: Component model of a SOA-based BPM platform for the EGI

The presented SOA-based BPM platform consists of three basic modules: Development module, Runtime environment module and Service registry module. As a representation of a model of the platform for the grid, the Runtime environment module has to be a part from the grid infrastructure. On the Figure 2 we present in details the basic modules of the platform with respect to their place into the grid infrastructure.

The Development module provides functionality for business process design, development and deployment. The Development module consists of two tools: Design module and Deployment module. Through the Design module the grid user can compose own business processes by choosing already developed Web services into the flow of the process. Through the Deployment module the grid user can deploy the designed business processes into the Runtime environment or to customize some of the steps for the business processes. The Deployment module can be used from grid developers for development of Web services for access to the EGI, for development of the monitoring model for the business processes or for generating graphical user interfaces for access to the business processes. Both the Design and the Deployment module are usually installed outside the grid on the users' computer.

The Service registry module is a registry for the developed web services and business processes. The module is also outside the grid and can be accessed from everyone. Nevertheless invoking Web service for access to the EGI from the registry require grid certificate.

The Runtime environment module includes service for business process execution (BP service), service for business rules management (BR service), service for business process monitoring (BM service) and service for human task management (HT service). All of the fourth services used common infrastructure for message exchange – the Enterprise service bus (ESB).

The BP service interprets the BPEL description if the business process and invoke and execute the Web services in it. The BP service also checks the business rules associated with the process. The BM service allows monitoring of the business process on key performance indicators during the execution. The key performance indicators are set, when the business process is designed.

The BR service provides functionality for changing business rules in runtime. The HT service handles human task management. Human tasks are three types: interactions from the type 'human - business process', interactions from the type business process – human' and interactions from the type 'human-human'. Example for human task is when the user initiates business process. The communication between all the services: the BP service, the BM service, the BR service and HT service is through the ESB.

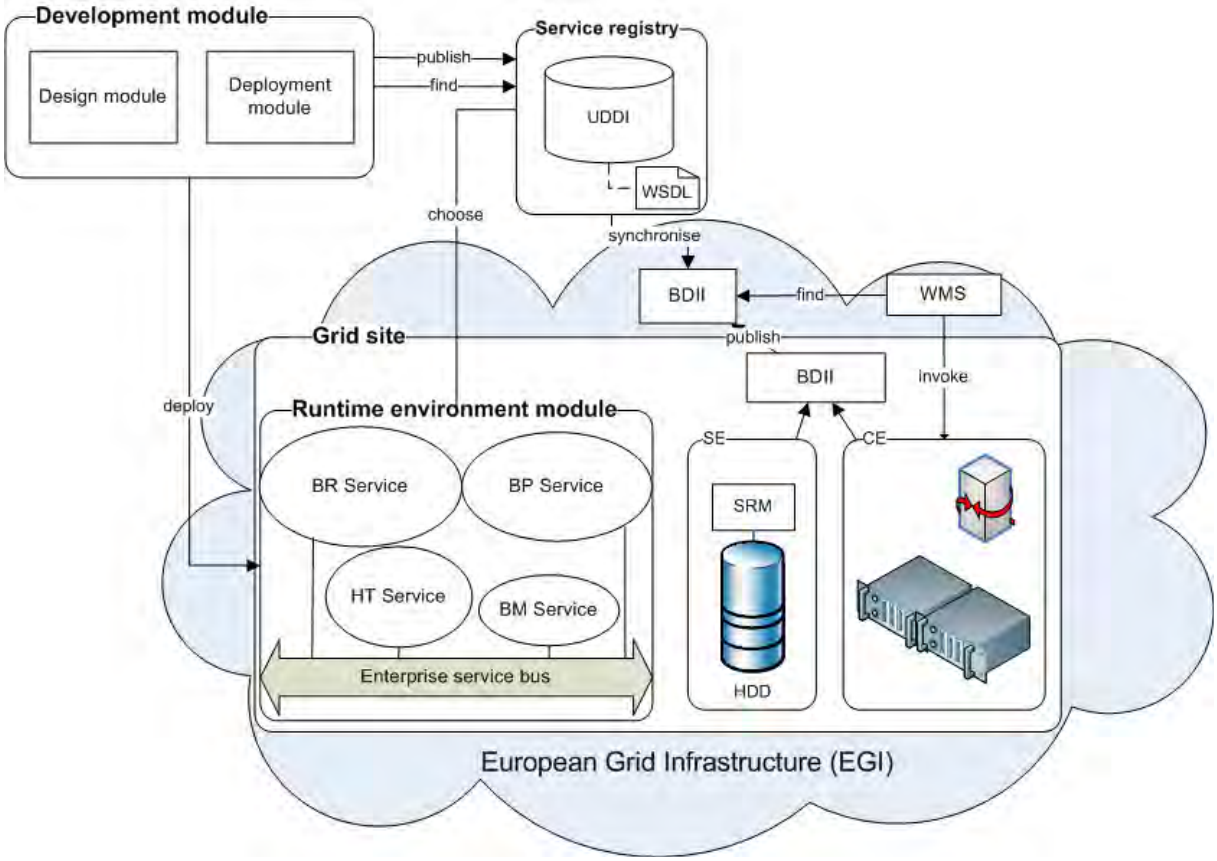


Figure 2: Modules and components of the platform with respect to their place into the EGI

The Runtime module consists of set of services. These services are physically distributed, but logically accessed through the same name. All of the services from the Runtime module are registered into the Service registry module. They are accessible as any other service from the registry. Unfortunately, the current realization of the EGI does not allow the Runtime environment module to be a part form the grid. In the next section we present some improvements to the presented model, which make it applicable for the EGI.

3. Architecture of a SOA-based BPM Platform applicable for the EGI

The presented SOA-based BPM platform relies on layer of existing Web services for access to the EGI. Unfortunately, the current realization of the g-Lite grid environment does not provide such Web services. If we conclude the limitations for realization of the above model are:

- The EGI is not service-oriented. Not all EGI services are service-oriented;
- The EGI does not provide service for composition. The grid environment g-Lite does not provide mechanism for services composition;
- The EGI can not be extended with ESB and the adjoining runtime services.

Because of this, an additional module was added to the architecture. The module provides adapters for access to the EGI services and applications and exposed them as Web services. The new module

represents the second layer of the presented architecture. The current implementation of the module provides Web services for access to g-Lite [18] and Web services for access to ROOT application [19]. The developed Web services are composable. The module is extensible. If other Web services are needed they can be developed and deployed into the module.

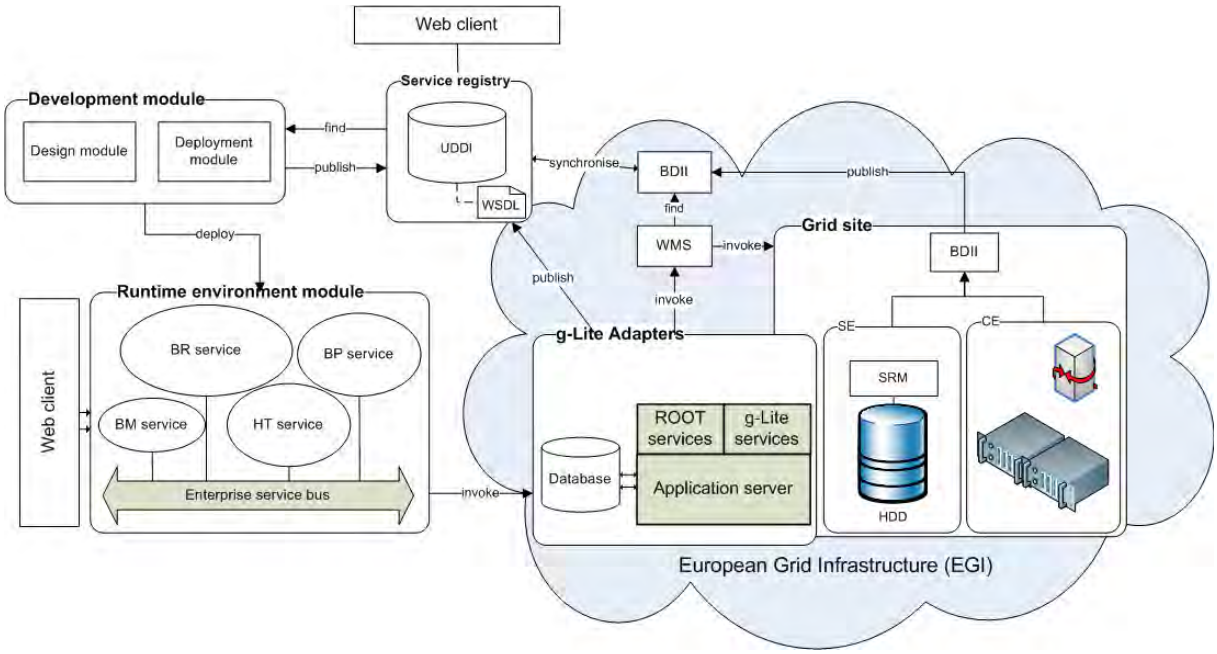


Figure 3: Additional module into the platform and its place into the EGI

On the Figure 4 we present example architecture of the SOA-based BPM Platform for the EGI.

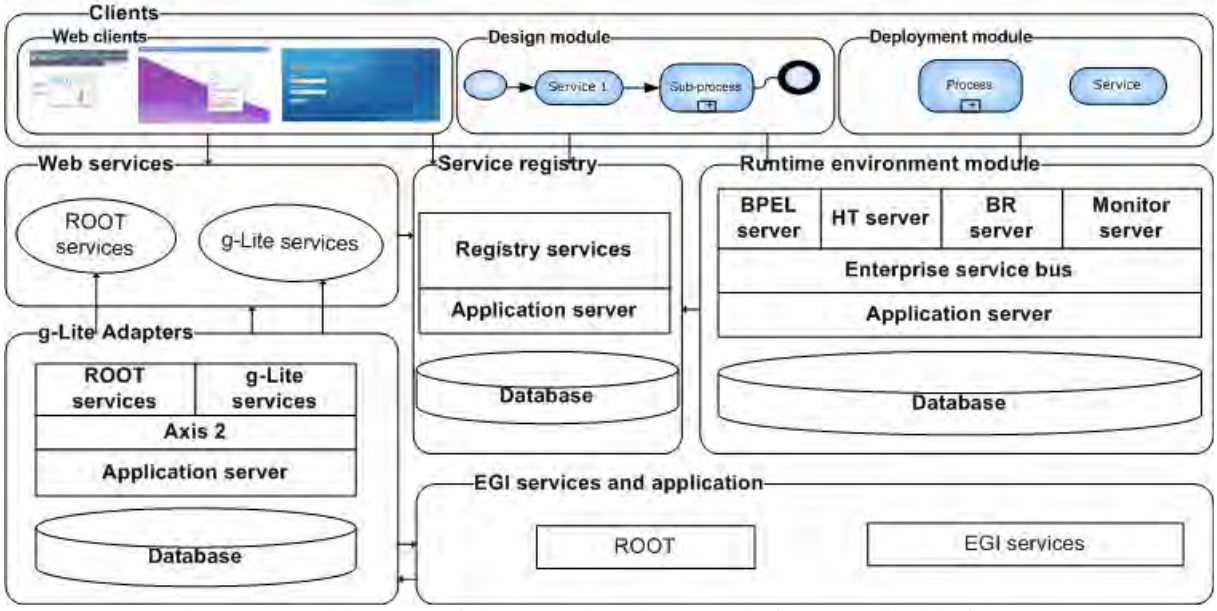


Figure 4: Architecture of the SOA-based BPM platform applicable for the EGI

Conclusion

The presented SOA-based BPM platforms cover all of the necessary criterions for business process composition and execution. In this article we present architecture of the platform applicable for the EGI. Features of the provided platform are: Web services support, BPEL support, registry support, HT support, BM support and BR support.

References

[1] T. Erl, Service-Oriented Architecture: Concepts, Technology, and Design, Prentice Hall Publishers, 2005.

[2] Web Service Architecture, http://www.w3.org/TR/ws-arch/

[3] Van der Aalst, M., at all., Weske, M. "Business Process Management: A Survey“, 2003, http://bpt.hpi.uni-potsdam.de/pub/Public/PaperArchive/bpm2003.pdf

[4] IBM SOA Foundations – WebSphere Software V7.0, http://www-01.ibm.com/software/solutions/soa/offerings.html

[5] Oracle SOA Suite 11g, http://www.oracle.com/us/technologies/soa/soa-suite/index.html

[6] JBoss Enterprise SOA Platform, http://www.jboss.com/products/platforms/soa/

[7] European Grid Infrastructure, http://www.egi.eu/

[8] g-Lite, http://glite.cern.ch/

[9] Dimitrov, V. T., Development of applications with service-oriented architecture for grid, ACM New York, 2008, Proceedings of the 9th International Conference on Computer Systems and Technologies (CompSysTech '08), Article No.14.

[10] Zhelev R. and V. Georgiev. A Generic Resource Framework for Cloud Systems, Proceedings of The 4th International Conference on Distributed Computing and Grid-technologies in Science and Education, June 28 - July 3, 2010 Dubna, Russia., pp. 268 – 278.

[11] Atanassov, E., Gurov, T., Dimitrov, D., "Job Track Service: Architecture and Features", SEE-GRID-SCI User Forum, 6-11 December 2009, Istanbul, Turkey, pp. 199-202, 2009.

[12] Atanassov, E., Gurov, T., Dimitrov, D., "Salute Grid Application using Message Oriented Middleware", AIP Conference Proceedings, vol. 1186, pp. 183-191, 2009.

[13] Goranova R. D., Service composition tools in g-Lite, Conference Proceedings of the Fifth International Conference ISGT, 2011, pp. 228-235.

[14] L. Kirchev, V. Georgiev and K. Boyanov, Workflow Management for a General Purpose Grid Platform of Commodity Computers, in Proceedings of the International Workshop on Network and GRID Infrastructures, Sofia, Bulgaria, 27-28 September, 2007. pp. 42 – 50.

[15] Goranova, R. D., Framework for service composition in g-Lite, American Institute of Physics, Conference Proceedings Volume 1404, 2011, pp. 218-224, ISBN 978-0-7354-0976-7, ISSN 0094-243X.

[16] ROOT, http://root.cern.ch/drupal/

[17] Pashov G., K. Kaloyanova, Comparison of Grid Resource Discovery Approaches, Third International Conference on Information Systems & Grid Technologies, 28 - 29 May 2009, Sofia, Bulgaria, pp. 138-147.

[18] Goranova, R., Goranov, G. Web Service Module for Access to g-Lite, American Institute of Physics, Proceedings of 4-th Conference AMITANS, 2012 (to appear).

[19] Goranova, R. D., Development of ROOT Services for Grid, American Institute of Physics, Conference Proceedings Volume 1301, 2010, pp. 661-668.

MODEL OF DATA STORAGE AND PROCESSING SYSTEM FOR "PIK" NUCLEAR REACTOR EXPERIMENTS[1]

A.P. Gulin, A.K. Kiryanov, N.V. Klopov,
E.G. Novodvorsky, S.B. Oleshko, Y.F. Ryabov
Petersburg Nuclear Physics Institute, 188300, Gatchina, Orlova Roscha, Russia
globus@pnpi.nw.ru

A new high-flux nuclear reactor PIK [1] is to be launched soon at Petersburg Nuclear Physics Institute, providing a foundation for new physics experiments. It also opens a way for international collaboration with other nuclear research centers. Parameters of its neutron beams and its experimental capabilities are unmatched, the only close alternative is a 58 MW HFR reactor at Institute Laue-Langevin (ILL), Grenoble.

Because of more than 50 planned experimental installations and various research teams, it is crucial to provide a uniform, handy and secure way to store, process and access data coming from experiments. In today's science world Grid technologies have gained a steady recognition as a way of connecting research centers around the world and providing distributed storage and computing resources for scientific collaborations.

A functional model for data storage and processing is being developed at PNPI (fig.1). It features a hierarchical schema for metadata covering all aspects of physics experiment and role-based access control. Each research study starts with a root object called "investigation" that has a team of people associated with it and represented in an access-control list (ACL). As study goes on, these people add child objects such as samples, datasets, software runs, etc. Datasets may have their own ACLs while other objects inherit one of their parent investigation (fig. 2).

Data storage model allows individual files of dataset to be placed on file servers, replicated and accessed via common protocols (fig. 3). It also features support for different types of file storages (e.g. SRM) and file integrity protection with checksums.

Data processing model allows distributed data processing on both local and remote Grid-enabled resources using standard Grid authorization and authentication mechanisms.

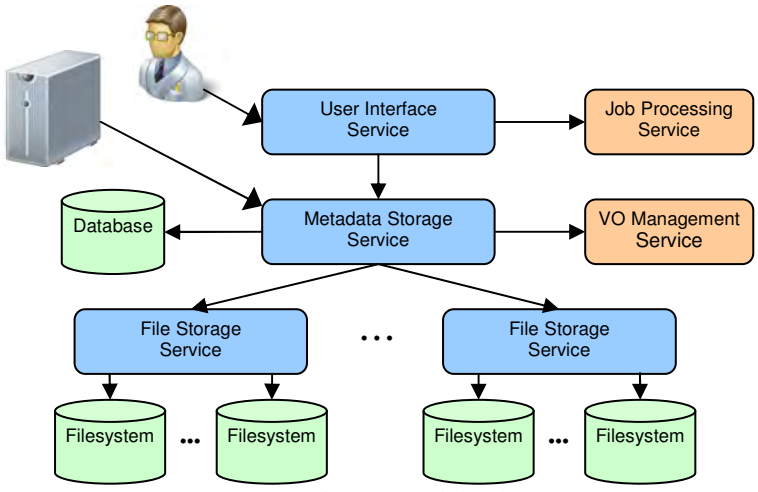


Fig. 1: Model schema

[1] This work was performed within the Federal Special Scientific and Technical Program (Project No. 07.514.12.4003).

Reference implementation consists of the following: metadata server engine with database, fileserver management engine, command line tools, application libraries and web-based user interface for metadata access and data processing. This implementation is intended for Linux distributions such as CentOS and Scientific Linux and widely uses standard tools and protocols:

- X509 certificates are used for user and inter-server authentication;
- HTTPS is used for most communications. Along with it, GridFTP is also available for file access;
- JSON data format is used for object data exchange;
- users and user groups are imported from VOMS server, which is used as a central point for user registration;
- CREAM is used as an interface to computing resources, allowing resource sharing via Grid.

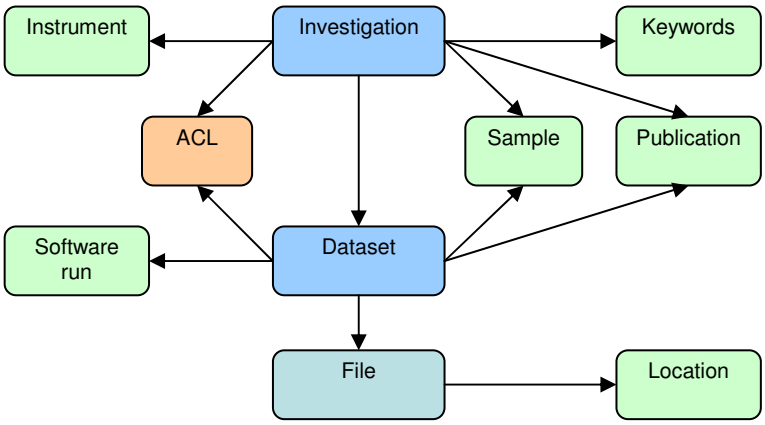


Fig. 2: Metadata model

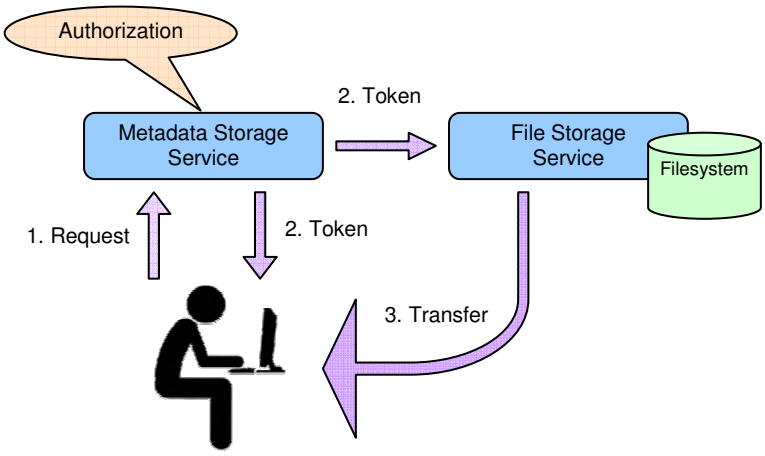


Fig. 3: Data storage model

User interface serves as a single access point for the user. It allows performing all kind of metadata management and file operations as well as job submission and control (fig. 4). Being web-oriented, user interface can be accessed from virtually everywhere, including portable devices. Two languages, Russian and English, are currently available. More may be added by minimal effort.

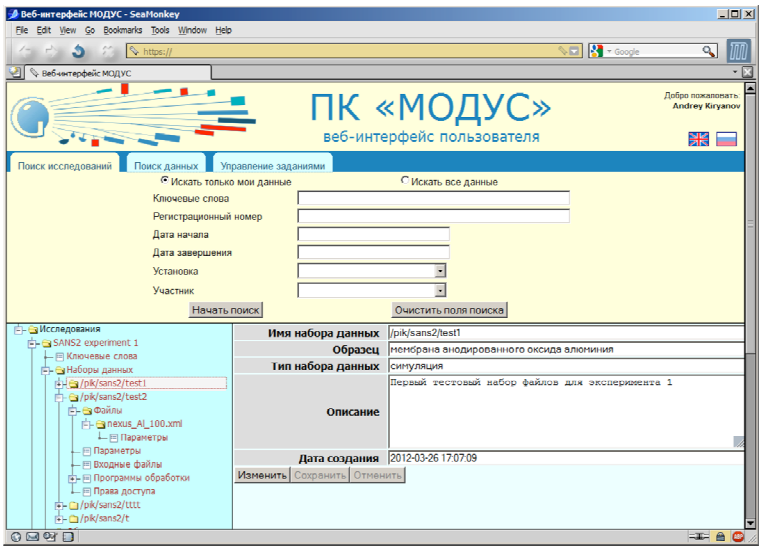


Fig. 4: User interface

Software simulators are used for testing purposes and validation. They produce data streams as real experiment workstations would do, thus allowing to simulate a full cycle of data acquisition and processing in conditions closest to real-life.

References

[1] http://nrd.pnpi.spb.ru/facilities/menu_pik.html

ON APPROACHES TO BUILDING PROBLEM-ORIENTED WEB-INTERFACES FOR APPLICATION SOFTWARE SUITES IN GRIDNNN

A.P. Gulin, A.K. Kiryanov, N.V. Klopov, S.B. Oleshko, Y.F. Ryabov
Petersburg Nuclear Physics Institute, 188300, Gatchina, Orlova Roscha, Russia
globus@pnpi.nw.ru

One of the most important features of Web-Based User Interface [1] for GridNNN was the existence of plug-in mechanism, which made it possible to embed application-specific user interfaces for various software packages right into the main interface frame. Internal API provided for this purpose relieves application interface developers of common yet complex tasks such as user authentication and authorization, job control, file transfer and management (fig. 1).

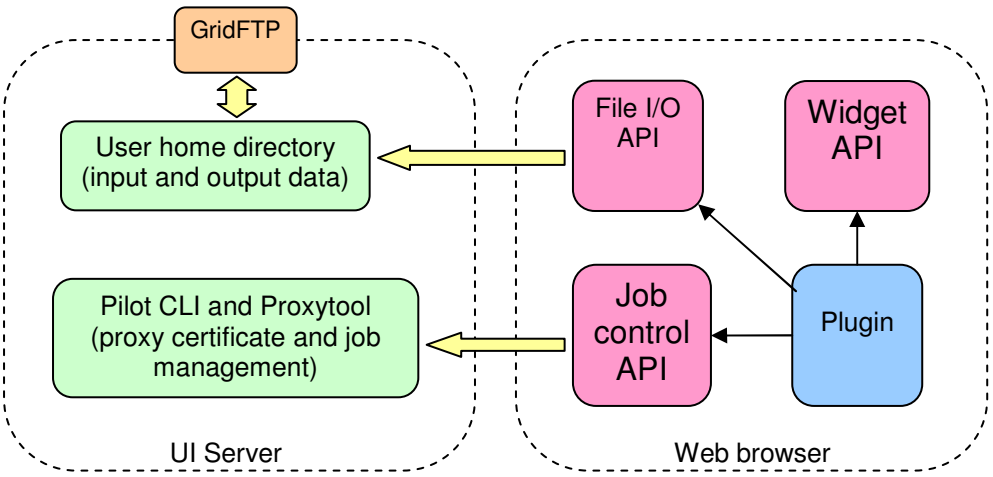


Fig. 1: GridNNN UI architecture

Regardless of the application area, complex software packages usually follow a common behavior:

- Software package consists of a main application (usually MPI) and a set of utilities;
- Input data may need to be prepared (e.g. converted from one of the common formats);
- There's a vast number of data processing options controllable via text configuration file;
- During run, a number of output files is produced, some of which are temporary or intermediate.

GridNNN user interface provides a following way for organizing user files:

- All user files are stored in a home directory;
- There are no tools or machinery to control external storages as well as applications requiring such machinery in GridNNN;
- It is considered convenient to store all files belonging to a specific application in a sub-directory named after it to avoid confusion with lots of files in a root of home directory;
- Output files are staged directly from computing resources and thus must be placed in unique subdirectories to avoid possible conflicts and name clashes.

In a Grid environment, interactive software runs are rare. Most of the time user submits a "job" that takes a bunch of input files and produce output ones. In order to organize these files, user workspace should be split into "projects", each of which is intended to represent a single research. Project may be seen as a folder, holding all the files, input and output, as well as software

configurations necessary to run a Grid job (fig. 2). User can run current project by simply pressing a button; a job will automatically appear in a job list of GridNNN interface where user can track its status. Output files along with a run log are delivered directly to the project folder.

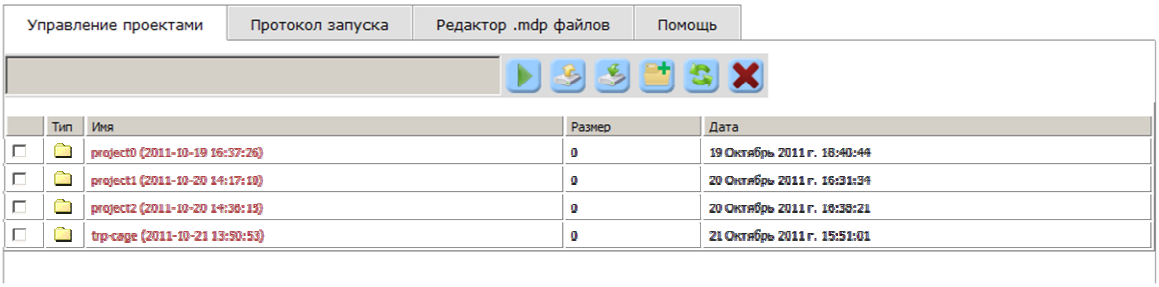


Fig. 2: List of projects in GROMACS user interface

Complex software packages usually consist of a number of tools such as preprocessors, data format converters, computational programs, etc. To simplify their use, each software run is divided into stages, which correspond to different programs and can be represented as either a small single Grid job or part of a bigger one. For example, first stage may convert input data from some common format into format understood by a specific software package. With input files of hundreds of megabytes in size this job is better be done on a Grid worker node rather than on a user workstation. Stages may be enabled and disabled individually. User interface should keep track of input and output files of each stage and warn a user in case of conflict or missing files (fig. 3).

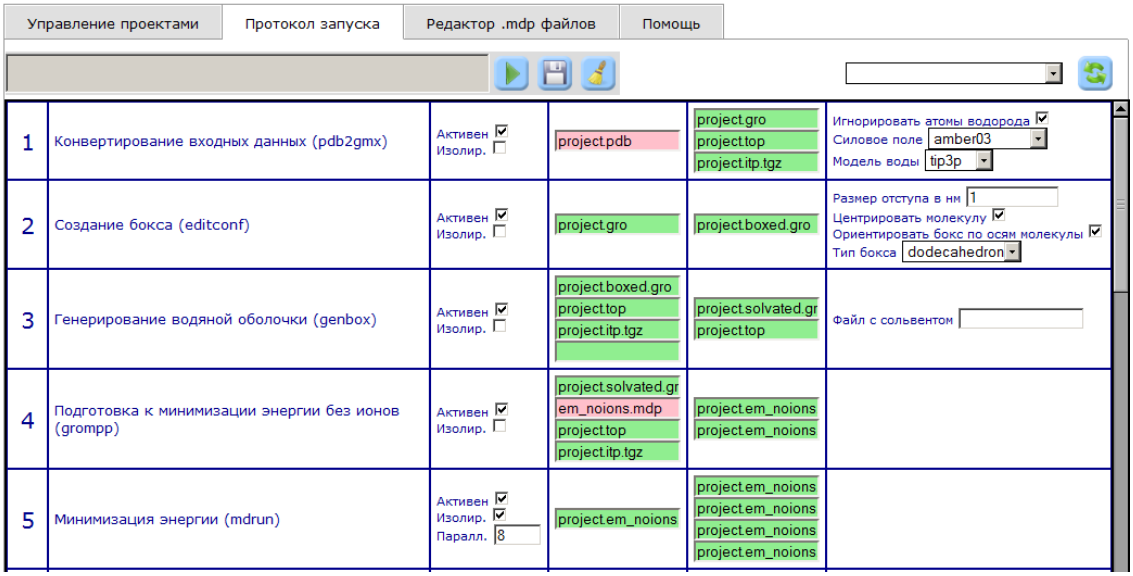


Fig. 3: Job stages with input and output files in GROMACS UI

Some software packages require quite large configuration files to be written in a plain text. Web interface can greatly simplify this task by providing a convenient configuration editor with in-place validation (fig. 4). Usually, a user experienced in specific application area.

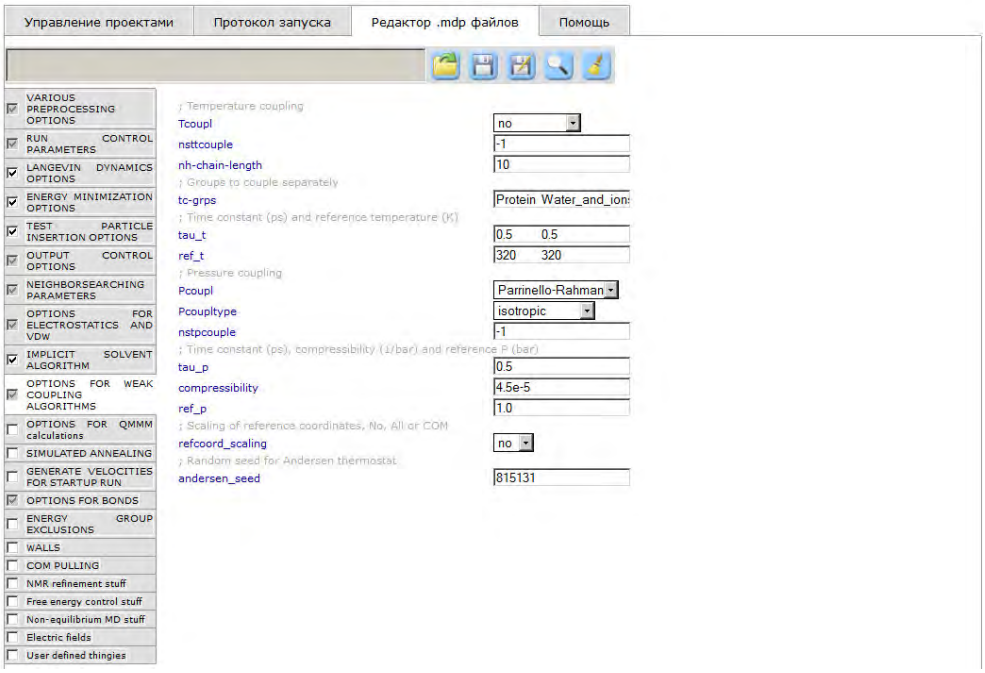


Fig 4: Configuration editor in GROMACS UI

Implementation of such an application-specific interface is available for GROMACS: a software package widely used in computational biology. Taking into account the fact that scientist can access web interface from virtually everywhere, even from his tablet computer, this approach brings scientific work to the real Grid level.

References

[1] A. P. Gulin, A. K. Kiryanov, N. V. Klopov, S. B. Oleshko, Yu. F. Ryabov: “Web-Based User Interface For GridNNN”. Proc. of Int. Conference “Distributed Computing and Grid-Technologies in Science and Education”, June 28 – July 3, 2010, Dubna, Russia, pp. 121-124.

WLCG TIER-2 COMPUTING INFRASTRUCTURE AT IHEP

V. Gusev, V. Kotlyar*, V. Kukhtenkov, E. Popova, N. Savin, A. Soldatov
State Research Center of Russian Federation Institute for High Energy Physics, Protvino, Moscow region, Russia
Victor.Gusev@ihep.ru, Victor.Kotlyar@ihep.ru, kvi@ihep.ru, Ekaterina.Popova@ihep.ru, Nikolay.Savin@ihep.ru, Anatoly@ihep.ru

RU-Protvino-IHEP site participates in the Worldwide LHC Computing Grid. The computing infrastructure serves for big four high energy physics experiments such as Atlas, Alice, CMS, LHCb. In this presentation we would like to talk about recent changes in the site infrastructure and software upgrades. Also we would like to present current status and future plans.

1. Introduction

RU-Protvino-IHEP site has participated in the Worldwide LHC Computing Grid from the very beginning, since 2003. In that time the first grid infrastructure services like CE, SE, WNs, UI on 16 two-core Pentium III 900MHz were installed and configured. LCG (LHC Computing Grid) and EDG (European Data Grid) grid middleware was used. IHEP participated in the EGEE I – EGEE III projects in NA2, NA3, NA4, SA1 [1] activities. After increasing network bandwidth to 100 Mb/s, then to 1 Gb/s and in the end to 10 Gb/s, the grid site in Protvino became one of the biggest Tier-2 site in Russia after JINR with 1k CPU (8800 HEP-SPEC06) and 800 TB disks space.

In the present time our site serves for big four LHC experiments (Atlas, Alice, CMS, LHCb) and many small experiments inside the Institute. We implement shared CPU schema that allows achieving 24x7 CPU resource usage and it makes resources usage more effective.

The main plan for the future is to become the first Tier-2 in Russia.

2. Site overview

From Grid point of view RU-Protvino-IHEP site has very simple architecture which is present on figure 1.

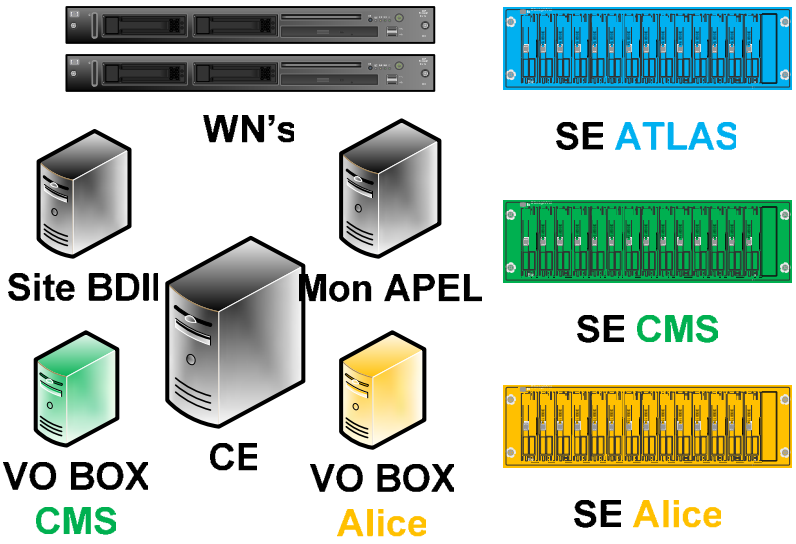


Figure 1: Grid architecture of IHEP site

* Corresponding author

It consists of the following:

- Computing Element based on CREAM-CE one for all Virtual Organisations;
- Three Storage Elements: dCache based for CMS and Atlas, and pure xrootd for Alice;
- Working Nodes with 4, 8, 24 cores and with 2 GB RAM per core;
- Two mandatory VO BOXes for CMS and Alice;
- And two grid services which are necessary to run a grid site – site BDII and APEL.

So we are maintaining an ordinary Tier-2 site nothing special as it may seem at a first glance. However, the complexity appears from three reasons:

1. We share resources for Grid and not Grid physical experiments;
2. We have a big size cluster with around 1000 CPU and 800 TB oriented to user analyses (many read of data for processing);
3. A team which is administrating the cluster is a core team in the IT department of IHEP and also maintains a network infrastructure and core network services.

Including all these circumstance, we get the more complex architecture for the site shown on figure 2.

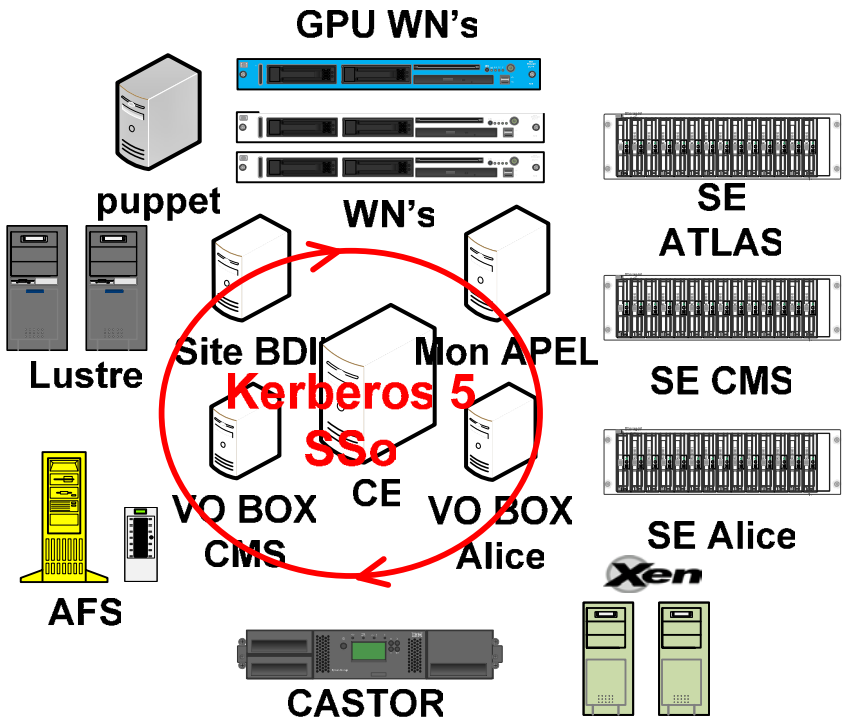


Figure 2: Additional services at grid site

As it might be seen, in the middle of the cluster is the Kerberos5 "Single Sign on" system. It is used by local users to get an access to the resources of the cluster directly without any grid services. It means that a batch system on the cluster should support Kerberos tickets and these tickets have to be forwarded to all Working Nodes too. As soon as we also use Andrew File System for user home directories we must provide support for AFS tokens which are based on Kerberos tickets.

The next major service is a Lustre parallel cluster file system which is shared across all working nodes to be able to allow local users perform data analysis us faster as possible.

And for the long data store and for the store of the RAW data we use CASTOR [2] (the CERN Advanced STORage manager). It is a HSM system that has been using in IHEP for six years. We store data on LTO4 and LTO5 tapes with a simple tape library and stand alone tape drives.

For the end the main administration system on the cluster is Puppet. We give up on using Quattor and started to use puppet instead as more elastic system for management.

We should mention that many services including site BDII, APEL, VO BOX CMS, Puppet and some others are placed under Xen hypervisors as virtual machines. This technique allows us to use resources as much as needed for the current setup and dynamically adjust them if it will be necessary in the future.

To summarize all above, we have got a site structure presented on figure 3. As it can be seen, there are several internal servers that are used for the site infrastructure. They are: network gateway servers which provide NAT (network address translation or masquerade) for the site internal network; DNS (Domain Name Servers) for this internal network and for caching DNS queries on the cluster; squid http proxy servers for caching CVMFS [3] (CERN Virtual Machine File System) requests and special frontier proxies for CMS and ATLAS to cache Oracle DB (data base) queries of experiments to the central repositories; and we have still i386 SLC3 subcluster as the part of our system but it is only for the internal experiments of IHEP; also we have a mix of GPU Nvidia Tesla computing systems.

Additional things to mention are middleware and software on the cluster. For the grid services and for WN's it is gLite 3.2 and base OS (operating system) on the cluster is Scientific Linux 5 64bit. For GPU's we use CUDA 4.2 [4]. There are Ansys 14.0 [5] and Mathematica 8.0 [6] installed on several nodes on the cluster and Intel® Fortran Composer XE 64bit is installed on UI (User Interface). All this software allows us a more flexibly use of our computing farm.

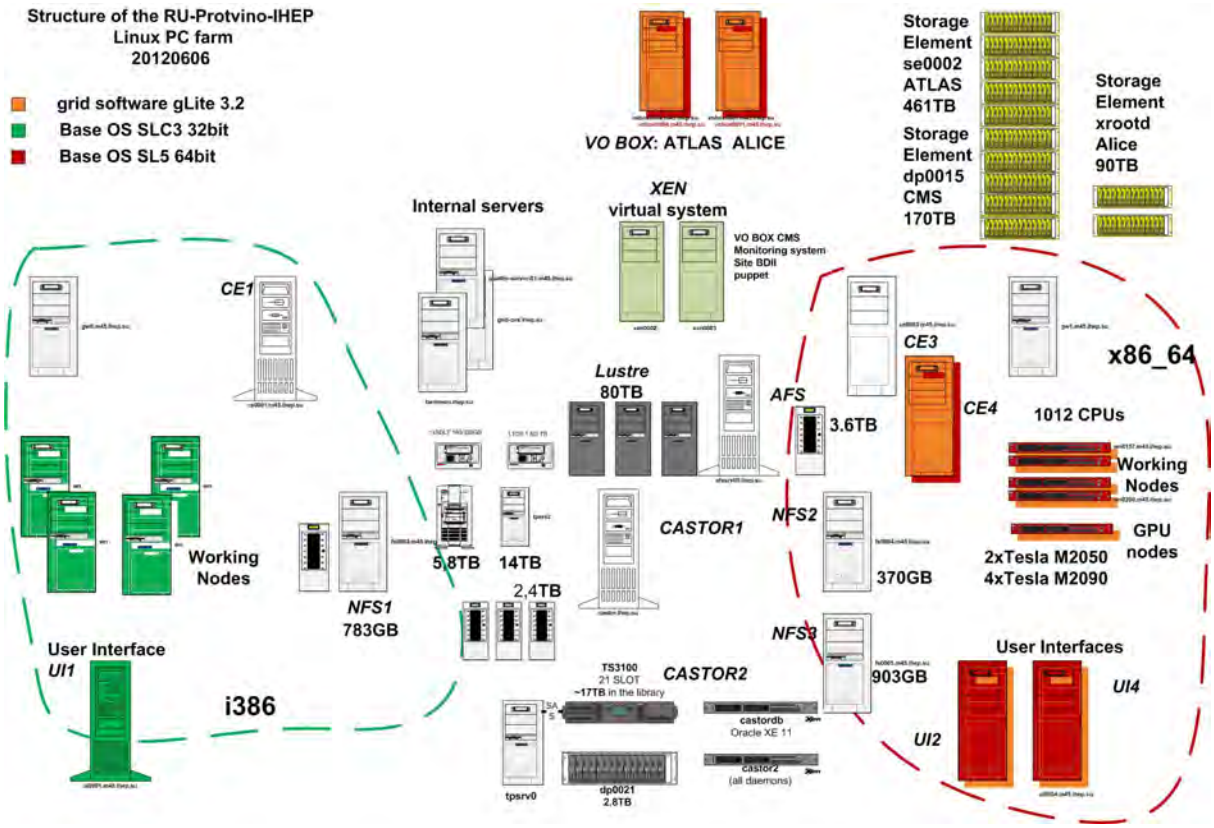


Figure 3: A structure of the site

3. Network overview

The next major part of the site is a network architecture. Here we used a model with internal and external networks and split all site to the hosts with two and hosts with only one access points. This model allows us to reach several goals:

- First of all we hide all working nodes from the direct access from Internet to increase security;
- The next one is that we use maximum bandwidth of available network interfaces and we do not mix different type of traffic in one network;

- Third thing is that it is easy add new resources to such structure by common blocks i.e. it is easy to scale whole infrastructure for the future needs;
- In the end such structure permits to use cheaper network interfaces on the cluster and avoid Infiniband and very expensive 10 Gb/s network cards or modules for devices.

Figure 5 shows the network architecture of the site. Before describing it we have to mention a physical location of hardware inside the computer centre of IHEP. It is physically spitted to several locations see figure 4. There are big working zones (4 and 6) and core network operating centre of the whole Institute (Core). Each zone has its own electrical power, cooling system and connectivity to the core network room. We try to distribute resources for experiments of the Grid and of the local groups in the Institute in such way that if one of working zones stopped to functional then other will still work for remaining VOs. For example if Zone 6 stops (by cooling reason or by power cut) then our site will lost only CMS and Alice and Atlas and LHCb will work in Zone 4. In general the site still works with decreased number of CPU and experiments.

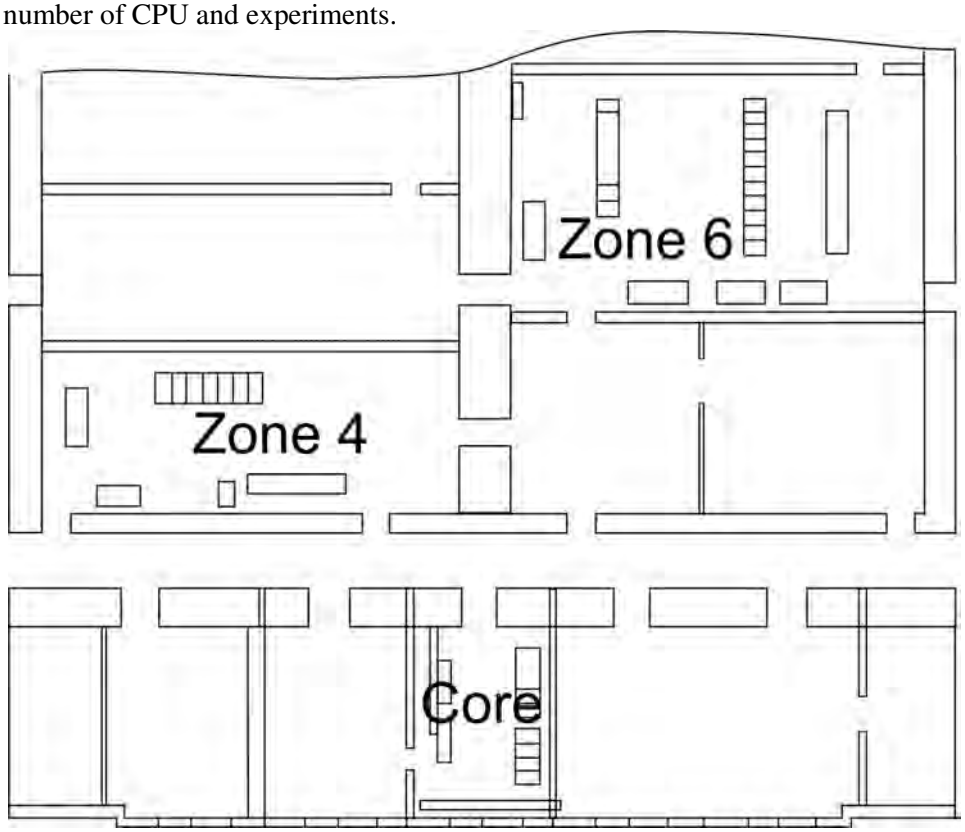


Figure 4: Site physical location

As described above, we have two network segments on the site. There are an internal network with IP address from 192.168.160.0 network and an external network with one of the Institute's subnets dedicated special for the Grid cluster. Every node and server of the Grid farm is connected to the internal network. This network is completely isolated physically from all other networks in IHEP and from Internet. Each subcluster from working zones gather connections on the one network switch with 120 1 Gb network interfaces. Two zones connect to each other by the link aggregation channel 2x10 Gb/s. In this way by connecting all WNs and storages to the one switch we use internal bus of the network device as shared high throughput channel for commutation. On the site used are 1 Gb/s or 2 Gb/s connections for working nodes and 2 Gb/s or 4 Gb/s connections for disk storages. The size of link depends on the number of cores for WNs and by the performance of the disk systems of the file storage servers.

For all core services external network is used. It is a special dedicated subnet of the Institute network. And the Grid site has a direct connect by 10 Gb/s from every working zone to the core router of IHEP. The Institute itself has 10 Gb/s connection to the Moscow backbone network for scientific research. Second 10 Gb/s connect is planned in near future.

Such a network structure allows us to use up to six 1 Gb/s links to data servers with cheap gigabit Ethernet cards.

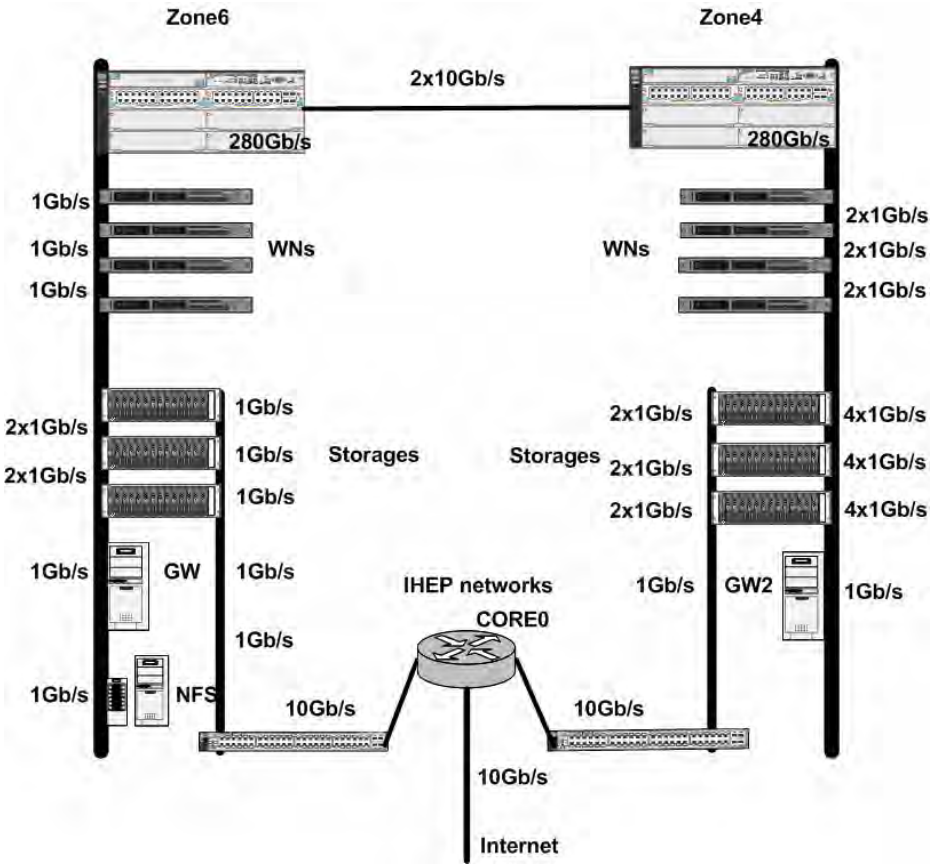


Figure 5: Site networking

4. Recent upgrades

In May 2012 RU-Protvino-IHEP site stepped over 1000 CPU slots. CPU (for CPU we assume cores with HP on) and disk resources were doubled from 400 CPU to 1000 CPU (8800 HEP-SPEC06) and from 360 TB to 800 TB. Such resources increase was a big stress test for the current infrastructure and showed bottlenecks of the services. The first main problem was in the default limit of maui scheduler to 1024 jobs per queue and was solved by recompiling maui scheduler (see figure 6).

```
diff /opt/maui-3.3.1/include/moab.h  /opt/maui-3.3.1.orig/include/moab.h
314c314
< # define MAX_MRES        2048
---
># define MAX_MRES        1024
```

Figure 6: Changing maui scheduler limits

The next big challenge was in CREAM-CE performance and stability. The hardware was upgraded to Intel Core7 server with 12 GB RAM. Some setup was made for Tomcat5 server by

increasing JAVA heap size. The new parameters for it are *"-Xms512m -Xmx4096m"*. Last thing is that we have enabled caching for mysql data base as it presented on figure 7.

```
diff /etc/my.cnf /etc/my.cnf.0
14,16d13
< query_cache_type = 1
< query_cache_size = 40M
```

Figure 7: mysql caching on CREAM-CE

To allow experiments use increased number of slots, we also made some changes:

- for Alice together with experiment management we switched our site to the peer-to-peer software distributing on the working nodes and we offloaded the central Lustre cluster;
- for ATLAS we moved shared library to the local disk on the working nodes and this dramatically improved situation with starting of Panda jobs. They did not have to use network shared area for LD_LIBRARY_PATH. The modifications are presented on figure 8.

```
cat /lustre/ihep.su/grid/atlas/setup.sh.local
export LD_LIBRARY_PATH=`echo $LD_LIBRARY_PATH |sed -e
"s/lustre\/ihep.su\/grid\/atlas\/lib/opt\/grid\/atlas\/lib/ "
```

Figure 8: Moving of the shared library for ATLAS

In the end we started to use a stripe RAID system for home and scratch directories on the working nodes under XFS [7] and found that CMS, LHCb, and ATLAS can not handle big partitions under XFS (we used 1.6 TB). Probably some part of software is build without 64 bit file support and it can not work with 64 bit incomes on the file system. We were forced to rollback to ext3 on such nodes.

5. A new storage for Atlas

To finish upgrade section, installation of the new storage system for ATLAS should be mentioned. In the Institute the latest gold release v.2.2.0 for dCache was installed. As a base OS we have chose Debian Linux which is supported by dCache distribution. Our primary goal was to install a system for the future that means an easy software upgrade and being able to handle several petabytes of data without modification of the infrastructure. Installed system is presented on figure 9.

It is worth to mention that we started to use a distributed system of dCache head nodes according to the latest recommendation of dCache for large (>1PB) setups. On name space node we started to use SSD for storing Name Space data base. Each pool node uses XFS partitions divided by 10 TB.

6. Site resources and usage

A current situation on the site with sharing and consuming resources is presented on figure 10. We have a fair share setup for the batch system queues for all big four LHC experiments and disk usage as presented on the right side of the figure. Accounting information [8] about consuming CPU on the farm in 2012 is shown on the left side. CMS was inactive several months on the farm due to the problems with CREAM-CE so its computing power was distributed across LHCb and Alice.

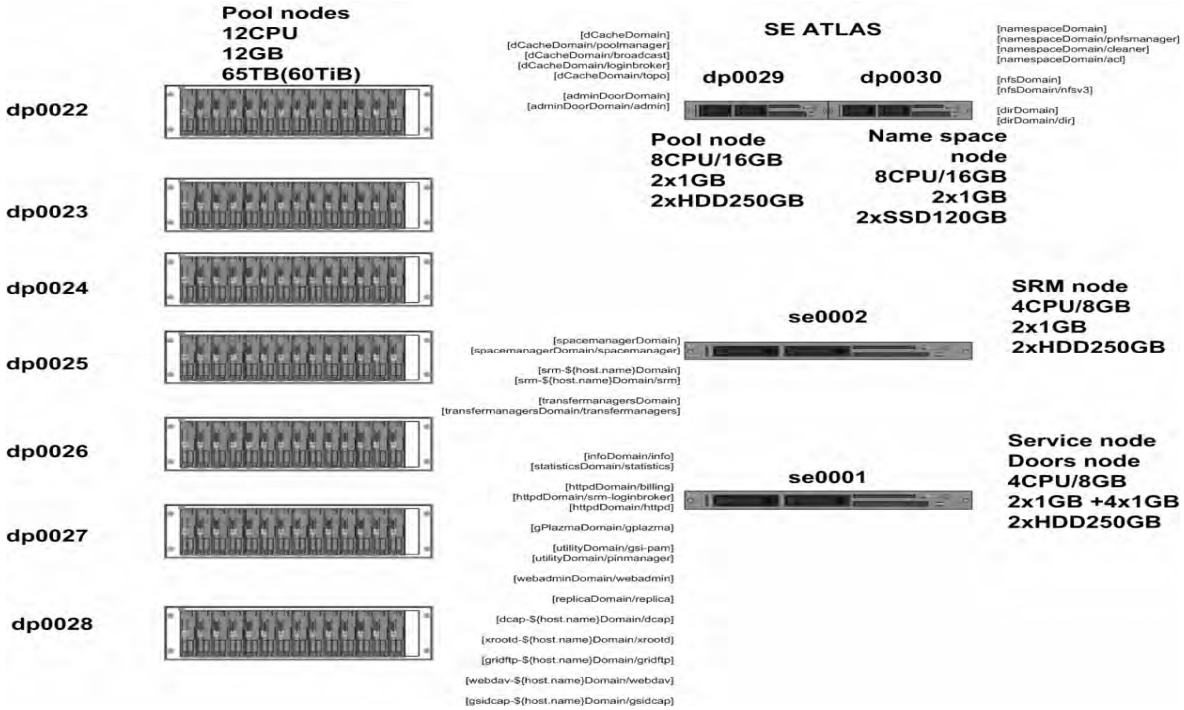


Figure 9: SE ATLAS at IHEP site

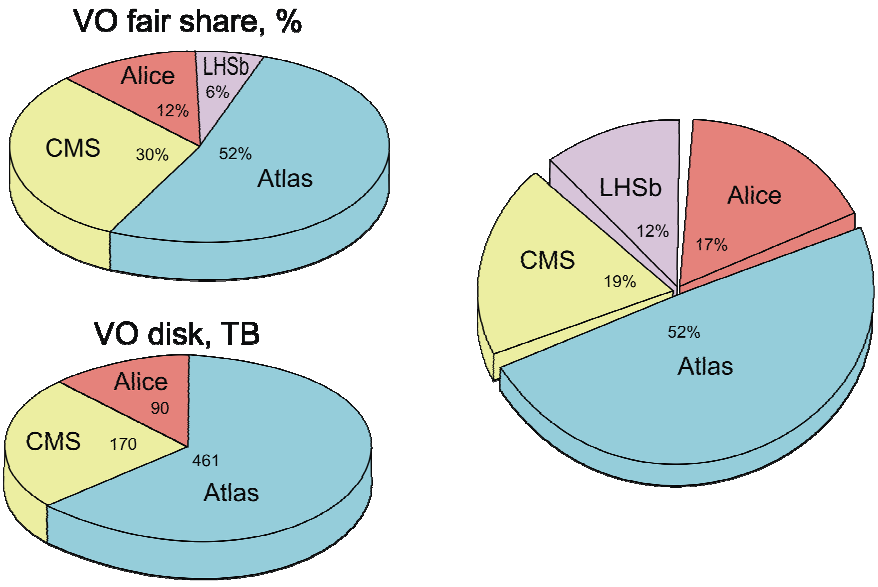


Figure 10: Site resources and usage

7. Future plans

We have some plans for the site development. At the end of 2012 we should upgrade resources from 1000 CPU to 1600 CPU and we should increase disk storages to 1200 TB, and for software we would like:

- split CREAM-CE servers by experiments;
- move CMS to Lustre and completely destroy NFS;
- upgrade CMS dCache to latest golden release 2.2.x;

- add CASTOR support for the local ATLAS group;
- upgrade cluster (especially WNs) to the latest EMI release.

For strategic plans we would like to achieve the physical maximum (limited by air conditioning and electric power) for our site in resources and network. It is 5000 CPU, 6000 TB disks and 8x10 Gb/s network bandwidth for external connections.

8. Conclusion

The WLCG Grid site at IHEP has a big potential of development for the future and it established reputation of the site with high availability and reliability in Russian segment of the Grid infrastructure. We use leading technologies in the computing science for computing clusters with batch systems to run jobs. There are many investigations and approbations inside the site infrastructure like cloud technology and GPU computing which are going to be a future in data centres. The generic goal is to become a number one Tier-2 in Russia.

References

[1] EGEE homepage http://public.eu-egee.org/
[2] CASTOR homepage http://castor.web.cern.ch/
[3] CVMFS homepage http://cernvm.cern.ch/portal/cvmfs/release-2.0
[4] GPU technology and CUDA http://developer.nvidia.com/cuda-downloads
[5] ANSYS mechanical home page http://www.ansys.com/Products/Simulation+Technology/Structural+Mechanics/ANSYS+Mechanical
[6] Mathematica home page http://www.wolfram.com/mathematica/
[7] XFS file system portal http://xfs.org/index.php/Main_Page
[8] Grid accounting system http://accounting.egi.eu/

OVERALL EXPERIENCE OF GRIDKA T1 OPERATIONS AND LHC EXPERIMENTS REPRESENTATION

A. Heiss, A. Petzold, M. Zvada
Karlsruhe Institute of Technology, Germany
{andreas.heiss,petzold,zvada}@kit.edu

The GridKa Computing Center at the Karlsruhe Institute of Technology is one of the biggest Tier-1 (T1) centers for the Worldwide LHC Computing Grid (WLCG) and one of the major resource providers in the EGI region NGI-DE. Since GridKa was established more than 10 years ago, GridKa staff has been closely cooperating with its different user communities also called virtual organizations (VOs) to ensure smooth operation and high availability of the resources and critical production services operated 24/7. This presentation will focus on overall experience how different experiments are represented in such a multi-VO computing infrastructure and what we are doing in order to keep GridKa running and serving T1 services successfully towards WLCG requirements.

1 The Grid and Steinbuch Center for Computing

Grid Computing is a key technology that provides scientists and engineers from research and industry worldwide distributed data and IT resources. Via high performance wide area networks grids integrate today's compute resources from desktops to cluster and visualization systems up to supercomputers. Also distributed data storage and archives from the terabyte to the multi-petabyte range as well as data of various types and disciplines. Transparent access from any location to such a distributed infrastructure allows to work on complex scientific and social problems and to collaborate in new interdisciplinary ways. Since many years the Steinbuch Center for Computing (SCC) department "Distributed Systems and Grid" (VSG from German term) contributes within national and international projects to the development and operations of this infrastructure, and works as a service and resource provider to the users [1].

Especially here in Europe, the European Grid Initiative (EGI) is mentioned, under whose umbrella national or international grid initiatives together form a common infrastructure [2]. The American equivalent to this is Open Science Grid (OSG), Asian countries and Australia are working together in the Asia Pacific Grid and Scandinavia has Nordic DataGrid Facility. None of these distributed infrastructures is complete itself rather than continuously evolving. However, despite the challenge of differences in the technologies, all resources to the user communities are available via standardized interfaces available worldwide.

1.1 VSG projects within SCC

The VSG department gives opportunity for diploma and doctoral thesis which are integral part of the grid and cloud computing evolution. In cooperation with the region Baden-Württemberg, VSG provides practical training of undergraduate students in the field of information technology. For example through participation in the winter semester 2011/12 the KIT first time offered lecture on "Distributed Systems: Grid and Cloud", and since 2003 has been hosting the annual Summer School for Grid and Cloud Computing [3].

In the context of the European Grid Infrastructure hierarchy, there are several Grid projects within the VSG department at SCC, for example:

- *European Grid-Initiative (EGI) / EGI-InSPIRE*: The European Commission in its 7th Framework Programme provided substantial support for a European Grid Initiative (EGI). Germany has responded to this initiative by the recently formed Gauss-Allianz e.V. consisting of 13 German supercomputing centers. The KIT is lead

manager and was instrumental in the project planning of EGI-INSPIRE (Integrated Sustainable Pan-European Infrastructure for Researchers in Europe).

- *National Grid Initiative (Deutschland) Germany (NGI-DE)*: NGI-DE consist of about 50 grid initiatives, which are involved in EGI. Within the NGI-DE, VSG works in the areas of "Project Office and Sustainability", "Central Monitoring", "Helpdesk", "Security".
- *GGUS – Global Grid User Support System*: GGUS provides a central platform where users as administrators ask technical questions and address their issues accordingly to the Grid Site support unit. These support units are distributed teams of experts - administrators of local and global services through application supporters to grid middleware developers - connected to whom such requests of appropriate workflows are assigned. The GGUS platform is now available and used in several very large European IT projects.

2 The GridKa Tier-1 Center for LHC

GridKa – Grid Karlsruhe, project founded as part of the other SCC activities is driven by the need of thousands worldwide scientists use the computing and storage resources for data analysis of LHC experiments at CERN. There have been years involved into the grid development in addition to the construction of the LHC accelerator and the four detectors, ALICE, ATLAS, CMS and LHCb, where GridKa is now significantly important piece of the World Wide LHC Computing Grid (WLCG) [4]. Together with the German nuclear and elementary particle physicists SCC established Grid Computing Center Karlsruhe (GridKa) as the German contribution to the LHC Computing was developed specifically for high data throughput of the LHC and other high-energy physics experiments. WLCG ties together resources from the European Grid Initiative (EGI), Open Science Grid in the United States (OSG), and the Nordic DataGrid Facility in Scandinavia.

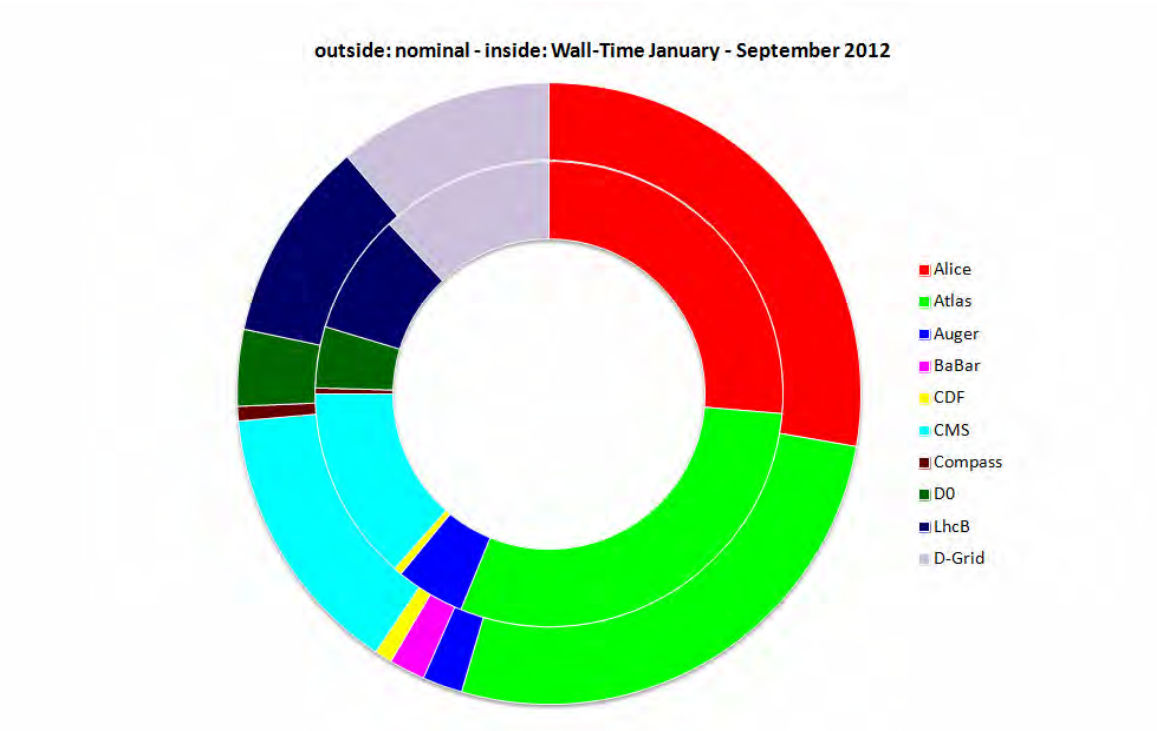


Fig. 1: GridKa computing resources share January-September 2012

In 2011, EGI sites provided the LHC VOs with more than 7 million kHS06hours [5] of CPU and more than 100 PB of online storage space. This is the largest single contribution of resources to

WLCG. The GridKa Tier-1 center is hosted by Steinbuch Centre for Computing at Karlsruhe Institute of Technology. GridKa was established in 2002 as a regional computing center for the LHC experiments which were still developing their computing models at that time and several other High Energy Physics experiments that were already taking data. Today, GridKa supports all four LHC experiment VOs and seven more VOs from High Energy Physics and Astroparticle Physics (Auger, BABAR, Belle, Belle2, CDF, Compass, DØ), and several other VOs from different fields of science. See Figure 1 for distribution of the resources from January till September 2012.

Among the 11 WLCG Tier-1 centers GridKa provides approximately 14% of the resources available to the experiments and it is the largest center supporting all four LHC VOs. In 2012, GridKa provides 130 kHS06 of CPU resources (more than 1200 compute nodes) split into two sub-clusters, 14PB of disk storage, and 17PB of tape storage to its users. For the regular archiving and reprocessing of the raw data of LHC, the experiment data flows via a 10 Gbit/s network connection imported from CERN while GridKa serves further connections to other Tier-1 and Tier-2 centers of the preprocessed data at Tier-1 level ready to be transferred. See Figure 2 with number running jobs per sub-cluster at GridKa over the year 2012.

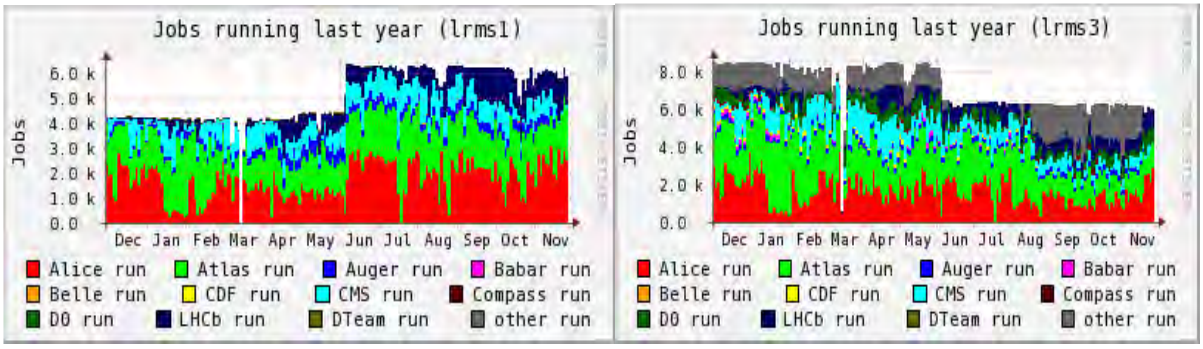


Fig. 2: Running jobs per VO in 2012 (~14k job slots in total)

3 Grid Core and Site Services

The amount of data from the LHC experiments with the latest algorithms and methods to (re-)processing is regularly ever-growing, so the computing power of GridKa increases yearly. For standardized measurement of computational power provided by all WLCG centers benchmark HEP-SPEC06 is used, where GridKa was deeply involved. These measurements helps in planning replacement an old and inefficient systems with new, more energy-efficient computers and thereby the overall computing power increases.

In addition to the resource management planning and accounting, GridKa provides high availability operations of grid services for the LHC and other user communities. It manages the Workload Management System (WMS) and compute element (CE), which forwards grid computing jobs to centers such GridKa.

Standardized Information Systems (Berkeley Database Information Index, BDII) deliver through Grid interfaces information about Grid computing centers, such as what virtual organizations are supported and whether free computing and storage resources are available. For data management there are file catalogs (LFC) and a file operated transfer service (FTS), which controls all data transfers between GridKa and its associated Tier 2 and Tier-1 centers. Different proxy services allow efficient access to database systems at SCC and at CERN. For community-specific services front-end VO-boxes are provided.

3.1 GridKA Certification Authority

The GridKa-CA [6] is present at KIT since 2002, as a member of the European Grid Policy Management Authority (EUGridPMA), an international organization, composed of over 40 countries.

Together with TAGPMA (North, Central and South America) and APGridPMA (Asia, Pacific, Australia) established a global network of the International Grid Trust Federation (IGTF).

Digital certificates contain a public key, additional information and the signature of the CA. The certificate body is accredited with signature of a public key belonging to a particular person. The certificates are used for authentication, encryption and decryption of sensitive data that is transmitted over the Internet and other networks. The GridKa-CA provides standard X.509 certificates to users, computers, and applications throughout Germany. The long list of organizations that use GridKa-CA certificates include both research institutions and universities, and industrial companies.

3.2 Continuous process optimization with ITIL

The coordination of global support needs well coordinated processes and workflows. The SCC has faced with a similar challenge. In merging the former Center of the University of Karlsruhe and the Institute for Scientific Computing of the Karlsruhe Research Center for SCC was clear that a data center in two locations requires special measures and coordination.

Therefore the decision was taken in the SCC within the rules of ITIL lifecycle stages "Service Strategy", "Service Design", "Service Transition" and "Service Operation" applying for continuous process optimization. An SCC internal ITIL Project was led by the department "ISM - IT Security and Service Management" initiative, which was also supported by the department of VSG. The focus in the grid environment was on the service process "Service Catalogue Management", the creation of service descriptions and the separation of grid services into so-called service modules and the service process "Incident Management", in particular towards the smooth 24/7 operation mode of GridKa [7].

To achieve improvements in terms of grid operation and support service in Germany "Incident Management" isn't enough, so keep maintenance of the German Tier-1 data center GridKa ITIL compliant, it requires in addition "Change Management" process. That is still work in progress though and on its way to the ITIL regulation fulfillment by SCC.

3.3 GridKa monitoring and 24/7 support

To monitor the function of the various grid services, VSG developed jointly with other research institutions monitoring system, which are largely based on the open source Nagios [8]. Nagios has been specially adapted to the needs of grid computing. However, as part of the modernization process and monitoring quality improvement, GridKa Tier-1 migrated all Nagios-based service to the Icinga [9], which is now used by the GridKa on-call engineer(s) (OCE) as a basic tool to solve problems or alarm local experts take an appropriate action.

At GridKa, we have two groups of OCEs: storage and grid services. Each group consist of 8-10 people and they are rotating on weekly basis. In addition, KIT has wide area network and infrastructure on-call service. Alarms are sent to mobile phones by Icinga plus GGUS alarm tickets are triggered by the incident thresholds configured within Icinga per service. Person on-call may or may not be an expert for the affected system (no experts expected on duty!). OCEs usually do their best try 'standard recipes' to fix the problem first and only if necessary they try to reach an expert.

Approximately 85% of problems could be solved without calling (additional) experts. Documentation is improved continuously, also people on-call gain experience by doing regular steps provided by 'recipes'. On-call services does not guarantee that problems can be fixed within few hours. The incident handling during on-call service hours is well illustrated on the Figure 3.

3.4 GGUS - Global Grid User Support System

Global Grid User Support [10], abbreviated GGUS, means globally distributed support system, which is managed and operated by a working group within the department of VSG. Scientists of international grid projects (e.g. EGI) or members of so-called virtual organizations such as the High Energy Physics (LHC experiments at CERN) use this system to report malfunctions related to the computing grid. It is also knowledge base for tips and tricks among site experts stored in the ticket database or place for providing important news.

Heart of GGUS is a complex web application that by providing web service interfaces allows

the integration of other helpdesk systems to form in its principle knowledge federation. Currently 15 external systems are linked via these interfaces with GGUS. To date, over 100 expert groups are spread over all the continents of the world, which provide round the clock support for the continuously growing community of grid users. By now approximately 80,000 Grid problems solved by 1,500 grid experts worldwide.

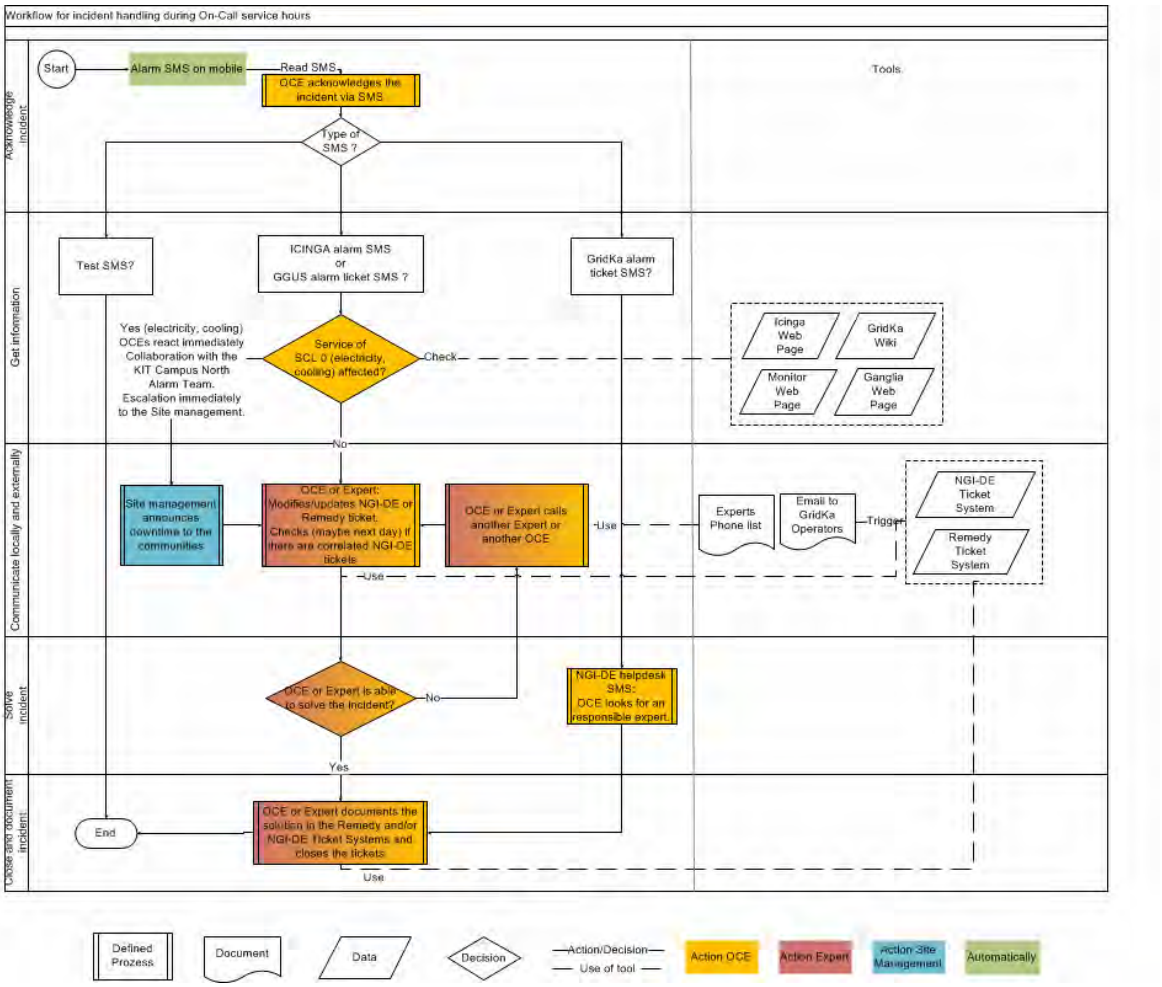


Fig. 3: Workflow for the incident handling during on-call service hours

4 LHC representation at GridKa

Several formalized support workflows, all based on ticket systems, have been set up within the LHC experiments, WLCG, and the major Tier centers. Experiment representation complements these workflows with experiment specific and proactive support.

In 2010, three positions of local VO representatives for ALICE, ATLAS, and CMS experiments were established. The representatives' duties are to represent both their respective LHC experiment at GridKa and GridKa within their respective LHC experiment [11].

4.1 Motivation and representatives' tasks

GridKa staff has been closely cooperating with its different user communities to ensure smooth operation and high availability of the resources and services. Even though all LHC VOs have used gLite-based services already for several years, there are major differences in the use of the components by the different experiment frameworks. These differences have a strong impact on the operations of the Tier-1 center.

The experiment representatives are fully integrated with GridKa staff and their experiments; within the latter, they mainly communicate with the computing experts, as the servers running at the Tier-1 are higher level ones. Both the experiments and GridKa benefit from the experiment representatives' inside knowledge of the details of the experiment specific computing and the on-site operations at the Tier-1.

The experiment contacts are responsible for several production critical services their respective experiment relies on at the Tier-1. These include site services like FTS (mainly used by ATLAS and CMS) and xrootd (used by ALICE) as well as other distributed experiment specific services.

The major task besides service administration is communication between the experiment and the site administrators. The experiment representatives serve as principal point of contact for the site administrators for all matters related to the VO, ranging from job behavior to storage system setup. At the same time the experiment contacts communicate with all parties within the experiment, e.g. Tier-0, Tier-1, and Tier-2 centers, users, and in particular experiment computing experts.

The close coordination among the experiment representatives has proved to be very beneficial for their daily work as well as intermediate and long term tasks. As the experiments share the Tier-1 infrastructure and many services, many issues affect several VOs. For instance, network problems are often discovered because of middleware service failures. The experiment representatives inform each other of these problems and together analyze the impact on the experiments and provide specific feedback to the local network administrators. In other cases, experiences of one VO with the effect of a VO workflow on the storage backend are shared among the on-site experiment contacts for the benefit of all VOs.

As part of the team of local administrators, the experiment representatives have privileges on the local infrastructure that would not have been granted to external representatives. Thus, they are enabled to quickly diagnose problems and assess the impact on the experiments' work much better than compared to a long chain of communication from externals to the local administrators and back. In addition, on-site representatives are perceived as internal members of the respective team by both the experiments and the site administrators. Thus, communication between VOs and sites has been improved on formal and informal levels.

The experiment representatives also contribute to organization of grid computing events as well as to the deployment and operation of services for the experiments. Also in this area, their work profits from their unique perspective which combines the view of the experiments and the Tier-1 center.

Experiment workshops like a collaboration-wide ALICE Tier-1/2 workshop or a face-to-face meeting of the grid administrators of computing centers of ATLAS DE cloud were organized by the experiment representatives at GridKa. In addition they make major contributions to the international GridKa summer school on grid and cloud computing.

Furthermore, experiment representatives are heavily involved in the deployment and operation of the glideInWMS [12] service for CMS and to the multi-VO meta-monitoring system HappyFace [13].

4.2 Feedback to on-site Representation Model

There has been positive feedback from the VOs represented at GridKa in several meetings and workshops, focusing on faster feedback and on better integration of GridKa into the experiments' computing groups. Service experts at GridKa highly appreciate to have expertise on the specific experiment workflows on-site and the easier and faster communication.

Contrary to formalized support via ticket systems, there are no obvious metrics for measuring the success of on-site experiment representation, as it uses a short and informal way of communication and as it provides proactive support. Its effectiveness is nevertheless reflected by the positive feedback mentioned above and the ongoing commitment by the experiments and the computing center to jointly co fund the positions.

5 Summary and Conclusion

In the long term future, we expect changes in the usage of the Tier-1 center both with respect to new user communities and with respect to new infrastructure as a service (IaaS) technology. If the current model of on-site VO representatives is still useful for computing centers that in the future will mainly work as IaaS providers depends largely on VO specific requirements which cannot be fulfilled with generic IaaS cloud resources. As long as VO specific knowledge is required to efficiently provide a service to the VO, it might be useful to have a VO representative integrated into the team of site administrators.

The GridKa Tier-1 center at its large profits very much from the on-site experiment representatives. Their work has resulted in improved availability and reliability of site services and in improved communication between the computing center and the experiments. In particular, the unique perspective offered by the integration in both the on-site team and the experiment enables the representatives to proactively address issues on most areas of site operations related to experiment work and even bring the new projects to participate on internally with significant outside GridKa visibility.

References

[1] SCC KIT News. Web site: http://www.scc.kit.edu/downloads/oko/SCC-News_2011_3_web.pdf
[2] European Grid Infrastructure. Web site: http://www.egi.edu
[3] GridKa School. Web site: http://gridka-school.scc.kit.edu
[4] Worldwide LHC Computing Grid. Web site: http://www.cern.ch/lcg
[5] HEPSPEC06. Web site: http://w3.hepix.org/benchmarks/doku.php
[6] GridKA-CA. Web site: http://www.gridka.de/ca
[7] ITIL and Grid services at GridKa: http://iopscience.iop.org/1742-6596/219/6/062018
[8] Nagios. Web site: http://www.nagios.org
[9] Icinga. Web site: http://www.icinga.org
[10] Global Grid User Support (GGUS). Web site: http://www.ggus.org/
[11] Experiment Representation at the WLCG Tier-1 Center GridKa. *PoS EGICF12-EMITC2*: http://pos.sissa.it/archive/conferences/162/032/EGICF12-EMITC2_032.pdf
[12] glideInWMS Project. Web site: http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.prd/index.html
[13] HappyFace Project. Web site: https://ekptrac.physik.uni-karlsruhe.de/trac/HappyFace

A STATUS REPORT ON CLUSTERING AND SERVICES DEPLOYED AT ISS DATE CENTER

F.L. Irimia, A. Sevcenco, B.A. Dumitru, I. Stan, S. Zgura
Institute of Space Science, ISS, Magurele, Romania
liviuirimia@spacescience.ro, bogdan.dumitru@spacescience.ro ,
adrian.sevcenco@spacescience.ro, bogdan.dumitru@spacescience.ro,
ionelstan@spacescience.ro, szgura@spacescience.ro

In this two part presentation we will describe the datacenter topology and the hardware used as well as clustering software services employed for HPC usage. We will briefly talk about main functions and proprieties of hardware components used and then we will browse through proprieties and implementation of MAUI cluster scheduler and TORQUE resource manager. In the second part of this paper we present the middleware used in ISS (Institute for Space Science) Data Center, hardware used for those middleware and a usage statistics for those middleware.

Datacenter architecture and clustering technologies used

Rocks Clusters was chosen to ease of deployment, management, maintenance and flexibility. The Rocks Clusters is a Linux distribution designed for high performance clusters. It is a project started in the year 2000. It is an open source operating system that was initially based on the Red Hat Linux distribution. Later versions of Rocks are based on the CentOS distribution (which comes with a modified interface for Anaconda to simplify mass installation.

Following its release, Rocks Clusters became one of the most used clustering operating system for commercial organizations, ranging from academic purposes to government use, used in more than 1400 clusters.

The physical assembly of a Rocks Cluster contains one or more of the following node types:

—Frontend nodes: these are the nodes exposed to the outside world, which include several services like DHCP, TFTP, NFS, MySQL, and HTTP. Also, on these nodes the users log in, submit and track their jobs.

—Compute nodes: these are the workhorse nodes that run the user submitted job.

In Figure 1 we present the typical architecture of a Rocks cluster.

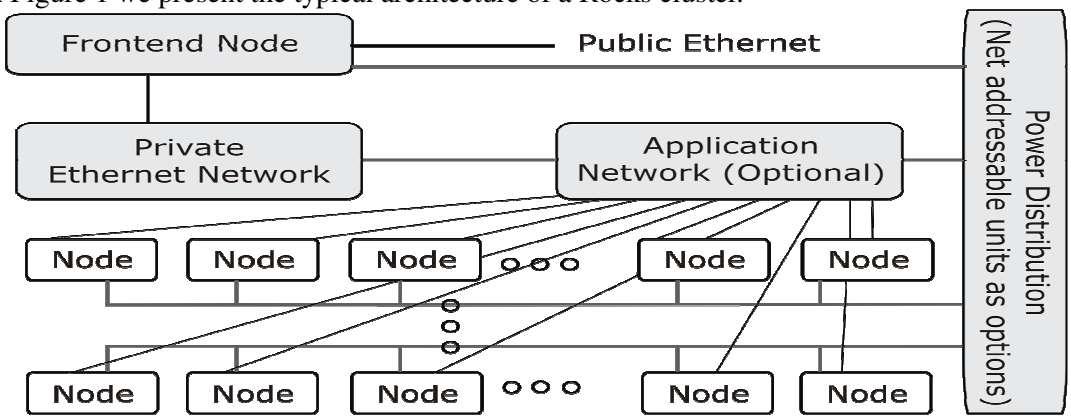


Fig. 1: Typical assembly of a Rocks cluster

On the compute nodes, the Ethernet Interface (in Linux it maps to eth0) must be connected to the clusters Ethernet Switch. This network is considered to be private. On the frontend node two Ethernet interfaces are required: eth0 must be connected to the same Ethernet network as the compute nodes and eth1 must be connected to a public Ethernet network.

Rocks use the well-known clustering tools “Maui” and “TORQUE”. “Maui” is a scheduler

with the properties of policy engine that allows sites control over when, where and how a clusters resources are allocated to jobs. It also optimizes the use of these resources, monitors system performance, diagnoses problems and manages the system. A scheduler is assigned the job of determining when, where and how jobs are run so it can maximize the output of the cluster. The decision that a scheduler takes is divided into three categories: traffic control, mission policies and optimization. "TORQUE" is a resource manager with a scheduler making requests to it. Resource managers provide the low level functionality to start, hold, cancel and monitor jobs. It is used primarily in batch systems and provides control over batch jobs and compute resources.

Underlying technologies

In this section we will summarize the underlying technologies deployed at ISS.

A. *DHCP* – **D**ynamic **H**ost **C**onfiguration **P**rotocol

The Dynamic Host Configuration Protocol (DHCP) is a network protocol that enables a server to automatically assign an IP address to a computer from a defined range of numbers configured for a given network. There are three methods of allocating an IP address:

- Dynamic allocation: the network administrator assigns a range of IP addresses to a requesting client.
- Automatic allocation: the DHCP server permanently assigns a free IP to a requesting client
- Static allocation: the DHCP server allocates an IP to a requesting client based on a table containing pairs of MAC-IP addresses pairs, which are manually filled in.

B. *PXE* – **P**reboot e**X**ecution **E**nvironment

PXE is an environment used to boot computers using a network interface independently of data storage devices or installed operating systems. It uses several network protocols like *Internet Protocol*(IPv4), *User Data Protocol* (UDP), *Dynamic Host Configuration Protocol*(DHCP) and *Trivial File Transfer Protocol*(TFTP).

Services deployed at ISS and usage statistics

1. gLite (in present EMI)

gLite is an middleware for grid computing used by the CERN LHC experiments and other scientific domains.

The project was started in 2004, and in May 2006 it become an official middleware, but in 2010 the project ended, and in present was taken by EMI (European Middleware Initiative).

The architecture contains security, user interface, computing element, storage element, information services, data management and workload management.

2. Computing element (or in gLite CREAM CE)

CREAM – Computing Resource Execution And Management services. It implements job management functionality at Computing Element level (allows to submit, cancel, monitor, etc). It can be use direct by the client or by some higher level services.

3. Storage element (or in gLite DPM)

DPM (Disk Pool Manager) is a lightweight solution for disk storage management.
The architecture contains two nodes type: head node and disk node (fig. 2).

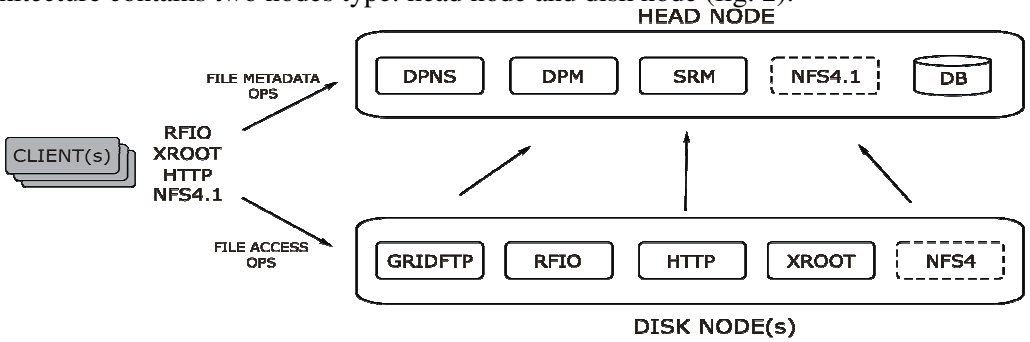


Fig. 2: Deferent components of the DPM and their basic interaction

The head node that is an entry point for clients, hosting and name server that include SRM, DPMS, and DPM domain and disk nodes that host the actual data, providing remote data; each of those nodes will run all data access demons (rfio, xroot, gridftp, ntd41).

4. Information services (or in gLite BDII)

BDII (Berkeley Database Information Index) consists of a standard LDAP (Lightweight Directory Access protocol) witch is update by an external process (fig. 3).

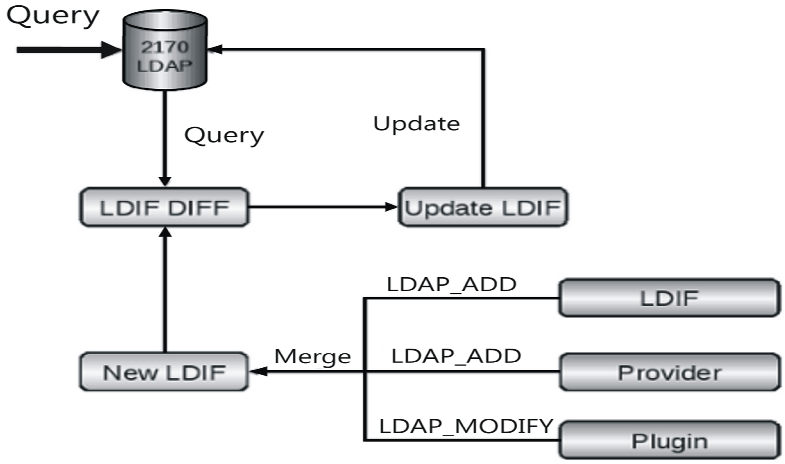


Fig. 3: BDII process

The update process obtains an LDIF (Lightweight Data Inter Change Format) from a number of sources and merges them. It then compares this to the contents of the database and creates an LDIF file of the difference.

AliEN (Alice Environment)

AliEn – is an open source grid framework using the combination of a web service and a distributed agent model – is started in 2000 and is run in present.

The basic components for AliEn are: file catalogue with metadata capabilities, data management tools for data transfer and storage, authentication and authorization, workload management system, interface to other grid implementation, root interface, monitoring.

AliEn architecture contains: external software, alien core components, and services and interfaces (fig. 4).

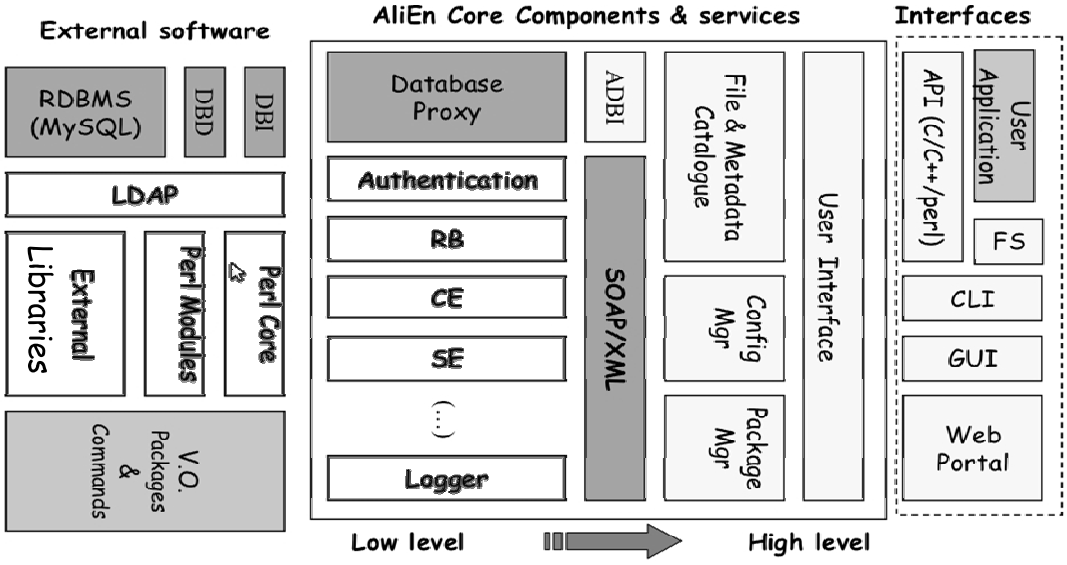


Fig. 4: The AliEn Architecture

In ISS we use for components and services CE, SE, authentication, etc and for interface web portal, GUI, CUI.

The AliEn nodes use pull architecture instated push like other middleware (gLite)(fig. 5).

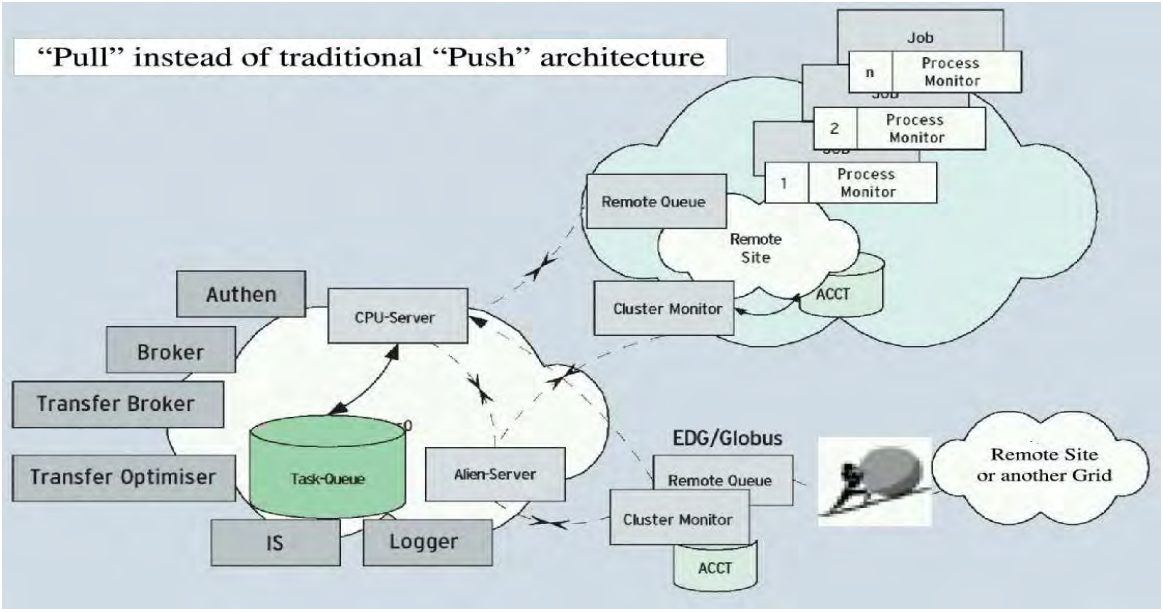


Fig. 5: Job flow architecture

Push – jobs are push to the cluster

Pull – on cluster are run some job agents that take the jobs from the grid queue.

AliEn xrootd

Xrootd is the data storage service with multiple functionality that is used by Alice for data storage.

Tools for serving data are: server demons, clients and xrootd protocol and provides high performance file based access, scale an servers and clients side, separate data flow and control flow.

Xrootd are network and thread management layer, protocol layer, file system layer and storage layer (fig. 6).

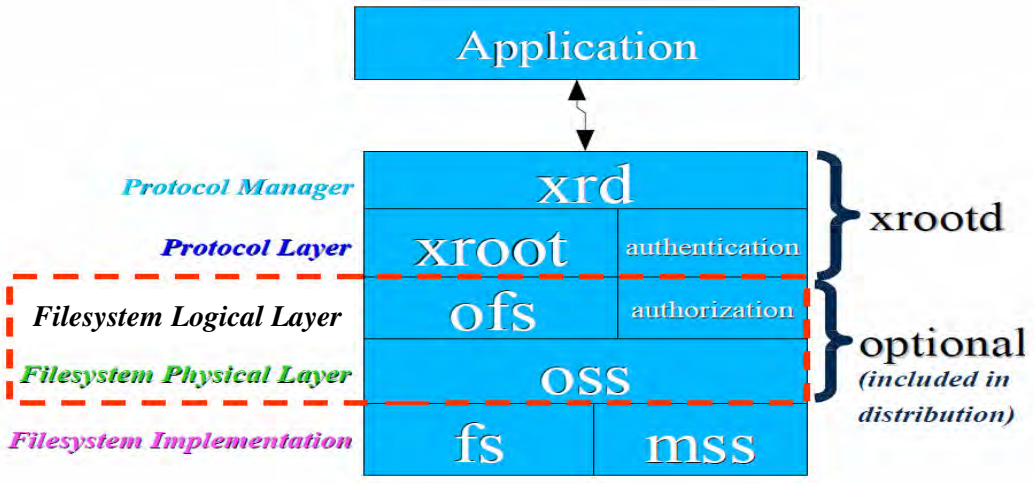


Fig.6: Xrootd Architecture

Hardware used

For Data Center are used servers and network switch from Supermicro: for computing it is used servers like Twin (fig. 7) and Twin^2(fig.8), for storage are used servers with 24, 36,45 trays (fig. 9) and for networking it is used an 10 Gbit as an aggregator and 1 Gbit switches liked with 10 Gbit links to aggregator for worker nodes (fig. 10).

Fig. 7: Servers Twin

Fig. 8: Storages Twin^2

Fig. 9: Storage

Fig. 10: Switch

The main scheme of the ISS Data Center is presented in fig. 11 below.

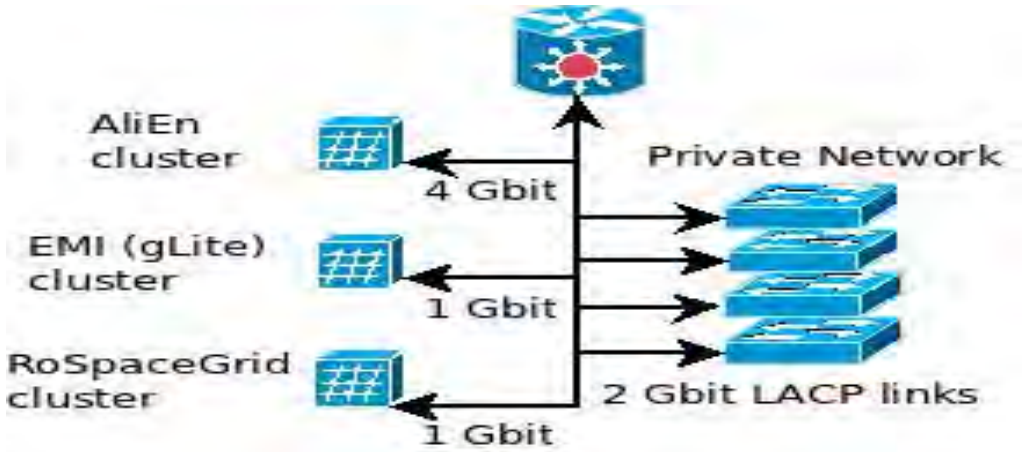


Fig. 11: ISS Data Center Scheme

A center gateway on which are linked the private network of the Institute and also of the cluster. The AliEn cluster has 4 Gbit link to the gateway and the others have 1 Gbit link.

LACP (Link Aggregator Control Protocol) provide an method that control the bounding of several physical ports together to form an single logical channel.

The nodes have Intel and AMD CPU at 1.8 – 2.3 GHz with 4 cores, HDD WD WD2002FYPS Enterprise on 2 TB and 2 Gb Ram per core.

The total power for our cluster is 400 cores with 2Gb Ram per core and 140 TB space.

Performance at ISS

We have 31% of the total number of jobs in Romanian Tire 2 Federation, 1% of Alice Computing (last year period) and 10 GigE upstream connection to DANTE network through RoEduNet.

In the last five years ISS had a total of 1270873 jobs done, with a minimal value of 2377 an average of 320264 jobs

On our storage devices we had a total of 238 TB of traffic data, with an average of 4.892 MB/s, while data sent from ISS totaled at 1.827 PB, with an average of 38.47 MB/s.

Conclusions

In this paper we presented the design and facilities of the Rocks Clusters Distribution and the underlying technologies employed at the ISS Data Center.

We also have a high availability – 88.68% of the total time, success ratio – 88.47% and high usage – 97.96%.

TORRENT BASE OF SOFTWARE DISTRIBUTION BY ALICE AT RDIG

V. Kotlyar[2], E. Ryabinkin[3], G. Shabratova[1], I. Tkachenko[3], A. Zarochentsev[4]
[1] *Laboratory of High Energy Physics, Joint Institute for Nuclear Research, Dubna*
galina@mail.cern.ch
[2] *State Research Center of Russian Federation Institute for High Energy Institute for High Physics, Protvino, Russia*
Victor.Kotlyar@ihep.ru
[3] *National Research Center "Kurchatov Institute", Moscow, Russia*
gridops@grid.kiae.ru
[4] *Saint-Petersburg State University, Saint-Petersburg, Russia*
andrey.zar@gmail.com

The experience of few RDIG sites in the implementation of such service for processing ALICE jobs will be presented in this report.

The GRID framework of LHC experiment ALICE – AliEn [1] is an open source framework built on Web Services and a Distributed Agent Model. In this model Job Agents are submitted onto a grid site to prepare the environment and pull work from a central task queue located at CERN. The communication between each ALICE site and central ALICE services is realized by ALICE-specific VO box. This is a single point contact. The deployment of job-specific software was performing from early AliEn days via PackMan [2]. This service at VO box simplifies deployment of job software, done onto a shared file system at site, and adds redundancy to the overall GRID system. Last year there was developing, testing and implementing a peer-to-peer method [3] based on BitTorrent for downloading job software directly onto each worker node at several ALICE sites. Today the main part of sites supporting ALICE migrates to the pear-to-pear download of application software.

PackMan usage for deployment of job-specific software

PackMan is a Transport Package building tool for packing up Templates, TVs, Snippets, Chunks and other Packages into a Transport Package. This software packages enables users to easily install and remove software on Linux. In case of PackMan application for ALICE GRID sites operate in such way:

- Jobs request Soft Ware from VO box service;
- VO box PackMan service pulls Soft Ware;
- Soft Ware deployed on shared area;
- Working Nodes read Soft Ware from shared area.

Figure 1 gives a scheme of traditional PackMan usage for software transfer to sites. This scheme has some advantages and disadvantages.

Advantages

There is necessary only one service/site managing for installation of require packages. The routine software builds with catalog & stored in AliEn is managed by Central Software.

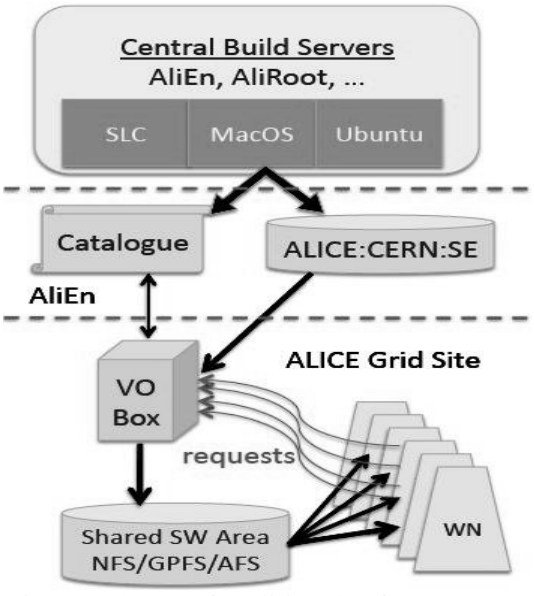


Fig. 1: Scheme of traditional software transport to working nodes with PackMan help

Disadvantages:

- Shared software area is a single point, which is a source of failure / bottleneck
- It is not simple to redeploy rebuilds of the same version. This can require active repairs per site
- Need to keep a short list of active software packages.

Peer-to-Peer method

Of the many p2p file-sharing prototypes in existence, Bit-Torrent is one of the few that has managed to attract millions of users. BitTorrent relies on other (global) components for file search, employs a moderator system to ensure the integrity of file data, and uses a bartering technique for downloading in order to prevent users from free riding. This method has been proposed by ALICE for deployment of job-specific software.

Basic Torrent details

The basic architecture of p2p data and principal scheme of operation with these data presented on Fig 2 and Fig 3 correspondingly. There are using a such definitions used in torrent method:

Tracker: map of seeders: files
Seeders: have & serve file
Leeches: pull & serve file chunks

In order to provide data integrity, file chunks contain hashes of original file.

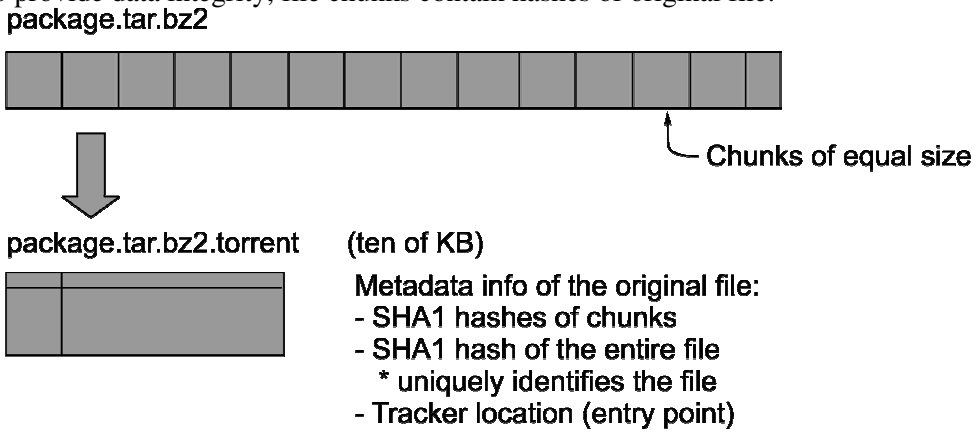


Fig. 2: Structure of p2p data

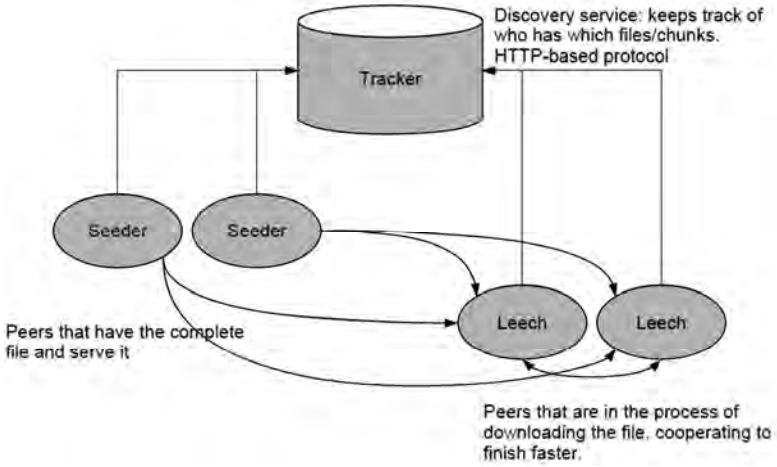


Fig. 3: Scheme of p2p operation.

Implementation pear-to-pear in GRID infrastructure of ALICE – AliEn

Fig 4 presents a principal scheme of pear-to-pear operation for uploading application software to working nodes of site.

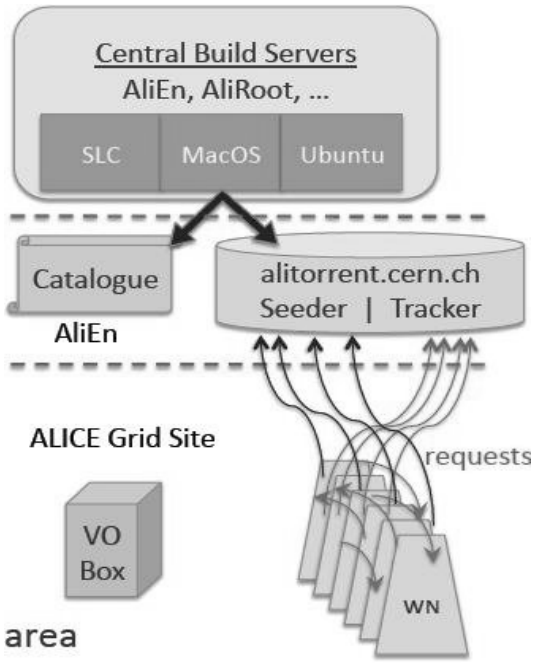


Fig. 4: Pear-to-pear upload of job-specific software

There are new duties applying to manage Central Software as according to p2p scheme VO box from the site side is not involved in the software deployment:

- New AliEn torrent storage has been added for management of p2p scheme.
- In addition to Catalogue there has been stored also seeder & tracker information.

Jobs now pull software from several sources:

a) central alitorrent.cern.ch seeder,
b) other worker nodes will fetch the content mostly from local nodes, if available,
c) worker nodes from site A are usually firewalled from site B, so no inter-site traffic,
d) if initial download is not possible via torrent, fall back to wget and then seed the fetched files.

In addition there is using special features of area2c by ALICE:

i) DHT (Distributed Hash Table) which let decentralize distributed lookup system,
j) Peer exchange. In this case information about local peers will be quickly propagated between peers,
k) LPD (Local Peer Discovery) with application of multicast mechanism to find out other peers in the local network.

So according to all these features, the system can work even without access to the central seeder and tracker.

For successful operation of p2p scheme, the following requirements have been applied to Fire Wall:

- Outgoing access from the WNs to alitorrent.cern.ch:{8088, 8092},
- Please don't allow incoming connection requests from the world to the WNs. But don't be surprised if they do talk to other outside nodes (users that have the package...),
- Allow WN-to-WN connections on at least by TCP,UDP/6881:6999 – aria2c listening ports and UDP,IGMP → 224.0.0.0/4 – local peer discovery.

These "tools" have been integrated in the *alien-installer* and http://alimonitor.cern.ch/packages. The usage of p2p upload (download) is activated by a flag in LDAP. This flag switches modes:

name=<CE_NAME>,ou=CE,ou=Services,ou=<SITE>,ou=Sites,o=alice,dc=cern,dc=ch
installMethod=Torrent

Some practical remarks

The volume of transported software does not exceed 400Mb/job (AliEn itself is packaged in a small (35MB) archive, AliRoot, Root & deps. : max. 300MB/job).

Network load is not so large. CERN seeder limited to 50MB/s. In practice the machine has an average of 8MB/s outgoing. So the fraction that goes to any particular site is negligible.

AliTorrent Software Deployment Advantage:

- Reduces problems associated with SW deployment
- Simplifies site operations by removing the PackMan VO box service. This action does not eliminate VO box model from ALICE Grid. It does eliminate site-specific VO box requirement
- Elimination of site-specific VO box allows for remote use of other Grid resources (for example OSG)
- Eliminate Bofleneck & single point failures

Applications:

Torrents@ALICE: technical details (Experience of RRC-KI and IHEP sysadmins):

Preface

Outline some practical points of using Torrent-based software distribution for ALICE VO as they are seen from the prospective of our Tier-2 site (RRC-KI).

We have been running Torrent-based software distribution since October 2011, so we have around a year of experience with this scheme and so far we had seen no major troubles connected with Torrents@ALICE.

If you have any questions, corrections, suggestions or other stuff; do not hesitate to ask, either during the presentation or by e-mail: gridops@grid.kiae.ru

Software

1. ALICE uses aria2c client, http://aria2.sourceforge.net/.
2. Job downloads the package with Torrent client from ALICE HTTP server (http://alitorrent.cern.ch/), unpacks and starts it.
3. Torrent description files (.torrent) are downloaded from the same server.
4. Torrent client ends when the job ends and we have a set of torrent downloads per each job: no shared cache.
5. Downloaded data lives inside the working directory of the job; in the case of CREAM CE it is CREAMxxxx directory that is removed automatically by the wrapper script.
6. The first downloaded item is the slim AliEn package for the LCG worker nodes.
7. After this, the pilot code is started and the usual sequences of operations are performed.
8. Local PackMan uses Torrent for downloading the needed software packages.

Firewall rules

Just as per aria2c manual, http://aria2.sourceforge.net/aria2c.1.html

```
# Actual downloads.
$IPTABLES -p tcp -m multiport --source <WNs> --dports 6881:6999 -j ACCEPT
# Distributed hash table.
$IPTABLES -p udp -m multiport --source <WNs> --dports 6881:6999 -j ACCEPT
# Peer discovery via multicast.
```

$IPTABLES -p udp --source <WNs> --destination 224.0.0.0/4 -j ACCEPT
$IPTABLES -p igmp --source <WNs>--destination 224.0.0.0/4 -j ACCEPT

Moving a site to the Torrent-based scheme

1. Tune the firewall on the WNs (and only them).
2. Announce to the ALICE mailing list. alice-lcg-task-force@cern.ch, that you're up to using Torrents.
3. Install the up-to-date AliEn on the VO-BOX locally (not on the shared file system). Of course, you can still use the shared file system, but there is no point in doing so if you have a reliable local disk.
4. LDAP entry for your site will be changed to use Torrents at PackMan.

Moving a site to the Torrent-based scheme

1. After LDAP modification it is wise to check that
2. you have no host entry
3. you have no forbidWnInstall entry in the active PackMan leaf for your site:
4. AliEn LDAP lives at ldap://aliendb06a.cern.ch:8389,
5. you're site's leaf is ou=<SITE>,ou=Sites,o=alice,dc=cern,dc=ch
6. PackMan sub-leaf is name=<NAME>,ou=PackMan,ou=Services
7. NAME is the value of attribute packman from the sub-leaf host=<VO-BOX FQDN>,ou=Config.
8. Now watch for new ALICE jobs and aria2c processes at your worker nodes.
9. When things are settled, you can get rid of the VO shared area for ALICE completely.
10. You're done.

Strong and weak points

1. Software distribution was slimmed down (fits in 1 GB per job) –**good**
2. Local disks are used for the software – **good**
3. Once you have some ALICE jobs at your cluster, torrent downloads are blazingly fast – **good**
4. You have multiple copies of software at a single node – **bad**
5. There is a limit for Torrent download rates (1 MB/s), so it will not fill up the network pipe – **good**
6. No monitoring what was downloaded and how fast – **bad**

References

[1] P. Saiz, et al., AliEn –*ALICE environment on the GRID*, Nucl. Instrum. Meth., A502 (2003) 437.

[2] http://packman.links2linux.org/

[3] R.J Porter, I.Saketer, C. Grigoras, et al, *Employing peer-to-peer software distribution in ALICE Grid Services to enable opportunistic use of OSG resources,* contribution 499 at CHEP2012, New-York, 2012; P. Saiz, et al, *AliEn: ALICE Environment on the GRID***,** contribution 516 at CHEP2012, New-York, 2012.

VIRTUAL ACCELERATOR: GRID-ORIENTED SOFTWARE FOR BEAM ACCELERATOR CONTROL SYSTEM

N.V. Kulabukhova[1], A.N. Ivanov[1], V.V. Korkhov[1], D.A. Vasyunin[1,2], S.N. Andrianov[1]

[1] *St.Petersburg State University, Universitetsky pr. 35, Peterhof, 198504, St. Petersburg, Russia, tel/fax: +7 (812) 428-71-59*

[2] *University of Amsterdam, Science Park 904, 1098 XH Amsterdam, the Netherlands*

The key idea of the Virtual Accelerator (VA) concept is the modeling of beam dynamics with the help of several software packages, such as COSY Infinity, MAD, etc., composed in pipelines and enacted on grid-enabled distributed computing resources.

The main use of the VA is simulation of beam dynamics by different packages with the opportunity to match the results (in case of using different solution methods for the same problem) and the possibility to create pipelines of tasks when the results of one processing step based on a particular software package can be sent to the input of another processing step.

The VA is considered as an information and computing environment and does not refer to real-time control systems. However, real-time control can be provided by connection to specialized software (e.g. Experimental Physics and Industrial Control Systems – EPICS). Such kind of VA is examined in [1] and [2] where authors emphasize on accelerator control development.

The general idea of the software implementation is based on the Service-Oriented Architecture (SOA) that allows using grid and cloud computing technologies and enables remote access to the information and computing resources. Distributed services establish interaction between mathematical models and a low-level control system.

The VA user interface allows getting solutions both from simulation models and from real accelerator machines. This approach gives researchers ability for system identification, parameter optimization, and result verification, which is impossible without computational models. The same approach to develop a virtual laboratory is discussed in paper [3] for nuclear physics applications.

The LEGO paradigm is used for the VA design. In terms of information technology it corresponds to object oriented design and component programming. Each object is represented as an independent component with own parameters and behavior. In paper [4] development of distributed computing systems based on this concept is examined in more detail.

Introduction

Contemporary control systems of complicated physics facilities, such as different accelerator complexes, thermonuclear reactors, etc., assume to use efficient scenarios to support operating mode. The development of workflows maintaining the work of different facilities is based on clear formalized mathematical models, describing appropriate processes on the one hand and effective software implementation on the other. The complexity of such facilities makes the models multicompound, and leads to the set of mathematical methods and formalizations. Variety of models and the computational complexity encourage to use distributed environment and appropriate methods, in particular Grid- and Cloud-technologies. It is necessary to distinguish two components: modeling (physical or mathematical) and software approach. The first one is to preliminary investigate (theoretically or experimentally) different effects of the installation. As the result the scenario for the control system of real facility if formed. The second component is responsible for the realization of the scenarios to achieve required operating modes.

In order to control large-scale accelerators efficiently, a control system with a virtual accelerator model was constructed by many facilities. In many papers by the notion of Virtual Accelerator an on-line beam simulator provided with a beam monitor scheme is meant. It works in parallel to real machine. The machine operator can access the parameters of the real accelerator through the client and then feed them to the virtual accelerator, and vice versa. Such a virtual machine

scheme facilitates developments of the commissioning tools; enable feasible study of the proposed accelerator parameters and examination of the measured accelerator data. That is the common scheme of virtual accelerators used in different laboratories. Until now there is no virtual accelerator working without a real machine. Our goal is to construct a Virtual Accelerator application that can be used independently of any machine.

Virtual Accelerator construction

The Virtual Accelerator is considered as a set of services and tools enabling transparent execution of computational software for modeling beam dynamics in accelerators on distributed computing resources. Users will get the access to VA resources by unified interface including GUI on different platforms. Figure 1 shows the scheme of VA.

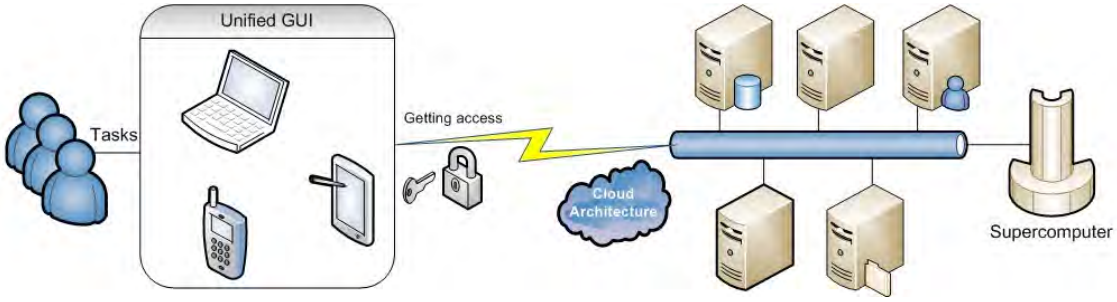


Figure 1: Schematic view of Virtual Accelerator environment

Virtual Accelerator as a computer model

Overview of the current literature allows us to formulate the following definition of the concept of Virtual Accelerator.

The key idea of Virtual Accelerator concept is the beam dynamics modeling by a set of several packages, such as COSY Infinity, MAD, etc., based on distributed computational resources, organized on Grid- and Cloud- technologies.

The main purpose of this environment is to conduct numerical experiments required to configure the actual physical facility during its start-up and ensure it is working properly in the relevant experiments. This purpose imposes certain restrictions on the resource potential of this complex.

Virtual Accelerator as a theoretical model

The main purpose of a virtual accelerator is to conduct computational experiments to simulate the beam dynamics using various software packages with the ability to compare the results of calculations (in case solutions of the same problem are obtained by various means), and the ability to create a task flow (solutions of one package can be used as input for the subsequent calculation in another package), see figure 2. In addition, an important part of the computational experiments are optimization problems sweeping over the possible parameters and system configurations in order to find the best option. In this case, technology of parallel computing and massively parallel computing systems can be used efficiently. Each of the above configurations may be executed independently of the others, along with them, together forming a parametric study of a given domain configurations of the accelerator.

The user has access to the resources of the virtual accelerator through a "single window" – a portal or some interface shell (for example, applications based on Java Webstart). In this interface the user selects a package or several packages in which wishes to carry out calculations, set the input data and parameters, and the task is run on the available computing resources. Access to resources can be provided on the basis of standard solutions used in the Grid-technology.

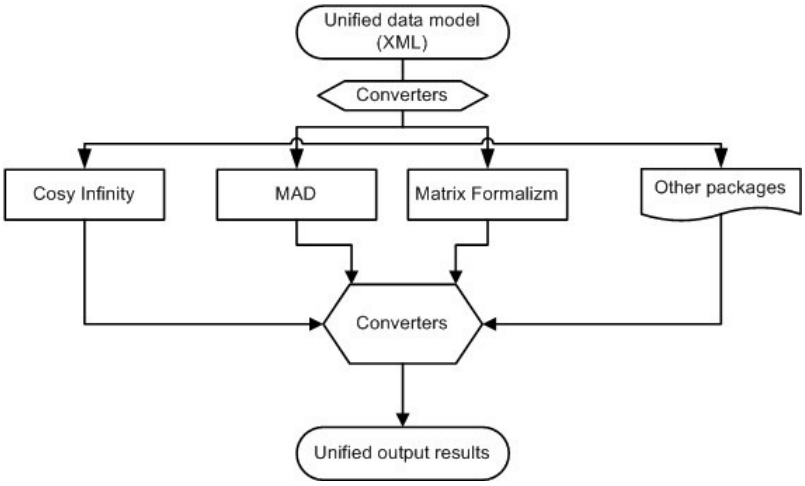


Figure 2: Computational experiment in VA

In this case, the resources to perform the calculations are taken from the common pool. At the time of peak load with a lack of resources a hybrid approach can be used, where the missing resources are taken from the Cloud [5]. Workflow management systems are required to control execution of workflows [6], as well as the conversion of data between the formats used by different packages must be taken into account. This requires the design of the experiment description language that can be translated into the language of each package for intermediate calculations.

Template using a Virtual Accelerator is as follows:

1. The user has access to the interface computer (authentication, authorization);
2. The user sets the initial conditions and parameters for the calculation. The options are:
   1) using "generic" description language that can be converted into a specific language used in the packages MAD, COSY, etc.;
   2) directly in the language of one of the packages, which will be used for calculations;
3. The user selects a package (or a set of packages) which will perform calculations;
4. The user instructs the system to run calculations using the packages and given initial data. This is done using either a dedicated resource (cluster, computer), which is selected manually or automatically selected resource based on information about requirements of the application.
5. After starting the calculation the user may wish to see the intermediate results. Depending on the abilities of packages, it may be possible to do it or not.
6. Most important thing is to track errors that occur. As practice shows, the most difficult is to figure out why something does not work. Carefully organized collection of error messages must be maintained. To collect this information, annotate data and results of computations so called provenance systems are used [7].
7. VA offers the possibility of organizing the flow of tasks – sequential running packages, where the next step uses the data obtained on the previous step. The means to convert data formats between packages must be provided.
8. Calculation results can be visualized by means of VA. It is particularly important to be able to visually compare the results of calculations of a problem in different packages.
9. A simple way to be able to restart the same calculation after a minor adjustment of parameters (which may be carried out after the analysis of the results).

It must be emphasized that the Virtual Accelerator is a modeling environment and is not directly related to the real accelerator control systems (eg, EPICS). However, the organization of communication between such systems is possible.

The task of running diverse software packages that have different requirements for the installed operating systems, libraries and other dependencies can be simplified by using the technology

of cloud computing. In this case, the virtual machine images ready to set up and configured simulation package can be deployed on provided computing resources. In addition, the use of Cloud enables the experiment in case of lack of computing resources in the Grid, as well as in the mode of "urgent computing", when it is necessary to get the results to a pre-set time.

Conclusions

This paper presents a prototype Virtual Accelerator environment used for modeling beam dynamics with the help of a number of software packages on grid-enabled distributed computing resources. We present some design concepts, discuss usage scenarios and prototype implementation. Some modules such as global optimization tools, simulation and numerical algorithm are completely developed, other are in a progress. The future development of the research can be based on writing software using different parallel techniques and complete implementation of the described approaches.

Some approaches that were described above were tested in the distributed computational environment at the faculty of Applied Mathematics and Control Processes on the department of Computer Modelling and Multiprocessor Systems.

References

[1] P.C. Chiu, C.H.Kuo, Jenny Chen, Y.S. Cheng, C.Y.Wu, Y.K.Chen, K.T. Hsu, “Virtual Accelerator Development for the TPS”, IPAC’10, Kyoto, Japan 2010, WEPEB019, p. 2728, http://www.JACoW.org.

[2] C. Gulliford, I. Bazarov, J. Dobbins, R. Talman, N. Malitsky, “The NTMAT EPICS DDS Virtual Accelerator for the Cornel ERL Injector”, IPAC’10, Kyoto, Japan 2010, WEPEB022, p. 2734, http://www.JACoW.org.

[3] V. Korkhov, D. Vasyunin, A. Belloum, S. Andrianov, A. Bogdanov, “Virtual Laboratory and Scientific Workflow Management on the Grid for Nuclear Physics Applications”, Proc. of the 4th Intern, Dubna, Russia 2010, p. 153.

[4] N. Kulabukhova, A. Ivanov, V. Korkhov, A. Lazarev, “Software for Virtual Accelerator Designing”, 13th International Conference on Accelerator and Large Experimental Physics Control Systems: Proceedings of ICALEPCS2011, Grenoble, France, WEPKS016 p. 816.

[5] [5] Rudolf Strijkers, Willem Toorop, Alain van Hoof, Paola Grosso, Adam Belloum, Dmitry Vasuining, Cees de Laat, Robert Meijer, "AMOS: Using the Cloud for On-Demand Execution of e-Science Applications, " in IEEE e-Science 2010 Brisbane, Australia: IEEE Computer Society, 2010.

[6] A. Wibisono, V. Korkhov, D. Vasunin, V. Guevara-Masis, A. Belloum, C. de Laat, P. Adriaans and L.O. Hertzberger, WS-VLAM: Towards a scalable workflow system on the Grid, Proceeding of the 16th IEEE International Symposium on High Performance Distributed Computing, June 27-29, 2007, Monterey Bay, California, USA.

[7] Michael Gerhards, Volker Sander, Torsten Matzerath, Adam Belloum, Dmitry Vasunin, Ammar Benabdelkader. Proceeding WORKS '11 Proceedings of the 6th workshop on Workflows in support of large-scale science Pages 57-66, ACM New York, NY, USA ©2011.

DISTRIBUTED TRAINING AND TESTING GRID INFRASTRUCTURE EVOLUTION

N.A. Kutovskiy[1,2]

[1] *Laboratory of Information Technologies, Joint Institute for Nuclear Research, Dubna*
[2] *National Scientific and Educational Centre of Particle and High Energy Physics of the Belarusian State University, 220040, Minsk, Belarus*
nikolay.kutovskiy@jinr.ru

During the last two years a distributed training and testing grid infrastructure (t-infrastructure) has been used for new several activities such as development of the monitoring tools for ATLAS Tier-3 sites and problem-oriented interfaces development for particular applications for the Russian grid network. Apart from that a few more grid sites were deployed at some organizations of the JINR Member States and integrated into the t-infrastructure. Ongoing activities and future plans are covered as well.

GLite/EMI-based testbed

In addition to the set of the grid sites mentioned in [1] two more grid sites, namely, KZ-ENU and UA-ILTPE hosted at L.N. Gumilyov Eurasian National University (Astana, Kazakhstan) and B.Verkin Institute for Low Temperature Physics and Engineering (Kharkov, Ukraine) respectively had been integrated into a gLite/EMI-based testbed of the t-infrastructure in 2011 (see Fig. 1). The deployed services at each of the recently added sites are given in Table 1.

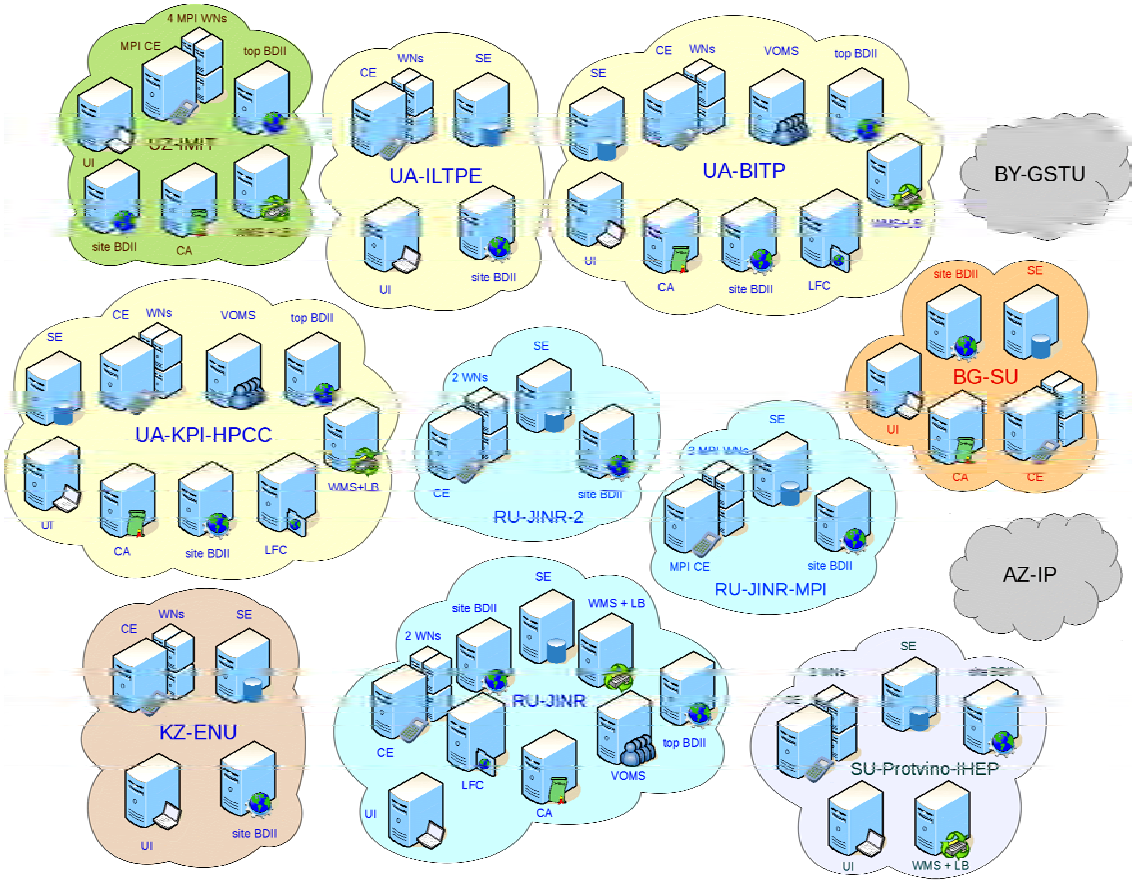


Fig. 1: A schema of the distributed training and testing grid infrastructure based on gLite/EMI middleware

Table 1. List of the deployed services at the recently added sites and their hosting organizations

site name	hosting organization	services
KZ-ENU	L.N. Gumilyov Eurasian National University (Astana, Kazakhstan)	UI, CREAM + 4 WNs, sBDII, DPM SE
UA-ILTPE	B.Verkin Institute for Low Temperature Physics and Engineering (Kharkov,Ukraine)	UI, CREAM + 2WNs, sBDII

An activity on the basic set of EMI services (UI, CREAM + some WNs, sBDII and DPM SE) deployment on the AZ-IP site hosted at the Institute of Physics (Baku, Azerbaijan) is on the closing stage. It is planned to integrate it into the t-infrastructure in the nearest future.

Besides, some preparation work has been done to install a grid site at the Pavel Sukhoi State Technical University of Gomel (Gomel, Belarus).

Testbed for the monitoring tools development for ATLAS Tier-3 sites

According to the ATLAS Tier-3 sites survey [2], several types of the local resources management systems (LRMS) and mass storage systems (MSS) are used on ATLAS Tier-3 sites:

- LRMS: PROOF, PBS, Condor, Oracle Grid Engine (OGE), LSF;
- MSS: XRootD, dCache, DPM, NFS, GPFS, Lustre.

Only few of these components have the embedded means able to provide a needed information for Tier-3 site administrators. Moreover, there is a necessity to aggregate Tier-3 monitoring information on the global VO level. Thus a monitoring suite has to be developed to fill that gap. In order to do that, an ATLAS Tier-3 monitoring task force was established [3] and a Tier-3 monitoring software suite (T3MON) proposal was presented [4].

The development of the software suite for the local site monitoring assumes the following activities:

- validation of the existing monitoring tools for each component in use,
- development and debugging of new monitoring tools.

The activities listed above imply the following:

- deployment of a separate testbed for each LRMS and MSS reported as being used on ATLAS Tier-3 sites,
- Ganglia server deployment (to gather, store and preview monitoring information on the testbed Ganglia monitoring system was chosen based on the survey results),
- Ganglia agents installation and configuration for a specific testbed,
- installation and validation of the additional Ganglia plug-ins for monitoring metrics collection as well as non-related to Ganglia monitoring tools.

Due to the reasons listed below, virtualization can be applied:

- 24/7 availability of the testbeds components with different LRMS and MSS running in parallel on dedicated physical servers would causes a sufficient hardware capacities;
- monitoring tools deployment and development as well as testbeds operation may require redeployment of a certain testbed or its parts;
- testbeds performance is not a critical issue for such tasks.

Virtualization allows a significant increase in the effectiveness of the hardware resources utilization as well as provides an ability to perform quickly and easily such operations as VMs creation from existing images/templates, VMs backup before significant changes and VMs restoration from backup if needed.

Since all components of each testbed can be run on linux (inside VM) as well as a physical servers and most components do not require own kernel extensions, the OS-level virtualization can be used which is more lightweight and faster than full hardware emulation or paravirtualization approaches. However, there are still some components which require own kernel extensions (e.g. Lustre, GPFS).

Among possible candidates, OpenVZ (http://openvz.org) as a solution for virtualization on the OS-level and Xen (http://xen.org) as a product providing a full hardware emulation, were chosen by the following reasons:

- stable and actively developing software with a sufficient tool set for VMs management and monitoring,
- strong and helpful community,
- good documentation,
- free software (GNU GPL license).

Besides, the services of the JINR training and testing grid infrastructure have been successfully running on OpenVZ-based VMs since 2006.

The deployment of the JINR testbed for ATLAS Tier-3 sites monitoring tools development started in February 2011 as part of the t-infrastructure and for the time being the following LRMS and MSS are running on it (see Table 2 and Fig. 2): PBS, Condor, XRootD, Lustre, PROOF, OGE.

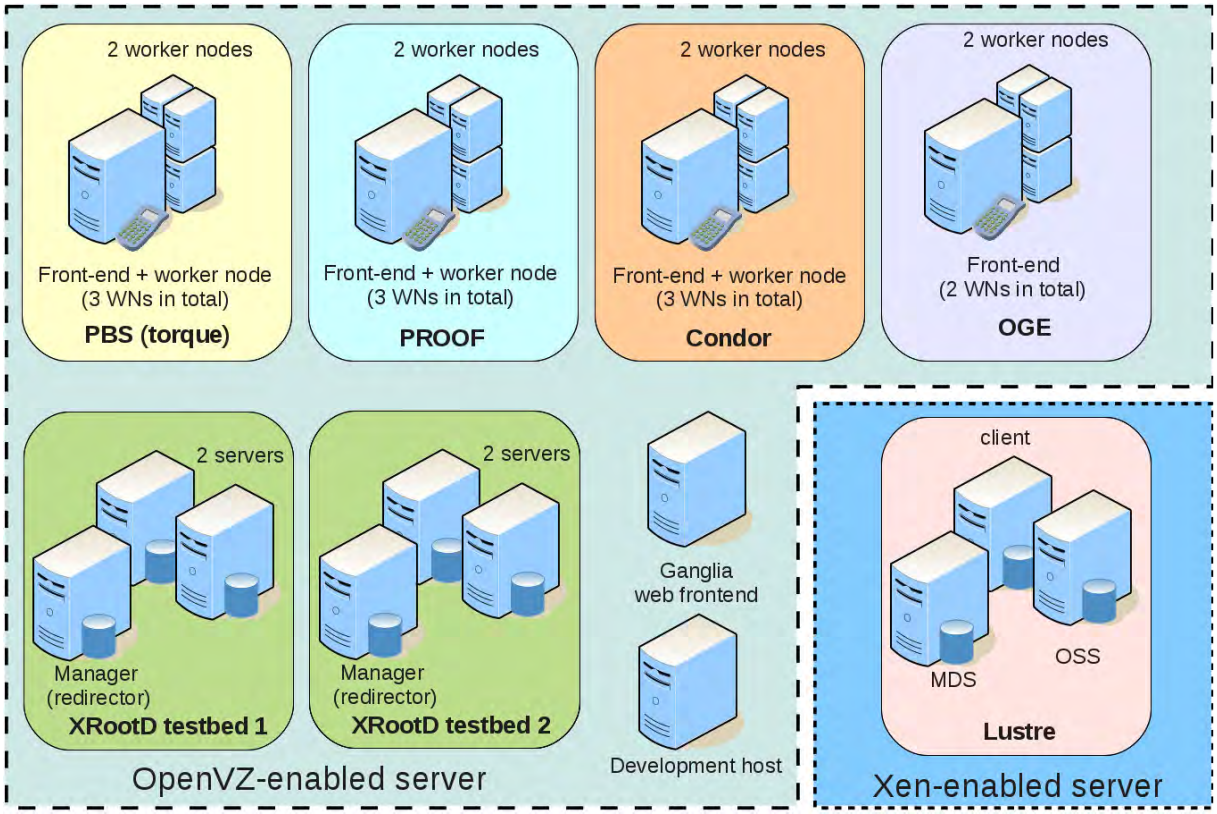


Fig. 2: A scheme of the LRMS and MSS testbeds distribution over the servers

In addition, the Ganglia and development servers are deployed. All testbeds excluding Lustre are running on single OpenVZ-enabled server (Dual-Core AMD Opteron Processor 2220 @ 2.8 GHz CPU frequency, 4 GB of RAM, 8 GB of swap partition, 3 HDDs with 250 GB capacity each combined as RAID0). Lustre services are deployed on Xen-based VMs on a separate server (Intel Core2 CPU 6400 @ 2.13 GHz, 2 GB of RAM and 4 GB of swap partition, single 250 GB HDD).

Table 2. A list of running services of the LRMS and MSS testbeds

Testbed name	Services
PBS	torque headnode (HN) + worker node (WN) + Ganglia (gmond, gmetad, webfronted) + jobmonarch, 2 torque WNs + gmond
PROOF	HN + gmond, 2 WNs + gmond

Testbed name	Services
Condor	HN + WN + gmond, WN + gmond, client + gmond
OGE	HN + Ganglia (gmond, gmetad, webfrontend), 2 WNs + gmond
XRootD 1	manager + gmond, server + gmond, server + gmond
XRootD 2	manager + gmond, server + gmond, server + gmond
Lustre	MDS + gmond + gmetad + gweb + t3mon-site-lustre, OSS + gmond, client + gmond

As a result, a set of Ganglia-based monitoring tools for the most popular LRMS and MSS used on ATLAS Tier-3 sites have been created and tested on the testbed's basis. These tools allow getting information on the Tier-3 sites operation both at local and global levels. Monitoring metrics sent from sites via an active messages queue are collected at CERN and then are presented at Dashboard (http://dashboard.cern.ch/) which is a single entry point to the monitoring data collected from the distributed computing systems of the LHC virtual organizations.

For more details on JINR testbed for ATLAS Tier-3 monitoring software tools development see [5].

Problem-oriented web-interfaces for Russian grid network

To simplify users' work in the Russian grid network (RGN), problem-oriented web-interfaces (POIs) for such applications as DL_POLY (molecular dynamics), Elmer (computer-aided engineering) and GEANT4-DNA (allows one to simulate biological damages induced by ionising radiation at the cellular and sub-cellular scale) were developed by a LIT JINR team in 2011 to use the mentioned software in the grid infrastructure.

POIs are intended to simplify users' work by providing a possibility for them to create and submit jobs relied on particular applications as well as to trace jobs status and get the output. They were developed as plug-ins for RGN graphical web-interface (web-GUI – a special service designed to be a single entry point for users to interact with RGN grid-infrastructure and its services) that was deployed on t-infrastructure and used for development and debugging. DL_POLY, Elmer and GEANT4-DNA applications were installed on a dedicated JINR cluster with 36 cores connected to RGN testbed through the grid gateway.

A screenshot of one of the POI is given on the Fig. 3.

More information on that topic can be found in [6].

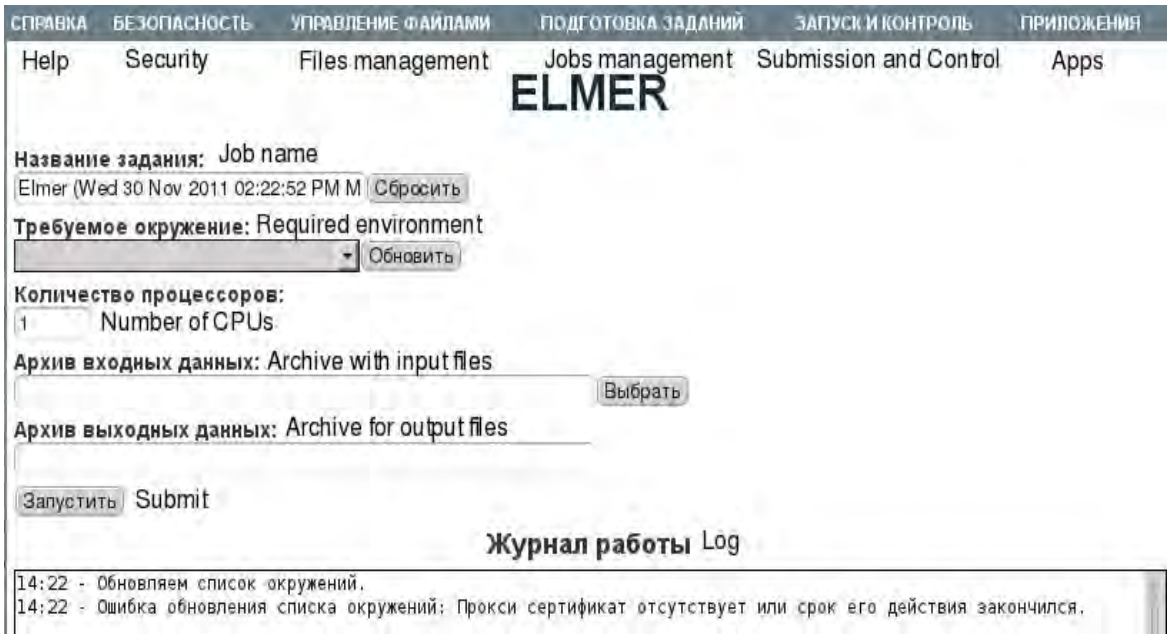


Fig. 3: A screenshot of the POI for Elmer application

Private IaaS

The t-infrastructure is rapidly developed because of being used for more and more tasks. The number of simultaneously running VMs varies from tens to hundred what causes a certain difficulties to manage all of them across physical servers manually. One of the possible solution could be to build a private cloud and migrate VMs into it (i.e. implement "infrastructure as a service approach", IaaS).

OpenNebula (http://opennebula.org/) was chosen as a platform for private IaaS. Since most of the VMs of the t-infrastructure are based on OpenVZ (apart from 3 Xen VMs) and OpenNebula does not support OpenVZ hypervisor, the OpenVZ driver for OpenNebula v2.2.1 has been developed in 2011 and recently updated by the team from Kyiv Polytechnic Institute (KPI) to the latest stable OpenNebula version 3.6. After some minor fixes and improvements are applied, it is planned to build a local private cloud at JINR and to migrate the t-infrastructure into it.

DesktopGrid segment

To use idle CPUs resources of the desktop PCs at JINR and organizations from its member states for computational tasks, one needs:

- to build a desktop grid (DG) infrastructure,
- to adopt applications to it.

A last item requires a testbed which can be deployed on the t-infrastructure and integrated within its gLite/EMI-based segment.

A first attempt to build a local DG testbed was undertaken in 2010.

A small DG testbed (BOINC server and few BOINC clients on the University Centre of JINR's PCs) was deployed .

To integrate DG segment into gLite/EMI-based testbed, a few more services need to be setup:

- bridge (3G-bridge),
- application repository (AR),
- gridftp-server.

Due to incomplete documentation and lack of support from the 3G-bridge developers, no much success has been reached yet. However, it is planned to take another attempt in the nearest future.

Trainings

The semestral educational courses have been conducted at the University "Dubna" and at the JINR University Centre. Apart from that, introduction lectures and short-term courses for participants of different international schools and practices were organized among which are the following:

- "JINR /CERN grid and advanced information systems" schools;
- practice for students from JINR member and associated states;
- introduction lecture for participants of training of young scientists from CIS, Russian student scientific and technical school "Personnel of the future".

Training courses for system administrators from JINR Member States have been conducted during the last 2 years as well. They are listed in Table 3.

Table 3. Conducted trainings for system administrators

Organization	City	Country	Dates	type
BITP, KPI	Kiev	Ukraine	05.09.11 - 09.09.11	in person
IP	Baku	Azerbaijan	13.07.11 - 31.08.11	in person
ENU	Astana	Kazakhstan	14.06.11 - 10.07.11	in person
NUM, SICT of MUST	Ulaanbaatar	Mongolia	17.05.11 - 20.06.11	in person
ILTPE	Kharkov	Ukraine	06.12.10 - till now	email

Conclusion

As one can see, there is a strong demand in different grid related activities that require a separate autonomous infrastructure. Following that necessity, a training and testing grid infrastructure was deployed with core services at LIT JINR. It has been successfully used for a wide spectrum of tasks. All its components are running on VMs. To simplify management across the physical servers, there is an activity to build a private cloud based on OpenNebula software and to migrate all t-infrastructure services into it.

There are also plans to upgrade grid services to their latest stable versions as well as to finish the integration of DG and gLite/EMI-based segments.

The ultimate goal is to have an infrastructure what could become a platform for training, research, development, tests and evaluation of modern technologies for distributed computing and data management.

More organizations are welcome to build their grid sites, integrate them into the t-infrastructure and use it for own training, development and other needs.

All t-infrastructure related work in 2011 and 2012 have been supported by the JINR grant for young scientists and specialists.

References

[1] Korenkov V.V., Kutovskiy N.A., “Distributed training and testing grid infrastructure” // Proceedings of the 4rd International conference “Distributed Computing and Grid-technologies in Science and Education” (GRID'2010), Dubna, 2010. P.148-152.

[2] D. Benjamin, Tier 3 Survey results, Report at ATLAS Software & Computing Workshop (29 November 2010 - 03 December 2010), CERN, Geneva.

[3] Brock et al., U.S. ATLAS Tier 3 Task Force, Preprint U.S. ATLAS, 2009.

[4] J. Andreeva et al., Tier-3 Monitoring Software Suite (T3MON) proposal, ATLAS note, 2011.

[5] S. Belov et al., VM-based infrastructure for simulating different cluster and storage solutions used on ATLAS Tier-3 sites, to appear in Proceedings of CHEP2012 conference, New York, USA, May 21– 25, 2012.

[6] Kutovskiy N.A., Lensky I.I., Semenov R.N., “Problem-oriented web-interfaces for Russian grid network”, ibid pp. 186-188.

PROBLEM-ORIENTED WEB-INTERFACES FOR THE RUSSIAN GRID NETWORK[1]

N.A. Kutovskiy[1,2], I.I. Lensky[1], R.N. Semenov[1]
[1] *Laboratory of Information Technologies JINR, Dubna*
[2] *National Scientific and Educational Centre of Particle and High Energy Physics of the Belarusian State University, 220040, Minsk, Belarus*
nikolay.kutovskiy@jinr.ru

To simplify users' work in Russian grid network, problem-oriented web-interfaces for such applications as DL_POLY, Elmer and GEANT4-DNA were developed by a LIT JINR team to use the mentioned software in the grid infrastructure. These interfaces provide a possibility for users to easily create and submit jobs that rely on a particular application. A testbed consisting of computing element, worker nodes and user interface was deployed for development purposes. A description of the developed problem-oriented web-interfaces, approaches applied and the testbed structure are covered.

The development of problem-oriented interfaces (POIs) is one of the tasks in a project established for building the Russian grid network (RGN, http://grid-russia.ru). POIs are intended to simplify users' work by providing a possibility for them to create and submit jobs relied on particular applications as well as to trace jobs status and get the output.

LIT JINR team developed POIs for the following applications:

1) DL_POLY (molecular dynamics),
2) Elmer (computer-aided engineering),
3) GEANT4-DNA (to simulate biological damages induced by ionising radiation at the cellular and sub-cellular scale).

POIs were developed as plug-ins for RGN graphical web-interface (web-GUI – a special service designed to be a single entry point for users to interact with RGN grid-infrastructure and its services) that was deployed on training and testing grid infrastructure (see [1] for more details on it) and used for development and debugging. DL_POLY, Elmer and GEANT4-DNA applications were installed on a dedicated JINR cluster with 36 cores connected to RGN testbed through the grid gateway.

A screenshot of one of the POI is given on fig. 1 and a schema of workflow is shown on fig. 2. All titles of the menu fields on the portal are in Russian but their translations in English are given on fig. 1. First, user has to prepare an archive with input files for application. Then he has to be authen-

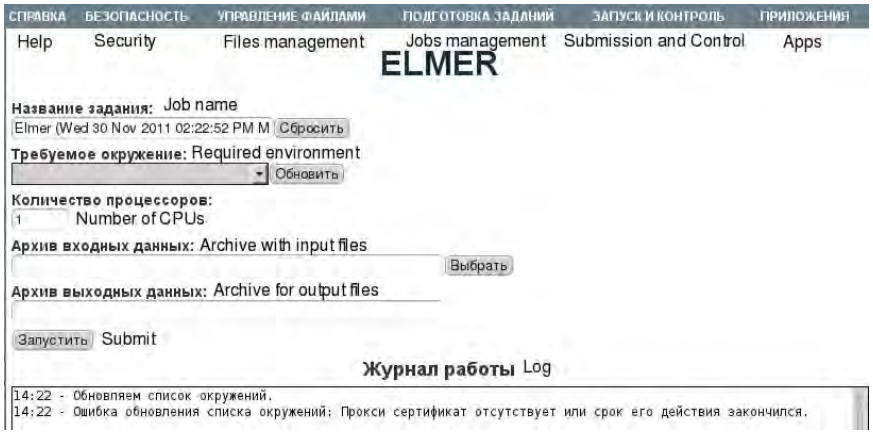


Figure 1: A screenshot of the POI for Elmer application

[1] Supported by the Ministry of Communications and Mass Media of the Russian Federation, cont. № 0173100007512000020_144316

ticated on the RGN portal and choose an appropriate POI in a portal menu. The field of the POI titled as “Job name” is for a job name which initially is generated automatically but can be changed by the user.

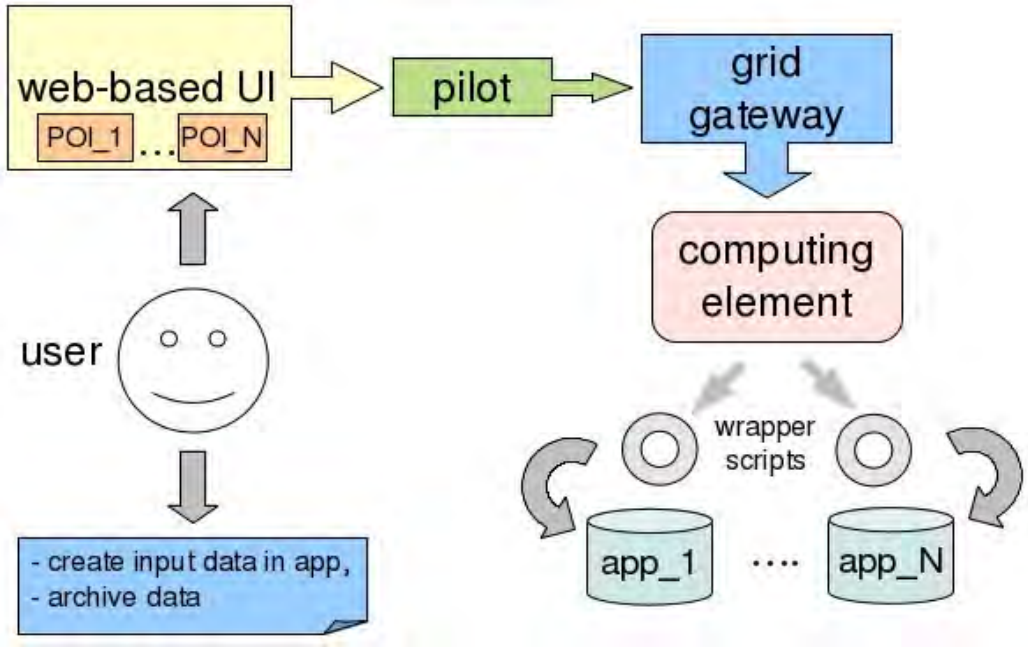


Figure 2: A schema of workflow

The field “Required environment” is for choosing a required environment to be set on the worker nodes of the resource center.

POIs for DL_POLY and Elmer applications have a field “Number of CPUs” where user has to set a number of required cores for MPI jobs (GEANT4-DNA is not MPI-enabled application). An archive with input data has to be prepared in advance in some external application since POIs do not provide such means. The created archive has to be uploaded to the portal in order to become available to be selected as an input file for the job in the field “Archive with input files”. Also the user can define a name for the archive with output files in the field “Archive for output files”. If it is not set then the POI will use a default one – output.tgz. As soon as all necessary values are set, the user has to click on “Submit” button to submit a job. A result of that operation is displayed in the “Log” window. If the job was submitted successfully, then the user can track its status in a corresponding screen which can be chosen in “Submission and Control” menu entry. The output archive of finished job becomes available in the user's home directory accessible through the portal in appropriate window. This is as for users' activities.

The internal workflow is the following. POI reads all user-defined data, generates a job based on them and produces a shell-script which is intended to be executed on the worker node(s) to perform some operations described below. After that job is passed to the RGN workload management service which is called “Pilot”. It tries to find resources matching job requirements (i.e. with specified version of particular application installed, sufficient number of cores, etc) based on the data provided by corresponding RGN information service. If Pilot managed to find such ones then the job is submitted there and the archive with input files as well as a set of scripts including helper shell-script are transferred there too. The helper shell-script is needed because MPI job is executed on the worker nodes with specified by the user number of computational units (CPUs or cores). It's OK if there is no need to do some preliminary and(or) subsequent operations before (after) application invocation on the worker nodes like e.g. extract input data for the archive or (and) pack output data into it. Without taking some additional measures, these pre- and post-operations would be performed on all CPUs

involved in the job execution (e.g. an extraction of the same archive would be performed by several concurrent processes doing exactly the same thing and writing data into same files what is not good). So the helper shell-script performs the following steps:

1) it is executed on the cluster in the number of instances corresponding to the specified by the user number of CPUs needed for the MPI job;
2) each instance of that helper script determines its own id and a master one (with id 0) performs all preliminary operations (e.g. extracting input files from the archive) while other instances are waiting until a certain file (lets call it a "signal file") is created which is a sign that necessary preparation is completed;
3) as soon as the signal file from the master process is created, the required application runs on a specified number of cores and each instance of helper script creates its own signal file whose existence is tracked by the master process;
4) when application is completed, all signal files except master's one are deleted by their parent processes;
5) master instance tracks all signal files and as soon as they disappear it performs post-processing operations (e.g. pack output files into archive).

The helper script was developed because of the lack of grid services functionality. There is another desirable feature to have: a possibility to run parametric jobs (the same executable file of the application is executed multiple times with different values of fixed set of parameters). Although it's possible to do that (e.g. to write a script which can generate one or several files with tasks description) it would be easier for users and developers to enable such functionality by using some extra options in job description file.

Conclusion

There is an activity in Russia to build a national grid infrastructure for high performance computing. One of the task in the framework of the corresponding project is to develop POIs aimed to simplify users' daily work in RGN. POIs for such applications as DL_POLY, Elmer and GEANT4-DNA were developed by LIT JINR. For that purpose the web-GUI service was deployed as a part of the training and testing grid infrastructure and it was able to interact with the RGN services as well as JINR cluster connected to RGN testbed through the grid gateway. Some useful functionality is not yet implemented in RGN services (a possibility to invoke shell-scripts before and after MPI job execution and to run parametric jobs) which could simplify users' and POI developers' work. Although there are few ways to achieve desired behavior hopefully the missing functionality will be realized in the future versions of RGN services.

References

[1] N.A. Kutovskiy, "Educational, training and testing grid infrastructure" // Proceedings of XIV conference of young scientists and specialists (OMUS'2010), Dubna, 2010, P.70-73.

DDM DQ2 DELETION SERVICE. IMPLEMENTATION OF CENTRAL DELETION SERVICE FOR ATLAS EXPERIMENT

D. Oleynik[1], A. Petrosyan[1], V. Garonne[2], S. Campana[2]
on behalf of the ATLAS Collaboration
[1] *Laboratory of Information Technologies, JINR, Dubna*
[2]*CERN, CH-1211 Geneva 23, Switzerland*
danila.oleynik@cern.ch

The ATLAS Distributed Data Management project DQ2 is responsible for the replication, access and bookkeeping of ATLAS data across more than 100 distributed grid sites. It also enforces data management policies decided on by the collaboration and defined in the ATLAS computing model.

The DQ2 Deletion Service is one of the most important DDM services. This distributed service interacts with 3rd party grid middleware and the DQ2 catalogues to serve data deletion requests on the grid. Furthermore, it also takes care of retry strategies, check-pointing transactions, load management and fault tolerance.

In this paper special attention is paid to the technical details which are used to achieve the high performance of service, accomplished without overloading either site storage, catalogues or other DQ2 components.

Special attention is also paid to the deletion monitoring service that allows operators a detailed view of the working system.

1. Introduction

ATLAS experiment [1] has recorded almost 5PB of RAW data since the LHC started running at the end of 2009. Many more derived data products and complimentary simulation data have also been produced by the collaboration and, in total, more than 94PB (300M of files) is currently stored in the Worldwide LHC Computing Grid by ATLAS. All of the data are managed by the ATLAS Distributed Data Management system, called Don Quixote 2 (DQ2)[2].

While deletion which might seem like a fairly straightforward activity on the surface, in a complex distributed environment, such as that managed by DDM, it is far from trivial. Dataset deletion requests on a particular site need to be done with care to ensure that:

- The dataset replica entry is deleted from the DDM central catalog.
- Corresponding files are physically deleted from storage.
- All file replica locations are removed properly from the local file catalog.

Each of these steps might fail, so it is necessary for the deletion service to have an internal state engine, which records the state of deletion for any dataset at a particular site. It is also necessary to throttle both catalog and physical deletion requests for files in order to prevent external services from being overwhelmed. In addition, there is also the additional complexity of overlapping datasets. If two datasets share files on a site and only one of them is deleted then the shared files should not be deleted, otherwise the remaining dataset would become incomplete. This requires some care in mapping dataset deletion requests onto file deletions.

Deletion rates in ATLAS DDM can reach significant levels, with millions of file deletions per day and terabytes of data being cleaned.

2. DQ2 Deletion Service architecture

While the number of stored data is growing steadily (fig. 1), increasing the number of sites involved in the ATLAS data processing, as well as increasing demands on the rate of release of storage space, architecture of Deletion Service shall be capable of scaling.

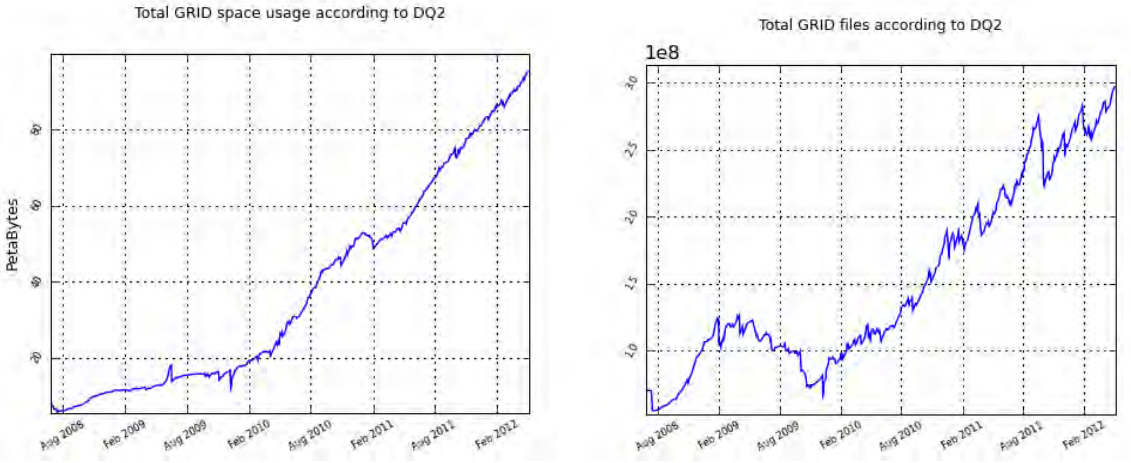


Figure 1: Growing of ATLAS data

To provide required scalability, Deletion Service was developed as a client - server application based on Web-service technology. Main components of deletion service are: Deletion client, Deletion server, Deletion agent and Deletion monitoring. Deletion client/server and Deletion agent together are the core part of Deletion Service. Deletion monitoring is a special web-based application providing reports about deletion process. Schematic interactions between components of Deletion service are shown in Figure 2.

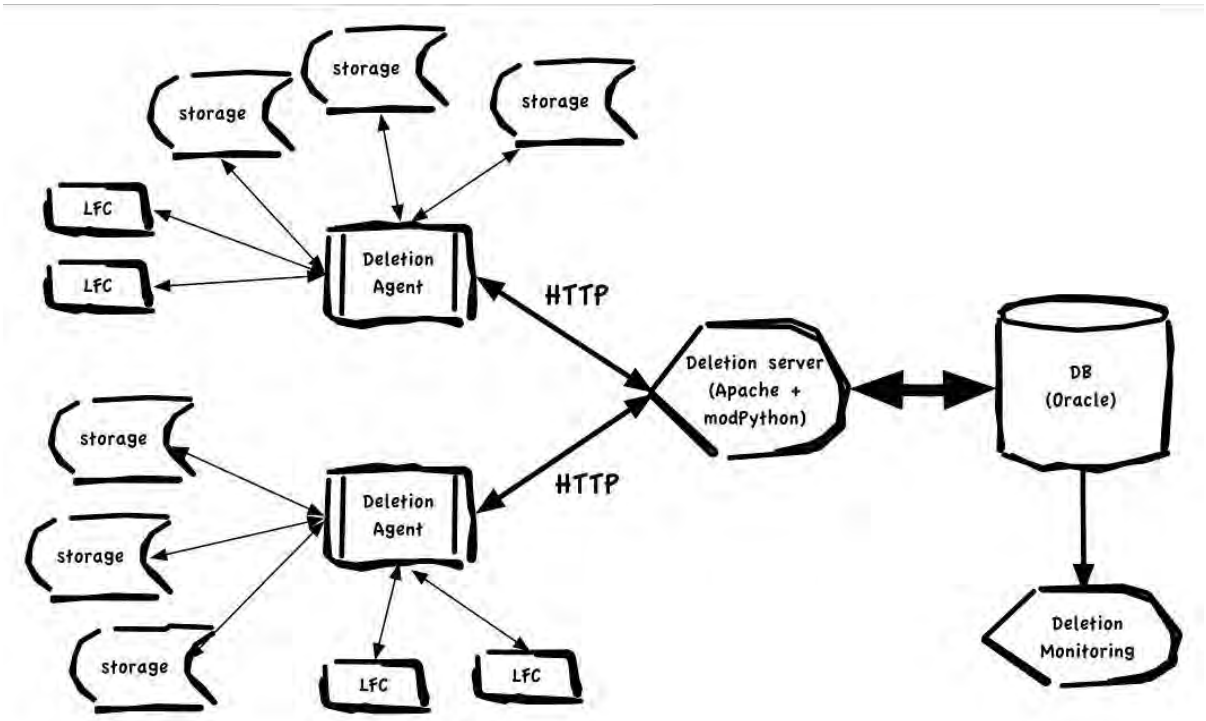


Figure 2: Interactions between components of Deletion Service

2.1. Deletion Client/Server

Server part of Deletion service encapsulates intercommunications with Oracle database backend. Deletion server is based on Apache web-server [3] with mod_python extension [4]. To speed up connections with the database, backend uses the connection pool component [5]. All these solutions allow one to achieve high service performance. This configuration of Apache web-server makes

possible maintaining of multiple concurrent requests to the database with fixed number of open connections to database.

A usual number of processed requests by web server is around 3 million per day. There is no direct relation between performance of Deletion service and load on the Deletion server due to most of instructions are realized as bulk operations.

Deletion client is a specialized http-client designed to simplify interaction with the Deletion server. The client is integrated with the Deletion agent module on API level and completely hides the lower-level protocols.

2.2. Deletion Agent

The main executive component of the Deletion service is Deletion agent. Before describing implementation of the Deletion Agent it is necessary to explain some concepts of DDM. DDM DQ2 realizes data management at dataset level; dataset is a group of logical aggregated files. There could be from one file to dozens of thousands files in a dataset. Deletion request, the initial instruction for the Deletion Service, contains information about which dataset should be deleted from the specified endpoint. Thus, the additional task of the Deletion Agent is defining correct file list for removal.

Thereby, it is possible to mark out three main operations for the Deletion Agent to perform:

1.Resolving list of files for removal

2.Removing files replica locations from LFC

3.Physical files deletion from storage

Three main procedures were developed to implement these functions. The created procedures operate as three independent concurrent processes. The architecture of Deletion agent is schematically shown on Figure 3.

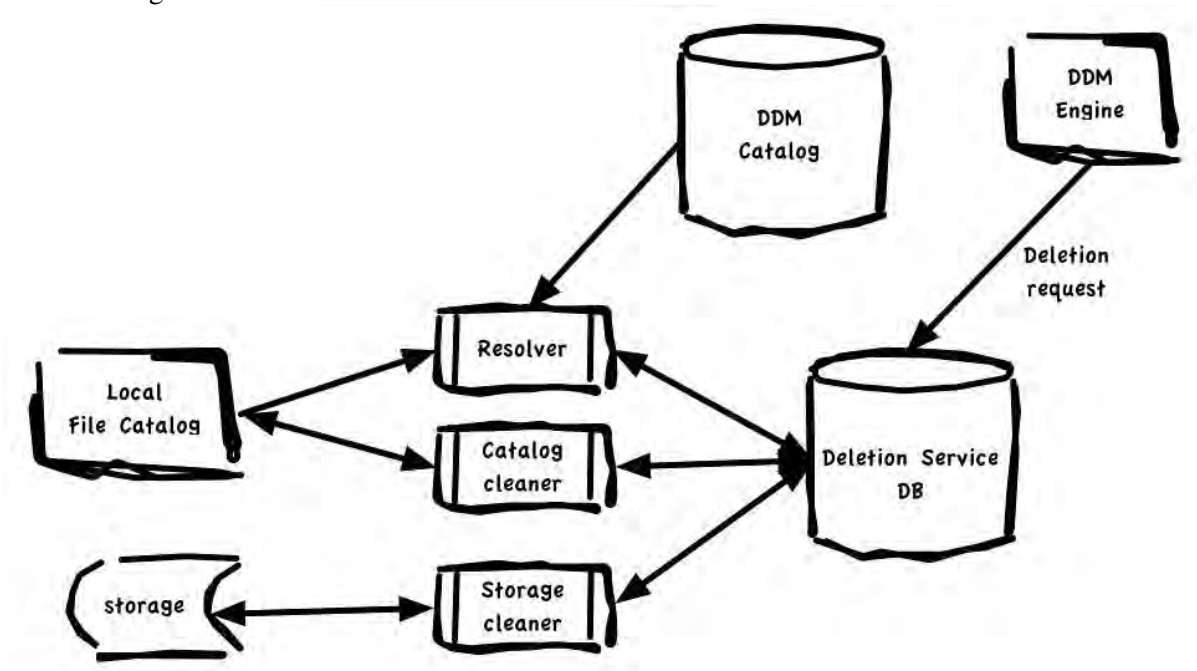


Figure 3: Deletion agent architecture

Procedure called “Resolver” defines the list of files to be deleted according to the request. For obtaining this list the DDM catalog is requested, the request is constructed in such a manner as to receive the list of not overlapped files only. The incoming list contains only names of the files; this information is not enough for deleting data from storage. For getting a full path to files (storage URL, SURL) LFC is requested. All acquired information is saved in a database for the subsequent operation. After this operation deletion request will mark as ready for deletion.

All requests marked as ready to deletion are processed by the procedure that cleans LFC records for the corresponded files (“Catalog cleaner”) and procedure of removing files from storage (“Storage cleaner”). The main algorithm of these procedures is similar: to select the chunk of files from a database, to execute an operation, to analyze the result, to update records in the database. The difference between algorithms consists in the features of interfacing with LFC (the asynchronous interface) and with storage systems (the synchronous interface). The asynchronous interface needs an additional check of operation’s results. Procedures of cleaning of LFC and storage works at file level, it gives the possibility to make checkpoints for each certain file.

“Catalog cleaner” and “Storage cleaner” manage only state of files in the database; the logic of dataset deletion operates by database triggers. This approach allows us to improve the performance of the service, and align deletion rate by eliminating the dependence on the size of the dataset.

To achieve high productivity, Deletion Agent is designed as a multithread application. Each site served by its own copy of the “Resolver”, the “Catalog cleaner” and the “Storage cleaner”. To improve the interaction with the database, bulk operations are used, and size of chunks is configurable for single sites and endpoints.

3. Deletion Monitoring

Deletion service is a high-priority service of ATLAS DDM, but most of the time only one person is engaged to the control over the work of the service. Tracking of removal of more than two hundred sites without the use of a specialized operator’s interface is almost impossible. To provide the operator’s information on how the service works, a special application was created – the system of monitoring of the Deletion Service [6].

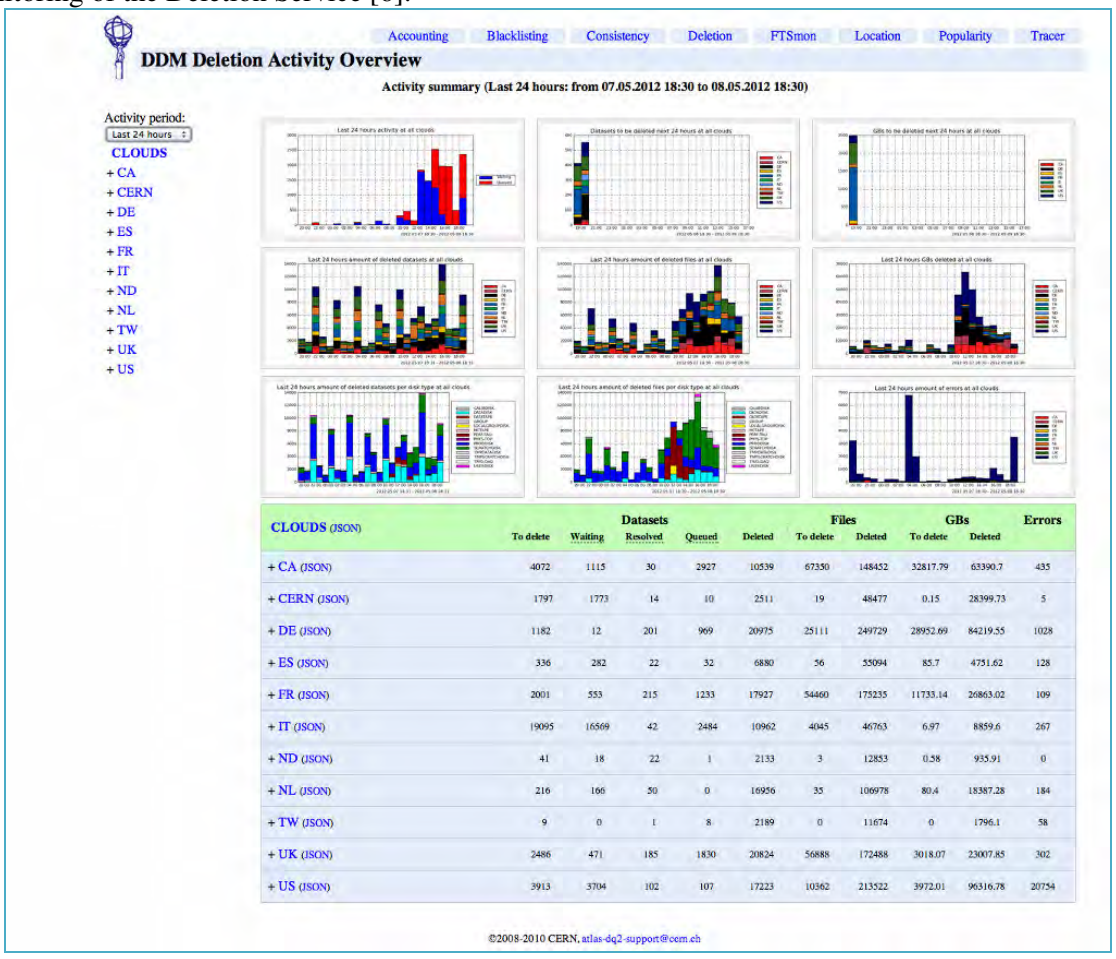


Figure 4: Deletion monitoring (Screen shot)

The monitoring system is implemented as an application running in the Internet environment. Web-platform was chosen to provide access to information about how the service works from anywhere in the world and be independent of the operating system installed on the operator's computer.

Deletion monitoring must meet the following requirements:

- high availability of the service;
- actuality of the information;
- intuitive user interface;
- summary graphical reports;
- detail of each event;
- minimum possible load on the database.

Several DDM DQ2 web-applications use Django framework [7], which, being written on Python, provides full compatibility with services system code. Therefore, Django is a natural choice of the platform for the deletion monitoring.

Deletion monitoring generates reports using jQuery [8] AJAX (Asynchronous JavaScript and XML) calls. BBQ plug-in [9] is used to maintain browser history, direct links and bookmarks.

Every report provides JSON (JavaScript Object Notation) export which can be retrieved by any third-party application.

The application is dealing with the heavily loaded database. To avoid extra load on the database, database connection pooling and Memcached [10] are used. Once being called, plot data saves in Memcached for next 10 minutes, 30 minutes, 1 hour or 4 hours depending on report period.

The application provides graphical and numeric reports representing deletion process at ATLAS sites. Screen shot of the application is shown on figure 4.

Summary information is presented at the cloud, site, and endpoint levels. Different report periods could be selected: last hour, last 4 hours, last 24 hours.

To provide detailed information about any event in the deletion process, system is instrumented by the dataset and error browsers with search interface. File reports reach highest detail level all the way to single operations as deletion from LFC and local storage system. This allows tracing of deletion events for every file in DDM.

4. Conclusion

The presented Deletion Service is used by ATLAS Distributed Computing since autumn of 2010. In 2011 some works for optimization of algorithms were done, that led a significant performance boost.

Deletion Service serves more than 120 sites with more than 700 endpoints. In usual operation it deletes 2-2,5M of files per day, which correspond to 250 - 300 TB per day. During the deletion campaigns when deletion was carried out on most sites, deletion rate achieved is more than 6M of files per day, reaching up to 300k files per hour (fig. 5).

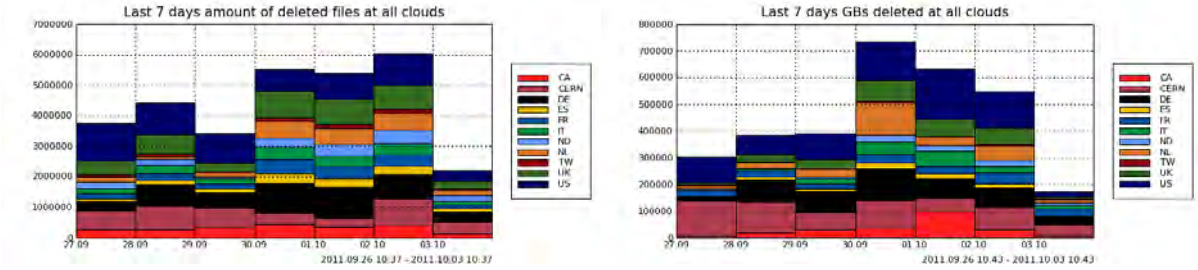


Figure 5: Deletion service performance during deletion campaign

Deletion monitoring is actively used by the ATLAS Distributed Computing operation team and site administrators.

Current implementation of the Deletion Service has a flexible configuration and can be easy tuned for better performance, but main limitation presently are performance and stability of storage systems and LFC Servers.

References

[1] The ATLAS Collaboration 2008 Journal of Instrumentation 3 S08003 URL
[2] http://stacks.iop.org/1748-0221/3/i=08/a=S08003
[3] Branco M, Cameron D, Gaidioz B, Garonne V, Koblitz B, Lassnig M, Rocha R, Salgado P and Wenaus T 2008 Journal of Physics: Conference Series 119 062017 URL
[4] http://stacks.iop.org/1742-6596/119/i=6/a=062017
[5] Apache http server project website, http://projects.apache.org/projects/http_server.html
[6] Mod_python Apache extension website, http://www.modpython.org/
[7] DB Connection pool component website, http://commons.apache.org/dbcp/
[8] Deletion monitoring page, http://bourricot.cern.ch/dq2/deletion/
[9] Django project web site, http://www.djangoproject.com
[10] Memcached project web site, http://memcached.org/
[11] jQuery project web site, http://jquery.com
[12] jQuery BBQ plug-in web site, http://benalman.com/projects/jquery-bbq-plugin/

ATLAS OFF-GRID SITES (TIER-3) MONITORING

A. Petrosyan[1], D. Oleynik[1], S. Belov[1], J. Andreeva[2], I. Kadochnikov[1]
on behalf of the ATLAS Collaboration
[1]*Joint Institute for Nuclear Research, Laboratory of Information Technologies, Dubna*
[2]*CERN IT/ES, CH-1211 Geneva 23, Switzerland*

ATLAS is a particle physics experiment on Large Hadron Collider at CERN. The experiment produces petabytes of data every year. The ATLAS Computing model embraces the Grid paradigm and originally included three levels of computing centers to be able to operate such large volume of data. With the formation of small computing centers, usually based at universities, the model was expanded to include them as Tier-3 sites. The experiment supplies all necessary software to operate typical Grid-site, but Tier-3 sites do not support Grid services of the experiment or support them partially. Tier-3 centers comprise a range of architectures and many do not possess Grid middleware, thus, monitoring of storage and analysis software used on Tier-2 sites becomes unavailable for Tier-3 site system administrator, therefore Tier-3 sites activity becomes unavailable for virtual organization of the experiment. In this paper ATLAS off-Grid sites monitoring software suite is presented. The software suite enables monitoring on sites, not covered by ATLAS Distributed Computing software.

1. Introduction

The ATLAS Distributed Computing activities concentrated so far in the "central" part of the computing system of the experiment, namely the first 3 tiers (CERN Tier-0, the 10 Tier-1s centers and about 50 Tier-2s). This is a coherent system to perform data processing and management on a global scale and host (re)processing, simulation activities down to group and user analysis.

Many ATLAS Institutes and National Communities built (or have plans to build) Tier-3 facilities. The definition of Tier-3 concept has been outlined. Tier-3 centers consist of non-pledged resources mostly dedicated for the data analysis by the geographically close or local scientific groups.

Tier-3 sites comprise a range of architectures and many do not possess Grid middleware, which would render application of Tier-2 monitoring systems useless [1].

2. Tier-3 task force

In March 2011 a proposal was approved, which describes a strategy of development monitoring software for non-pledged resources: Tier-3 Monitoring Software Suite (T3MON) [2]. T3MON software package should meet the requirements of the ATLAS collaboration for global monitoring of ATLAS activities at Tier-3 sites, and the needs of Tier-3 site administrators. The solutions implemented in frames of this project are expected to be generic, so other Virtual Organizations (VO), within or outside of LHC experiments, can use them.

Software suite should:

- allow a site administrator to monitor local Tier-3 fabric(s);
- provide a global monitoring view to the services provided by the Tier-3 center, namely:
  - data transfers to the site and between sites;
  - data processing and analysis.

Main components of the system are:

- a software suite for local site monitoring – "T3MON-SITE";
- information system which should aggregate and visualize data from distributed Tier-3 sites at a global VO level – "T3MON-GLOBAL".

The main requirements for "T3MON-SITE" package are simple installation and support, intuitive user interface. The package should provide a low level monitoring of all site resources, status and performance of the hardware components, activities of the VO at the site. The toolkit should also include monitoring of data files located at the site. The toolkit should foresee a possibility for

propagation of the aggregated monitoring metrics of the VO activities at the site to the VO central Tier-3 monitoring system ("T3MON-GLOBAL").

Central Tier-3 monitoring should be based on data collected from the local monitoring systems at Tier-3 sites. These data contain aggregated monitoring metrics of VO job processing and data transfer at a given Tier-3 site. The service must be scalable and has a minimal impact on the local resources. The set of the monitoring metrics as well as its granularity have to be defined by ATLAS.

3. Implementation of "T3MON-SITE"

In the light of the results of Tier-3 survey [3] and in accordance with the requirements, we developed a package based on the Ganglia [4] monitoring system. Ganglia is an open-source package used for real-time monitoring of large UNIX clusters. Each node in a Ganglia system runs a daemon that reports on the state of its host in the form of performance metrics including memory, CPU, load, disk and network statistics. Collectors gather data produced by the daemons and store it in round-robin database. The information is typically presented in the form of plots via a web-server, but can be also obtained in XML format and consumed by various clients.

The main development effort was concentrated on enabling plug-ins for PROOF [5] and XRootD [6] monitoring through Ganglia. PROOF, The Parallel ROOT Facility, is an extension of ROOT (a framework for data processing) intended to parallelize certain class of tasks and could be considered as an alternative to batch systems for physics analysis purposes. XRootD is a highly scalable architecture and services for data access; it is widely used for distributed data handling and federation.

The PROOF plug-in contains a job accounting database, which is used to provide Ganglia with status information and send hourly messages containing data on file and job statistics, to the Dashboard [7], the system for monitoring of distributed computing systems of the LHC virtual organizations. In this case ActiveMQ [8] acting as message broker.

The XRootD monitoring makes use of both the summary and detailed XRootD monitoring streams produced by the XRootD servers. The monitoring daemons receive monitoring data as UDP packets, and after processing the information is displayed in Ganglia and sent through ActiveMQ. Figure 1 shows T3MON-SITE dataflow.

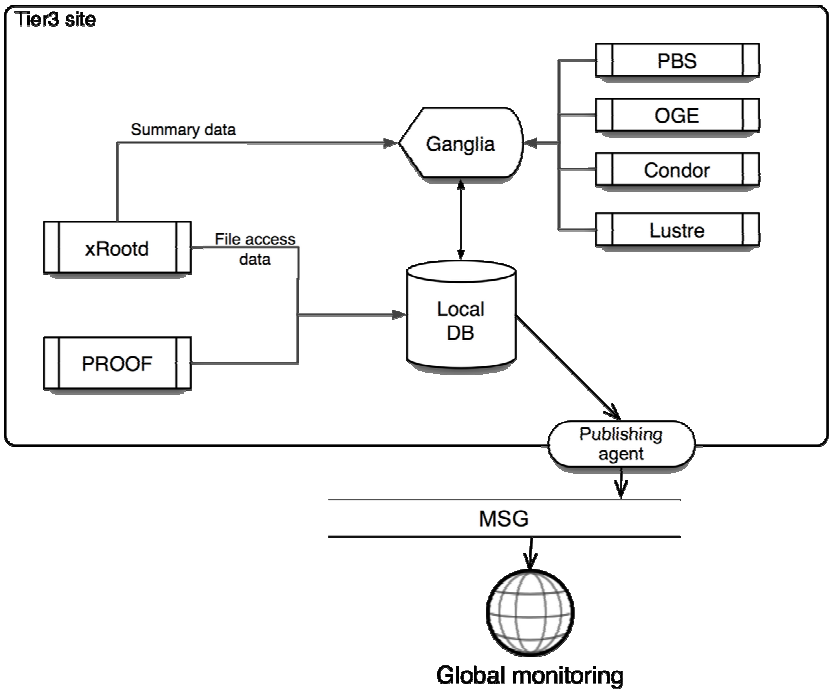


Figure 1: T3MON-SITE dataflow

The methods and framework developed for implementing the PROOF and XRootD plug-ins facilitated further development of plug-ins for monitoring of other software on Tier-3 sites, namely workload management solutions Condor [9], Torque [10], OGE [11] and distributed file system Lustre [12].

Condor plug-in utilizes the database of the Quill, the operational data logging system for Condor, to provide monitoring data for Ganglia and Dashboard. Quill is a natural part of Condor and gives enough information to monitor Condor daemons and queues as well as job statistics.

PBS/Torque queue and server status is obtained using the PBSQuery Python library, while job information is parsed directly from PBS accounting logs.

Lustre monitoring plug-in is based on reading the information provided in the virtual filesystem */proc/fs/lustre* and reporting it to Ganglia.

Job information from Condor and PBS/Torque is sent upward to Dashboard through ActiveMQ, while both status and job information is displayed in Ganglia.

4. Implementation of "T3MON-GLOBAL"

The central Tier-3 monitoring system is based on monitoring data published by Tier-3 sites and should provide a global picture of how ATLAS uses Tier-3 resources. The necessary condition for the development of the central Tier-3 monitoring system is consistent registration of the Tier-3 sites in the ATLAS Grid Information System (AGIS) [13]. Another important factor is encouraging the Tier-3 user community to use data transfer and job submission systems which are instrumented for reporting the monitoring data, for example Ganga, Athena, DDM DQ2.

The system consists of several components (see Figure 2):

- Publishing agents, which run at Tier-3 sites, interact with the local monitoring systems, aggregate and publish monitoring metrics to the message bus. As a transport layer, we use the Apache ActiveMQ messaging system. Apache ActiveMQ is an open source messaging system which was recently evaluated as a standard messaging solution for the WLCG infrastructure.
- Data collector receives information through ActiveMQ message broker. Collected data is being recorded in the central data repository (based on HBase, the Hadoop database [14]).
- Data presentation layer includes an interactive user interface and an API for data export. Tier-3 views are enabled in the existing Dashboard ATLAS monitoring systems, namely ATLAS DDM Dashboard, ATLAS Global job monitoring and Dashboard Transfer monitoring.

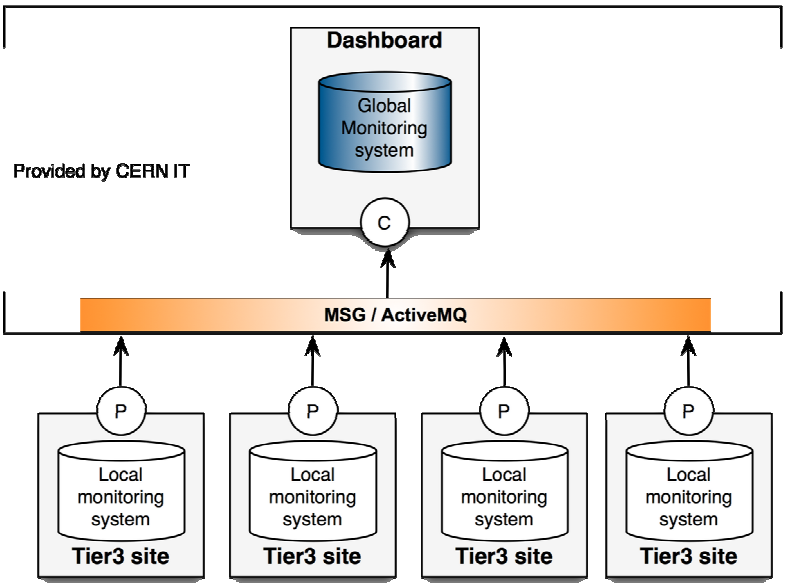


Figure 2: T3MON-GLOBAL logical scheme

5. Project infrastructure

There are two main goals for the infrastructure of T3MON project. The first one is to deliver monitoring packages' distribution to site administrators in the most clear and convenient way. The second one is to give to the developers means to manage code base, distributions, documentation, and product testing procedures and have a feedback from end users.

T3MON monitoring tools are distributed as RPM [15] packages via special YUM [16] repository. These packages are built and tested for Scientific Linux 5 and Python 2.6 as a main target software platform. To install required packages, site system administrator has just to set up few YUM repositories configuration files (for T3MON stable and externals repositories, standard EPEL and DAG) and install desired tools from T3MON. Completely the same way is used by developers on the testbed; packages to be tested are taken also from testing and nightly repositories. All the project repositories are rebuilt once a day to provide fresh packages versions for the corresponding SVN packages releases.

Packages build and code release and versioning system is based on the tools developed within Dashboard project [7]. Originally based on standard tools as Python's *distutils* package and SVN versioning and revision control system, Dashboard's code management and build system significantly extended their functionality and flexibility towards better version management, packages build and deployment. In T3MON project this system was slightly improved (support for custom Python version, remotely located external packages, extension of build scripts flexibility, and some other technical points).

6. JINR Tier-3 testbed

The development of a software suite for local site monitoring assumes the following activities:

- validation of existing monitoring tools for each of the used component;
- development and debugging of new monitoring tools.

The activities listed above imply the following:

- deployment of separate testbed for each of the LRMS and MSS reported as being used at ATLAS Tier-3 sites;
- Ganglia servers deployment;
- Ganglia agents installation and configuration for a specific testbed;
- installation and validation of the additional ganglia plug-ins for monitoring metrics collection as well as non-related to ganglia monitoring tools.

For estimating this aims, a testbed based on VM technology was deployed in JINR. PBS/Torque, Lustre, Condor, XRootD and PROOF clusters were installed. Testbed is based on virtualization technology, at the moment there are 18 virtual machines, all run on one physical server (AMD Athlon 64x2 Dual Core 3800+, 4Gb RAM, 320Gb HDD). Test load suite provides:

- job events;
- random submissions with configurable frequency;
- adjustable memory usage;
- CPU load;
- file events;
- uploading file to storage (random size, random time);
- remote file existence check;
- deletion of the file after configured period of time.

Ganglia packages were installed at each cluster; all clusters are being shown via one main web interface [17].

7. Conclusion

The monitoring tools developed within T3MON project allow having information on Tier-3 sites operation both on local and global levels. Most popular batch systems and mass storage systems used on real Tier-3 sites are supported.

On the level of site, there are several features quite useful for site's administration. There is a detailed monitoring of the local fabric (overall cluster or clusters monitoring, monitoring each individual node in the cluster, network utilization). Job processing is monitored based on information from batch system. For the mass storage system the created tools make possible to watch such significant parameters as total and available disk space, number of connections, I/O performance.

On the global level, T3MON provides file transfers information and jobs statistics to the Dashboard. It permits having a view of general operation of such heterogeneous non-pledged and even off-grid resources as Tier-3 sites are.

T3MON software was installed and tested on volunteer sites such as Tier-3 sites at JINR, BNL and on the test site of Kyiv Polytechnic Institute in Ukraine.

Installation and configuration instructions can be found on the project's home page [18].

References

[1] R. Brock et al., U.S. ATLAS Tier 3 Task Force, Preprint U.S. ATLAS, 2009.
[2] J. Andreeva et al., Tier-3 Monitoring Software Suite (T3MON) proposal, ATLAS note, 2011.
[3] D. Benjamin, Tier 3 Survey results, Report at ATLAS Software & Computing Workshop (29 November 2010 - 03 December 2010), CERN, Geneva.
[4] Ganglia monitoring system, http://ganglia.sourceforge.net/
[5] PROOF, Parallel ROOT Facility, http://root.cern.ch/drupal/content/proof
[6] XRootD, http://xrootd.org/
[7] J. Andreeva et al., Experiment Dashboard for Monitoring of the LHC Distributed Computing Systems, J. Phys.: Conf. Ser. 331 (2011) 072001.
[8] ActiveMQ, message broker, http://activemq.apache.org/
[9] D. Thain et al., Distributed Computing in Practice: The Condor Experience, Concurrency and Computation: Practice and Experience, Vol. 17, No. 2-4, pp. 323-356, Feb-Apr, 2005.
[10] Torque resource manager, http://www.adaptivecomputing.com/products/open-source/torque/
[11] Oracle Grid Engine, http://www.oracle.com/us/products/tools/oracle-grid-engine-075549.html
[12] Lustre distributed file system, http://lustre.org
[13] A. Anisenkov et al., ATLAS Grid Information System, to appear in Proceedings of CHEP2012 conference, New York, USA, May 21– 25, 2012.
[14] HBase, the Hadoop database, http://hbase.apache.org/
[15] RPM package manager, http://www.rpm.org/
[16] YUM software package manager, http://yum.baseurl.org/
[17] Monitoring web interface example on testbed, http://vm01.jinr.ru/ganglia/
[18] T3MON project home, https://svnweb.cern.ch/trac/t3mon/wiki/T3MONHome

META-MONITORING WITH THE HAPPYFACE PROJECT[1]

S. Röcker, A. Burgmeier, M. Heinrich, G. Quast, G. Vollmer, M. Zvada
Karlsruhe Institute of Technology,
Kaiserstraße 12, 76131 Karlsruhe, Germany
steffen.roecker@kit.edu

The efficient operation of large computing centers requires sophisticated tools for the monitoring of local computing infrastructure. The flood of information stemming from different monitoring sources unnecessarily delays the identification of problems and complicates administration.

The meta-monitoring tool HappyFace [1] offers a solution by aggregating information from different sources and providing a user-friendly overview of all relevant information. The modular setup of the framework allows adaption to site specific needs and easy extension with custom modules. This article describes the HappyFace project and its usage and gives an overview about current development.

1. Introduction

The discovery of a new particle with a mass of about 125 GeV - a candidate for the long-sought Higgs boson - at the Large Hadron Collider (LHC) at CERN would not have been possible without the Worldwide LHC Computing Grid (WLCG). The WLCG is a global collaboration linking grid infrastructures and computer centers worldwide to enable thousands of scientists all over the world to store, analyze and distribute the approximately 25 petabyte of data generated each year by the LHC experiments. Centrally collected data at CERN is distributed to 11 Tier-1 centers around the world for further storage and processing. Each of these computing centers provides data storage and processing power through standardized interfaces which enables grid software to distribute the workload equally on all available sites. High availability for each site is therefore crucial and requires extensive monitoring. However, monitoring such a site, consisting of a multitude of different systems, is a complex task.

Grid sites provide their services through the interplay of heterogeneous software. Each service has to be monitored individually and often comes with its own monitoring solution providing unstructured data. Correlation between error sources becomes difficult as each monitoring solution provides different output and has a distinct graphical interface. To gather all relevant information one has to check multiple monitoring systems and change their specific visualization settings. Additionally, most available monitoring solutions are difficult to use for non-experts and often have high latency due to their database backends. All this unnecessarily increases the amount of time required for people on shift monitoring the site.

2. Meta-Monitoring

A solution to the problem of monitoring multiple sources is meta-monitoring. A meta-monitoring system aggregates information from different, already existing sources and provides a coherent overview. A meta-monitoring system should provide the following properties:

- **Aggregation** All relevant information should be collected and aggregated.
- **History** All previous collected information should be stored to allow the access of the system status at a specific time in history or over a specific time range.
- **Usability** All important information should be visible at a glance and the system should allow comfortable navigation to access detailed information.
- **Warning functionality** Warnings and errors should be highlighted separately.

[1]This work has been supported by the Helmholtz Alliance "Physics at the Terascale"

- **Customizability** Test modules should be easy to configure and customize and the system should be extensible by custom modules.

The user base of such a system primarily consists of site operators and administrators who want automated site checks and a quick overview of their system. Additionally, grid users can access a less detailed overview in order to recognize possible problems with a site in the case of grid jobs with errors.

3. Meta-Monitoring with HappyFace

The HappyFace project is a meta-monitoring software developed in order to monitor the GridKa Tier-1 center at the Karlsruhe Institute of Technology for the CMS Collaboration. For an overview of the GridKa Tier-1 see the article "Support for the CMS experiment at the TIER-1 Center in Germany" in these proceedings.

HappyFace provides a modular framework to gather information from different monitoring sources, process this information and provide an overview of all relevant information. It allows real-time site monitoring for shifters and experts. All relevant information from different sources is fetched in a defined time interval and processed. The processed information is displayed graphically with a powerful rating system. Automatic alarms can be set up to raise an alarm when a specific module emits a warning or changes its status.

A screenshot of the HappyFace framework used to monitor the GridKa Tier-1 center is shown in Figure 1.

Several other German WLCG sites (both ATLAS and CMS) use HappyFace to monitor their sites for users and site admins. The CMS collaboration is using HappyFace to centrally monitor the batch systems of all CMS Tier-1 and Tier-2 sites. HappyFace has been used for more than three years and is stable, reliable and well tested.

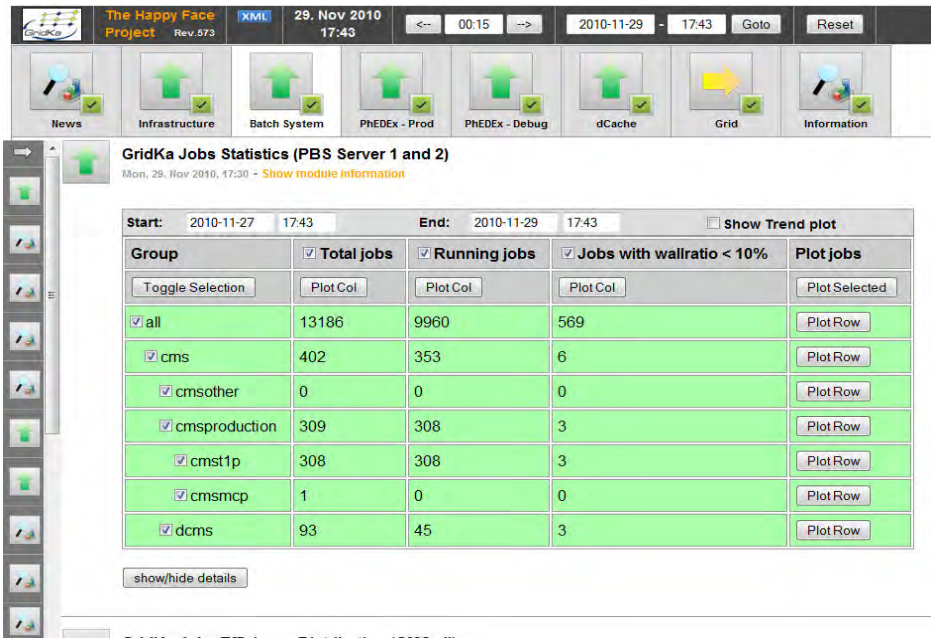


Figure 1: Screenshot of the web interface

4. The architecture of the HappyFace framework

The HappyFace framework version 2 consists of a core and several modules, written in Python. The workflow of the framework, shown in Figure 2, will be briefly described in the following.

The core provides all essential functionality such as a database backend and generating the web output. The core periodically executes all active modules, stores the retrieved files on disk, saves the module output in a database and initializes and renders the final web output. The web interface is written in PHP and JavaScript. The output of the modules is rendered via PHP code embedded in the Python code of the modules. This PHP code queries the database and generates the final HTML output. The output of the modules can be classified into different categories. Each category has its own status derived according to different selectable algorithms from the status of the individual modules belonging to this category. The status of the monitored site at an arbitrary past date, available in the database, can be retrieved by using the history navigation bar on top of the website.

Each module is written to test a specific service on a site. Many modules needed to monitor a typical WLCG site already exist and can be adapted easily by changing a configuration file. Custom modules can be written easily for site specific needs. More detailed and technical descriptions of the HappyFace core and the available modules can be found in Ref. [2] and [3].

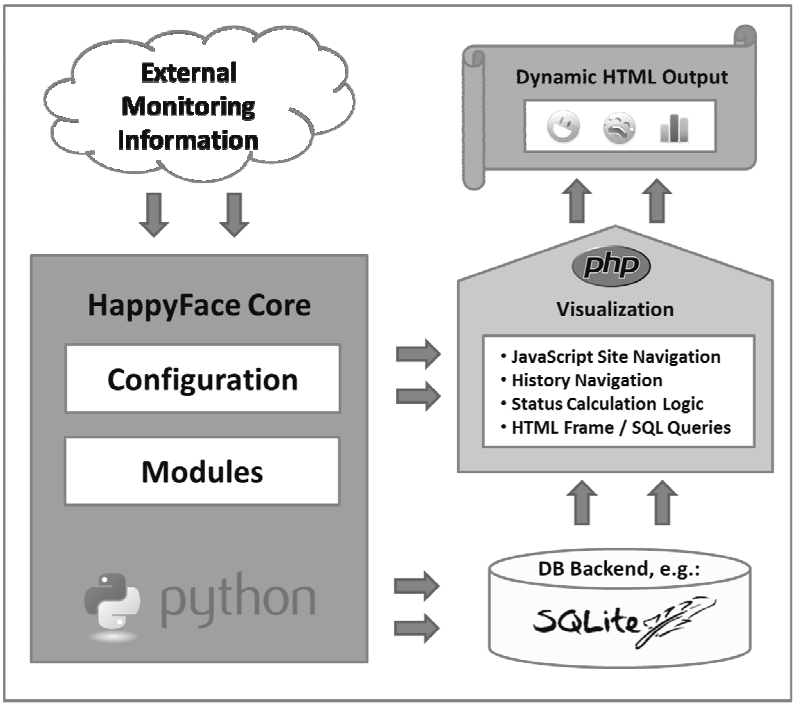


Figure 2: Schematic overview of the framework

5. Current development

The current version of HappyFace has been used for more than three years in production mode. It is stable, reliable and well tested. However, over the years some functionality has been identified which could be improved by taking a different approach. Software packaging and release management is difficult in the current version due to the mixing of core and module functionality. Some functionality has been added later in the development process and implementing specific features would require a large modification of the core system. Therefore, the best solution was to keep and support the stable version of HappyFace and start the development of a new version from scratch. The result is HappyFace version 3.

HappyFace 3 (HF3) is written in pure Python and no longer requires any PHP code for the web output. The web output is generated by a template engine (Mako) and either served directly by the web framework CherryPy or via an interface to Apache (or any other web server of choice supporting WSGI). This enables developers to focus on the code development while the output can be modified by changing a simple HTML template, thus reducing the code size of modules by up to 50%. A new database interface has been introduced which uses the abstraction scheme SQLalchemy and currently

provides backends for SQLite and PostgreSQL. The module code and configuration is strictly separated. The code is documented from the beginning of the development via in-line comments and the documentation is visualized via Sphinx, a documentation generator. The progress of the development is already very advanced. The core of HF3 is in good shape and most of the important modules have been ported and adapted. A first release of HF3 is expected later this year.

The ubiquity of mobile devices allows shifters to monitor a site at any moment in time no matter where in the world they currently reside. This allows instantaneous reaction which improves the readiness of a site. One drawback of the current HappyFace version 2 is the large webpage which gets rendered all at once and requires a high loading time for mobile devices. Therefore, the new version 3 renders each category separately and will be specifically tested with mobile devices. Another project considered for mobile devices is the development of a HappyFace monitoring program for Android mobile devices, also called app. This app will display the status of HappyFace, which is exported via XML, on a widget and alert the user via vibration or sound when a status changes to a critical level.

6. Conclusion

The HappyFace project provides a modular framework for meta-monitoring. HappyFace gathers information from different monitoring sources, processes this information and provides an overview of all relevant information. Today, several German WLCG sites (both ATLAS and CMS) use HappyFace to monitor their sites for users and site admins. Furthermore, a HappyFace instance is used by the CMS collaboration to centrally monitor the batch systems of all CMS Tier-1 and Tier-2 sites. The current version 2 of HappyFace has been used for more than three years and is stable, reliable and well tested. A new version 3 of HappyFace is currently written from scratch. The HappyFace core system is already implemented and the currently most used modules are already ported. A first release of HappyFace version 3 is expected later this year.

References

[1] HappyFace: https://ekptrac.physik.uni-karlsruhe.de/trac/HappyFace/

[2] V. Büge et al, “Site specific monitoring of multiple information systems – the HappyFace Project”, 2010 *J. Phys.: Conf. Ser.* **219** 062057

[3] V. Mauch et al, “The HappyFace Project”, 2011 *J. Phys.: Conf. Ser.* **331** 082011

BUILDING A HIGH PERFORMANCE MASS STORAGE SYSTEM FOR A WLCG TIER-1 SITE

V. Sapunenko[1,4], L. dell'Agnello[1], A. Cavalli[1], D. Gregori[1], A. Prosperini[1], P.P. Ricci[1], F. Noferini[2,3], E. Ronchieri[1], V.Vagnoni[3]

[1]*INFN-CNAF, Viale Berti Pichat 6/2, Bologna I-40127, Italy*

[2]*Centro E. Fermi,* Piazza del Viminale 1 – 00184, *Rome, Italy*

[3]*INFN Sezione di Bologna, via Irnerio, 46, I-40126 Bologna - Italy*

[4]*Corresponding author: vladimir.sapunenko@cnaf.infn.it*

The INFN Tier-1 at CNAF is one of the biggest European computing sites, operational since 2005, processing and providing storage resources for about 12% of all LHC data. Mass storage at the CNAF Tier-1 was initially provided by CASTOR (CERN Advanced Storage). Then we have moved to a more manageable and efficient solution, as a result of several years of case studies, software development and tests. This solution, called the Grid Enabled Mass Storage System (GEMSS), is based on a custom integration between a fast and reliable parallel file system (the IBM General Parallel File System, GPFS), with a complete integrated tape backend based on the Tivoli Storage Manager (TSM), which provides Hierarchical Storage Management (HSM) capabilities. Access to Grid users is provided by the Storage Resource Manager (StoRM), i.e. a standard SRM interface, and in case of the ALICE experiment by Xrootd. Since the start of the Large Hadron Collider (LHC) operation, all the LHC experiments have been using GEMSS at CNAF for both disk data access and long-term archival on tape media. Moreover, GEMSS has become the standard solution for all the other experiments hosted at CNAF, allowing the definitive consolidation of the data storage layer. Our choice has proven to be very successful during the last years of production, with continuous enhancements, accurate monitoring and effective customizations according to end-user requests. In this paper a description of GEMSS is reported, addressing manageability, administration and monitoring issues. We also discuss the solutions adopted in order to grant the maximum availability of the service. Finally, we summarize the main results obtained during the last years of activity, showing the reliability and the high performances that can be achieved using GEMSS.

1. Introduction

As the LHC experiments produce petabytes of data every year, they require not only to promptly archive acquired data, but also the ability to access those data at any time from any location. Data management becomes more and more resource demanding and still represents a challenge in High Energy Physics (HEP) computing.

The first issue is the amount of data to handle, amounting to several Petabytes of data (online and near-line) which need to be accessed at any time from thousands of concurrent processes. The second one is the required aggregated data throughput: on both Local Area Network (LAN) and Wide Area Network (WAN), it is on the order of several GB/s. The requirement of having all data archived on tape and the limited disk capacity leads to the necessity of an advanced Mass Storage System (MSS) capable of moving huge amount of data from disks to tapes and vice versa in very efficient manner.

Independent experiments (with independent production managers and end-users) concur for the usage of disk and tape resources. Chaotic access can lead to traffic jams, which must be taken into account as quasi-ordinary situations. The MSS needs to have the following features:

- Grid-enabled;
- characterized by high performance;
- modular design;
- stability and robustness;
- capability to manage several tens of PB of data;
- simplicity of installation and management;

• 24x7 operation with limited manpower;
• centralized administration.

The INFN-CNAF computing center hosts the Italian World-wide LHC Computing Grid (WLCG) Tier-1 site, the largest Italian computing facility employed in the LHC distributed computing infrastructure [1]. Since 2008 we have started our tests and the consequent production activities focused on the integration of the IBM General Parallel File System (GPFS) [2] disk storage infrastructure with the IBM Tivoli Storage Manager (TSM) [3], aiming at realizing a full Hierarchical Storage Management (HSM) system, that we named GEMSS [4]. GEMSS uses StoRM as Grid Storage Resource Manager (SRM) [5], an interface to the WLCG world distributed under GSL license with standard RPM packages. The GEMSS system is currently used as the Tier-1 storage solution for all the LHC experiments and other HEP experiments, like BaBar (SLAC) and CDF (Fermilab), the astro-particle physics experiments VIRGO and ARGO, the AMS, GLAST and PAMELA satellites, the MAGIC telescope and others.

In terms of resources INFN-CNAF is currently using a model where all our hardware storage systems (disk and tape drives) are accessed through Storage Area Network (SAN) [6] switches and Linux servers running Scientific Linux as operating system and equipped with redundant HBA (Fibre Channel Host Bus Adapter). This has demonstrated so far to be a robust, stable and very flexible approach.

At the moment a total of 8.4 PB net used disk space (available to the end user) is managed by GEMSS, and this number is going to increase to 11.2 PB by the end of 2012. The following storage hardware devices compose the whole SAN:

- 7 Data Direct Networks (DDN) S2A 9900 systems for a total of 7 PB (equipped with 2 TB SATA disks) served by about 40 disk servers with 10 Gb/s Ethernet connection to the LAN network;
- 7 EMC2 CX3-80 + 1 EMC2 CX4-960 for a total of 1.4 PB (1 TB SATA disks) served by about 90 disk servers with 1 Gb/s Ethernet connection to the LAN network.

An Oracle SUN SL8500 tape library with a total of uncompressed 14 PB tape space is also used in production with 20 T10KB drives from Oracle (100 MB/s of bandwidth and 9000 1TB tape cartridges) and 10 T10KC drives (Oracle, 200 MB/s of bandwidth and 1000 5TB tape cartridges). The connections between tape drives and servers are done via subset of the Fiber Channel SAN, which is referred to as Tape Area Network (TAN).

The storage resources are also accessible from the WAN via GridFTP servers under SRM control. In the LAN environment 13000 CPU cores take advantage of the POSIX compliant GPFS client, since they access the shared file system as it were local to the nodes.

2. **Storage group tasks**

The storage group is managing a number of tasks including (but not limited to): Disk storage administration (GPFS, GEMSS), Tape library administration (ACSLS, TSM), SAN maintenance and administration, installation and configuration of I/O servers, management of services (SRM, FTS, DB), monitoring of all hardware (HW) and software (SW) components, procurement, HW life circle management and basic HW support.

In order to be able to perform the above mentioned tasks we require:

- Fault tolerance and redundancy everywhere, but avoiding resource trashing;
- “Active-Active” configurations as much as possible, so that the load of failed elements is distributed over remaining ones (SAN, servers, controllers);
- Monitoring based on NAGIOS, which includes automated recovery procedures via the so called “event handlers”, permitting automated restart of failed services, isolation of failed components, e-mail notifications and so on.

With the aim of minimizing the management efforts, we are trying to use the lowest number of objects in our environment, using few but big storage systems (of the order of 500-800 TB per system) and I/O servers equipped with 10 Gb/s Ethernet network adapters.

In order to verify and validate any change in the hardware and software layers without

impacting the production, a small and dedicated cluster with all functionalities that are needed in the production environment has been setup as a testing facility (testbed) [7]. The testbed has proven to be very important to validate new versions of StoRM and GEMSS, ensuring the operation of new hardware and avoiding unexpected malfunctioning in the event of an upgrade of TSM and GPFS. It was also used to test procedures for moving data between different storage pools, avoiding unexpected behaviors of hardware or software in production. In the last use case, thanks to the testbed, we were able to move from the T10K-B to T10K-C drive technology, making the upgrade of the TSM server in production with few working hours. Besides the use of new drives, we also moved the data contained in the old media to new ones. The testbed has become an essential tool for the administration of production systems. It is used not only for development purposes, but also for testing all configuration changes, software components upgrades or deployment of patches.

3. The Software Components

CASTOR [8] was the "traditional" solution for Mass Storage at CNAF for all VO's since 2003. Large variety of issues, both at set-up/admin level and from the users' perspective (complexity, scalability, stability), was observed, with a large operational overhead. In 2006 we started to search for a potentially more scalable, performing and robust solution.

The most important milestones in the startup of this project have been:

- Q1 2007: after massive comparison tests [9], GPFS was chosen as the only solution for disk- based storage (it was already in use at CNAF for a long time before this test);
- Q2 2007: release of StoRM (developed at INFN), implementing SRM 2.2 specifications;
- Q3-Q4 2007: StoRM/GPFS in production for D1T0[1] storage class (SC) for LHCb and Atlas, with clear benefits for both experiments and a significant reduction of load on CASTOR;
- End 2007: a project started at CNAF to realize a complete Grid-enabled HSM solution based on StoRM/GPFS/TSM.
- In 2008 we realized the first prototype of D1T1[2] SC in production for LHCb experiment.
- At the end of 2009 the first D0T1[3] SC was in production for CMS.

The main software components which are composing our MSS are:

- GPFS as a Clustered Parallel File System;
- TSM as tape backend and HSM system;
- StoRM as SRM system;
- GEMSS as interface between StoRM, GPFS and TSM;
- NAGIOS [10] as alarm and event handling;
- QUATTOR [11] as system configuration manager;
- LEMON [12] as monitoring tool.

GPFS: General Parallel File System from IBM. It is Clustered (providing fault tolerance and redundancy) and Parallel file system with high scalability. Widely used in industry, very well documented and supported by user community and by IBM. Always providing maximum performance and there is no need to replicate data to increase availability. Running on AIX, Linux (RH, SL) and Windows. This software is not bound to IBM hardware.

TSM: Tivoli Storage Manager from IBM. It's very powerful, simple, database management (IBM's db2) is hidden to the administrator. It also includes built-in HSM functionality and transparent data movement from disk to tape and vice versa. Widely used in industry; Technical support either from IBM or from user community.

StoRM: Implementation of the SRM solution designed to leverage the advantages of cluster file systems (like GPFS) and standard POSIX file systems in a Grid environment. Developed at INFN-CNAF, StoRM provides data management capabilities to access and transfer data among

[1] D1T0 storage class means one copy on disk and no copy on tape
[2] D1T1 storage class means one copy on disk and one copy on tape
[3] D0T1 storage class means one copy on tape while the disk is only used as temporary buffer

heterogeneous and geographically distributed data centers, supporting direct access (native POSIX I/O calls) to shared files and directories, as well as other standard Grid access protocols. StoRM is adopted in the context of the WLCG framework.

GEMSS: Grid Enabled Mass Storage System. It is an integration of GPFS, TSM and StoRM. It combines GPFS and TSM specific features with StoRM to provide a transparent Grid-enabled HSM solution. An interface between GPFS and TSM has been implemented to perform an intelligent tape- ordered file recall to minimize mechanical operations in tape robotics (such as mount/unmount, search/rewind operations). In addition, StoRM has been extended to include the SRM methods required to manage the tapes. This approach permits to minimize management efforts and increase reliability. More details about GEMSS can be found in [13].

NAGIOS: Open Source software for monitoring purposes. It is a client/server system, which is capable of sending e-mails and SMS notifications in case of alarms and can also perform restart of daemons. We have been developed several custom controls and checks such as those for the ORACLE SL8500 library, for services like StoRM and GridFTP, and for GPFS [14].

QUATTOR: It is a system administrator tool which provides unattended (automated) installation, configuration and management of sets of servers, it also includes modules for specific software services configuration.

LEMON: It is a monitoring tool developed at CERN, available for Linux providing plots and historical information of different server quantity (e.g. CPU and network load, memory occupancy, etc).

GPFS is a clustered file system that provides a scalable POSIX file access. The GPFS clients (i.e. the farm Worker Nodes in our case) do not need a direct connection to the storage backend thought the SAN, but they access the data using the GPFS Network Shared Disk (NSD) mechanism [15]. The NSD disk servers have redundant Fiber Channel connections to the SAN and provide GPFS transparent access to the clients, as they are directly responsible for the I/O operation on the disks. In addition, GPFS works as a real cluster, so the NSD disk servers can provide the same level of service also in case of hardware/software failures of some components. At present, a number of roughly 130 disk servers is used in production for accessing the whole disk space area, subdivided into different GPFS clusters. A total of seven clusters is used, one cluster for each of LHC experiments, one dedicated to SuperB and BaBar and two shared between non-LHC users. Another GPFS cluster comprising only clients is used for the farm Worker Nodes. In general, for each experiment, a disk only file system and file system with HSM features are provided. The farm computing nodes and the User Interface nodes statically mount these file systems (which are POSIX compliant) and the access to the data is performed using the file protocol as they were local to the nodes. Therefore roughly 12000 CPU cores corresponding to a computing power of about 125 kHS-06[4] are currently directly accessing via POSIX (file) protocol the GEMSS-managed file systems.

The use of intensive parallel I/O over all the disk servers is useful for optimizing the performance and for a proper distribution of the load over the different hardware storage boxes.

The main elements of the TSM system are the master TSM server and several HSM nodes that are directly responsible for moving the data to and from the tape drives and run the TSM Storage Agents [16]. The TSM server is the core component, it relies on a database to store the metadata information and it also provides the space management services to the HSM nodes. The TSM server stores all the information on a dedicated shared disk volume, and a cold stand-by machine is ready for replacing the main server in case of hardware or software failure, in order to provide a fast recovery of the service. The TSM Storage Agents enable LAN-free data movements on the HSM nodes, using the dedicated TAN Fiber Channel connections to communicate with the drives, and this greatly improves the performances avoiding traffic congestion on the LAN when moving data between the disk and tape media. In our setup, 13 HSM nodes are enough for providing all the data movements with optimal

[4] The HS-06 or HEP-SPEC06 is the HEP-wide benchmark for measuring CPU performance that has been developed by the HEPiX Benchmarking Working Group. For details *http://w3.hepix.org/benchmarks/doku.php*

performance. In order to avoid contention and for redundancy purposes, at least two HSM nodes are dedicated for each GPFS cluster. Since the GPFS NSD disk servers are separated from the HSM nodes, it is possible to interrupt the tape access service for maintenance while keeping the disk service online: this is very useful in case of tape library failures or scheduled upgrades of the software.

4. GEMSS from the end-users perspective

The ATLAS experiment is one of the biggest users, with almost ¼ of all our resources for both storage space and CPU power.

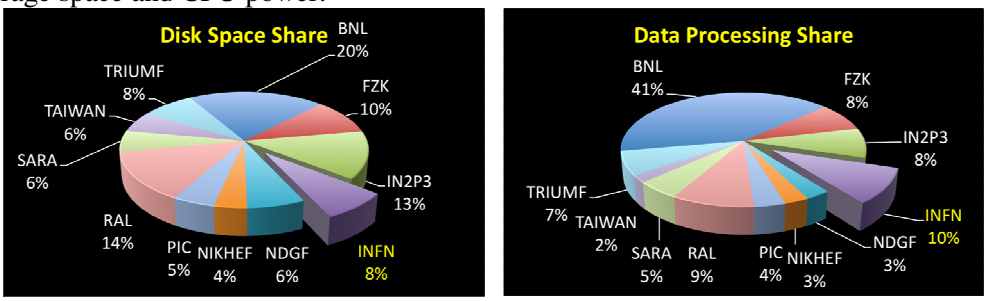


Fig. 1: Disk share and data processing share of ATLAS between Tier-1 sites (excluding CERN)

Its storage infrastructure is based on just a few hardware components, with 2.3 PB of disk space allocated for experiment (fig. 2): 3 DDN S2A9900 storage systems, 8 I/O servers, 2 metadata servers, 4 GridFTP servers, 5 StoRM servers and 2 HSM servers. All WAN data transfers to and from ATLAS GPFS file systems are performed by 4 GridFTP servers, which are configured as clients in GPFS cluster with direct connection (via Fibre Channel) to the storage system. 8 I/O servers are NSD servers, which are providing accesses for all GPFS clients like worker nodes and user interface nodes, which do not have direct connection to the storage system. The HSM nodes are configured in the same way as the GridFTP servers and are having direct access to the storage. In addition, they are connected to the tape drives via dedicated Host Based Adaptor (HBA) to the Tape Area Network (TAN).

In such a way we are separating three different data flows of different access patterns: sequential (WAN transfers and disk to tape transfers), and mostly random in case of access from the worker nodes and user interfaces.

The system, as shown in the figure 3, is routinely delivering up to 6 GB/s to clients on the LAN and up to 2 GB/s to clients on the WAN (which is actually limited by the 2x10 Gb/s WAN connection).

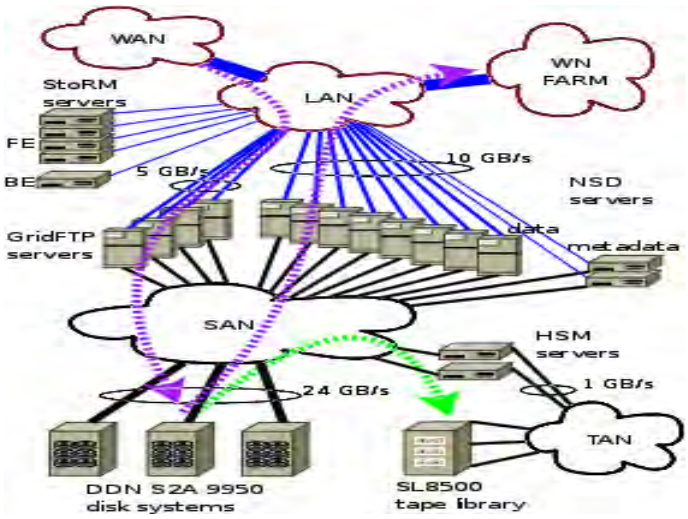


Fig. 2: Data flow and hardware components allocated for ATLAS

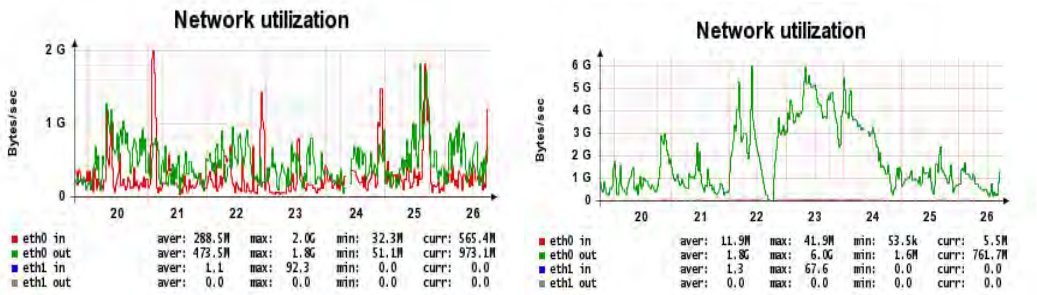


Fig. 3: Weekly statistics of network bandwidth used by ATLAS over WAN (left) and over LAN (right)

Another LHC experiment, but with quite different requirements, is ALICE. In contrast with other LHC experiments, ALICE is not using SRM for remote access or for data transfers. Everything is done using the Xrootd protocol [17] within the AliEN (ALIce ENvironment) [18] framework. Xrootd is designed to provide fault tolerant location and access to files distributed throughout cluster and WAN environment by employing peer-to-peer-like mechanisms. AliEn is a lightweight Open Source Grid Framework using the combination of a Web Service and Distributed Agent Model. It started within the ALICE Off-line Project at CERN for simulation, reconstruction, and analysis of physics data of the ALICE experiment.

From the hardware point of view, ALICE setup is quite similar to the ATLAS one: 1.3 PB of disk space subdivided into two file systems: one disk-only (960 TB) and another (385 TB) used as a buffer to tape. Both file systems are located on one DDN S2A9900 storage system, with 8 I/O (Xrootd) servers, 2 GPFS metadata servers and 2 HSM servers.

In contrast with ATLAS setup, ALICE does not use POSIX-like file access from computational nodes, so there is no need for powerful NSD servers. In this cluster we have left only two low-end NSD servers to support eventual remote access and cluster management tasks. All real I/O being performed on the Xrootd servers, which are mounting the same GPFS file systems and directly accessing the storage system via Fiber Channel connection. The HSM nodes are configured in the same way as in ATLAS case.

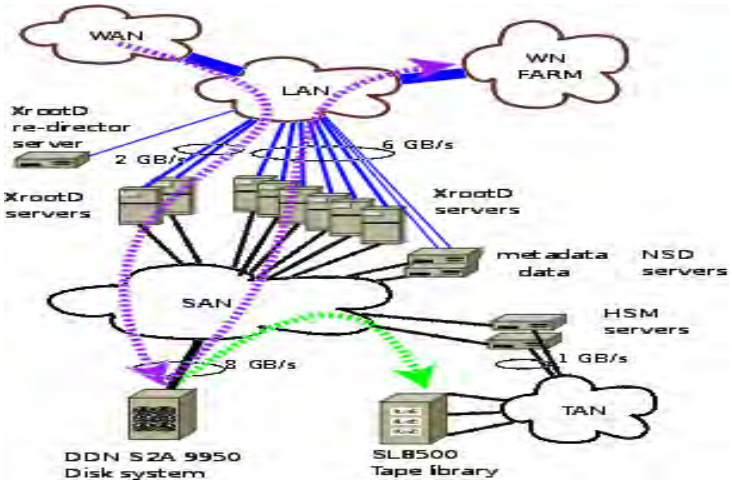


Fig. 4: Data flow and hardware components allocated for ALICE

The system, as shown in the figure below, is routinely delivering about 2 GB/s to clients on the LAN and up to 1 GB/s to clients on the WAN.

The implementation of the ALICE data access, based on Xrootd with GEMSS, is very simple. In fact the main requirement for Xrootd is the POSIX access to the file system, which is natively provided by GPFS. For the T1D0 instance of the ALICE storage the development of a Xrootd plugin

was needed, in order to manage recalls from tape, whereas for the T0D1 instance the system works without any further customization. The performance reached by the system is very satisfactory. For the T0D1 instance a throughput larger than 2 GB/s is provided to the Worker Nodes and the system guarantees a redundancy of all the files both at the Xrootd and at the GPFS levels.

In particular, the redundancy at the GPFS level allows to realize a perfect balance of the throughput over all the servers, independently of the kind of files requested. This is clearly seen from the AliEn monitoring tool (fig. 5 and 6). Plot on the top of fig. 5 represents incoming traffic (WAN data transfers), plot on the bottom showing outgoing (read) traffic produced by local and remote analysis jobs. Different colors represent traffic to or from different servers

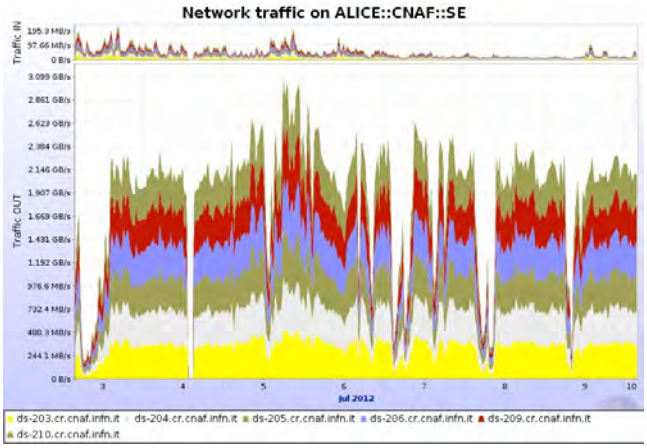


Fig. 5: Weekly statistics of network bandwidth used by ALICE at CNAF seen by MonALISA (ALICE monitoring tool) to and from disks-only storage system

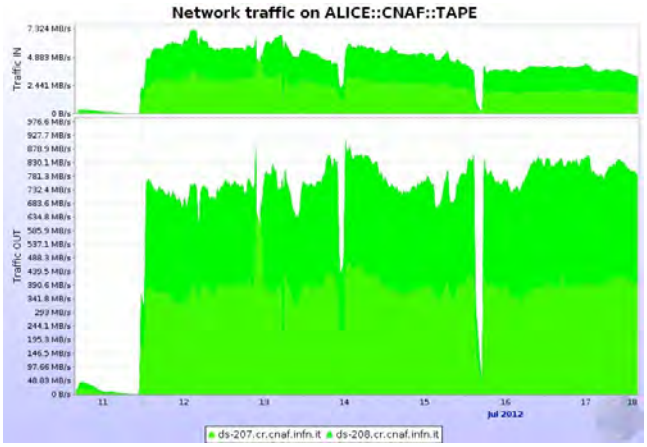


Fig. 6: Weekly statistics traffic to and from GPFS file system with tape backend

5. Conclusions

We implemented a full HSM system based on GPFS and TSM, able to satisfy the requirements of WLCG experiments operating the Large Hadron Collider. StoRM, the SRM service for GPFS, has been extended in order to manage tape support. An interface between GPFS and TSM was realized in order to perform tape recalls in an optimal order, so achieving great performances. A plugin for Xrootd allowed to interface Xrootd and GEMSS. GEMSS is the storage solution used in production in our Tier-1 as a single integrated system for all the LHC and non-LHC experiments. All the services are redundant both at hardware and software levels in order to guarantee a true 24x7 support.

During these last years of production, results from the point of view of experiments have shown the reliability of the system and its high performance with a moderate effort.

References

[1] Bird I et al. 2005 LHC computing Grid Technical design report (CERN LHCC-2005-024) also available online http://cdsweb.cern.ch/record/840543/files/lhcc-2005-024.pdf

[2] IBM General Parallel File System Administration and Programming Reference Version 3 Release 2 SA23-2221-01.

[3] IBM website references for Tivoli Storage Manager info and documentation https://www.ibm.com/developerworks/wikis/display/tivolidoccentral/Tivoli+Storage+Manager and http://www-01.ibm.com/software/tivoli/products/storage-mgr/

[4] Andreotti D et. al. INFN-CNAF Tier-1 Storage and Data Management Systems for the LHC Experiments, 2011 J. Phys.: Conf. Ser. 331 052005.

[5] Info about StoRM available online http://storm.forge.cnaf.infn.it/ Corso E et al. StoRM: A RM Solution on Disk Based Storage System Proceedings of the Cracow GridWorkshop 2006 (CGW2006), Cracow, Poland, October 15–18, 2006.

[6] Tate J., Lucchese F. and Moore R. Introduction to Storage Area Networks SG24-5470-03 IBM RedBook avaliable online https://www.redbooks.ibm.com/redbooks/pdfs/sg245470.pdf

[7] D. Gregori et al. "INFN Tier-1 Testbed Facility", CHEP2012.

[8] Info about Castor available online http://castor.web.cern.ch/

[9] Bencivenni M et al. A comparison of Data-Access Platforms for the Computing of Large Hadron Collider Experiments IEEE Transactions on Nuclear Science, Volume 55, Issue 3, Part 3, pp. 1621–1630 (June 2008) ISSN: 0018-9499.

[10] Info about NAGIOS available online http://www.nagios.org

[11] Info about QUATTOR available online http://quattor.sourceforge.net

[12] Info about LHC Era Monitoring available online http://lemon.web.cern.ch

[13] P.P. Ricci at al. “The Grid Enabled Mass Storage System (GEMSS): the Storage and Data management system used at the INFN Tier1 at CNAF”, CHEP 2012.

[14] D. Gregori et al. "INFN-CNAF Monitor and Control System", CHEP 2010.

[15] IBM General Parallel File System Advanced Administration Guide Version 3 Release 2 SC23-5182-01.

[16] IBM Tivoli Storage Manager for SAN for Linux Version 6.2: Storage Agent User’s Guide SC23-9799-01.

[17] Info about xrootd available online http://xrootd.slac.stanford.edu/ A. Dorigo, P. Elmer, F. Furano, and A. Hanushevsky, in WSEAS Trans. Comput., Apr. 2005.

[18] Info about AliEn available online: http://alien.cern.ch Saiz, P. and others, AliEn – ALICE environment on the GRID, Nucl. Instrum. Meth., A502 (2003) 437-440.

ATLAS DISTRIBUTED COMPUTING AUTOMATION

J. Schovancová[1,*], F. H. Barreiro Megino[2], C. Borrego[3], S. Campana[2], A. Di Girolamo[2], J. Elmsheuser[4], J. Hejbal[1,5], T. Kouba[1], F. Legger[4], E. Magradze[6], R. Medrano Llamas[2], G. Negri[2], L. Rinaldi[7], G. Sciacca[8], C. Serfon[4], D. C. Van Der Ster[2]
on behalf of the ATLAS Collaboration

[1] *Institute of Physics, Academy of Sciences of the Czech Republic, Na Slovance 2, CZ-18221 Prague 8, Czech Republic*
(*) Email: schovan@fzu.cz
[2] *CERN, CH - 1211 Geneva 23, Switzerland*
[3] *Universidad Autonoma de Madrid, Madrid*
[4] *Ludwig-Maximilians-Universitat Muenchen, Fakultaet fuer Physik, Am Coulombwall 1, DE - 85748, Garching, Germany*
[5] *Czech Technical University in Prague, Fac. of Nuclear Sciences and Physical Engineering, Brehova 7, CZ-11519 Prague 1, Czech Republic*
[6] *II. Physikalisches Institut, Georg-August Universitaet Goettingen, Friedrich-Hund-Platz 1, D-37077, Goettingen, Germany*
[7] *Instituto Nazionale Fisica Nucleare, Viale Berti Pichat 6/2, Bologna I-40127, Italy*
[8] *Albert Einstein Center for Fundamental Physics and Laboratory for High Energy Physics, University of Bern, Bern*

The ATLAS Experiment benefits from computing resources distributed worldwide at more than 100 WLCG sites. The ATLAS Grid sites provide over 100k CPU job slots, over 100 PB of storage space on disk or tape. Monitoring of status of such a complex infrastructure is essential. The ATLAS Grid infrastructure is monitored 24/7 by two teams of shifters distributed world-wide, by the ATLAS Distributed Computing experts, and by site administrators. In this paper we summarize automation efforts performed within the ATLAS Distributed Computing team in order to reduce manpower costs and improve the reliability of the system. Different aspects of the automation process are described: from the ATLAS Grid site topology provided by the ATLAS Grid Information System, via automatic site testing by the HammerCloud, to automatic exclusion from production or analysis activities.

1 Introduction

The Large Hadron Collider (LHC) at CERN has been delivering stable beams colliding at the centre-of-mass-energy 7TeV since March 2010 and at the center-of-mass-energy 8TeV since April 2012. ATLAS Experiment [1], one of the general purpose detectors of the LHC, has accumulated over 4 PB of RAW data over past 2 years. ATLAS benefits of the World-wide LHC Computing Grid (WLCG Grid) to process data and simulations.

The ATLAS Distributed Computing [2] (ADC) infrastructure is a complex and heterogeneous system: The ATLAS grid resources (CPU resources, storage systems, network links) are spread over more than 120 computing centres distributed worldwide. ATLAS grid computing centres host their storage either on disk or tape systems, with different flavours of storage systems, and heterogeneous CPU resources available to accommodate over 100k job slots. ATLAS grid sites are organized within three different flavours of grid: EGI, OSG, and NorduGrid. To provide a good quality of service to the ATLAS Collaboration, the operations team of the ATLAS computing resources has to be able to easily

identify issues with the infrastructure, and to address these issues. Challenging task to address requests for the monitoring of the ADC infrastructure is addressed by the ADC Monitoring team [3]. Even more challenging task to monitor the ADC infrastructure is covered by the ADC Operations teams: various ADC Shift teams [4], ADC Experts, and site administrators.

ATLAS sites may or may not be part of 3 ATLAS Activities: Data transfers, Data processing, Distributed analysis.

In Section 2 we describe motivation for automation efforts within ADC team. In Section 3 we briefly describe benefits of the ATLAS Grid Information System. In Section 4 we detail on what functional tests are available for automation.

2 Repetitive tasks & need for automation

There are 3 Shift teams monitoring status of the ADC infrastructure, each team focusing on different aspects of ADC Activities. The Shift teams are backed up by 2 ADC Experts. Site issues are addressed by site administrators.

Whenever a shifter on duty identifies an issue with the ADC infrastructure, he/she creates a bug report to expert, or activity requester, or to the site. There were over 6700 GGUS tickets created to the ATLAS grid sites since 1st January 2010, which leads to an average rate of 7 tickets per day. This amount of bug reports represents huge manual effort carried out by the ADC Operations team, ranging from the issue investigation by the ADC Shifter, creation of the bug report, addressing the issue by the site administrator or activity requester, resulting in functional testing of the reported service, and putting the service back into production for ATLAS Activities. The amount of manual work is the main motivation for automation of well known issues.

3 ATLAS Grid Information System

The ATLAS Grid Information System [5] (AGIS) collects site information from the GOCDB [6] and the OIM [7], and exposes it in a way convenient to the experiment. The AGIS provides topology information about the ATLAS grid sites, about services at sites, about downtimes of those services.

This unique information collection available in AGIS enables the ATLAS experiment to map between physical resources (CEs, SEs, LFCs, etc.) and ATLAS activity endpoints (PanDA [8] queues workload management endpoint, DDM [9] spacetoken endpoints), and additional logical layer in AGIS provides availability information of an ATLAS Activity at a particular site based on availability of subsequent physical resources at that site.

Having written what useful set of information AGIS provides, ATLAS benefits from several collectors, which collect downtime information for ATLAS activity endpoints, and exclude those activity endpoints for downtime period from corresponding ATLAS Activities.

First example of such a collector is the DDM collector, which excludes DDM spacetoken(s) from Data transfer activity, with granularity of sub-activities such as write/read/deletion, when a downtime of underlying SE starts, and re-enables those DDM spacetokens for the sub-activities once the SE downtime is over.

Second example of a collector taking action when a service is on downtime, is the Switcher. The Switcher manipulates Panda queues when a downtime of a CE or a SE affects Data processing or Distributed analysis activity at an ATLAS grid site.

Both collectors, DDM collector and Switcher, take automatic action ca 30 times per week. The main benefit of both collectors is saving ADC Operations manpower when a site declares unscheduled downtime, secondary benefit is for scheduled downtime.

Third example of a collector is the DDM space collector, which based on DDM spacetoken occupancy excludes a DDM spacetoken for write when a very small fraction of its size (several TBs) is left. When a fraction of free space at that DDM spacetoken is cleaned, at least up to limit which enables uninterrupted ATLAS Activities at that site, DDM spacetoken is enabled for writing again.

4 Functional tests for Services and Activities

ATLAS experiment runs a continuous flow of functional tests at each site. The tests are marginal with respect to normal ATLAS Activity at a site. Fraction of functional tests with respect to the overall activity is of the order of percent.

ATLAS experiment uses the HammerCloud [10], [11] framework to test how a site performs in the Data processing and Distributed analysis Activities. The HammerCloud test jobs simulate behaviour of an usual ATLAS data processing or analysis job. The HammerCloud uses the same environment as usual ATLAS jobs, access input data and installed SW in the same way, and stages out the output data in the same way. The HammerCloud then provides a very useful probe in the site health for real Activities. When several HammerCloud tests fail, site is excluded from an Activity for period of time, and recovered for that Activity once a set of jobs in a row succeeds. The HammerCloud takes ca 240 exclusion/recovery actions per week. The HammerCloud framework is used as the recovery framework for the Switcher exclusions.

ATLAS experiment probes NxN endpoint-to-endpoint transfers functionality with the Sonar [12] test. Purpose of this testing is to find optimal path for the transfers. In the past, ATLAS used strictly hierarchical topology of DDM endpoints. ATLAS sites are grouped in 10 clouds, each cloud is a set of geographically-close grid sites. The most powerful site in each cloud is a Tier-1 site. There are usually several Tier-2 sites in each cloud. Cloud may host also Tier-3 grid sites (sites with no pledge to WLCG).

In the past transfer between 2 Tier-2 sites, which belong to different clouds, was possible only through 2 Tier-1 sites, transfer path then was T2(Cloud A) → Tier-1 (Cloud A) → Tier-1 (Cloud B) → Tier-2 (Cloud B). This transfer path may not be very optimal, due to 3 additional sites being filled with data on the way.

Currently, ATLAS relaxes a bit the strictly hierarchical tier mode, and direct transfers between Tier-2 sites from different clouds with a very good network connectivity are enabled.

About 20 ATLAS grid sites are taking part in the LHCONE [13] network project. Such sites are running perfSonar [14] tests. As of September 2012 there is no automatic action taken based on perfSonar test results.

ATLAS uses the WLCG SAM framework [15] to test resources [16] registered in GOCDB or OIM. Currently, the SAM test results are used as additional sanity check when manual recovery of a service is necessary.

Conclusion

The ATLAS experiment has been successfully collecting data for more than 2 years. ATLAS data is processed and analysed at more than 120 grid sites distributed worldwide, taking into account Tier-1s, Tier-2s, and Tier-3 sites. The ATLAS Distributed Computing successfully fulfils its mission to deliver data to the ATLAS physicists. Current monitoring tools enable the ADC Operations team address issues in a timely manner. Level of automation of the ADC Operations helps to save manpower, and to focus on more urgent issues first.

Acknowledgements

Jaroslava Schovancová gratefully appreciates support from the Academy of Sciences of the Czech Republic. Support from the grant LA08032 of the MEYS (MŠMT), Czech Republic, and the grant SVV-2012-265309 of the Charles University in Prague is greatly acknowledged. This work was partly supported by the JINR and FZÚ AS CR Common Project "The GRID infrastructure for the physics experiments".

References

[1] The ATLAS Collaboration, "*The ATLAS Experiment at the CERN Large Hadron Collider*," *JINST* **3** (2008) S08003.

[2] Jézéquel S. et al. for the ATLAS Collaboration, "*ATLAS Distributed Computing Operations: Experience and improvements after 2 full years of data-taking*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012

International Conference.
[3] Schovancová J. for the ATLAS Collaboration, "*ATLAS Distributed Computing Monitoring tools after full 2 years of LHC data taking*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[4] Schovancová J. et al. for the ATLAS Collaboration, "*ATLAS Distributed Computing Shift Operation in the first 2 full years of LHC data taking*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[5] Anisenkov A. et al. for the ATLAS Collaboration, "*AGIS: The ATLAS Grid Information System*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[6] J. Gordon *et al.*, "*GOCDB, A Topology Repository For A Worldwide Grid Infrastructure*", *J. Phys. Conf. Series* **219** (2010) 062021.
[7] R. Pordes *et al.*, "*New science on the Open Science Grid*", *J. Phys. Conf. Ser.* **125** (2008) 012070.
[8] Maeno T. for the ATLAS Collaboration, "*PanDA: Distributed production and distributed analysis system for ATLAS*", *J. Phys. Conf. Ser.* **119** (2008) 062036.
[9] Branco M. et al. for the ATLAS Collaboration, "*Managing ATLAS data on a petabyte-scale with DQ2*", *J. Phys. Conf. Ser.* **119** (2008) 062017.
[10] Van Der Ster D. C. et al. for the ATLAS Collaboration, "*Experience in Grid Site Testing for ATLAS, CMS and LHCb with HammerCloud*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[11] Legger F. for the ATLAS Collaboration, "*Improving ATLAS grid site reliability with functional tests using HammerCloud*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[12] Campana S. for the ATLAS Collaboration, "*Evolving ATLAS computing for today's networks*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[13] Fisk I, "*New computing models and LHCONE*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[14] Laurens P. et al., "*Monitoring the US ATLAS Network Infrastructure with perfSONAR-PS*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[15] Rodrigues De Sousa Andrade P. M. et al., "*Service Availability Monitoring framework based on commodity software*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.
[16] Di Girolamo A. et al., "*New solutions for large scale functional tests in the WLCG infrastructure with SAM/Nagios: the experiments experience*", to appear in Proceedings of the Computing in High Energy and Nuclear Physics 2012 International Conference.

COMPUTING FACILITIES FOR SMALL PHYSICS ANALYSIS GROUP: EXAMPLES AND CONSIDERATION

A.Y. Shevel
High Energy Physics Division
Petersburg Nuclear Physics Institute
Andrey.Shevel@pnpi.spb.ru

A small physics group (3-15 persons) might use a number of computing facilities for the analysis/simulation, developing/testing, teaching. The instances of the small clusters for **Nuclear Chemistry Group** at State University of New York campus Stony Brook (http://www.sunysb.edu) and for **High Energy Physics Division** at Petersburg Nuclear Physics Institute (http://hepd.pnpi.spb.ru/) are briefly described. It is discussed different types of computing facilities: collaboration computing facilities, group local computing cluster (including colocation), cloud computing. The author emphasize the growing variety of different computing options and growing role of the group owned computing cluster of micro size.

Introduction

Usually members of a physics group have computer accounts on large computing facilities which are supported by the physics collaborations. Such the facilities have certain rules: who can the access to the computing installation in which scale, and for which purpose. As the result the or registration procedure takes some time. On other hand short term students and/or visitors might need for computer account just temporarily. Finally physics group needs in addition to the collaboration computing infrastructure more agile and flexible computing infrastructure completely under group control for several purposes:

- to keep common group data (papers, drafts, programs, fraction of experimental data, etc);
- to test new/modified simulation or/and analysis software/algorithms;
- to give the account for short time visitors/students who needs to do something in analysis;
- any other possible requirements, in particular as good gateway for remote large computing cluster(s).

We have to take into account the grows of the CPU computing power each year. If we pay attention not only to CPU but to the whole computing cluster we can find for example such the estimation "... *DOE centers have historically delivered average improvements in computing capability of 40%-80% per year with relatively flat budget*" [14] page 125. All above means that dozen of modern computing nodes in 2012 is more powerful than hundred servers in 2002.

Obviously such a small computing installation is used to be the good complement to large computing facility.

The computing needs can be considered in various ways [from point of view of the small group]:

- to use big[1] centralized cluster (here we mean collaboration cluster);
- cloud computing;
- own group local cluster (might be in two instances)
  - co-location of your cluster hardware somewhere else [16].
  - location of the group computing cluster in group office space with all responsibilities for air conditioning, electrical power, hardware support, etc;

Many pros and contras for each above options were discussed earlier [1]. Here it is assumed

[1]The cluster sizes: big, large = more than 1000 machines; middle size = until 1000; small = until 100; micro = O(10).

that that physics group is using more than one cluster to get the computing task done. In a range of papers such group owned computing clusters are referenced as clusters Tier[2]-3 [15]. Further in this paper we will analyze the own local computing cluster and cloud computing facilities: now and in nearest years.

Usually small physics group has limited financial resources. This fact does impose many restrictions on the cluster architecture.

The cluster has to be:

- cheap (useful consideration on the true cluster ownership cost is in [9]);
- consisted of reliable hardware;
- not demanding intensive watching/maintenance.

Other requirements – the implication of the desire to decrease the maintenance efforts:

- compatibility (architecture and base **OS**) with collaboration cluster environment (as in **Atlas** or **CMS** and other **CERN** collaborations for example), in particular same set of application software as in the collaboration cluster.

From above we see that group owned computing cluster is not possible to be large or even mid-range, it is quite small = micro cluster. The good configuration of the group owned cluster might consist of 5-15 modern machines (multicore CPUs, 2-3 GB of main memory per core, 10-20 TB or more of disk space per machine) and fastest network interconnect. Such the group cluster can help to get more flexibility when using several remote computing facilities: collaboration cluster(s), public/private cloud computing, etc.

The situation in different physics groups might differ from each other. Here we shall discuss the concrete group cluster solutions for Nuclear Chemistry Group (**NCG**) at **SUNYSB/**Chemistry and for High Energy Physics Division (**HEPD**) at **PNPI**.

Local computing cluster at SUNYSB/Chemistry

The computing cluster in **NCG** is appeared in 2000 or bit earlier. At that time all the machines (30+) had 512MB of main memory and Dual 500 MHz CPUs. This cluster was used for program development, test analysis, student work, etc. More than 70 registered users and around 3-5 are quite active. More detailed information about the cluster is available from [1].

To reduce downtime for the cluster it is good to buy and install special equipment **KVM switch over IP** to do many control actions (switching **on** and **off** of any machine in the cluster, get access to the console of any machine, etc) remotely over Internet. In another words the group might use remote help from external experts. However in cases described here the idea is not implemented yet.

As the batch system we use pair of torque/maui from http://www.supercluster.org.

Due to security reasons (no regular maintenance for) the cluster is available from only specifically defined network domains.

Because the cluster is located in relatively large room with good ventilation there is no needs for air conditioner. After years of experience we found that the University electric power grid is quite stable.

The basic **OS** (Scientific Linux with same RPM set as on **RACF**) installation procedure and basic configuration are semiautomatic: there is a couple of scripts with use of kickstart as initial step and another step consisting of script for post kickstart configuration. No virtualization technique was used in the cluster.

In our circumstances the users mailing list does form kind of thinking engine for various methods how to use the cluster for concrete tasks. The mailing list is located in the Google.com (i.e. somewhere in ***cloud)***.

Local computing cluster at High Energy Physics Division (PNPI)

The computing cluster in High Energy Physics Division started from a very small cluster

[2]In grid like computing infrastructure around **LHC** it is defined several Tiers: Tier-1, Tier-2, Tier-3, and so on. The difference is mainly determined by expected functionality (ability to accept and maintain policy of Virtual Organization (VO), implement distinguished service for different VO, existing of backup facility, etc).

consisting of three servers in February 1998. Details of the initial implementation are available from [2]. The cluster passed through many upgrades in hardware and software though it remained quite small or micro size. Now the cluster consists of 5 hardware servers with 20 virtual machines (i.e. fully virtualized) and around 27 TB of disk space. The OS is Scientific Linux 5.7. There are about 150 registered users on the cluster; about 50 users logged many times per month and about 15 users do use cluster every day. Virtual tools permit to use specific configurations for specific user needs, e.g. it is possible to use CERNVM for a range of physics collaborations.

There is home made backup scheme for user home directories (not for the data). One experienced person spends part time to keep the cluster up. The cluster room is equipped with air conditioner, UPSs, UDP.

In two computing cluster examples for **HEP** (**SUNYSB** and **PNPI**) we might see main similar trend: the desire to reduce the cluster **T**otal **C**ost of **O**wnership (**TCO**). **TCO** includes everything: cost of hardware and deployment, electricity power, man power, software and hardware support, any operation cost, cost of upgrades, etc. In this context it is not bad to take a look at ***cloud computing***.

Cloud computing

The cloud computing is hot topic in **IT** around 5 years. Many successful experiments with clouds have been performed [3, 10, 14]. It is not quite trivial paradigm though which has a lot of instance types in government and private sectors. The quote below is part of cloud computing definition (most consistent) I copied from [8].

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.

That was just beginning of the definition[3] but it gives main idea.

Many tens of cloud services are available with little difference in character and style of service and policy to pay for the service, e.g. [4, 5].

Some physicists are afraid to use public cloud computing service because the public cloud is out of their control (for instance the service could be down forever due to business or/and political issues). That is true. On other hand we can consider the control capability as the reliability of the access to the cloud. Can we think that public cloud service is 100% reliable all the time? The true answer is no. Unfortunately we have to say the same about any other instance of computing service of any kind. At the same time the small groups do have often not so reliable local computing which depends on unstable enthusiast activity. In many cases for even middle term time frame (2-5 years) local computing service is most probably less reliable than public cloud computing service. If you are worrying for the reliability of you data being safe - the obvious conclusion is to use combination of all mentioned options.

Several successful testbeds with using the cloud computing for production simulation in **HEP** have been carried out, e.g. **ATLAS** [3] and **STAR** [10] (latter work has many deep and smart observations of the experience with computing grid and cloud computing architectures). The success does depend on a lot of details, in particular on the computing infrastructure components and theirs parameters which are “under hood” of computing cloud. In work [10] authors were urged to do additional conversions of **VM** images, may be due to the lack of the open standards in the field. In other cases [10, 12] authors did find that tested public cloud has not so good computer hardware parameters as they expected. Also it has to be taken into account computing cloud initiatives and plans in government [6].

Conclusion

The small computing/information installations are already on the way to use the clouds. The moving to the cloud does eliminate for small physics group cluster hardware maintenance task, but not application software and data structure maintenance. Also to achieve maximum effect of using the cloud you can not ignore good understanding of cloud hardware, architecture, OS details.

To compare cluster of micro size and large cluster with many hundreds and more of servers, someone might see a lot of similarities: everywhere you need security, proper OS and applications

[3]Whole definition is explained in two pages or so.

configurations, reliable hardware, etc. However there are difference in between clusters of different size. For example, in micro cluster the common strategy is to buy and deploy computing nodes or/and components with parameters which do fit your task for this cluster (not always cheapest products). At the same time in largelhuge clusters the usual strategy is to buy and use cheapest computing nodes and other components.

We are emphasizing specifically the clusters of micro size because if we take a look at a range of all size clusters we might see the more servers in the cluster the more spending and efforts to support it. With more powerful cluster you need additional stuff and additional activity to meet more complicated conditions including more strong regulations from the public authority: fire safety, information security, insurance, etc: all mentioned factors increase **TCO** significantly. There are many reasons that for midrange computing clusters the **TCO** will grow faster with number of hardware servers than number of servers. It leads to the idea, that two main types of the computing clusters would have long live: huge clusters with many thousands of servers, often referenced as data center (like plant) which has a lot of users (actually such the cluster is used as computing cloud) and micro clusters which can be deployed in almost any office and used by the small group of users.

In light of above experience the group owned cluster is used as important gateway to public or private (i.e. collaboration) cloud computing. The number of public and private cloud computing instances is growing significantly each year. That means the importance of suitable gateway to different clouds for small physics group is growing as well.

References

[1] Damian Reynolds, Andrey Y Shevel *PHENIX technical note tn-452.0* http://www.phenix.bnl.gov/phenix/WWW/publish/shevel/tech-reports/ClusterPaper-2011-04-11.pdf

[2] Batch Computing Facility based on PCs - http://hepd.pnpi.spb.ru/CSD/CSDPublications/proc_043.pdf

[3] Jan-Philip Gehrcke et al, ATALS@AWS, http://iopscience.iop.org/1742-6596/219/5/052020

[4] A Drive http://www.adrive.com/

[5] Amazon Elastic Compute Cloud (Amazon EC2) - http://aws.amazon.com/ec2/

[6] Vivek Kundra, U.S. Chief Information Officer, "Federal Cloud Computing Strategy", http://www.cio.gov/documents/Federal-Cloud-Computing-Strategy.pdf

[7] Cloud computing at TeraGrid - http://www.rcac.purdue.edu/teragrid/resources/

[8] NIST definition of Cloud Computing - http://www.nist.gov/itl/cloud/upload/cloud-def-v15.pdf

[9] The True Cost of HPC Cluster Ownership - http://www.clustermonkey.net//content/view/262/1/

[10] Jerome Lauret et al. *From grid to cloud: STAR experience* http://computing.ornl.gov/workshops/scidac2010/papers/data_j_lauret.pdf

[11] Jerome Lauret et al. *Contextualization in practice: the Clemson experience* http://pos.sissa.it/archive/conferences/093/027/ACAT2010_027.pdf

[12] Keith R. Jackson et al. *Performance analysis of high performance computing applications on the Amazon Web Services Cloud* http://www.lbl.gov/cs/CSnews/cloudcomBP.pdf

[13] Migrate your Twiki to Google Sites (using Google Sites API and Perl) http://blog.famzah.net/2010/05/30/migrate-your-twiki-to-google-sites-using-google-sites-api-and-perl/

[14] The Magellan Report on Cloud Computing for Science – U.S. Department of Energy, Office of Advanced Scientific Computing Research, December, 2011 http://www.science.energy.gov/~/media/ascr/pdf/program-documents/docs/Magellan_Final_Report.pdf

[15] OSG Tier3 Twiki https://twiki.grid.iu.edu/bin/view/Tier3/WebBook

[16] Colocation centre http://en.wikipedia.org/wiki/Colocation_centre

IMPLEMENTATION OF COMMON TECHNOLOGIES IN GRID MIDDLEWARES[1]

O. Smirnova[1], B. Kónya[1], C. Aiftimiei[2], M. Cecchi[3], L. Field[4], P. Fuhrmann[5], J. K. Nilsen[6], J. White[7]

[1] *Dept. of Physics, Lund University, Box 118, 22100 Lund, Sweden*
[2] *INFN, via Marzolo 8, I-35131 Padova, Italy*
[3] *INFN-CNAF, viale Berti Pichat 6/2, Bologna, Italy*
[4] *CERN, CH-1211 Genève 23, Switzerland*
[5] *DESY, Notkestrasse 85, D-22607 Hamburg, Germany*
[6] *Dept. of Physics, University of Oslo, P. b. 1048 Blindern, N-0316 Oslo, Norway*
[7] *Helsinki Institute of Physics, Helsinki, Finland*
oxana.smirnova@hep.lu.se

Large variety, and often incompatibility, of Grid middlewares stem from the fact that the original technology lacked open standards accepted by the community. In Europe alone, it resulted in three substantially different middleware stacks, ARC, gLite and UNICORE, as well as a dedicated storage solution dCache. Interoperability between them has been the goal of many efforts, including a number of working groups of the Open Grid Forum. As a result, several commonly accepted specifications were developed. The European Middleware Initiative (EMI) project became a framework in which these, and other common approaches, are implemented. This paper gives an overview of such common solutions in the areas of compute, security, infrastructure and data management, including both server and client tools. The compute area focusses in consolidation of standards and agreements through a unified interface for job submission and management, a common format for accounting, and the wide adoption of GLUE schema version 2.0. The security area is working towards a unified security model and lowering Grid entry barriers by accepting users' institutional credentials. One of the highlights of the infrastructure area is the consolidation of the information system services via the creation of a common information backbone. The data area is focusing on implementing standards to ensure interoperability with other grids and industry components and to reuse already existing clients in operating systems and open source distributions.

I. INTRODUCTION

The European Middleware Initiative (EMI) [1] is a close collaboration of four major middleware providers, ARC [2], gLite [3], dCache [4] and UNICORE [5]. It aims to deliver a consolidated set of middleware components for deployment in EGI [6] and other distributed research computing infrastructures, extend the interoperability and integration between grids and other similar systems, strengthen the reliability and manageability of the delivered services and establish a sustainable model to support, harmonize and evolve the middleware. The progress of EMI is driven by requirements of the scientific communities relying on its solutions. As the ultimate result of its software development activity, EMI will deliver a consolidated middleware distribution of modular inter-compatible components with unified interfaces offering advanced functionalities that can be swapped depending on what kind of feature set is needed. This unified software stack will consist of interoperable solutions for the core capabilities needed to operate and manage a distributed computing infrastructure.

The EMI development roadmap is divided into three phases (roughly one year each):

[1] This work was partially funded by the EMI project under European Commission Grant Agreement INFSO-RI-261611

1. Initial integration and agreements on important technical aspects, as well as new component design and early implementations and numerous improvements of existing ones.
2. Intensive development, completion of consolidation plans, delivery of several harmonized solutions based on the existing agreements.
3. Completion of all the open development, hardening of existing EMI features, improving non-functional aspects such as reliability, usability and interoperability, integration of common libraries and other new common products with the rest of the EMI software portfolio.

In following this roadmap, EMI is guided by the key principle of converging to common standards in all applicable areas. While pre-EMI middlewares often implemented proprietary solutions, from protocols to deployment schemes, EMI strives to achieve unification based on standard solutions used in software development elsewhere. When it comes to grid standards, OGF [7] is both the source of specifications, and the target for EMI contributions. Figure 1 shows common client-side solutions developed by EMI, and illustrates usage of standards and common agreements in server-side solutions in each technical area.

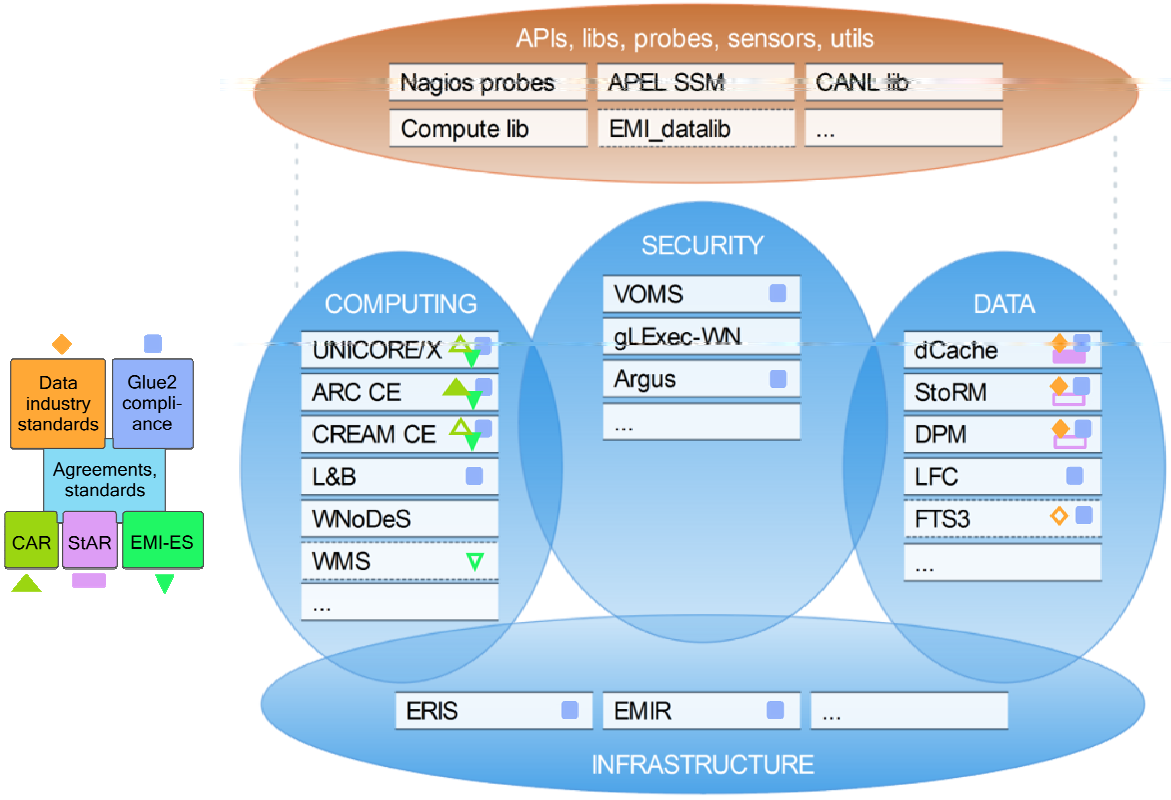


Figure 1: Overview of major EMI products and common technologies and standards involved in their implementation (indicated by different figures). Products are grouped as common client-side (top of the figure) and server-side (middle and bottom). Server-side products are grouped by technical areas.

In addition to using standard protocols and interfaces, EMI products also rely on standard build, testing and distribution tools. Most EMI products will be available via generic Linux distribution channels.

In what follows, we present several examples of successful definition and implementation of common standards and solutions, covering the four technical areas of EMI: *Compute*, *Data*, *Security* and *Infrastructure*.

II. COMPUTE AREA

EMI compute area includes various computing elements, workload manager and the corresponding client libraries and command line tools. As such, it is one of the most challenging areas with respect to integration and standardization.

A. GLUE2 support

GLUE2 is a proposed recommendation standard by OGF [8], which introduces information model for grid entities. Although the specification was released before EMI, and was endorsed by all EMI contributors, practical implementation of it was not complete. Grid services offered different incompatible information, which was one of the main obstacles to interoperability. Fully implementing GLUE2 became thus one of the key goals of EMI. The EMI Computing Elements (ARC CE, gLite CREAM and UNICORE/X) now fully support publication of local-level resource information expressed according to GLUE2. The remaining task before completing move to GLUE2 in compute area is to implement GLUE2 support in the match-making modules and client tools. Particularly, this concerns implementation of a new module in the WMS [9], responsible for querying over LDAP a GLUE2 enabled BDII [10] and fetching information into the WMS internal cache.

B. Common job submission and management method (EMI-ES)

Another problem with pre-EMI computing elements was incompatible interfaces for job submission and management. This is addressed by implementation of the EMI Execution Service (EMI-ES) agreement [11] in all the EMI computing services and their clients. EMI-ES offers a Web-service interface with integrated support for data staging, delegation capability, common state model, common job description, GLUE2-based service and activity description, and other advanced features. This common job management interface is one of the most distinguished developments of the project, and it will allow, also fostered by the migration to the common authentication library in the compute area components, seamless execution of complex workflows to HPC and HTP environments through a single entry point, for example, the gLite WMS.

Delegation and authorization aspects still need to be finalised, especially for what concerns interoperability with ARC/gLite and UNICORE. The final goal is to have each client of each of the three middleware solutions able to send and manage jobs to each different computing service and, conversely, to have all computing services able to accept jobs sent by each different client. Figure 2 illustrates how this opens a possibility to simplify development of a common client that can make use different grid infrastructures in a straightforward fashion.

C. Argus-based common authorization

As an important EMI agreement, Argus [12] was selected as the common authorization service for EMI components. All relevant EMI compute area components are now capable of interacting with Argus. This is accomplished either by using the available API, or by making use of the standard-conformant public interface offered by Argus.

III. SECURITY AREA

Grid security already before EMI was to large extent based on common standards, like X.509, albeit with few non-standard extensions and in places obsolete. ARC and gLite security models are fully compatible, while the UNICORE one is somewhat different, particularly with respect to delegation usage and details of Virtual Organisation-based authorisation. The challenges in the security area were thus to implement the latest relevant specifications and streamline already fairly common approaches.

A. Lowering the security credential handling barrier

A key security development in EMI is to make the security credential management more accessible to ordinary users. This is to be achieved by introducing simplified management of security credentials via reducing the complexity of handling certificates and integrating different security

mechanisms like Shibboleth [13] and Kerberos [14] across the EMI stack. This development will allow users to use their own authentication system to access a Grid. The goal of this activity is to lower the barrier of accessing distributed computing infrastructures using institutional or federated institutional authentication systems and to enable the usage of EMI components and services with other security infrastructures such as Kerberos or Shibboleth. In order to enable this access, a new security service, the Security Token Service (STS) is needed to translate these external credentials into the X.509 credentials needed by most Grid infrastructures.

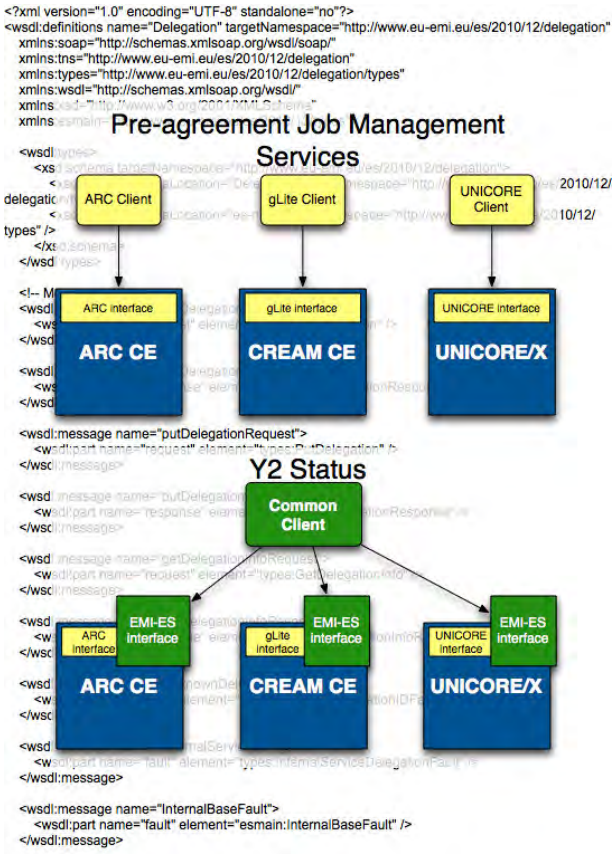


Figure 2: Job management interfaces offered and used by various EMI compute are products. Before the EMI-ES agreement, compute services and their clients were relying on middleware-specific communication channels (top scenario). With the implementation of EMI-ES in 2nd phase (Y2), the EMI release of the compute products comes with support for the common job management protocol in addition to the pre-EMI possibilities. The common EMI job management client utilizes this interface (bottom scenario).

The simplified management of credentials is to be fulfilled by the development and deployment of the EMI Security Token Service (STS). The STS implements the service defined by the standard WS-Trust specification. STS is a Web service that issues security tokens, a collection of claims, for the authenticated clients. As the clients can authenticate to the service using different security token formats, the service can be seen as converting a security token from one format into another. As such the STS is used to bridge different trust domains.

Current STS implementation is already capable of issuing the X.509 certificates, the first use-case consisting of an issuance of an X.509 certificate based on a security token from another security domain: username and password in this case.

B. Common authentication library (CANL)

The STS service, as well as other security area EMI products, will exploit the EMI Common Authentication Library (CANL), supporting X.509 and optionally, for the future, SAML. Such library has been defined, and API definition is available for Java, C and C++ [15]. Main features of CANL include: credentials handling, trust store handling, name constraints checking, standard CRL handling and OCSP [16] support (on-line revocation), SHA2 support, proxy operations such as verification and generation, proxy CSRs, proxy utilities, partially unified error codes and messages, and PKCS 11 support. The implementations of the libraries have almost completed and prototype versions of CANL

for all the three languages were included into the EMI Matterhorn release. In EMI phase 3, all major gLite, UNICORE, ARC and dCache products will adopt the common authentication library.

IV. INFRASTRUCTURE AREA

EMI infrastructure area covers information system components, as well as service monitoring solutions and accounting probes and publishers. As such, it is strongly linked to the compute area and in many aspects relies on implementation of the GLUE2 information model.

A. EMI Service Registry (EMIR)

Prior to the EMI project there was no common solution to discover ARC, gLite, UNICORE or dCache service instances from a common information source. The middleware stacks were confined into their own information systems. A common information system backbone, a service registry shared by the middleware stacks therefore was identified as the most important missing service, a critical missing component needed for the harmonization and convergence of the EMI products.

To address this issue, the EMI Registry (EMIR) was designed [17,18] and is being implemented. EMIR offers unified service discovery, independent of the service nature. It ensures high availability through quorum-based, replicated Global Service Registry (GSR) database which keeps track of available services. Service providers register to and push information to Domain Service Registries (DSR), which are in a hierarchical relation to GSR. In the 3rd phase of EMI, all EMI services will be provided with registration modules that take care of sending service information to the EMIR registry. In addition, modifications need to be implemented in information consumers, namely, service discovery clients.

B. Harmonized resource level information (ERIS)

Once a service is discovered, Grid tools should be able to obtain information about it in a standard manner. The EMI Resource Information Services (ERIS) is a common component for obtaining information directly from services. It effectively consolidates the existing resource-level products and ensures interoperability through the agreement on a common information model and interface. It was decided that ERIS should provide an LDAPv3 interface to GLUE2 information, minimising transition efforts, since gLite, ARC and dCache services are already capable publishing GLUE2 information via LDAPv3 interface. The missing part to be developed during the 3rd phase of EMI is a solution for UNICORE services that currently are not covered by ERIS.

C. Accounting consolidation

When it comes to accounting, EMI is concerned about producing and transporting accounting records to accounting servers such as APEL [19]. Prior to EMI, formats of Grid accounting records were in very basic state, with storage accounting record format not existing at all. EMI defined and submitted to OGF both the Compute Accounting Record (CAR) [20] and Storage Accounting Record (StAR) [21]. CAR is mainly a profile of the existing OGF UR specification, defined reflecting practical, financial and legal requirements of resource consumption, including CPU time, wall-clock time and memory usage, adding support for user groups or Virtual Organisations, and encompassing both single and aggregated records. StAR is a new specification, inspired by OGF UR but focussing on reporting used storage space (if no space change occurs, no record is sent). Like CAR, it supports Virtual Organisations.

CAR-enabled accounting service is already deployed by EGI, and EMI is working on updating its accounting clients to submit CAR-formatted records via APEL's Secure Stomp Messenger (SSM). Storage accounting record publisher producing StAR already exists for dCache, and will be developed for other storage elements (DPM and StoRM). These will also be able to submit records to APEL.

V. DATA AREA

EMI data area covers a range of products dealing with file storage and transfer, such as storage elements, data movement service, data catalogue and client-side data libraries and utilities. This area

benefits from using one of the oldest Grid standards, OGF SRM, already before EMI. Main challenges are in consolidation of the multitude of client tools and in moving from the rather obscure Grid File Transfer Protocol towards widely used and supported HTTPS and WebDAV.

A. EMI datalib: consolidation of the data access libraries

The ARC and gLite data access libraries have sufficient functionality in common to justify a merge of them. This includes, but is not limited, to storage control (SRM), storage access (e.g. gsiFTP) and information protocols. Common EMI data library, EMI_datalib, has been designed by re-using and streamlining existing components [22].

The proposed architecture will offer the POSIX-like interface as provided by gLite GFAL-2, and the higher-level file-based interface provided by ARC's libarcdata2 library. In addition, a transfer interface will be made available in GFAL-2 to handle initiation and monitoring of 3rd party transfers. In this architecture, libarcdata2 will use the POSIX-like interface of GFAL-2 through a GFAL-plug-in. Clients requiring POSIX-based byte-wise data access can use the GFAL-2 library directly, while file based clients for data transfer, such as the lcg_util, ARC CLI and ARC CE, will use the libarcdata2 file-based interface. Additionally, FTS3, parts of the lcg_utils and ARC CLI will use the 3rd party transfer library.

EMI_datalib is expected to be fully implemented and adopted by the EMI data area products during the 3rd phase of EMI. GFAL-2 and the ARC data library plug-ins for GFAL-2 are already implemented at a prototype level, and the GFAL-2 library is available for beta testing in the EMI Matterhorn release.

B. FTS3: next generation file transfer service

FTS3 is a replacement of the gLite File Transfer Service (FTS) [23], addressing a number of existing FTS shortcomings. FTS3 will rely on the common EMI_datalib, thus being able to support standard transfer protocols, such as e.g. HTTPS. In addition, it will be decoupled from the underlying database specifics, and will implement a range of new features. Full authorisation model of FTS3 is yet to be defined. Development of FTS3 is ongoing, and a working prototype is expected to become available during EMI phase 3.

C. Supporting WebDAV

Recently WebDAV [24] has become a more and more requested protocol. Therefore EMI decided to support this protocol in the EMI Storage Elements and the LHC File Catalog (LFC).

Traditionally, gsiFTP has been preferred as Grid file transfer protocol for a number of reasons: it natively supported multi-streams transfer, 3rd-party transfer, and credential delegation. Still, similar functionalities can be achieved with HTTP, by using Content-Range HTTP header, WebDAV COPY functionality, and Gridsite delegation service, respectively.

WebDAV protocol support is implemented for DPM and dCache. DPM implementation is based in two widely used Open Source tools: Apache2 and mod_dav. It is fully functional, though a number of improvements will still be made during the last phase of EMI. The dCache team decided to use the Milton libraries [25] to support the WebDAV protocol, while gPlazma2 offers X.509 certificate and user/password authentication.

Traditionally access to the LFC has been done using a custom protocol, preventing any standards based client from browsing the catalogue. As part of EMI work to move towards standard protocols, new component was added, exposing the name server metadata via WebDAV. Standard operations like browsing directory contents, renaming files or directories, removing files or any other expected file system operation are now available to LFC users using standard clients – browser or command line tools. In addition, LFC now also exposes the base HTTP standard, allowing GET redirections from the catalogue to different storage services with corresponding replicas registered in the catalogue. This offers the user a global access experience to all the data registered in a given LFC catalogue, using standard HTTP clients and without having to know any details regarding available file replicas.

VI. CONCLUSION AND OUTLOOK

EMI is heavily engaged in implementing community standards and common approaches in all its technical areas: compute, security, infrastructure and data. In cases when existing standards were implemented (such as, for example, GLUE2 or WebDAV), convergence is already largely achieved. In other cases EMI had to come with new specifications (EMI-ES, CAR, STAR etc), which are yet to be fully implemented.

In addition to the components described in detail in this article, several other convergence efforts and feasibility studies are ongoing in EMI. In compute area, one should mention compute client harmonisation and various aspects of scheduling: common parallel execution framework, common approaches to node-exclusive or multi-core scheduling, as well as common characterisation of different classes of jobs. In security area, development of encrypted storage layer for ordinary storage elements is ongoing. In infrastructure area, development of Nagios probes for service monitoring is well on track, covering most EMI services. In data area, promising work on global storage federation based on HTTP and WebDAV should be mentioned.

The last year of EMI is dedicated to bring to completion development and integration activities. With the final release EMI aims to achieve its technical objective which at a high level can be outlined as delivering a production-level consolidated middleware distribution, consisting of modular components with unified interfaces. The components will cover a wide range of functionalities, providing universal building blocks for infrastructures of various complexity and specialization.

References

[1] European Middleware Initiative Web site URL http://www.eu-emi.eu
[2] Ellert M *et al* 2007 Future *Gener. Comput. Syst.* **23** 219-240 ISSN 0167-739X.
[3] Laure E *et al* 2005 Middleware for the next generation Grid infrastructure *Proceedings of Computing in High Energy Physics and Nuclear Physics 2004* (publised version from CERN) pp.826.
[4] dCache Web site URL http://www.dacache.org
[5] Unicore Web site URL http://www.unicore.eu
[6] EGI Web site URL http://www.egi.eu
[7] Open Grid Forum Web site URL http://www.ogf.org
[8] GFD.147 GLUE 2.0 Specification Web site URL http://ogf.org/documents/GFD.147.pdf
[9] Cecchi M *et al* 2010 The gLite Workload Management System *J. Phys.: Conf. Ser.* **219** 062039.
[10] BDII, the Berkeley Database Information Index Web site URL https://twiki.cern.ch/twiki//bin/view/EGEE/BDII
[11] EMI Execution Service (EMI-ES) Specification, Web site URL, http://cdsweb.cern.ch/record/1452918
[12] Argus authorization system Web site URL https://twiki.cern.ch/twiki/bin/view/EGEE/AuthorizationFramework
[13] Shibboleth Web site URL http://shibboleth.internet2.edu/
[14] Kerberos, The Network Authentication Protocol.Web site URL http://web.mit.edu/kerberos/
[15] EMI Common Authentication Library (CANL) Web site URL, https://twiki.cern.ch/twiki/bin/view/EMI/EmiJra1T4SecurityCommonAuthNLib
[16] OCSP Online Certificate Status Protocol Web site URL http://tools.ietf.org/html/rfc2560
[17] EMIR design Web site URL http://cdsweb.cern.ch/record/1449141?ln=en
[18] Memon S *et al* 2012 EMIR: An EMI Service Registry for Federated Grid Infrastructures *Proceedings of 1st European Grid Infrastructure (EGI) Community Forum/ 2nd European Middleware Initiative (EMI) Technical Conference Muenich, Germany 2012.*
[19] Jiang, M *et al* 2011 An APEL Tool Based CPU Usage Accounting Infrastructure for Large

Scale Computing Grids *Data Driven e-Science* eds Lin, S and Yen, E Springer New York pp175-185 DOI 10.1007/978-1-4419-8014-4_14.
[20] Definition of the Compute Accounting Record (CAR), Web site URL http://cdsweb.cern.ch/record/1449764
[21] EMI StAR – Definition of a Storage Accounting Record Web site URL http://cdsweb.cern.ch/record/1452920
[22] Nilsen J K *et al* 2012 EMI datalib, EMI datalib - joining the best of ARC and gLite data libraries *submitted to Proceedings of CHEP 2012.*
[23] The gLite File Transfer Service Web site URL https://wiki.chipp.ch/twiki/pub/LCGTier2/FTSlinks/transfer.pdf
[24] WebDAV protocol Web site URL http://tools.ietf.org/html/rfc4918
[25] Milton libraries Web site URL http://http://milton.io

MATHCLOUD: FROM SOFTWARE TOOLKIT TO CLOUD PLATFORM FOR BUILDING COMPUTING SERVICES[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

MathCloud is an open research project led by CGTDC ISA RAS investigating the use of distributed service-oriented environments for scientific research. It uses high-level decomposition of computational problems and service-oriented architecture in order to support the problem solved by composition of distributed computing services. The paper presents a software platform being developed within MathCloud project, its current state and future directions.

1 Introduction

Modern scientific research is closely related to complex computations, analysis of large amounts of experimental data, use of unique equipment and collaboration within distributed research projects. The scientists are increasingly faced with the lack of resources required for running day-to-day research on their computers, whether information resources, computing power or applications. Often the necessary resources can be found on remote servers, computing facilities or colleagues' desktops, administratively and geographically distributed. Modern networks and distributed computing technologies enable wide-scale sharing and reuse of such resources among scientists thus increasing productivity of research. The most striking examples are World Wide Web and global grid infrastructures.

Web is the largest distributed system that provides access to a wealth of information resources. The success of the Web is due to a number of important features of its architecture. The client-server model provides inalienability of resources from their owners who can quickly update the content of resources and control access to them. At the same time, the use of unilateral hypertext links makes it easy to refer to web resources, without the need for the participation of their respective owners. Web is based on open standards, allowing anyone to create independent implementations of servers and clients.

Global grid infrastructures that emerged in 2000s are focused on the integration of high performance computing resources to support research projects. However, despite the impressive amount of aggregated resources, the range of grid users and applications is relatively narrow. This is due to inherent complexity of grid middleware, low-level interfaces, and lack of convenient services that enable the researcher to formulate the problem to solve via familiar interface, and taking on the responsibility for the launch and management of computations in grid.

The problem of the technological gap between the researcher and the computing infrastructure existed before. Suffice it to recall the classical supercomputers and shared computing facilities. Not every researcher wrote parallel programs - usually he used an existing computing package for the solution of his problem. In this case, in addition to struggle with the chosen package, the researcher had also to master the subtleties of working with the command line and the batch system of supercomputer. About 30 years ago it was taken for granted, but in the eyes of the modern user it looks the same as a text web browser - awkward and archaic. Without radically changing their interface, scientific computing infrastructures have grown and become more complex inside and harder to use.

The outlined problem is not a lack of computing infrastructures themselves, but rather indicates the need for higher-level systems, which operate on specific classes of problems and hide the complexity of an underlying infrastructure from researchers. Emerging grid portals and scientific

[1] The work is supported by RFBR (grants 11-07-00543-a, 11-07-12045-офи-м-2011, 10-07-00176-a) and RAS Presidium (Programme №14).

gateways are the first step in this direction. However, the transition to the new level is only possible by creating a holistic approach to the construction of such systems, based on a detailed analysis of the needs of researchers.

MathCloud [1] is an open research project led by CGTDC ISA RAS investigating the use of distributed service-oriented environments for scientific research. The project is based on the observation that the vast majority of computing problems faced by researchers can be reduced to a single or a composition of several well-known problem classes. Using a service-oriented approach one can build a distributed environment which provides researchers with access to services to solve these common classes of problems, as well as ready tools for service composition in order to solve complex problems.

The proposed approach is based on the concept of Service-Oriented Science [2] introduced by Ian Foster in 2005 to refer 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.

One of the main goals of MathCloud project is the development of a software platform for building service-oriented scientific environments, including tools for building, publication and composition of computing services. The key requirements for this platform are ease of use, following standards, and openness. MathCloud platform is based on generally accepted approaches, standards and technologies, such as the REST architectural style [3], HTTP protocol, JSON format and Java platform. The platform defines a unified interface (REST API) of computing service, allowing the creation of alternative implementations.

2 Computing Service as a Web Function

The core concept behind MathCloud is a service which represents a stateless, asynchronous "web function" with a set of input parameters passed to service by client and a set of output parameters returned back to client. This model is suitable for algorithmic resources and provides for scalability and fault-tolerance. Services are implemented as RESTful web services [4] with a unified API supporting service introspection, asynchronous request processing and passing data files by links.

In accordance with the principles of REST, the interface of service is formed by a set of resources identified by URIs and accessible via standard HTTP methods (Fig. 1).

	GET	POST	DELETE
SERVICE_URI	Get service description	Submit new request → Get JOB_URI	
JOB_URI	Get job status and results		Cancel job Delete job data
FILE_URI	Get file data		
SERVER_URI	Get list of services provided by server		

Fig. 1: REST API of MathCloud service

As a primary data representation format for the REST interface JSON has been chosen because of a more compact representation of data structures in comparison to XML and tight integration with

JavaScript language simplifying creation of Ajax based web interfaces. A known disadvantage of JSON is the lack of standard tools for description and validation of the JSON data structure, comparable to XML Schema. However, there is an active ongoing work on such a format called JSON Schema [5], which is used for describing service parameters.

3 MathCloud Platform

In its current state the MathCloud platform represents a software toolkit consisting of the following tools for building, deployment, discovery and integration of computing services using the described REST API.

Service Container

The availability of ready and easy to use tools for creating services is very important for expanding the range of potential service developers. To simplify the process of service development, a service container is implemented (Fig. 2). The container provides a hosting environment for services and implements the described REST API.

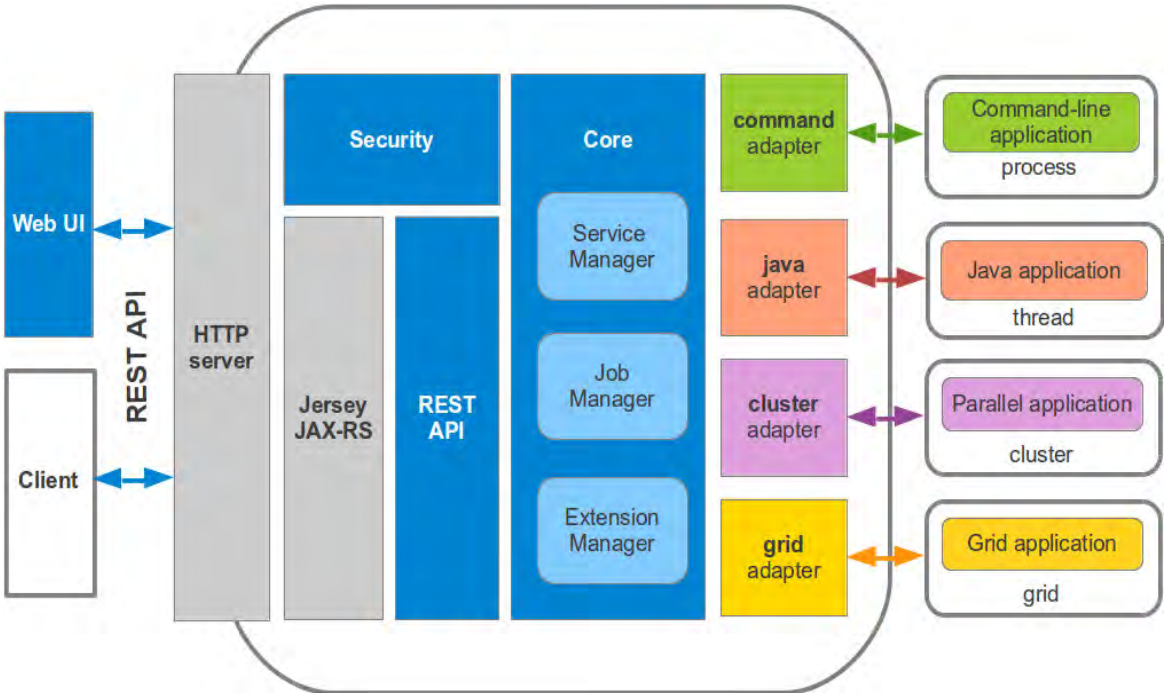


Fig. 2: Service container architecture

The service container simplifies the service development and deployment by providing ready-to-use adapters for command-line, Java, cluster and grid applications. This makes it easy, in many cases without writing a code, to transform into services a wide range of existing 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. Besides, the support for pluggable adapters allows one to attach arbitrary service implementations and computing resources. For example, such an adapter can implement access to the grid infrastructure by transforming service requests to jobs submitted to grid.

Service container is based on Jersey library, a reference implementation of JAX-RS (Java API for RESTful Web Services) specification. The container uses built-in Jetty web server for interaction with service clients. Incoming HTTP requests are forwarded to Jersey and then to service container. The communication between Jersey and the container is implemented by means of Java classes that correspond to resources from the described REST API interface.

Everest is processing client requests in accordance with configuration information. The ServiceManager component maintains a list of services deployed in the container and their configuration. The JobManager component manages the processing of incoming requests. The requests are converted into jobs and placed in a queue served by a configurable pool of handler threads. During job processing, handler thread invokes adapter specified in the service configuration.

The components that implement processing of service requests (jobs) are provided in the form of pluggable adapters. Each adapter implements a standard interface through which the container passes request parameters, monitors the job state and receives results. The adapter implementation usually converts a service request to an execution of external application.

Each service deployed in container is published via REST API. In addition, the container provides a complementary web interface allowing users to access services via a web browser.

Service Catalogue

After the service is built and deployed in the service container, it can be published using the service catalogue component which supports discovery, monitoring and annotation of services. The catalogue is implemented as a web application with interface and functionality similar to modern search engines.

A service can be published in the catalogue by providing a URI of the service and tags describing it. The catalogue retrieves service description via its URI, performs indexing and stores description along with specified tags in a database.

The catalogue implements a familiar search query interface with optional filters and supports full text search in service descriptions and tags. Search results consist of short snippets of each found service with highlighted query terms and a link to full service description. In order to provide current information on service availability the catalogue periodically pings published services. If a service is not available, it is marked accordingly in search results.

Workflow Management Service

In order to simplify composition of services into various application scenarios, a workflow management system is implemented. The system supports description, storage, publication and execution of workflows composed of multiple services. Workflows are represented as directed acyclic graphs and described by means of a visual editor. The described workflow can be published as a new composite service and then executed by sending request to this service. The system hides from users the low-level details of service calls and data transfer between services, leaving only the need for correct connection of services with each other. This enables rapid development of new applications and services by users without distributed programming skills.

Fig. 3 shows the interface of the workflow editor. It is implemented as a web application in JavaScript language. Thus the editor can be used without installation on any computer running a modern web browser. The interface inspired by Yahoo! Pipes provides easy-to-use tools for building workflows by connecting services with each other. The right side of the editor contains a list of available services and other basic blocks from which the user can compose a workflow. The upper part is a main menu that provides access to basic operations with workflows, such as opening, saving, running, etc. The main area of the editor contains a graphical representation of workflow.

Security

All platform components use a common security mechanism (Fig. 4) for protecting access to services. It supports authentication, authorization and a limited form of delegation based on common security technologies.

Authentication of services is implemented by means of SSL server certificates. Authentication of clients is implemented via two mechanisms. The former is a standard X.509 certificate. The latter is Loginza service which supports authentication via popular identity providers (Google, Facebook, etc.)

or any OpenID provider. Authorization is supported by means of allow and deny lists which enable service administrator to specify users which should or should not have access to a service.

A common security challenge in both grid and service-oriented environments is providing a mechanism for a service to act on behalf of a user, i.e. invoke other services. The most important use case in MathCloud is a workflow service which needs to access services involved in the workflow on behalf of a user invoked the service. For such cases a proxying mechanism is implemented by means of proxy list which enable service administrator to specify certificates of services that are trusted to invoke the service on behalf of users. This approach is more limited but it provides a more light-weight solution in comparison to the proxy certificates used in grids.

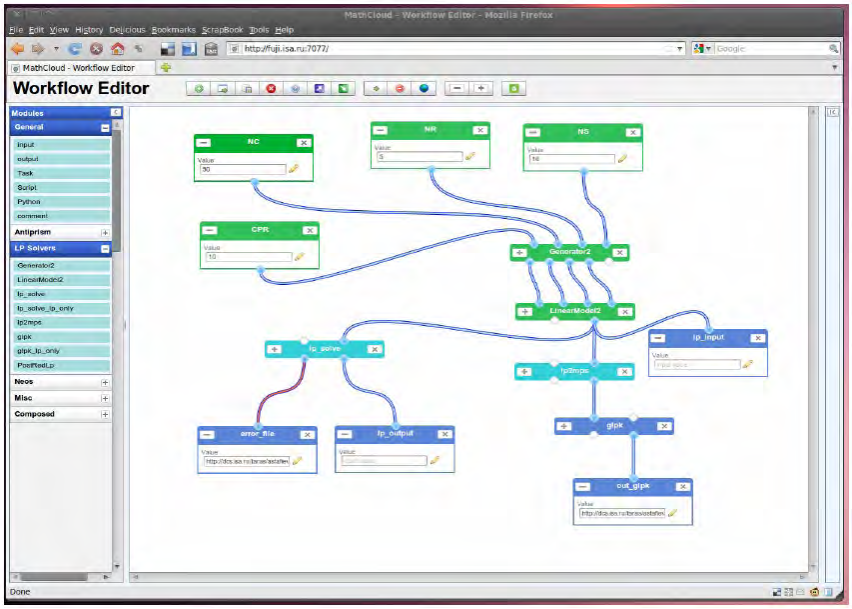


Fig. 3: Graphical workflow editor

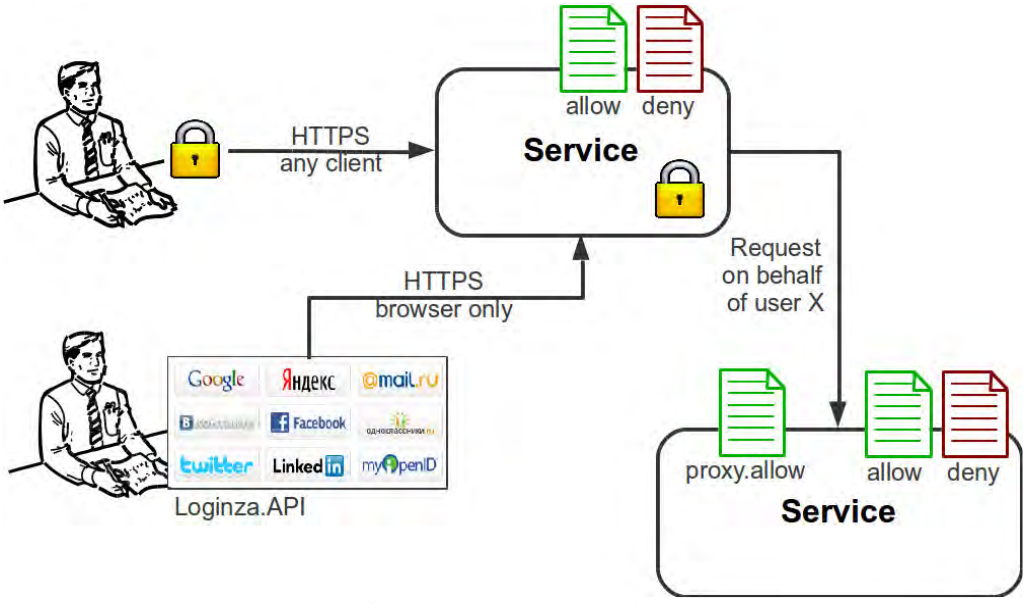


Fig. 4: Security mechanism

4 Future Directions

The future development of MathCloud platform will be focused on building a hosted Platform-as-a-Service (PaaS) for computing services based on described software toolkit. The cloud version of the platform will provide the same functionality for building, deployment, discovery and integration of computing services.

This approach has several advantages in comparison to the current toolkit distribution. For service developers it reduces a development time since there is no need to download, install, configure and update software on their servers. For platform developers it simplifies management, update and support of platform components since there is only one copy of the software. Thus the cloud platform will streamline development and foster innovations for both sides.

The proposed direction also has some challenges. The biggest of them is how to connect the cloud platform with external computing resources to run service jobs, since it is both unrealistic and unpractical to provide such computing infrastructure within the platform. The proposed approach is to provide means for service developers and users to pass credentials to the platform to access computing resources on behalf of them. This approach has some security and trust issues which will be investigated in the future research.

References

[1] MathCloud project. http://mathcloud.org/

[2] I. Foster. Service-Oriented Science // Science. – 2005. – V. 308, N. 5723. - P. 814-817.

[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] JSON Schema. http://www.json-schema.org/

DEPENDABLE JOB-FLOW DISPATCHING AND SCHEDULING IN VIRTUAL ORGANIZATIONS OF DISTRIBUTED COMPUTING ENVIRONMENTS[1]

V.V. Toporkov, A.S. Tselishchev, D.M. Yemelyanov, A.V. Bobchenkov
National Research University "MPEI",
ul. Krasnokazarmennaya 14, Moscow, 111250 Russia,
ToporkovVV@mpei.ru, {pjcrew, yemelyanov.dmitry, groddenator}@gmail.com

1 Introduction

Heterogeneity, changing composition, different owners of different nodes whose computing time is partially shared by users turn the organization of a distributed computational environment into an especially difficult task. Utility Grid [1], multi-agent systems [2] and cloud computing [3] are types of distributed environments where usage of economic mechanisms is seen as promising. Those economic mechanisms are designed to solve tasks like resource management and scheduling of user jobs in a transparent and efficient way. Within the context of any used economic model the interests of different participants of a distributed computing environment (such as end-users or node owners) are often contradictory. Since the resources of distributed environment such as Grid are non-dedicated, it is assumed that node owners may have local job flows (their own tasks) and global job flow (which is formed by external user jobs) competing for limited computational resources of the node. Elaboration of pricing rules which are used to calculate a fee for node computing time usage and take into account user-required quality of service (QoS) is also a very serious problem [1-3]. An overview of various approaches to this problem is given in [4]. Heuristic algorithms for resource selection based on user-given utility function are described in [5]. Some resource management models offer simple search and selection of resources required by a user [6] and do not support any optimization. Others do not take into account features related to global and local job competition, the competition among users and other characteristics of distributed environments with non-dedicated computational resources [7]. A resource broker model [1-5] dynamically employs various economic policies which perform resource management which is decentralized and application-specific and have two parties: node owners and brokers representing users. Another common trend is related to virtual organizations [7-9] with central schedulers providing job-flow level scheduling and optimization. While former type of resource management is well-scalable, the simultaneous satisfaction of various application optimization criteria submitted by independent users is unreachable in essence and also can deteriorate such integral quality of service rates as total execution time of a sequence of jobs or overall resource utilization. The latter type, virtual organizations naturally restrict the scalability. However, scheduling based on uniform and controlled rules for allocation and consumption of resources makes it possible to improve the efficiency of resource usage and find a tradeoff between contradictory interests of different participants.

In this work, we propose a two-level model of the resource management system which is functioning within a virtual organization (VO). Resource management is implemented with a hierarchical structure consisting of a metascheduler and subordinate job schedulers that are controlled by the metascheduler and in turn interact with resource managers (e.g., with batch job processing systems). The application-level optimization begins when job-flow level optimization is finished. Such a flexible structure coupled with complex metascheduling approach enables multiaspect resource

[1] This work was partially supported by the Council on Grants of the President of the Russian Federation for State Support of Leading Scientific Schools (SS-316.2012.9), the Russian Foundation for Basic Research (grant no. 12-07-00042), and by the Federal Target Program "Research and scientific-pedagogical cadres of innovative Russia" (State contracts 16.740.11.0038 and 16.740.11.0516).

management and makes possible to control dynamic priority of job execution, resource selection and provide multicriterial optimization both on the job-flow scale and for specific job, according to its submitter requirements and optimization criteria. Hence, we may speak not only of a scheduling algorithm but rather of a scheduling strategy that is a combination of various methods of external and local scheduling. Such a mechanism allows finer control and higher overall resource management efficiency in a distributed computing environment. *Resource* is defined as an abstract computational entity, which can be used for execution of one and only one *task*. The complex set of connected interrelated tasks form a *job*. In some applications jobs require co-scheduling and resource co-allocation on several resources [10-13]. In this case resource allocation has a number of substantial specific features caused by autonomy, heterogeneity, dynamic content changes, and node failures [6-9]. In our model jobs are submitted to the system by end-users. The proposing approach is more or less the same as used in gLite Workload Management System, where Condor is used as a scheduling module [14]. But the significant difference between the approach proposed in this work and well-known scheduling solutions for distributed environments such as the Grid [1, 3-7] is the fact that the execution strategy is formed on a basis of formalized efficiency criteria, which efficiently allows one to reflect economic principles of resource allocation by using relevant cost functions and solving a load balance problem for heterogeneous processor nodes. At the same time the inner structure of the job is taken into account when the resulting schedule is formed. Thus, two approaches are uniquely combined in a proposed two-tier model.

This work is organized as follows. Section 2 overviews model components and metascheduling workflow. In section 3 a strategy search is formalized. Section 4 contains simulation results. Section 5 summarizes the work and describes further research topics.

2 Basic notions and informal model components description

Let us define basic model components presented in this work.

- VO, that defines resource co-allocation dispatching strategies, pricing policies and resource load-balancing mechanisms.
- Heterogeneous hierarchical computational environment that contains computational resources (Grid nodes, CPUs or others) with different performance indices. Each resource is considered as non-dedicated (i.e. it can have its own internal schedule and these schedules are sent to application-level schedulers upon request).
- Metascheduler, which implements resource management strategies and policies of the virtual organization.
- Application-level schedulers that analyze internal job structure and schedule single tasks.

The VO in our model of distributed computational environment includes three independent parties with their own interests.

- End-users of services provided within the VO such as computation services. End-users take steps to make resource requests to the environment, according to resource performance, time and budget estimations needed for running custom user jobs.
- VO administrators that set up resource usage policies to optimize scheduling and improve load balance. The administrators control metascheduler process running in the environment which is in fact the part of VO infrastructure software. Thus they are directly responsible for managing the parameters of higher level resource management.
- Owners of computational nodes that comprise the environment network and hardware base of the distributed computing environment. The owners offer part of their nodes computing time to VO for a fee. Computational nodes provide the only type of distributed resources used in our model.

Each computational node of the heterogeneous environment is mapped to a computational *resource line* in the metascheduler resource management routine. Several resource lines are combined into a virtual resource domain. Each resource line has two static attributes which are its performance P and its base price tag F for a computing time unit. The performance is an inherent parameter of a node and the base price tag is assigned by its owner. The dynamic characteristic of a node is represented

with its local schedule which is a list of slots available for reservation. This list is sent to metascheduler by request. A slot is a continuous interval of time and is described with three parameters: its start time, its length and its fee [10-12]. The fee is calculated when the metascheduler applies its pricing policies taking in account resource type, slot length etc.

A resource request is a set of a few constraints determined by a user which correspond to the properties of the respective user job. They include: minimal performance requirement for computational nodes, P_{min}; maximal price tag for a single timeslot, F_{max}; number n of simultaneously reserved timeslots; minimal slot length; the internal structure of a job as a directed acyclic graph (DAG), where vertices represent single tasks and edges represent data dependencies [13]; deadline for the job execution. A job may require more than one timeslot if it includes several segments that can be executed in parallel way, for instance. Then the user specifies the number of reserved timeslots and minimal performance requirement that applies for them all. The whole job budget is determined by the timeslot number and the maximum price per timeslot. The minimal timeslot length requires an additional explanation. This is the minimal time estimated by the user which is required to complete job execution given the performance of the nodes meet the minimal requirement P_{min}. Hence, the metascheduler and the user share the responsibility since the probability of being run successfully for a job equally depends on primary user estimates and overall scheduling quality.

The hierarchical model of the computational environment implies two-tier scheduling (Fig. 1). On the job-flow level the set of independent jobs is distributed between resource domains according to dispatching strategies and economic criteria. Schedule on this level is defined by the metascheduler as a slot set for each job, which is optimal in terms of a whole job set. Application-level schedulers receive the list of resources which were meant to execute the job on and a strategy, which defines the rule used to execute tasks of a concrete job. On this level an optimal slot and specific resource are defined for each single task in a job, thus, making it possible to take internal job structure into account. On the job-flow level all end-user jobs are initially submitted into the global queue. The metascheduler can manage one or more job-flows which become sub-queues of the global queue. The mechanism of distribution of jobs between job-flows can be random or based on current load and actual efficiency of scheduling in certain job-flows. Scheduling process in each job-flow is performed by identical scheduling instance. We consider a single job-flow case.

The metascheduler works in cycles which are quanta of its process. For each cycle it has the following information: information about distributed computing environment as a set of resource lines and the global job queue.

What it needs then is a batch of jobs which is a ranked job list and a subset of available slots for a specific virtual resource domain and a certain timeframe which is called a scheduling interval. The length of the batch and the scheduling interval are parameterized by VO administrators. Jobs are fetched into the batch accordingly to several variables, such as the maximum price tag, deadline, and the number of failed scheduling attempts for a job. These variables being weighted and added up determine job rank according to which it takes a position closer to head or tail of a batch. The preparation phase ends and the actual scheduling process is executed as follows (see Fig. 1). The metascheduler analyzes available slots and finds an optimal slot combination to accommodate every job in a batch using economic criteria. The budget and the deadline defined by the end-user are considered during this step. The algorithms for this step were detailed in [10-12]. After the domain is determined metascheduler defines the strategy for each job. For example as shown on Fig. 1, the user, who has sent the job *i* has the higher budget than the one who has sent the job *k*. The strategy for *i* may be expressed as "*execute as soon as possible*" while the strategy for *k* may be expressed as "*execute as late as possible within the defined deadline*". These jobs are later sent to application-level schedulers and the application-level scheduling begins. Application-level schedulers query internal schedules for all the resources which were selected during step 2 for each job, analyze the job DAG and form a resulting schedule for every task according to the strategy from step 2. These schedules must support interruptions and delays and should be optimal in terms of the defined criteria (i.e. cost or resource load). The criterion for the job i would be to minimize execution

cost within the defined budget, criterion for the job k would be to maximize average resource load while meeting the defined deadline. As shown on Fig. 1, jobs i and k are scheduled to be executed on the same set of resources at once. Application-level schedulers are guaranteeing that there are no collisions between the tasks which were scheduled during step 3 and local tasks, which may have priority over the job-flow from step 1.

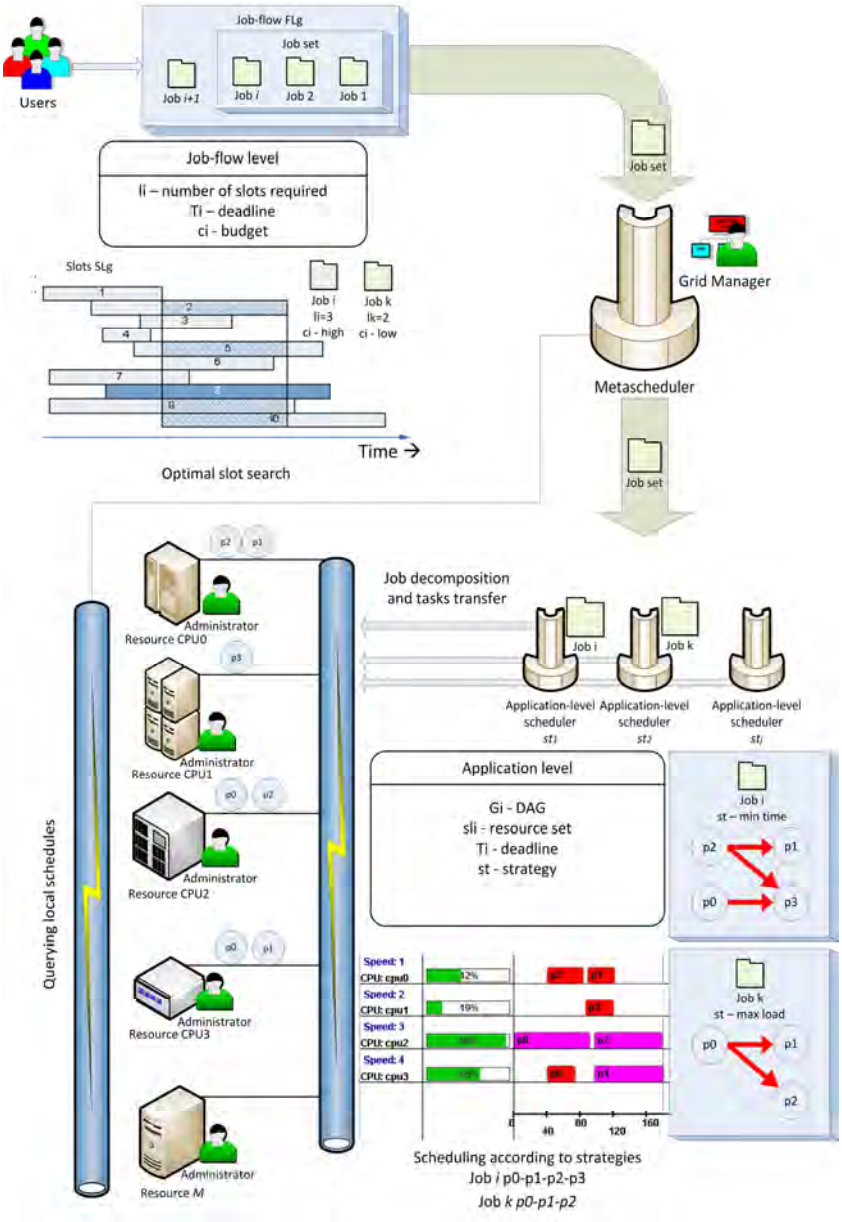


Fig.1: Model components

3 Formalization of scheduling

Let us note a global resource set $\boldsymbol{R_g = \{ r_p, p = 1,..,M \}}$, which includes all resources. A global job-flow is a set of jobs received by the metascheduler in time: $FL_g = \{l_i, c_i, T_i, G_i, i = 1,.., I\}$, where the job i is represented as l_i – the amount of resource slots required, c_i - the maximal budget end-user is ready to allocate for execution of the job, T_i – deadline, G_i – the job DAG. Metascheduler at

any time point may query each resource, receive its local schedule and build a set of slots S_{gt} – idle time intervals. Let us introduce a set of strategies $ST = \{st_l, l = 1,..,L\}$, which are based on economic criteria and defined by Grid-managers and developers. Let SL be a set of K slots suitable to execute a subset of jobs $FL_p \subseteq Fl_g$. A slot set is considered as suitable for the job i if the execution is possible in terms of the resource number, the budget c_i and the deadline T_i. It is assumed that for every job there is at least one suitable slot set $sl_i \in SL, sl_i = k, k \in \{1,..,K\}$. On a job-flow level for each job the metascheduler aims at finding a slot set sl_i and a strategy st_i for which the value of the function $g_i(sl_i)$, that defines whether the slot set is being effective for the job i, would be optimal [11]. The internal job structure G_i is not taken into account at this time. The mechanism to define $g_i(sl_i)$ which was developed in the previous works [10-12] is now improved. According to the resource request it is required to find a "window" with the following description: n concurrent time-slots providing resource performance rate at least P and maximal resource price not higher than F_{max} should be reserved for a time span T_i (the resource request type was described in more detail above). The length of each slot in the window is determined by the performance rate of the node on which it is allocated. Thus as a result we have a window with a "rough right edge" (Fig. 2). In addition, the criterion of selecting the most suitable set of slots could be specified. This could be the minimum cost, the minimum runtime or, for example, the minimum power consumption criterion. The window search is performed on the list of all available system slots sorted by their start time in ascending order (this condition is necessary to examine every slot in the list and for operation of search algorithms of linear complexity [10-12]).

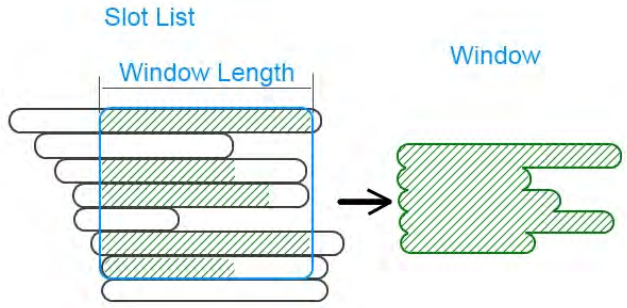


Fig. 2: Window with a "rough right edge"

The scheme of a search for a window that meets the requirements and effective by the given criterion can be represented as follows.

1. From the list of available system slots the next suitable slot s_k is extracted and examined. Slot s_k suits, if following conditions are met: **a)** resource performance rate $P(s_k) \geq P$ for slot s_k; **b)** slot length (time span) is enough (depending on the actual performance of the slot's resource) $L(s_k) \geq T_i * P(s_k)/P$. If conditions **a)** and **b)** are met, the slot s_k is successfully added to the window list.

2. A current window start time is a set equal to the start time of the last added slot.

3. Slots whose length has expired considering new window start time T_{last} are removed from the list. The expiration means that remaining slot length $L'(s_k)$, calculated like shown in **step** 1°**b**, is not enough assuming the k-th slot start is equal to the last added slot start: $L'(s_k) < (T_i + (T_{\text{last}} - T(s_k)))P(s_k)/P$, where $T(s_k)$ is the slot's start time. Any combination of the remaining slots can form a window of necessary length.

4. If the number of slots m in the current window is greater or equal to n, it is required to select n slots, effective on the specified criteria and at the same time satisfying the total cost and deadline restrictions. Suppose the window W of size n with a target criterion value equal to crW was selected. The problem of selecting efficient window consisting of n slots in the case of $m > n$ will be described below.

5. The target criterion value crW of window W is compared with the cr' – the current best target criterion value for all previously found windows. If $crW < cr'$ (in case of a minimization problem) the window W announced as a new window-candidate and crW becomes the new best criteria value: $\boldsymbol{cr' = crW}$. Go to **step 1**°.

6. The algorithm ends after the last available slot is processed. The result of the algorithm is the window-candidate with the best target criteria value.

The described algorithm can be compared to the algorithm of maximum/minimum value search in an array of flat values. The expanded window of size m "moves" through the ordered list of available system slots. At each step any combination of n slots inside it (in case when $n \le m$) can form a window that meets all the requirements to run the job. The effective on the specified criteria window of size n is selected from this m slots and compared with the results in the previous steps. By the end of the slot list the only solution with the best criteria value will be selected. Consider the problem of selecting a window of size n with a total cost no more than S from the list of $m > n$ slots (in case when $m = n$ the selection is trivial). The maximal budget is counted as $S = Ft_s n$, where t_s is a time span to reserve and n is the necessary number of slots. The current extended window consists of m slots $s_1, s_2, ..., s_m$. The cost of using each of the slots according to their required length is: $c_1, c_2, ..., c_m$. Each slot has a numeric characteristic z_i the total value of which should be minimized in the resulting window. Then the problem could be formulated as follows:

$$a_1 z_1 + a_2 z_2 + ... + a_m z_m \to \min,\ a_1 c_1 + a_2 c_2 + ... + a_m c_m \le S,$$

$$a_1 + a_2 + ... + a_m = n,\ a_r \in \{0,1\},\ r = 1,...,m.$$

Additional restrictions can be added, for example, considering the specified value of deadline. Finding the coefficients $a_1, a_2, ..., a_m$ each of which takes integer values 0 or 1 (and the total number of '1' values is equal to n), determine the window with the specified criteria extreme value. Job-flow level scheduling ends here.

Application-level schedulers receive following input data.

- The optimal slot set sl and the description of all corresponding resources: $R = \{r_j, j = 1,..,J\} \subseteq R_g$.
- The directed acyclic information graph $G = \{V, E\}$, where $V = \{v_i, i = 1,..n\}$ is a set of vertices that correspond to job tasks, for each of those execution time estimates τ_{ij}^0 on each of resources in R are provided, E – is a set of edges that define data dependencies between tasks and data transfer time intervals.
- The dispatching strategy st, which defines the criterion for a schedule expected
- The deadline T_i or the maximal budget c_i for the job (depends on a dispatching strategy and $g_i(sl_i)$.

The schedule which is being defined on an application level is presented as follows: $Sh = \{[s_i, f_i], \alpha_i, i = 1,..,n\}$, where $[s_i, f_i]$ is a time frame for a task i of a job and α_i - defines the selected resource. Sh is selected in the way that the criterion function $C = f(Sh)$ achieves an optimum value.

The *critical jobs method* [13] which is used to find the optimal schedule and to define f consists of three main steps: 1) forming and ranging a set of critical jobs (longest sets of connected tasks) in the DAG; 2) consecutive planning of each critical job using dynamic programming methods; 3) resolution of possible collisions.

A detailed algorithm description is presented in [13].

4 Simulation results

The two-tier model described in the sections 2 and 3 was implemented in a simulation environment on two different and separated levels: on the job-flow level, where job-flows are optimally distributed between resource domains and on the application level, where jobs are decomposed and each task is executed in an optimal way on a selected resource. Job-flow level metascheduling was simulated in a specially implemented and configured software that was written to test the features of the two-tier resource management. An experiment was designed to compare the performance of our job-flow level metascheduling method with other approaches such as FCFS and backfilling. Let us remind that our scheduling method detailed in works [10] and [11] involves two stages that backfilling does not have at all, namely, slot set alternative generation and further elaboration of specific slots combination to optimize either time or cost characteristic for an entire job batch. Backfilling simply assigns "slot set" found to execute a job without an additional optimization phase. This behavior was simulated within our domain with random selection from an alternative slot, each job having one or more of them. So two modes were tested: with optimization and without optimization.

The experiment was conducted as follows. Each mode was simulated in 5000 independent scheduling cycles. A job batch and environment condition was regenerated in every cycle in order to minimize other factor influence. A job batch contained 30 jobs. Slot selection was consistent throughout the experiment. If a job resource request could not be satisfied with actual resources available in the environment, then it was simply discarded.

For optimization mode as well as for no-optimization mode four optimization criteria or problems were used: 1) maximize total budget, limit slot usage; 2) minimize slot usage, limit total budget; 3) minimize total budget, limit slot usage; 4) maximize slot usage, limit slot budget.

Optimization mode, which is using additional optimization phase after slot set generation, wins against random slot selection with about 13% gain in the problem 1 whose concern is about maximizing total slot budget thus raising total economical output per cycle and owners' profits.

Optimization mode wins against random slot selection with about 10-12% gain for the problems 2-4. The experiment results show the advantage of the critical jobs method usage in a two-tier scheduling model compared to consecutive application-level scheduling: while the scheduling cost for a job is more or less the same, 1000 jobs are planned 25% faster.

Consider another experiment: while changing the length of the scheduling interval, we will estimate the proportion of successfully distributed jobs. The length of the scheduling interval is equal to $L = l * h, h = 1.0,..,2.6,$ with step 0.2, where l is the length of the longest critical path of tasks in the job and h is a distribution interval magnification factor. There were carried 200 experiments for each h (bold points on Fig. 3). Analysis of the Fig. 3 shows that increasing the scheduling interval (relatively to the execution time of the longest critical path on the nodes with the highest performance) is accompanied by a significant increase in the number of successfully distributed jobs. The detailed study of this dependence can give a priori estimates of an individual job successful distribution probability.

In the next experiment we will consider a dependence of successful distributions number and the number of collisions per experiment on the level of resource instances availability. The experiments were performed in conditions of limited resources using the specific instances of the resources. The number of resources J in each experiment was determined as $J = j * N$, where j – factor (x-axis) and N – number of tiers in the graph. Fig. 4 shows results of the experiments with different j values and $N = 3,5,7$.

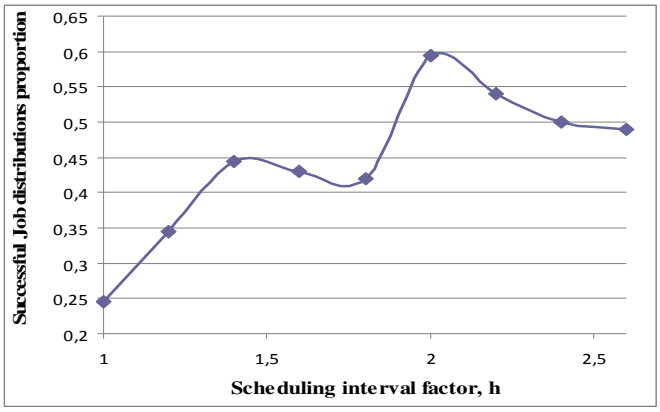
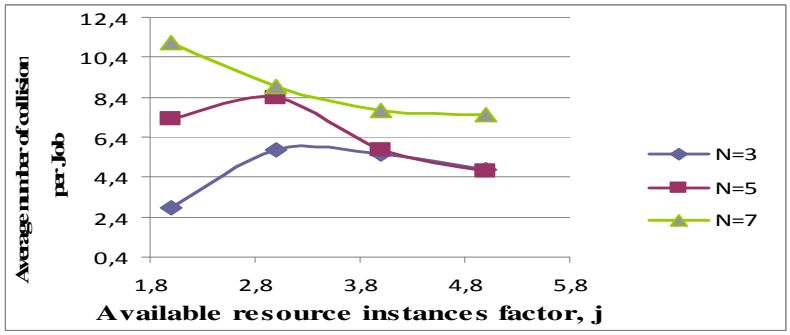
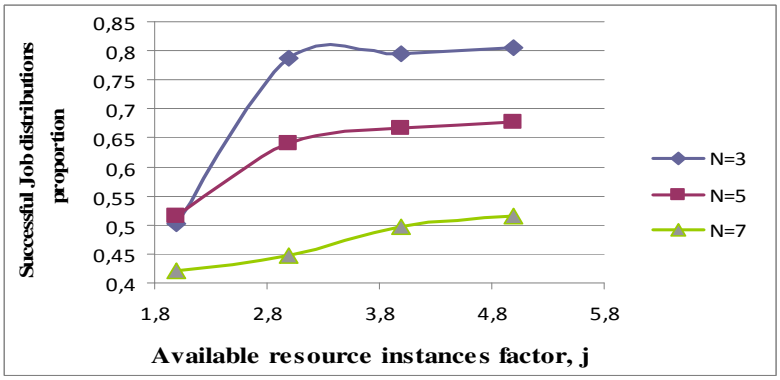


Fig. 3: Dependence of the proportion of the successful job distributions on the length of the distribution interval

The obtained dependencies (Fig. 4) suggest that the collisions number depends on the resources availability. The lower the number of resource instances and the greater the number of tiers in the graph – the more collisions occurred during the scheduling. At the same time the number of resource instances affects the successful distribution probability. With a value of $j > 4$ (that is, when the number of available resource instances is more than 4 times greater than the number of tiers in the graph) all cases provide the maximum value of successful distribution probability. These results are subject of future research of refined strategies on a job-flow level.



(a)



(b)

Fig. 4: Simulation results: resource dependencies of collisions number (a) and successful job distribution proportion (b)

5 Conclusions and future work

In this work, we address the problem of independent job-flow scheduling in heterogeneous environment with non-dedicated resources.

Each job consists of a number of interrelated tasks with data dependencies. Using the combination of existing methods with a number of original algorithms the resulting schedules are computed. These schedules meet the defined deadlines and budget expectations, provide optimal load-balance for all the resources and follows virtual organization's strategies, thus, allowing to achieve unprecedented quality of service and economic competitiveness for distributed systems such as Grid. The experiments which were conducted are showing the efficiency of methods developed for both job-flow and application level scheduling. The model proposed is showing the way these methods and advantages can be converged in one place making it possible to achieve the main goal.

Future research will include the simulation of connected job-flow and application levels and experiments on real Grid-jobs in order to get finer view on advantages of the approach proposed.

References

[1] Garg SK, Buyya R, Siegel HJ (2009) Scheduling parallel applications on utility Grids: time and cost trade-off management. Proc of ACSC 2009, Wellington, New Zealand: 151-159.

[2] Tesauro G, Bredin JL (2002) Strategic sequential bidding in auctions using dynamic programming. Proc of the First international joint conference on Autonomous agents and multiagent systems: part 2, ACM New York, NY, USA: 591 – 598.

[3] Garg SK, Yeo CS, Anandasivam A, Buyya R (2011) Environment-conscious scheduling of HPC applications on distributed cloud-oriented data centers. J. of Parallel and Distributed Computing. 71(6): 732-749.

[4] Buyya R, Abramson D, Giddy J (2002) Economic models for resource management and scheduling in Grid computing. J. of Concurrency and Computation: Practice and Experience. 14(5): 1507–1542.

[5] Ernemann C, Hamscher V, Yahyapour R (2002) Economic scheduling in grid computing. Proc of the 8th Job Scheduling Strategies for Parallel Processing. Eds D.G. Feitelson, L. Rudolph, U. Schwiegelshohn. Heidelberg: Springer, LNCS. 2537: 128-152.

[6] Voevodin V (2007) The Solution of Large Problems in Distributed Computational Media. Automation and Remote Control. Pleiades Publishing, Inc. 68(5): 773-786.

[7] Kurowski K, Nabrzyski J, Oleksiak A et al. (2003) Multicriteria aspects of Grid resource management. Grid resource management. State of the art and future trends. Eds J. Nabrzyski, J.M. Schopf and J. Weglarz. Kluwer Acad. Publ.: 271–293.

[8] Toporkov V (2009) Application-level and job-flow scheduling: an approach for achieving quality of service in distributed computing. Proc of PaCT 2009, LNCS 5698. Berlin, Heidelberg: 350 – 359.

[9] Toporkov VV (2009) Job and application-level scheduling in distributed computing. Ubiquitous Comput Commun J 4: 559-570.

[10] Toporkov V, Toporkova A, Bobchenkov A, Yemelyanov D (2011) Resource selection algorithms for economic scheduling in distributed systems. Procedia Computer Science. Elsevier. 4: 2267-2276.

[11] Toporkov V, Yemelyanov D, Toporkova A, Bobchenkov A (2011) Resource co-allocation algorithms for job batch scheduling in dependable distributed computing. Dependable Computer Systems. Springer-Verlag, AICS. V. 97. Berlin, Heidelberg: 243-256.

[12] Toporkov V, Bobchenkov A, Toporkova A, Tselishchev A, Yemelyanov D (2011) Slot selection and co-allocation for economic scheduling in distributed computing. Proc of the 11th Intern. Conf. on Parallel Computing Technologies. Springer-Verlag, LNCS. 6873. Berlin, Heidelberg: 368–383.

[13] Toporkov VV, Tselishchev AS (2010) Safety scheduling strategies in distributed computing. Intern. J. of Critical Computer-Based Systems. 1(1/2/3): 41-58.

[14] Cecchi M, Capannini F, Dorigo A et al. (2010) The gLite Workload Management System. Journal of Physics: Conference Series 219(6): 062039.

ADVANCEMENTS IN BIG DATA PROCESSING IN THE ATLAS AND CMS EXPERIMENTS[1]

A.V. Vaniachine
on behalf of the ATLAS and CMS Collaborations
Argonne National Laboratory, 9700 S Cass Ave, Argonne, IL, 60439, USA

The ever-increasing volumes of scientific data present new challenges for distributed computing and Grid technologies. The emerging Big Data revolution drives exploration in scientific fields including nanotechnology, astrophysics, high-energy physics, biology and medicine. New initiatives are transforming data-driven scientific fields enabling massive data analysis in new ways.

In petascale data processing scientists deal with datasets, not individual files. As a result, a task (comprised of many jobs) became a unit of petascale data processing on the Grid. Splitting of a large data processing task into jobs enabled fine-granularity checkpointing analogous to the splitting of a large file into smaller TCP/IP packets during data transfers. Transferring large data in small packets achieves reliability through automatic re-sending of the dropped TCP/IP packets. Similarly, transient job failures on the Grid can be recovered by automatic re-tries to achieve reliable 6σ production quality in petascale data processing on the Grid.

The computing experience of the ATLAS and CMS experiments provides foundation for reliability engineering scaling up Grid technologies for data processing beyond the petascale.

1. Introduction

Today, various projects and initiatives are under way addressing the challenges of Big Data. For example, the data produced at the LHC and other advanced instruments present a challenge for analysis because of petascale data volumes, increasing complexity, distributed data locations and chaotic access patterns.

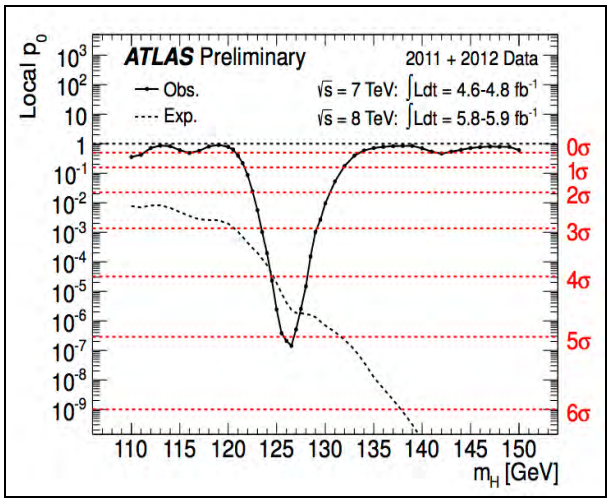


Figure 1: The Higgs boson discovery culminated many years of search for new physics phenomena driven by the Big Data revolution in Grid computing technologies [1]

2. Big Data Processing at the LHC

To address petascale Big Data challenge, the LHC experiments are relying on the computational infrastructure deployed in the framework of the Worldwide LHC Computing Grid. Following Big Data processing on the Grid, more than 8000 scientists analyze LHC data in search of discoveries. Culminating the search, a seminar at CERN on July 4 presented the results of the Higgs boson search at the LHC. Figure 1 shows the significance of the search achieved by the ATLAS experiment in combination of several Higgs decay channels. A standard for discovery – the five sigma significance of the result – corresponds to the 3 10^{-7} probability of the background fluctuation to mimic the Higgs signal (local p_0 value).

[1] Invited talk presented at the V International Conference on "Distributed computing and Grid-technologies in science and education" (Grid2012), JINR, Dubna, Russia, July 16–21, 2012.

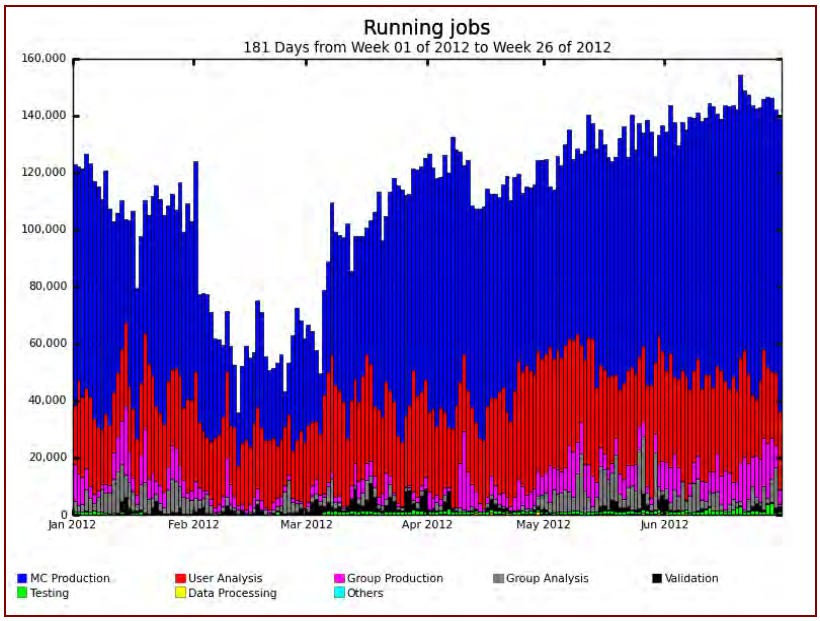


Figure 2: During first half of 2012, the number of concurrent Grid jobs in the ATLAS experiment routinely exceeded the level of 100,000. These jobs were running at the CERN Tier-0 site, ten large Tier-1 sites, and more than eighty smaller Tier-2 sites [1]

The observed signal significance is related to the LHC luminosity resulting in high data acquisition rates and petabytes of recorded data volumes, which requires significant computing power to process. Depending on conditions, it takes 3–6 10^6 core-hours to processes one petabyte of LHC data. Even higher computing power is required to produce simulated events required for the signal and background selection studies. Speakers at the CERN seminar acknowledged the role of Grid computing technologies in the discovery. It would have been impossible to release physics results so quickly without the outstanding performance of the Grid, including the CERN Tier-0 site. The ATLAS Grid resources were fully used. The number of running jobs often exceeded 100,000 including simulations, user analysis and group production (Figure 2). Figure 3 shows simulation capabilities of Grid computing in the CMS experiment.

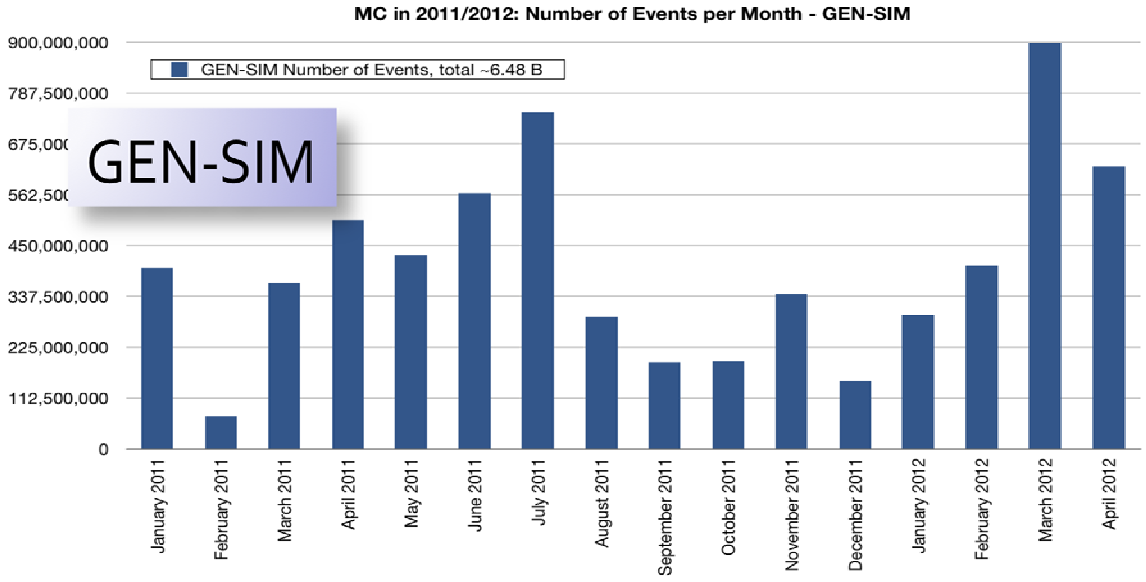


Figure 3: Thanks to the Grid computing capabilities, the CMS experiment achieved the sustained simulations rate at the level of 400,000,000 events per month [2]

3. Genesis

Figure 4 shows the time evolution of the Universe. As the temperature decreased with time, transitions took place. Among the Big Data experiments at LHC, the ALICE experiment studies the QCD phase transition, the ATLAS and CMS experiments probe the electroweek phase transition by observing the Higgs properties. It is possible that the matter-antimatter symmetry was broken earlier via the CP–violation mechanism [4]. However, the CP–violation in the Standard Model is too small for baryogenesis indicating new physics beyond the Standard Model. Future experiments in search for new physics (SuperB and Belle II) require Big Data technologies. In a high rate scenario, the Belle II experiment acquires data at the rate of 1,800 MB/s – at the level expected for all LHC experiments combined [5]. The storage needs grows from 50 PB to 600 PB in six years of the SuperB experiment [6]. Both experiments adopted Grid computing for Big Data processing [7, 8].

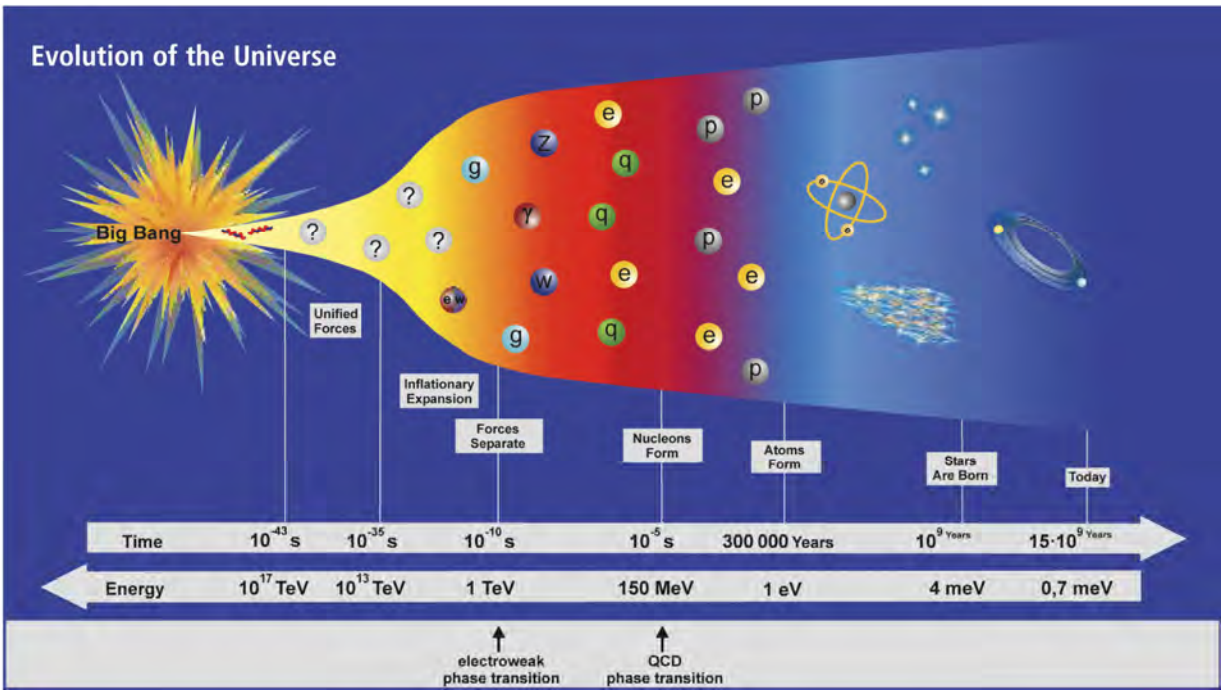


Figure 4**:** Evolution of the Universe [3]

4. Six Sigma Quality for Big Data

In industry, Six Sigma analysis improves the quality of production by identifying and removing the causes of defects. A Six Sigma process is one in which products are free of defects at 0.3 10^{-5} level because an industrial Six Sigma process corresponds to the mathematical 4.5σ after taking into account the 1.5σ shift from variations in production.

In contrast, LHC Big Data processing achieves 6σ quality in a true mathematical sense – the 10^{-8} level of defects. Figure 5 shows why physics requires 6σ quality during Big Data processing. In comparison to known physics processes, the production rate of new phenomena is very small. To select interesting data, LHC experiments employ hardwired multi-level data selection mechanisms (online trigger) followed by flexible offline selections (offline data processing). The production rates for Higgs physics at the LHC energies result in low selection rates – few events are selected out of billions recorded; experiments cannot loose even one event. Figure 6 shows number of events (after all selections) in the "golden" Higgs discovery channel illustrating that the event selection at the LHC is indeed at the 10^{-9} level [10].

Failure recovery by re-tries achieves production quality in Big Data processing on the Grid at the 6σ level [11]. No events were lost during the main ATLAS reprocessing campaign of the 2010 data that reconstructed on the Grid more than 1 PB of data with 0.9 10^{9} events. In the last 2011 data

reprocessing, only two collision events out of 0.9 10^9 events total could not be reconstructed. (These events were reprocessed later in a dedicated data recovery step.)

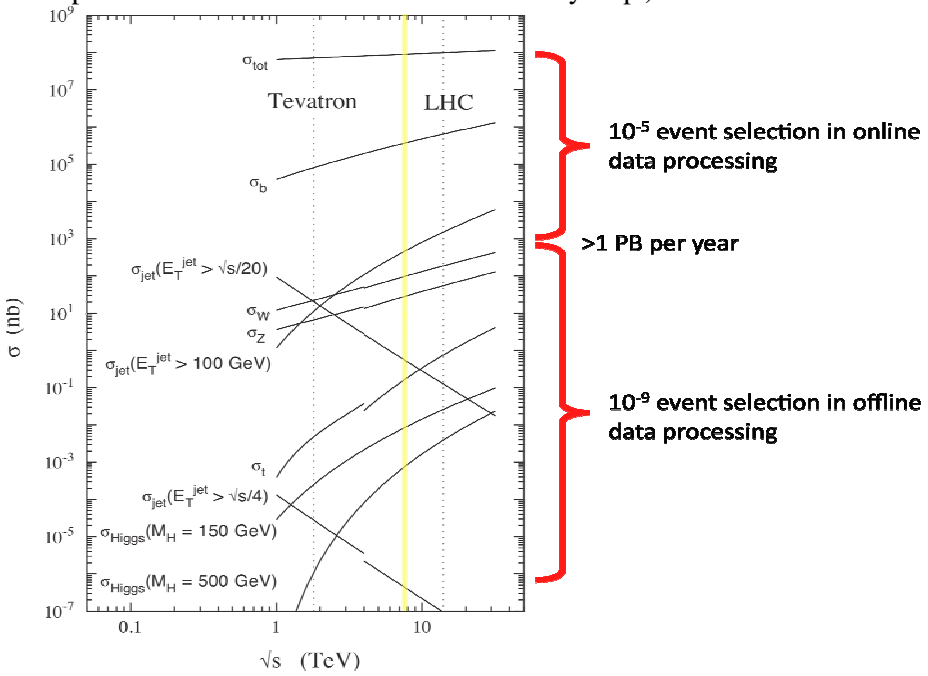


Figure 5: The offline event selection rate in LHC physics exploration [9] requires 6σ quality during Big Data processing at LHC energies shown by the yellow band

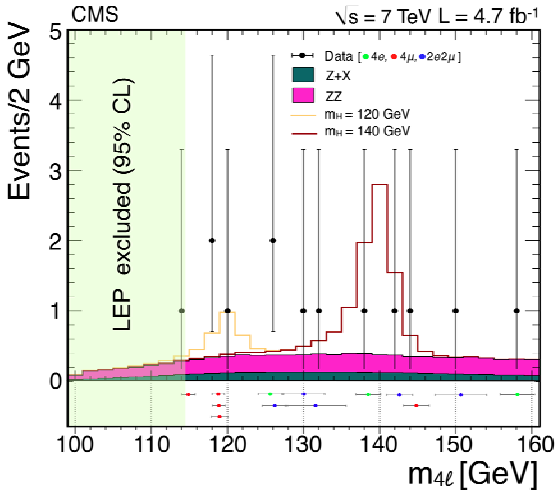


Figure 6: The event selection rate in the Higgs to four leptons channel [10]. Few events were selected out of a billion

Later, silent data corruption was detected in six events from the reprocessed 2010 data and in one case of five adjacent events from the 2011 reprocessed data [11]. Corresponding to event losses below the 10^{-8} level, this demonstrates sustained 6σ quality performance in Big Data processing.

5. Big Data Processing Techniques

In Big Data processing scientists deal with datasets, not individual files. Thus, a "task" – not a "job" – is a major unit Big Data processing on the Grid. Splitting of a large data processing task into jobs is similar to the splitting of a large file into smaller TCP/IP packets during the FTP data transfer. Splitting data into smaller pieces achieves reliability by re-sending of the dropped TCP/IP packets. Likewise, in Big Data processing transient job failures are recovered by re-tries. In file transfer, the TCP/IP packet is a unit of checkpointing. In high energy physics Big Data processing, the checkpointing unit is a job (e.g., PanDA [12]) or a file (e.g., DIRAC [13]).

Many high-performance computing problems are tightly-coupled and require inter-process communications to be parallelized. In contrast, high energy physics computing often called embarrassingly parallel, since the units of data processing – physics events – are independent. However, the event-level checkpointing rarely used today, as its granularity is too small for Big Data

processing in high energy physics. The next generation system needs the event-level checkpointing for Big Data.

6. Reliability Engineering for ATLAS Big Data Processing on the Grid

LHC computing experience has shown that Grid failures can occur for a variety of reasons. Grid heterogeneity makes failures hard to diagnose and repair quickly. Big Data processing on the Grid must tolerate a continuous stream of failures, errors and faults. The failure detection and performance prediction are considered open areas of research by many [14].

While fault-tolerance mechanisms improve the reliability of Big Data processing in the Grid, their benefits come at costs. Reliability Engineering provides a framework for fundamental understanding of the Big Data processing on the Grid, which is not a desirable enhancement but a necessary requirement.

6.1. Failure Recovery Cost

Job resubmission avoids data loss at the expense of CPU time used by the failed jobs. In 2010 reprocessing, the CPU time used to recover transient failures was 6% of the CPU time used for the reconstruction. In 2011 reprocessing, the CPU time used to recover transient failures was reduced to 4% of the CPU time used for the reconstruction. Figure 7 shows that most of the improvement came from reduction in failures in data transfers at the end of a job.

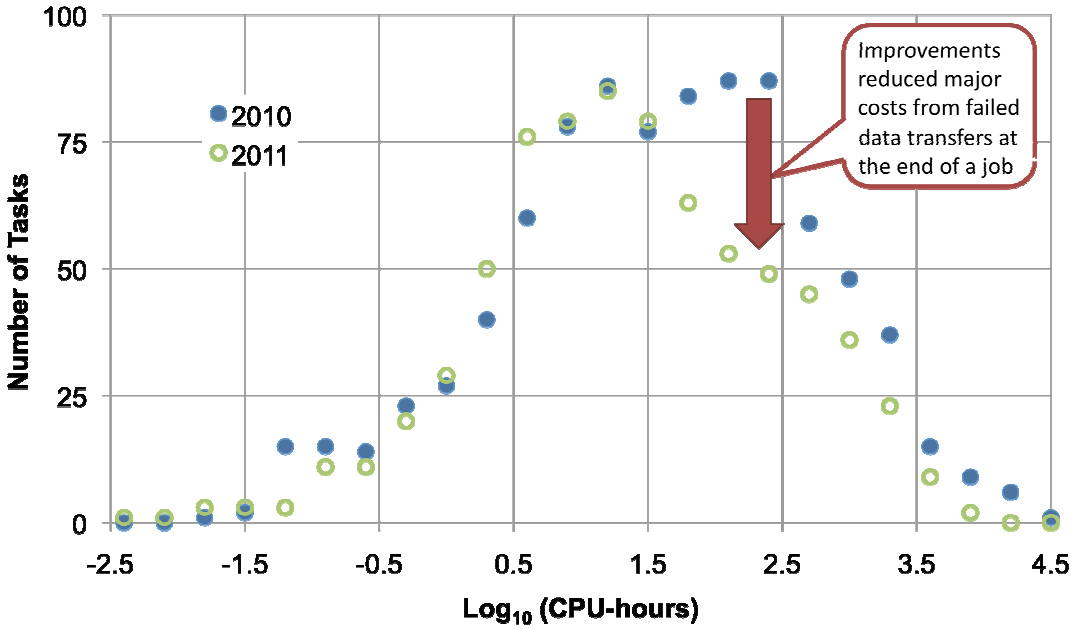


Figure 7: Distribution of tasks vs. CPU-hours used to recover job failures follows a multi-mode Weibull distribution

6.2. Time Overhead

Fault-tolerance achieved through automatic re-tries of the failed jobs induces a time overhead in the task completion, which is difficult to predict. Transient job failures and re-tries delay the reprocessing duration. Workflow optimization in ATLAS Big Data processing on the Grid and other improvements cut the delays and halved the duration of the petabyte-scale reprocessing on the Grid from almost two months in 2010 to less than four weeks in 2011 [11]. Optimization of fault-tolerance techniques to speed up the completion of thousands of interconnected tasks on the Grid is an active area of research in ATLAS.

7. Summary

The emerging Big Data revolution drives new discoveries in scientific fields including nanotechnology, astrophysics, high-energy physics, biology and medicine. In Big Data processing on the Grid, physicists deal with datasets, not individual files. A task (comprised of many jobs) became a

unit of Big Data processing. Reliability Engineering provides a framework for fundamental understanding of Big Data processing on the Grid, which is not a desirable enhancement but a necessary requirement. Fault-tolerance achieved through automatic re-tries of the failed jobs induces a time overhead in the task completion, which is difficult to predict. Reduction of the duration of Big Data processing tasks on the Grid is an active area of research in ATLAS.

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 supported in part by the U.S. Department of Energy, Division of High Energy Physics, under Contract DE-AC02-06CH11357.

References

[1] F. Gianotti for the ATLAS Collaboration: *Status of Standard Model Higgs searches in ATLAS.* Talk given at CERN Seminar, Geneva, Switzerland, July 4, 2012. https://indico.cern.ch/conferenceDisplay.py?confId=197461

[2] J. Incandela for the CMS Collaboration: *Status of the CMS SM Higgs Search*. Talk given at CERN Seminar, Geneva, Switzerland, July 4, 2012. https://indico.cern.ch/conferenceDisplay.py?confId=197461

[3] The brochure of the John von Neumann Institute for Computing. http://www2.fz-juelich.de/nic/Publikationen/Broschuere/elementarteilchenphysik-e.html

[4] A. D. Sakharov: *Violation of CP invariance, C asymmetry, and baryon asymmetry of the universe.* Journal of Experimental and Theoretical Physics, **5** (24) 1967.

[5] M. Sevior: *Belle II at SuperKEKB*. Talk presented at the 36th International Conference for High Energy Physics (ICHEP2012), Melbourne, Australia, July 4 – 11, 2012.

[6] D. Del Prete: *Computing at SuperB*. Talk presented at the 36th International Conference for High Energy Physics (ICHEP2012), Melbourne, Australia, July 4 – 11, 2012.

[7] Z. Dolezal: *BELLE-II and SuperKEKB Status.* Talk presented at the 4th SuperB Collaboration Meeting, La Biodola, Isola d'Elba, Italy, May 31 – Jun 5, 2012. http://agenda.infn.it/conferenceOtherViews.py?view=standard&confId=4880

[8] F. Bianchi: *Computing Closeout*. Talk presented at the XVII SuperB Workshop and Kick Off Meeting - La Biodola, Isola d'Elba, Italy, May 28 – June 2, 2011. http://agenda.infn.it/conferenceOtherViews.py?view=standard&confId=3352

[9] ATLAS Collaboration: *ATLAS high-level trigger, data-acquisition and controls: Technical Design Report.* LHCC Report CERN-LHCC-2003-022, Geneva: CERN, 2003, 369 p.

[10] CMS Collaboration: *Search for a Standard Model Higgs boson produced in the decay channel $H \rightarrow ZZ \rightarrow 4l$ in pp collisions at $\sqrt{s} = 7$ TeV*. Phys. Rev. Lett. **108** (2012) 111804.

[11] A. Vaniachine for the ATLAS Collaboration: *ATLAS detector data processing on the Grid.* Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC), 2011 IEEE, p. 104.

[12] K. De: *Status and evolution of ATLAS workload management system PanDA*. Invited talk presented at the V International Conference on "Distributed computing and Grid-technologies in science and education" (Grid2012), JINR, Dubna, Russia, July 16–21, 2012. http://grid2012.jinr.ru/docs/de_panda-grid2012.pdf

[13] A. Tsaregorodtsev: *DIRAC middleware for distributed computing systems*. Invited talk presented at the V International Conference on "Distributed computing and Grid-technologies in science and education" (Grid2012), JINR, Dubna, Russia, July 16–21, 2012. http://grid2012.jinr.ru/docs/Tsaregorodtsev_DIRAC_Dubna_Grid_2012.pptx

[14] J. Schopf, A. Chervenak, I. Foster et al.: *End-to-End Data Solutions for Distributed Petascale Science*. CTWatch Quarterly, 3 (4) 2007.

APPLICATION OF DATA GRID TECHNOLOGY FOR SHARING SCIENCE OUTREACH RESOURCES IN CHINA[1]

Zhang Zuli, He Hongbo, Xiao Yun
Computer Network Information Center,
Chinese Academy of Sciences, 100190, Beijing, China

Along with the rapid development of science outreach practices in China, a large number of digital science outreach resources have been accumulated in various research institutes and organizations. However, the efficiency in the use and sharing of these resources is still low, unable to meet the increasing demand from the general public. This paper describes the advantages of data grid platform applied to peer-to-peer network environment and proposes the data grid architecture of science outreach resources. "China Science Outreach Resource Grid" has been built based on the grid middleware, with its organizational structure and service system composed of backbone grid resource node, dynamic resource node, grid manage node and grid portal. This data grid architecture eventually integrates geographically distributed storage systems as a whole, and forms a manageable distributed resource sharing model. This paper argues that the implementation of "China Science Outreach Resource Grid" brings a simple and effective idea and a feasible way to build and share trans-regional and socialized science outreach resources.

Introduction

In recent years, science outreach has been rapidly developed. A lot of research and education institutions accumulated large amounts of digital science resources. However, the information barrier among various departments, results in the slow growth of effective resources, and also affects the development of quality resources. The quantity and quality are unable to meet the growing demand from the general public. On the other hand the public is hard to access those resources. The use efficiency of resource is not high, especially for local science organizations.

The data grid provides a great technical means at for sharing and applying science outreach resources. The data grid technology does not change the ownership, form or location of the physical data resources. This is a new idea for sharing digital science outreach resources. To this end, the China Association for Science and Technology has set up the task of building science resource sharing grid, trying to apply the data grid technology to the science field. This will allow the sharing of existing distributed digital science resources without changing the ownership, and make these resources dynamically access to the grid and be easily used by others.

In order to make the data grid platform tailored to the needs of science education and public outreach, we suggest that three problems should be solved:

1. Convenient access to the resource node. Resource nodes can conveniently access or exit the grid. This is not only particularly suitable for the majority of local science institutions, but also facilitates the promotion and application of the science resources of the grid.

2. Reasonable resource scheduling program. Resource nodes in the provision of services are likely to receive the download requests of a number of grid customers. On the other hand, users (grid customers) may also download resources directly from those who have already downloaded part of the same resource to gain a higher speed. In this case, the grid service needs to adopt a certain strategy, which can create a reasonable download request, the priority sequence, and able to switch to a better resource nodes of the network conditions to be downloaded.

3. Unified diversity data access. The grid system brings together a large number of distributions, heterogeneous data resources of multiple administrative domains. The display form and storage form

[1] Supported by the special fund of China Association for Science and Technology (CAST) & CNIC.

of the data resources are differ, stored in document form or in the database. So, the grid needs a unified access which can reduce the grid user access to data complexity and inconvenience.

Structure

The China Science Outreach Resource Grid (CSORG) is designed for cross-organizational and cross-platform resource sharing. The owners of these resources which are in the grid are equal participants. The resources they shared are stored in their servers instead of being submitted to a centralized portal platform, and the ownership won't be changed during resource sharing. The client can get and share grid resources with applications installed at the client side. The grid can provide faster network access, greater storage capacity and more convenient access to resources.

Based on features discussed above, the overall structure of the grid platform is designed as shown in figure 1.

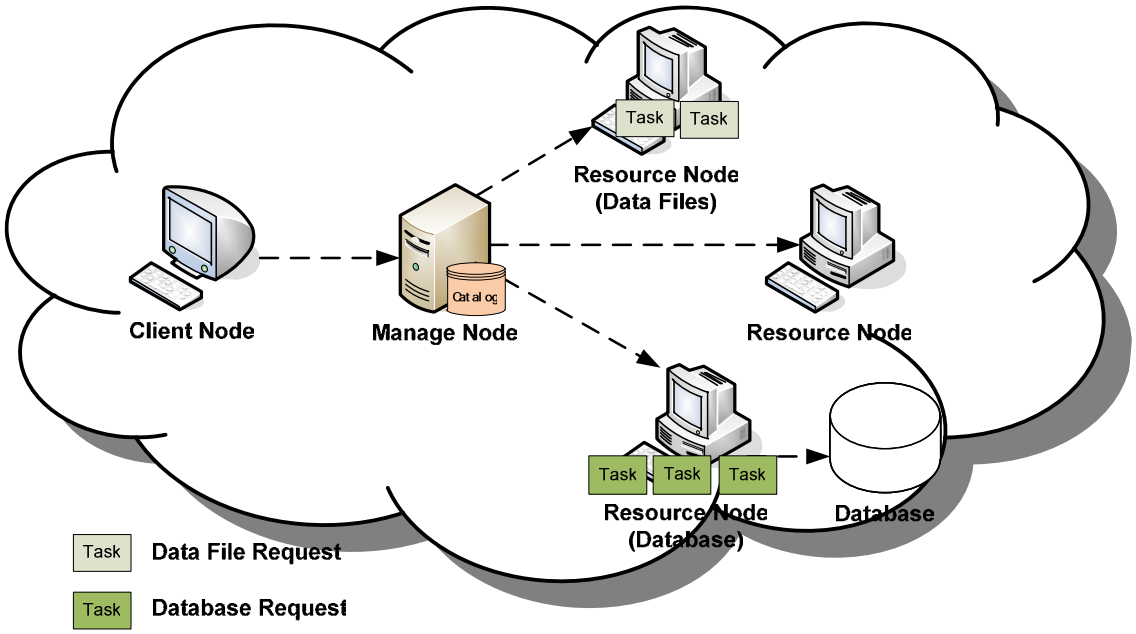


Fig.1: Overall structure of China Science Outreach Resource Grid(CSORG)

The overall structure of the grid platform is composed of three parts including the client, the manage node(server node) and resource nodes:

Client is the main resource user. Resources downloaded by the client will become seeds for other clients to download. The client is not the true owner of the resources, unable to get rewards for sharing resources from the grid platform. Resource sharing activities of the client must be authorized by the Manage node.

Manage Node, the bridge for communication between the resource nodes and the client, is a centralized manager of the information about the resources and the node hosts. The manage node contains the data catalog with the location index of all shared resources. It provides resources index, resource ownership management, node maintenance and resource requests scheduling.

Resource node is the main owner of the resource file. The original resource is published on the hosts which have real ownership of the resources and get rewards from the grid platform when the resources are used. These nodes communicate with the manage node and handle the distribution over the request. The transmission of resources is initiated by the resource node, which can reduce the safety validation.

The manage node, the resource node and the grid client connect with each other and transfer data through the grid middleware. Figure 2 shows the deployment structure of the component units and the main interactions in the grid platform:

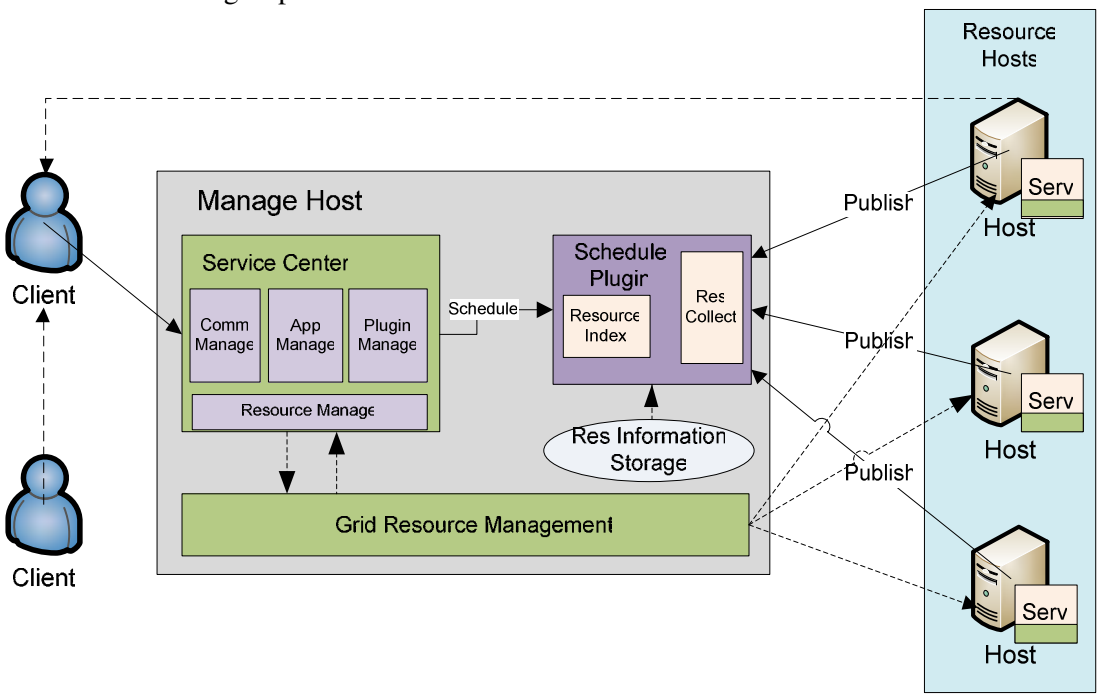


Fig.2: The deployment structure and the main interactions

Grid middleware is mainly composed of the scheduling service center and the grid resource management. The scheduling service center is only deployed on the manage node. There is a direct interaction between the service center and the client program to handle customers' resource requests, and to coordinate and control individual requests and the resource scheduling process. The grid resource management is at the bottom of the grid middleware service, which provides the necessary information about the resources for the upper scheduling service center. The grid resource management is deployed in each grid node to manage the service program running on each node, and to control and monitor their status.

The scheduling service is a plug-in management service developed for scheduling service center, which provides the necessary weights for the resource request scheduling, collects the information about the resources published by the resource nodes, stores and organizes the resource data (maintains the ownership information of resources), and provides resource-related services (such as search services). When the resource scheduling service receives a resource acquisition request, it retrieves and matches all of the resource positions in the grid, selects the appropriate grid location according to the scheduling policy, and then notifies the resource requesting party. The requesting party downloads or accesses the resource with peer-to-peer connection.

Main process

The first step for the client and the resource nodes is to sign in to the management node. The platform will verify their information, and the resource nodes will automatically register the resources to be shared in the data catalog. Details are shown in figure 3 and figure 4.

After signing in to the management node, the client can send resource request tasks, and the system will process the tasks in four steps:

1. Search the catalog to get a list of resource nodes containing the resources required.
2. Sort the list according to the load of nodes and the distance between the client and node.
3. Use GRM module to verify the sequence and return the verified sequence to the client.

4. The client will establish P2P connection to get resources according to the sequence. Details of the submit process are shown in figure 5.

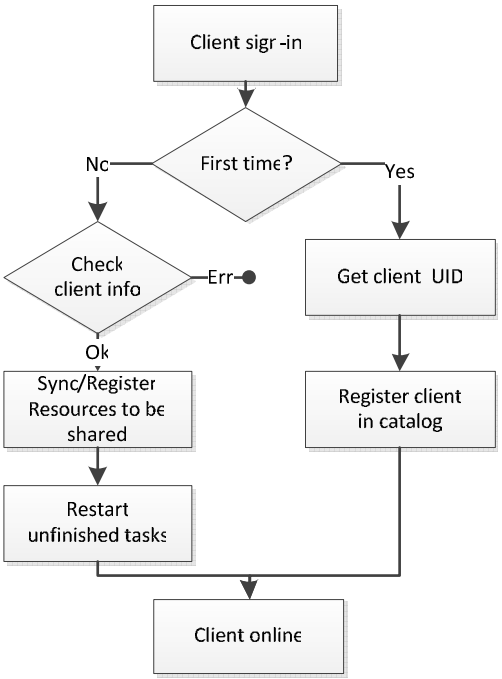


Fig.3: Sign in process of the client

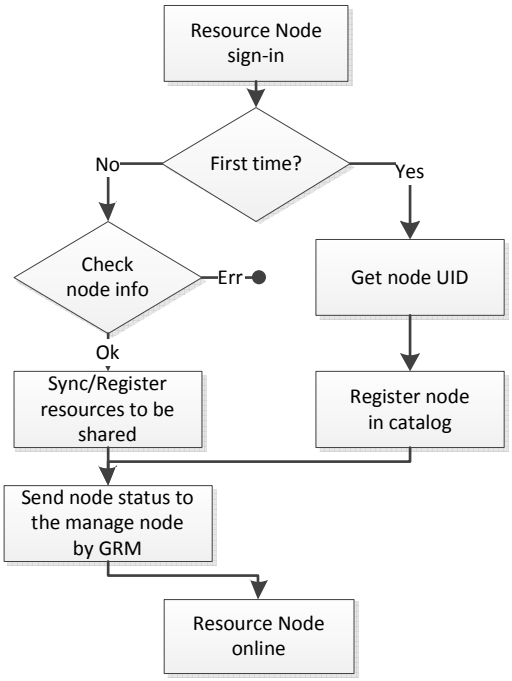


Fig.4: Sign in process of resource node

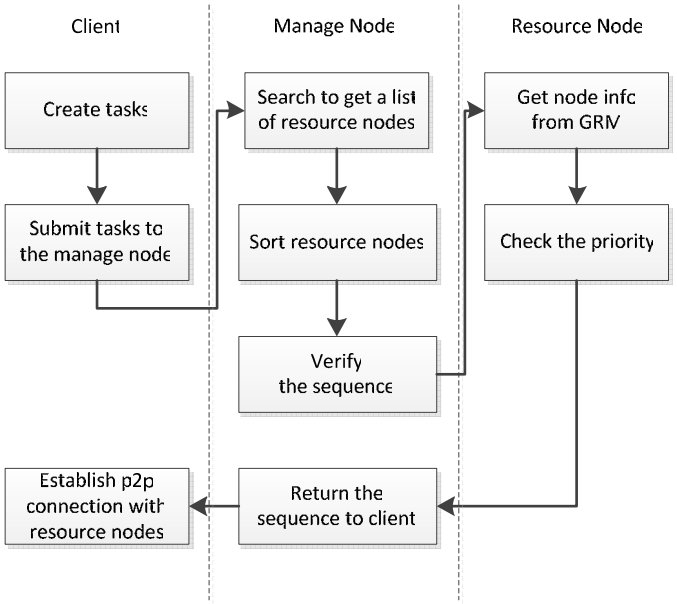


Fig.5: Submit process of tasks

Summary

"China Science Outreach Resource Grid" consists of one server grid node and dozens of resource gird nodes. This platform provides open access to more than one hundred thousand resources including pictures, videos, lectures, books, animations, etc.

More importantly, resources from the society including those contributed by individuals can be shared as dynamic resource nodes, which create a positive social atmosphere for sharing and using science outreach resources. Up to now, the grid has collected over 100,000 science resources.

This paper studies the structure and main process of "China Science Outreach Resource Grid", and provides a feasible solution and technical architecture for the future science resource sharing and related work. To a certain extent, the resource sharing platform design has eliminated the barriers of data sharing, and opened up the resource exchange channel.

References

[1] ZHAO Shan. Integration Methodology for China Digital Science and Technology Museum. Science and Technology Review. 2008.

[2] ZHANG Xiguang DENG Dali. Survey on progress in scientific data grid. Application Research of Computers.2009-10.

[3] Wang Xiaoning Xiao Haili. Software integration and application infrastructure in China National Grid. J.Huazhong Univ. of Sci. & Tech. (Natural Science Edition).2010.

[4] Zou Y, Zha L, Wang X. A layered virtual organization architecture for grid. The Journal of Supercomputing, 2010.

[5] Ma Yongzheng Sun Peng Nan Kai. A CNGrid-based collaboration platform and its multi-discipline grid applications. J.Huazhong Univ. of Sci. & Tech. (Natural Science Edition).2010.

[6] China Science Outreach Resource Grid. http://grid.kepu.cn

СОЗДАНИЕ В ОИЯИ АВТОМАТИЗИРОВАННОЙ СИСТЕМЫ ОБРАБОТКИ ДАННЫХ УРОВНЯ TIER-1 ЭКСПЕРИМЕНТА CMS НА LHC[1]

Н.С. Астахов, С.Д. Белов, А.Г. Долбилов. В.Е. Жильцов, В.В. Кореньков, В.В. Мицын, Т.А. Стриж, Е.А. Тихоненко, В.В. Трофимов, С.В. Шматов
Объединенный институт ядерных исследований, Дубна

Введение

"Компактный мюонный соленоид" (Compact Muon Solenoid - CMS) [1] является одним из двух многоцелевых экспериментов, созданных для работы на пучках ускорительного комплекса Большого адронного коллайдера (Large Hadron Collider - LHC) [2] в Европейском центре по ядерным исследованиям (ЦЕРН) [3]. Программа физических исследований CMS охватывает большой спектр задач физики элементарных частиц и направлена на проверку стандартной модели на новом масштабе энергий (до нескольких ТэВ), поиск бозона Хиггса, суперсимметрии и других сигналов новой физики за пределами стандартной модели, а также проведения комплекса экспериментальных исследований в области физики тяжелых ионов [4]. Установка CMS представляет собой многоцелевой экспериментальный комплекс, способный регистрировать широкий спектр возможных экспериментальных сигналов в диапазоне энергий и масс рождаемых частиц вплоть до нескольких ТэВ [5].

Большой адронный коллайдер обеспечивает столкновения пучков протонов с частотой до 40 МГц, что приводит к потоку данных из точки взаимодействия более 40 ТБ в секунду. Система последовательного отбора событий в режиме реального времени (trigger) уменьшает поток записываемых в ходе эксперимента данных до 100 Гц. Таким образом общий объем экспериментальных данных (часто называемых "сырыми'), подлежащих хранению и последующей детальной обработке и анализу, достигает более 1 ПБ в год.

Обработка и хранение данных в CMS возлагается на распределенные вычислительные центры, объединенные в многоуровневую структуру с помощью технологий всемирный вычислительный грид-среды для LHC (WLCG – Worldwide LHC Computing Grid) [6].

В марте 2011 г. предложение о создании центра уровня Tier-1 для четырех (ALICE, ATLAS, CMS и LHCb) экспериментов LHC выдвинуло Министерство науки и образования РФ, которое было поддержано дирекцией ЦЕРН. Для реализации этого проекта в том же году была принята целевая федеральная программа "Создание автоматизированной системы обработки данных экспериментов на Большом адронном коллайдере (БАК) уровня Tier-1". Проект предусматривает создание в России центра уровня Tier-1 с распределенной ответственностью - центр для поддержки экспериментов ALICE, ATLAS, и LHCb организуется на базе вычислительного комплекса Национального исследовательского центра "Курчатовский институт" (Москва), а для поддержки эксперимента CMS - в Лаборатории информационных технологий ОИЯИ.

Автоматизированная система обработки и хранения данных (АСОД) CMS в Объединенном институте ядерных исследований (ОИЯИ) предназначена для работы в составе глобальной грид-системы БАК WLCG и нацелена на проведение полного цикла обработки физической информации, получаемой в ходе проведения эксперимента, обеспечения работ по моделированию физических процессов, защищенного хранения и приема/передачи данных в другие центры WLCG.

[1] Работа выполнена в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы» (Гос.контракт №07.524.12.4008).

CMS использует распределенную модель компьютинга для данных всех типов – “сырых”, реконструированных, Монте-Карло (моделирование). Это влечет за собой ответственность за безопасность и обслуживание данных региональных центров.

В настоящее время модель компьютинга CMS включает в себя центр уровня Tier-0 в ЦЕРН, семь центров уровня Tier-1, расположенных в крупнейших мировых копьютерных центрах и связанных высокоскоростной сетью, и множество центров уровня Tier-2 и Tier-3, созданных в различных научных организациях (рис. 1). Кроме того, в модель компьютинга входят центры CMS по набору, контролю и быстрой обработки данных, размещенных как в непосредственно в ЦЕРН, так и в некоторых удаленных региональных центрах (например, в ОИЯИ и МГУ).

Вычислительная система CMS включает семь региональных центров уровня Tier-1:

- T1_DE_KIT в Технологическом институте Карлсруэ (KIT), Карслруэ, Германия
- T1_ES_PIC в Центре научной информации университета Барселоны (PIC), Барселона, Испания
- T1_FR_CCIN2P3 в Национальном институте ядерной физики и физики частиц (IN2P3), Лион, Франция
- T1_IT_CNAF в Национальном центре по исследованию и развитию информационных технологий и телематике (INFN), Болонья, Италия
- T1_TW_ASGC в Академии грид-технологий (ASGC), Тайбей, Тайвань
- T1_UK_RAL в Лаборатории Резерфорда — Эплтона (RAL), Дидкот, Великобритания
- T1_US_FNAL в Национальной ускорительной лаборатории им.Э. Ферми (FNAL), Батавия, США

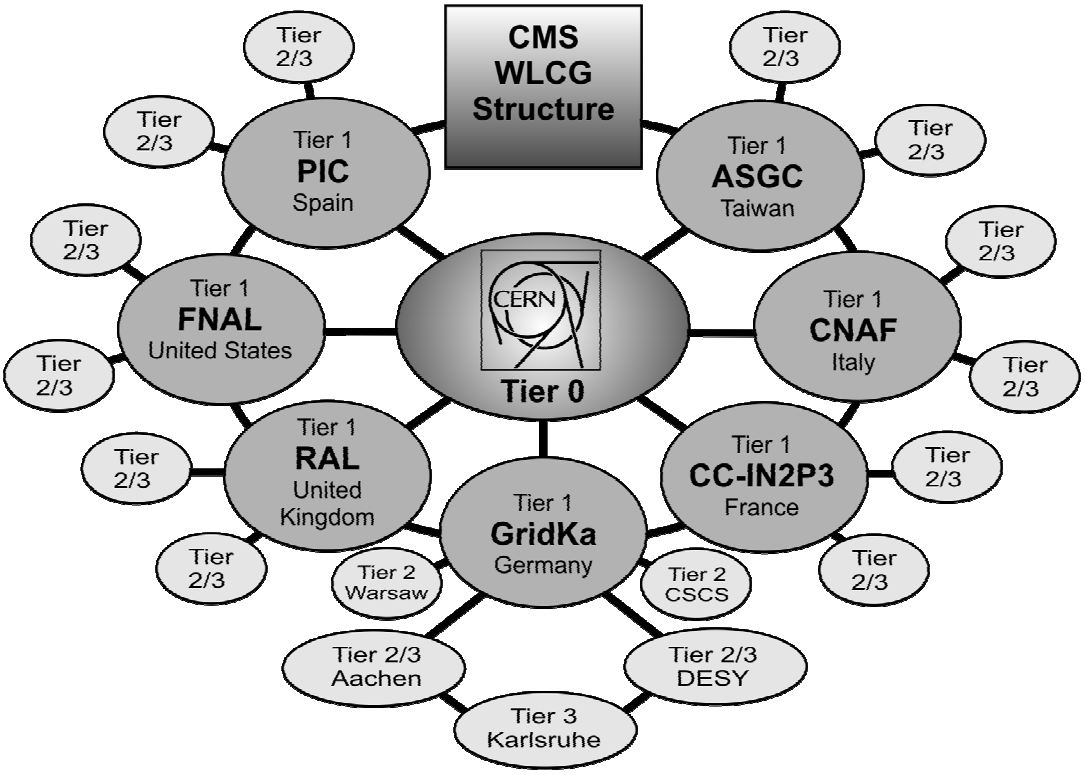


Рис.1 Структура компьютерных центров CMS

Иерархия центров и задачи центров каждого уровня определены в Меморандуме WLCG (WLCG Memorandum of Understanding – WLCG MoU) [7].

Вычислительная модель RDMS CMS

Создание автоматизированной системы обработки данных эксперимента CMS на LHC уровня Tier-1 для эксперимента CMS на базе ОИЯИ диктуется активной позицией ОИЯИ, ряда ведущих научных центров России, бывших советских республик и некоторых других стран-участниц ОИЯИ в этом эксперименте. С самого старта проекта ученые из институтов России и стран-участниц ОИЯИ выступают в качестве единой группы, пользующейся высоким авторитетом в коллаборации CMS. Это содружество групп ученых и специалистов из институтов России и других стран-участниц ОИЯИ получило широко известное название RDMS – аббревиатура от Russia and Dubna Member States (Россия и страны - участницы Дубны). Сегодня RDMS – неотъемлемая и важная составная часть коллаборации CMS. Организация сотрудничества RDMS, объединяющего усилия многих институтов и научных школ, позволила физикам России и стран-участниц ОИЯИ нести полную ответственность за ряд детекторов установки CMS. Кроме этого физики RDMS активно участвуют в разработке программы физических исследований, реконструкции и отборе событий и создании базового математического обеспечения и компьютинга.

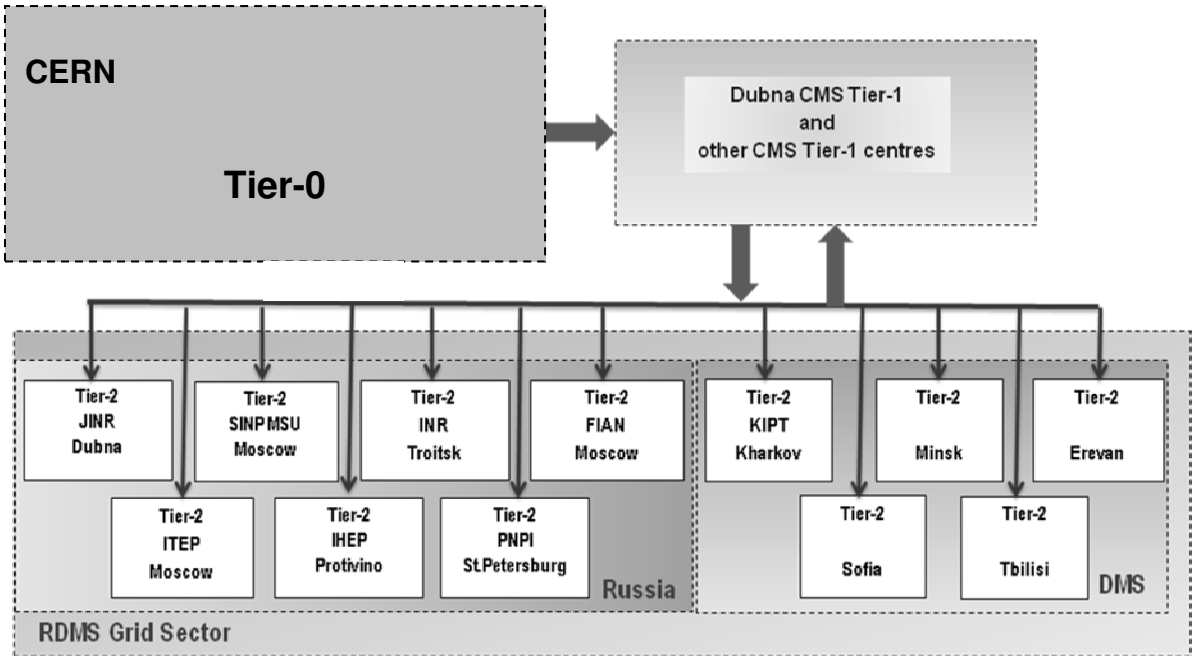


Рис.2: Структура компьютерных центров RDMS CMS

В ОИЯИ успешно создана принятая экспериментами LHC грид-инфраструктура как часть распределенного кластера RuTier2 (Russian Tier-2) (рис.2) [8-9]. Этому предшествовал длительный период тестирования соответствующего программного обеспечения (ПО) в ОИЯИ и российских институтах, являющихся членами консорциума RDIG (Russian Data Intensive Grid).

В течение 2011-2012гг. для эксперимента CMS на грид-сайте ОИЯИ выполнено 35,8% от общего числа задач консорциумом RDIG (55,2% от общего процессорного времени консорциума, затраченного на CMS).

Роль центров CMS уровня Tier-1

В соответствии с вычислительной моделью CMS [10] уровень Tier-1 предназначен для долговременного хранения данных и преобразования «сырых» данных, поступающих с детекторов экспериментальной установки, и подготовки их для их последующего анализа на уровне Tier-2. Физический анализ данных на уровне Tier-1 не предусмотрен.

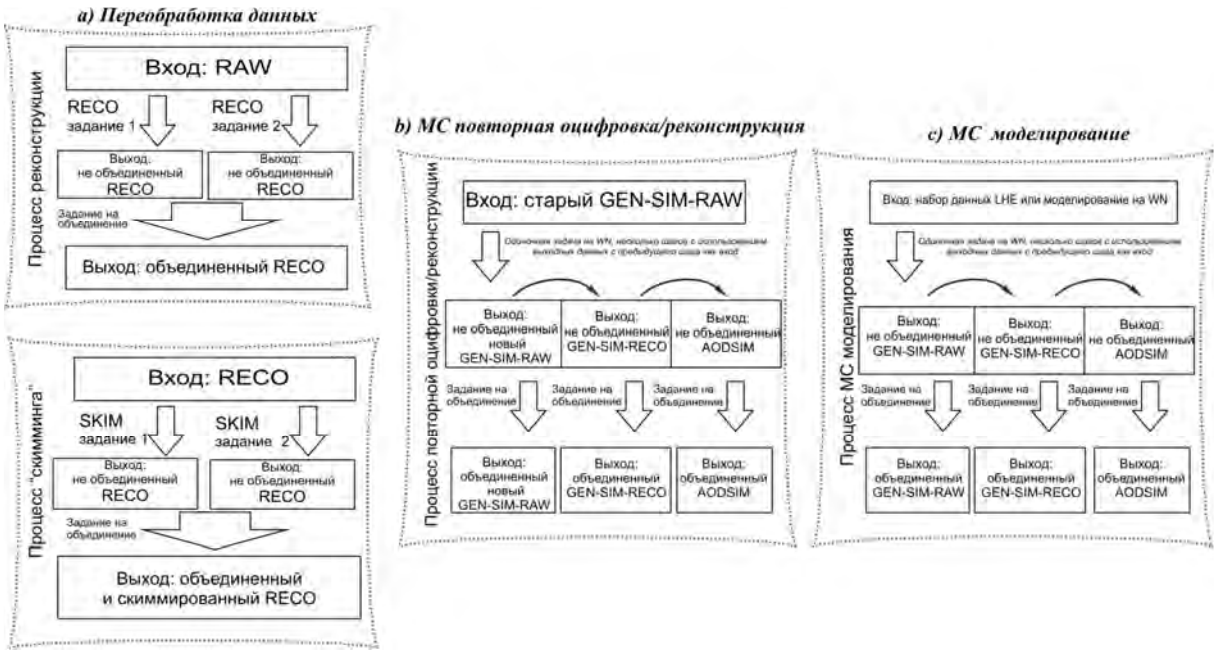


Рис.3: Схема обработки и создания данных CMS в процессе: а) реконструкции данных, б) переоцифровки и переобработки данных моделирования в) моделирования данных

В вычислительной модели CMS на уровне Tier-1 выделяются два основных типа потоковой обработки заданий: повторную реконструкцию данных (рис. 3 а) и повторную оцифровку/реконструкцию данных моделирования (рис. 3 б). В процессе повторной реконструкции данных, сайты Tier-1 обрабатывают сырые RAW данные с помощью более нового программного обеспечения и/или с учетом обновленных констант калибровки и констант пространственного выравнивания детекторных систем, более точной информации о состоянии установки во время набора данных. Выходные данные записываются как в формате RECO (RECOnstructed data), так и формате AOD (Analysis Object Data). RECO – данные, содержащие значения параметров физических объектов (треков, вершин взаимодействия, струй, электронов, мюонов, фотонов и т.д.), а также кластеров и хитов, реконструированных с помощью различных алгоритмов из RAW данных. Они являются выходным потоком данных из Tier-0 для перераспределения на различные Tier-1. Объем одного события составляет 0.4 МБ. Эти данные могут быть использованы для анализа, но неудобны из-за своего большого размера. AOD представляют собой выборочный набор информации из RECO данных и включает только значения параметров физических объектов (треков, кластеров, вершин взаимодействия, струй, электронов, мюонов, фотонов и т.д.). AOD имеет значительно меньшие, по сравнению с RECO размеры (0.12 МБ на одно событие) и используется для восстановления окончательной топологии физического события и окончательного физического анализа.

Также на Tier-1 выполняется отбор по заданным критериям событий из реконструированных данных (skimming). Подобный отбор может осуществляться из “сырых” RAW данных или уже из реконструированных RECO данных. Эти события также проходят через процесс обработки и объединения, как и во время повторной реконструкции, после чего записываются в файлы формата RECO или в объединенном формате (RAW-RECO) (рис. 3 а).

В центрах уровня Tier-1 так же, как и в случае экспериментальных данных происходит переобработка данных моделирования с помощью более новых версий программного обеспечения и/или с учетом обновленных констант калибровки и констант пространственного выравнивания детекторных систем. Входные данные типа GEN-SIM-RAW подвергаются повторной оцифровке (часть данных GEN-SIM) для получения обновленных версий

смоделированных данных типа RAW, которые в дальнейшем проходят повторную реконструкцию (на выходе данные типа GEN-SIM-RECO и/или AODSIM).

Таким образом, основные функции центра Tier-1 включают:

- прием экспериментальных данных из центра уровня Tier-0 в объеме определенном соглашением по WLCG (WLCG MOU);
- архивирование и защищенное хранение части экспериментальных RAW данных;
- выполнение последовательной непрерывной обработки данных;
- отбор событий и формирование набора данных по заданным критериям (skimming);
- повторная обработка данных (reprocessing) с помощью нового ПО и калибровочных констант и констант пространственного выравнивания частей установки CMS, другие задачи;
- создание AOD данных;
- распределение (передача) наборов данных (datasets) RECO и AOD на другие центры Tier-1/ Tier-2/ Tier-3 для дублированного хранения (replication) и физического анализа;
- повторное моделирование с помощью нового ПО и калибровочных констант и констант пространственного выравнивания частей установки CMS, защищенное хранение смоделированных событий.

Во время работы LHC набранные экспериментом RAW данные передаются на все сайты Tier-1 постоянно, а во время перерывов в работе ускорительного комплекса идет передача данных, набранных на космических лучах. Каждый Tier-1 центр берет на себя ответственность за сохранность ассоциированного с ним основного набора данных. Конкретный Tier-1 центр может иметь только доступную копия набора данных, что должно позволить любому пользователю CMS получать доступ к ним.

В соответствии с обязательствами перед пользователями, связанными с финансирующими структурами, которые поддерживают центры Tier-1, центры могут предоставлять свои мощности для региональных групп (сообществ). Эти функции, однако, не должны нарушать способность сайта выполнить обязательства перед всей коллаборацией.

Сервисы центров CMS уровня Tier-1

Центры уровня Tier-1 должны обеспечивать широкофункциональные высокоскоростные и высоконадежные сервисы модели компьютинга коллаборации CMS на основе грид-интерфейсов WLCG и высокоуровневых сервисов CMS. При этом ожидается высокий уровень функциональности и поддержки.

Сервисы центров CMS уровня Tier-1 подразделяются на системные (недоступные для пользователей) и сервисы, доступные для пользователей (сервисы пользователей). К системным сервисам относятся:

- сервисы системы массового хранения данных,
- сервисы системы безопасности сайта,
- сервисы системы регистрации и сервисных приоритетов,
- сервисы системы баз данных.

Сервисы пользователей зависят от вычислительных ресурсов и сервисов системного уровня: массовые системы хранения данных; обеспечения безопасности сайта; приоритетность и эккаунтинг. Они включают:

- сервисы архивации данных,
- сервисы дисковой системы хранения данных,
- сервисы доступа к данным,
- сервисы системы реконструкции данных,
- сервисы системы анализа данных,
- другие пользовательские сервисы.

Инфраструктура АСОД БАК CMS состоит из следующих компонентов (сервисов):

Подсистема хранения данных (*Local Worker Nodes*), предназначенная для хранения экспериментальных данных, поступающих от экспериментов БАК из центра уровня Tier-0 (ЦЕРН), смоделированных данных, поступающих из центров уровня Tier-2, а также выходных данных задач по обработке данных, выполняемых на АСОД БАК CMS. Согласно модели данных CMS объем файлов с данными не может превышать 10 ГБ, поэтому к системе хранения данных предъявляется требование по обеспечению не менее 10 ГБ дискового пространства на вычислительный процесс. Также система должна обеспечивать оперативную память RAM не менее 2 ГБ на процесс.

Подсистема вычислительных узлов *(Computing Elements)*, предназначенная для обработки данных, поступающих от экспериментов БАК. Данная подсистема представляет собой набор сервисов, входящих в программное обеспечение промежуточного уровня ГРИД (gLite), для обеспечения доступа грид-задач к локальным ресурсам центра Tier-1 (LRMS, система массового счета batch). Обычно система предоставляет доступ к набору очередей (queues) задач вычислительных узлов. Число вычислительных узлов должно соответствовать ожидаемой загруженности системы. Задачи обычных пользователей не допускаются к счету на ресурсах центров уровня Tier-1, доступ к набору очередей ограничен с помощью подсистемы распределения ролей (t1 access roles).

Подсистема передачи данных (*FTS*) между центрами различных уровней. Данный сервис также реализуется с помощью инструментов gLite. Передача данных осуществляется посредством создания виртуальных каналов с возможностью указания последовательностей и приоритетов передачи. Согласно вычислительной модели CMS вся передача данных между различными центрами уровня Tier-1 и между центрами уровня Tier-1 и ассоциированными центрами уровня Tier-2 осуществляются с помощью сервиса FTS.

Управление передачей и хранением данных (*CMS VOBOX*) на уровне наборов и потоков данных CMS (datasets) осуществляется посредством инструментов проекта PhEDEx, которые включают в себя:

- База данных управления передачами (Transfer management database - TMDB).
- Передающие агенты, управляющие передачей файлов между сайтами, миграцией данных на локальных хранилищах, проверкой контрольных сумм переданных данных.
- Управляющие агенты, обеспечивающие размещение файлов, в соответствии с подписками сайта на данные.
- Локальные агенты, обеспечивающие обработку файлов после их поступления на сайт или перед отправкой файлов с сайта: слияние файлов, регистрация файлов в каталогах, помещение информации о файлах в базу данных управления файлами.
- Мониторинг передачи и отображение результатов с помощью веб-интерфейса.

Сервис (сервер) кэширования запросов (*Frontier Local Squid Cache Server*) к базе данных калибровочных констант. Задачи, запускаемые на рабочих узлах вычислительного кластера обращаются к центральной базе констант калибровок. Для снижения нагрузки на серверы этой базы, ускорения получения данных и снижения сетевого трафика, доступ к ним осуществляется через промежуточные узлы, построенные на основе кэширующего прокси-сервера squid. Локальная установка CMSSW конфигурируется с указанием списка прокси-серверов, обслуживающих данный сайт (являющихся либо частью самого сайта, либо находящихся на ближайших сайтах). Для надежности каждый центр уровня Tier-1 должен поддерживать по крайней мере 2 два подобных сервера, каждый из которых обеспечивает одновременный запуск порядка 800-1000 задач (jobs slots).

Системное и прикладное программное обеспечение (*CMS software server*) для обеспечение вычислительного окружения для счётных задач. Задачи, приходящие на

вычислительные элементы системы содержат только конфигурационные файлы, все вычислительные модули предоставляются сайтом. Соответственно, одна из функций Tier-1 — обеспечить поддержку набор версий (релизов) специализированного ПО CMS (CMSSW) и его зависимостей, обновлять этот набор и обеспечивать прозрачный доступ к нему со стороны вычислительных элементов системы и системы хранения данных. Размер репозитория ПО CMSSW центра уровня Tier-1 должен быть не менее 200 ГБ.

Система распределения нагрузки и интерфейс между гридом и локальными счётными очередями, обеспечивающие обмен информацией и командами между различными устройствами и подсистемами разрабатываемой Системы, а также между системой и смежными системами, а также центрами уровня Tier-2 WLCG в России и мире.

Сетевая инфраструктура CMS Tier-1 в ОИЯИ включает в себя подсистему LHC OPN (рис.4), предназначенную для организации выделенных линий передачи данных, связывающей центры Tier-1 и Tier-0. Пропускная способность LHCOPN между Tier-0-Tier-1 и между Tier-0-Tier-1 составит 2 Гбит/с в конце 2012 и будет увеличена до 10 Гбит/с в 2014 (см. таблицу 2). ОИЯИ также подключен к академическим сетям с пропускной способностью 2x10 Гбит/с, обеспечивающие соединение Tier-1 ОИЯИ с центрами уровня Tier-2/Tier-3.

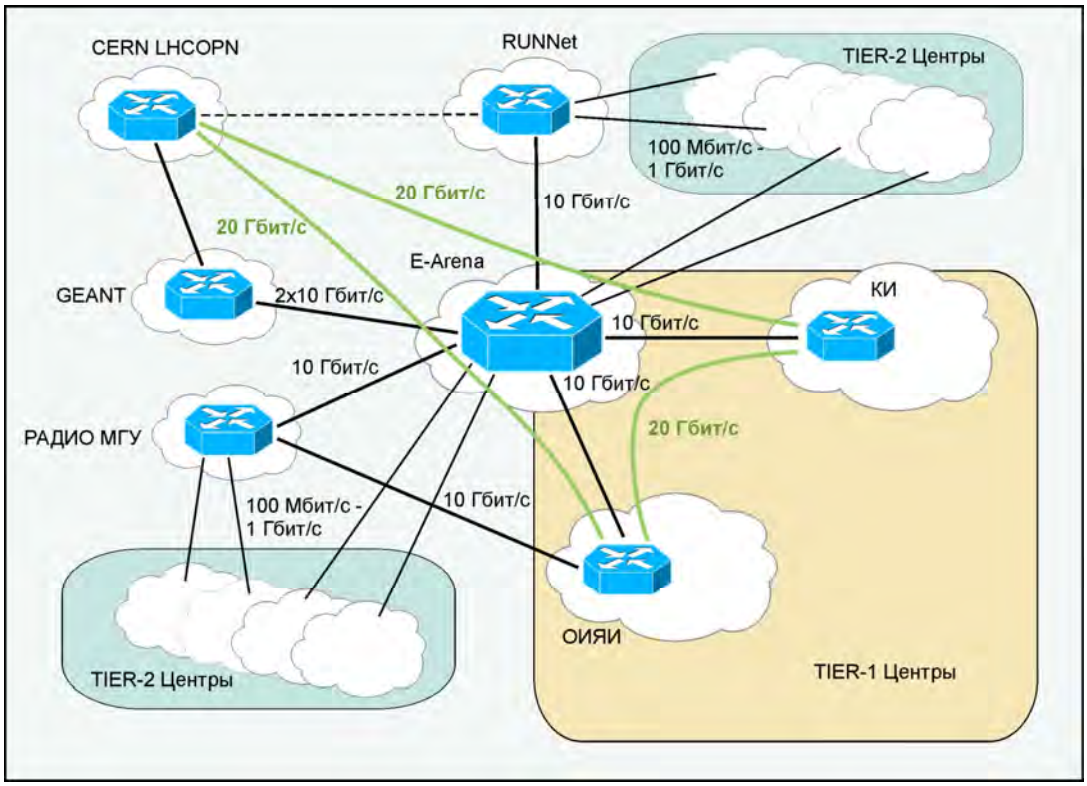


Рис.4: Подключение ОИЯИ к сетевой инфраструктуре WLCG

Таблица 2. Ожидаемый рост полосы пропускания сетевой инфраструктуры ОИЯИ

Год:	2012	2013	2014
LHCOPN, Гбит/с	2	10	10
WAN, Гбит/с (Российская академическая сеть и GEANT2)	2x10	2x10	2x10

Ввод в эксплуатацию и сертификация центра

Для оценки готовности центра к выполнению возложенных на него задач в компьютерной модели эксперимента CMS предусмотрена специальная программа ввода центра в эксплуатацию и контроля его работоспособности, включающая последовательную цепочку тестов различных узлов и сервисов центра. Результаты этих тестов могут быть использованы, во-первых, для контроля неполадок и их устранения персоналом центра, и во-вторых, экспериментом CMS для автоматической сертификации центра до состояния его готовности для работы в составе компьютерной инфраструктуре эксперимента. Для этого установлена периодичность проведения этих тестов, что наряду с требованиями сервисов WLCG, позволяют обеспечить стабильную работоспособность всей системы.

Для оценки готовности центров используются следующие параметры:

- время за день, в течении которого успешно отработали все соответствующие тесты;
- процент успешно запущенных задач анализа данных в день;
- количество соединений с другими сайтами для передачи запрашиваемых данных.

Результаты тестов суммируются в итоговую таблицу статуса центров - Site StatusBoard (SSB) [11] (рис.5). Предусмотрена возможность соотнести результаты работы задач анализа данных с информацией о текущих проблемах в работе сайтов, предоставляемых системой мониторинга, с тем чтобы обеспечить экспертов указанием на конкретное место в полном наборе данных мониторинга CMS. Эта информация может часто меняться в связи с постоянно улучшающимся пониманием того, что именно наилучшим образом диагностирует проблемы. SSB позволяет гибко отображать динамическую структуру, в которой информация хранится в виде колонок базы данных, используя имя сайта в качестве ключа. Колонки заполняются процессом, использующим информацию из внутренней базы данных CMS Dashboard [12], из информационной системы WLCG и из текстовых файлов в Web. Колонки могут формироваться и добавляться с помощью Web-интерфейса, что не требует специальных разработок. Изменение во времени содержимого колонок можно отобразить графически или получить в формате XML. Информация нескольких колонок может быть отображена в виде одного "отображения", как показано на Рис. 5.

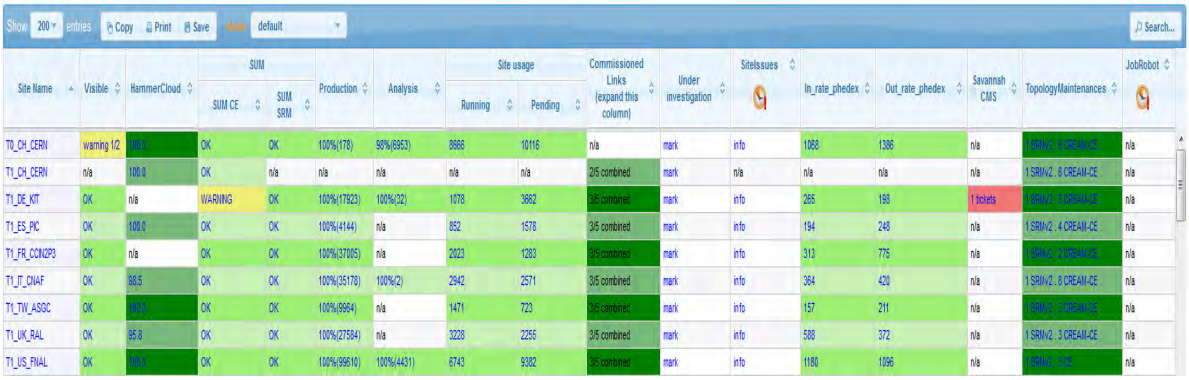


Рис. 5: Отображение статуса сайтов: каждая строка соответствует сайту, в колонках показана готовность каждого сайта

Установка и тесты дисковых и вычислительных серверов

Планируется установить:

- 1200 CPU - ноябрь 2012
- 660 ТБ дискового пространства - ноябрь 2012
- 1600 CPU - ноябрь 2013
- 3168 ТБ искового пространства - ноябрь 2013
- Добавочно 1600 CPU - октябрь 2014
- Добавочно 1056 ТБ дискового пространства - октябрь 2014

Установка и тест ленточных накопителей

Планируется установить:

- Ленточная система общим объемом 72 ТБ – ноябрь 2012
- Ленточная система общим объемом 5720 ТБ - ноябрь 2013
- Добавочно 1600 ТБ ленточного и дискового пространства - октябрь 2014

Сетевое подключение

Ниже приведен план интеграции создаваемого Tier-1 в LHC OPN:

Этап	Дата
Интеграция в LHC OPN 2 Гбит/с	Декабрь 2012
Функциональные тесты OPN 2 Гбит/с	Февраль 2013
Интеграция в LHC OPN 10 Гбит/с	Июль 2013
Функциональные тесты OPN 10 Гбит/с	Август 2013

Тесты передачи данных

Каждый компьютерный центр CMS должен поддерживать достаточное количество сертифицированных линий передачи данных. В соответствии с требованиями CMS каждый центр уровня Tier-1 должен быть связан со всеми другими центрами того же уровня, с центром уровня Tier-0 и с большинством центров уровня Tier-2 (в настоящее время доля сертифицированных линий связи Tier-1-Tier-2 достигает 80% от максимально возможного колличества). Специальная процедура проведения тестов передачи данных CMS (Debugging Data Transfers) определяет следующие критерии для сертификации линий связи центров Tier-1:

- скорость передачи данных в/на центры уровня Tier-0/Tier-1 должна быть не менее 20 МБ/с круглосуточно,
- скорость передачи данных в/на центры уровня Tier-2 должна быть не менее 5 МБ/с круглосуточно.

Тесты передачи данных из ЦЕРНа в ОИЯИ будут проведены в 2013 году - по OPN каналам 2 и 10 Гбит/с. Тесты должны будут показать способность принимать и хранить сырые данные CMS в объемах, соответствующих объявленным возможностям сайта Tier-1 ОИЯИ для каждого тестового периода.

Тесты WLCG- и VO-сервисов

С самого начала, в соответствии с требованиями WLCG и экспериментов на LHC, создаваемый Tier-1 должен обеспечить определенный набор сервисов для всех экспериментов. В частности, Tier-1 ОИЯИ должен обеспечить:

- систему авторизации и безопасности WLCG (GSI, Argus, gLExec),
- вычислительные элементы - Computing Element (CREAM CE) и рабочие узлы - Worker Nodes,
- элементы хранения - Storage Elements (дисковые и ленточные),
- мониторирование и систему регистрации (Nagios, APEL),
- систему управления загрузкой (WMS),
- систему регистрации активности и учета использования ресурсов - Logging and Bookkeeping (LB),
- информационную систему (BDII),
- службу передачи файлов (FTS),
- базовые сервисы (NTP, DNS, logging and auditing),

- прокси-ферму http,
- сервисы, специфические для каждой виртуальной организации (VO), в частности, для CMS: PhEDEx.

План работ предусматривает тестирование всех сервисов в ОИЯИ на всех этапах создания Tier-1. Интеграция в структуру SAM/Nagios и тесты доступности и надежности будут произведены на этапе создания прототипа (декабрь 2012), и в дальнейшем проверка доступности и надежности развернутых ресурсов будет производиться постоянно, с первых дней запуска Tier-1.

Основные тесты доступности и надежности планируется выполнить в конце 2013 года, когда Tier-1 ОИЯИ достигнет запланированного объема 10% всех узлов WLCG уровня Tier-1 в совокупности.

Тест доступности сайты (Site availability)

Грид-сервисы в WLCG тестируются с помощью структуры SAM (Service Availability Monitor) [13], которая периодически запускает тесты всех сервисов в инфраструктуре. SAM является одним из основных источников информации о работе Грид, которая используется для определения доступности сервисов. В CMS структура SAM используется для запуска собственных тестов вычислительных элементов (CE) и системы управления ресурсами (SRM) на сайтах. Для каждого типа сервиса определен один или несколько критических тестов. В соответствии с заданным интервалом времени, доступность сервиса определяется как доля времени, в течение которого сервис прошел все критические тесты, а доступность сайта - как доля времени, в течение которого по крайней мере один сервис данного типа доступен на сайте. Критические тесты для CE и SRM приведены в таблице 3.

Таблица 3

Результат	Минимальное время непрерывной работы (месяцев)	Начало	Конец
Исполнение 85% задач	2	Дек. 2013	Февр. 2014
Доступность 98% объема хранения	2	Дек. 2013	Февр. 2014
Работа 98% сервисов WLCG и VO	2	Дек. 2013	Февр. 2014

К концу 2012 года усредненная доступность Tier-1 ОИЯИ ожидается на уровне 70%, с учетом запланированных отключений. Доступность будет доведена до 85%, в соответствии с требованиями CMS для сайтов Tier-1, в середине 2014 года, когда сайт Tier-1 ОИЯИ будет развернут в полном масштабе.

Тест запуска задач (Job Robot)

Еще одним дополнительным методом тестирования является запуск задач, подобных реальным задачам анализа данных. Отличие его от тестов SAM заключается в том, что статистика в этом случае существенно выше (~600 задач/(сайт в день)), а также в том, что поскольку используемые данные могут быть распределены по многим дискам, нагрузка на систему хранения тоже выше, что дает большую достоверность результатов теста. Для реализации этого метода была разработана система Job Robot, которая автоматически запускает задачи, используя систему запуска задач анализа данных CRAB [14].

Через регулярные интервалы времени создается новая задача анализа данных со специальным набором данных (dataset). Затем эта задача разбивается на несколько задач, которые запускаются как набор gLite WMS [15]. Каждая такая задача выполняет некоторую тривиальную обработку части данных. Результаты запуска задач классифицирутся как успешное выполнение, отказ по причине сбоя самой задачи и отказ по причине ошибки

программного обеспечения грид. На сайтах уровня Tier-1 система Job Robot может использоваться для эмуляции отбора по заданным критериям событий из реконструированных данных (data skimming). Эта операция интенсивно читает данные, и тем самым сильно зависит от функционирования инфраструктуры хранения данных.

Ежедневная статистика системы Job Robot используется для определения доли успешных запусков для каждого сайта. В настоящее время запускается около 25000 задач на примерно 60 центрах, что не доходит до максимально возможной загрузки центров. Однако, система Job Robot может быть настроена так, чтобы использовать все доступные для CMS ресурсы центров полностью и затем сравнить результат с тем, что заявлено сайтом, что позволяет обнаружить узкие места в работе сервисов.

Оперативное обслуживание и поддержка центров уровняTier-2

В соответствии с вычислительной моделью CMS, центр Tier-1 ОИЯИ будет

- проводить операционные работы, в соответствии с требования WLCG [16]
- принимать согласованные объемы сырых данных и данных моделирования;
- обеспечивать доступ к данным из других центров уровня Tier-2/Tier-3 инфраструктуры WLCG;
- обеспечивать работу каналов FTS для российских центров Tier-2, включая мониторинг передачи данных.

Более детально задачи будут определены в мае 2013 года.

Tier-1 ОИЯИ будет осуществлять оперативное обслуживание и поддержку региональных центров и пользователей. Это обслуживание должно быть доступно в официальное рабочее время (9:00 – 18:00 MSK), а также должно включать поддержку с помощью электронной почты и в некоторых случаях - выезд специалистов в региональные центры, требующие помощи. Поддержка включает в себя:

- консультации по развертыванию типичного грид-центра;
- помощь по специфическим проблемам грид-сервисов;
- поддержка в разрешении инцидентов, связанных с безопасностью;
- распространение положительного опыта работы.

Заключение

Генеральный план проекта по созданию CMS Tier-1 центра в ОИЯИ состоит из трех основных этапов в 2012-2014 гг. Первый этап нацелен на создание в конце 2012 г. прототипа работающего центра, с общей мощностью компьютерных ресурсов на уровне 10% от номинального центра Tier-1. Второй этап должен быть завершен к концу 2013 г. созданием полномасштабного центра уровня Tie-1 (фаза I). Последующие увеличение компьютерных ресурсов будет осуществляться в течении 2014 г. (фаза II) (см. таблицу 4).

Таблица 4. Ожидаемый рост компьютерных ресурсов CMS Tier-1 центра в ОИЯИ

	2012	2013	2014
CPU (HEPSpec06)	14400	28800	43200
Диски (Terabytes)	660	3168	4224
Ленты (Terabytes)	72	5700	8000

Литература

[1] CMS Collaboration, CMS, the Compact Muon Solenoid: Technical proposal", CERN-LHCC-94-38

[2] http://lhc.web.cern.ch/lhc/; http://public.web.cern.ch/public/en/lhc/lhc-en.html

[3] http://public.web.cern.ch/public/

[4] G.L. Bayatian et al. (CMS Collab.), Journal of Physics G: Nucl. Part. Phys. 34, 995 (2007);

[5] S. Chatrchyan et al. (CMS Collab.), JINST 3 S08004 (2008)

[6] LHC Computing Grid Technical Design Report. CERN-LHCC-2005-024, 2005; Worldwide LHC Computing Grid (WLCG), http://lcg.web.cern.ch/LCG/public/default.htm

[7] Worldwide LHC Computing Grid Memorandum of Understanding, http://lcg.web.cern.ch/lcg/mou.htm

[8] V. Gavrilov at al., RDMS CMS Computing activities before the LHC startup, see Proc.of this Int.Conference “Distributed Computing and GRID-technologies in Science and Education, Dubna, 2012, ibid????.

[9] V. Gavrilov et al., RDMS CMS data processing and analysis workflow, in Proc. of XXIII Int. Symp. on Nuclear Electronics & Computing (NEC`2011), Dubna, 2011, pp.148-153.

[10] C. Grandi, D. Stickland, L. Taylor, CMS NOTE 2004-031 (2004), CERN LHCC 2004-035/G-083; CMS Computing Technical Design Report, CERN-LHCC-2005-023 and CMS TDR 7, 20 June 2005.

[11] R. Rocha et al., Experiment Dashboard for Monitoring of the Computing Activities of the LHC Experiments on the Grid, Grid Computing. Nuclear Science Symposium, IEEE (Dresden), October 2008; http://dashboard.cern.ch/cms/

[12] J. Andreeva et al., "Dashoard for the LHC Experiments", Proceedings of International Conference on Computing in High Energy and Nuclear Physics (CHEP 07), J.Phys.Conf.Ser.119:062008, 2008.

[13] A. Duarte, P. Nyczyk, A. Retico, D. Vicinanza, "Monitoring the EGEE/WLCG Grid Services", J. Phys.: Conf. Ser. 119 (2008) 052014.

[14] D. Spiga et al., "The CMS Remote Analysis Builder (CRAB)", LNCS vol. 4873, pp. 580-586 (2007).

[15] P. Andreetto et al., "The gLite workload management system", J. Phys.: Conf. Ser. 119 (2008) 062007.

[16] http://wlcg.web.cern.ch/grid-operations

ТЕСТИРОВАНИЕ ПРОИЗВОДИТЕЛЬНОСТИ PVM И MPI С МИГРАЦИЕЙ ПРОЦЕССОВ MOSIX В РАСПРЕДЕЛЕННОЙ ВЫЧИСЛИТЕЛЬНОЙ СРЕДЕ

А.В. Богданов[1], Е Мьинт Найнг[2], Пья Сон Ко Ко[2]

[1]*Институт высокопроизводительных вычислений и интегрированных систем, Санкт-Петербург, Россия*
bogdanov@csa.ru.
[2]*Санкт-Петербургский государственный морской технический университет*
yemyintnaing@gmail.com pyaesonekoko@gmail.com.

В настоящем сообщении описываются преимущества схемы миграции процесса для улучшения использования вычислительных ресурсов, а также возможности получить существенные ускорения при выполнении параллельных и многозадачных приложений. Мы провели несколько экспериментов с PVM и MPI, популярными средами программирования для параллельных вычислений, которые используют статическое планирование процесса. Эти тесты были выполнены под управлением операционной среды MOSIX, с и без преимущественной схемы миграции процесса. Результаты этих тестов наглядно доказывают преимущества использования миграции процесса. Кроме того, в статье анализируется файловая система DFSA и определяется производительность MFS с и без DFSA.

Введение

Современные компьютерные технологии позволяют создавать относительно дешевые многомашинные комплексы с общим математическим обеспечением и общими вычислительными ресурсами - вычислительные кластеры. Такие системы обеспечивают относительно низкую стоимость вычислений, хорошо масштабируемы, имеют высокий уровень надежности, имеют апробированные инструментальные средства для конструирования, отладки и анализа параллельных программ [1]. В сообщении рассматривается MOSIX технология организации кластерных систем и средства тестирования её производительности, обсуждаются особенности и возможности. MOSIX — системное ПО для UNIX-подобных ОС, таких как Linux, состоящее из адаптивных алгоритмов разделения ресурсов. Это позволяет множеству однопроцессорных (UP) и SMP узлам использоваться для работы под единым управлением. Алгоритмы разделения ресурсов MOSIX разработаны в соответствии с использованием ресурсов узлов в режиме реального времени. Это достигается миграцией процессов с одного узла на другой, преимущественно и прозрачно, для балансировки загрузки (load-balancing) и предотвращения переполнения памяти. Целью такой организации вычислений является увеличение суммарной производительности и создание удобной многопользовательской и разделяемой по времени среды для запуска последовательных и параллельных приложений [2]. Она поддерживает и интерактивные процессы и пакетные задания. MOSIX можно рассматривать как операционную систему мульти-кластера, которая включает автоматическое обнаружение ресурса и динамическое распределение рабочей нагрузки, обычно она загружается на уединенных компьютерах с большим количеством процессоров [3]. Система MOSIX обеспечивает автоматическое перемещение процессов между ними. Она расширяет ядро Linux механизмами миграции и предоставляет комплект управляющих утилит, предназначенных для настройки системы распределения процессов, а также для отладки и контроля. Особенностью MOSIX является отсутствие централизованного управления — каждый узел кластера может работать как автономная система и самостоятельно управлять вычислительными процессами. Это позволяет динамически конфигурировать кластер, наращивая или сокращая количество узлов без остановки системы. При необходимости MOSIX

допускает использование статического управления, что позволяет накладывать явные ограничения для достижения максимальной эффективности конкретной кластерной архитектуры [4].

Файловая Система MOSIX (MFS)

Ключевым компонентом любой распределенной системы является файловая система. Как и в централизованных системах, в распределенной системе функцией файловой системы является хранение программ и данных, а также предоставление доступа к ним по мере необходимости [5]. Здесь анализируется файловая система прямого доступа MOSIX (DFSA),условие, которая может улучшить производительность кластерных файловых систем, позволяя миграционному процессу непосредственно получить доступ к файлам в его текущем расположении. Такая возможность, в объединении с соответствующей файловой системой, могла бы существенно увеличить производительность ввода-вывода и уменьшить сетевую перегрузку, перемещая процесс, интенсивно использующий средства ввода-вывода, на файловый сервер. DFSA подходит для кластеров, которые управляют пулом общих дисков. С DFSA можно переместить параллельные процессы с клиентского узла на файловые сервера для параллельного доступа к различным файлам. Чтобы протестировать ее производительность, мы будем использовать файловую систему MOSIX (MFS), которая разрешает непротиворечивые параллельные операции на различных файлах [6].

Файловая система прямого доступа DFSA была разработана, чтобы уменьшить дополнительные издержки ввода-вывода, переориентируя системные вызовы миграционного процесса. Это было сделано, чтобы ограничить выполнение большинства таких системных вызовов локально на текущем узле процесса. В дополнение к DFSA новый алгоритм, который учитывает операции ввода-вывода, был добавлен к системе балансировки нагрузки MOSIX. В результате этих изменений процесс, который выполняет от умеренного до большого объема операций ввода-вывода, мигрирует на узел, который и занимается, в основном, операциями ввода-вывода. Еще одно очевидное преимущество состоит в том, что у процессоров, задействованных как в операциях ввод-вывод, так и для вычислений, появляется большая гибкость для такой их миграции с соответствующих узлов, которая лучше балансирует нагрузку. Система MOSIX распределена и масштабируема, поскольку она позволяет выполнять много процессов одновременно, и ей доступны различные файлы, которые были предварительно выделены различными узлами до выполнения параллельных процессов. Улучшения MOSIX реализованы в ядре операционной системы, без изменения интерфейса UNIX, и они полностью прозрачны для прикладного уровня [6].

Производительности PVM и MPI с и без миграции процессов MOSIX

Для организации распределенной вычислительной среды наиболее часто используются следующие программные средства: MPI, MOSIX, PVM. PVM является популярной средой программирования, которая позволяет пользователям объединять подключенные к сети компьютеры и находить устройства, которые можно использовать для организации вычислений. Ее основные преимущества - поддержка неоднородных сетей и машин, возможность организации динамического процесса и управления отдельной виртуальной машиной, а так же простота и эффективность библиотеки и пользовательского интерфейса. Главная цель использования PVM — это повышение скорости вычислений за счет их параллельного выполнения. Функционирование PVM основано на механизмах обмена информацией между задачами, выполняемыми в ее среде. Основные недостатки PVM - это статическое закрепление задач за узлами, которое приводит к неспособности эффективно ответить на изменения загрузки узлов, и предположение, что все рабочие станции имеют близкие скорости [2].

MPI — это стандарт на программный инструментарий для обеспечения связи между отдельными процессами параллельной задачи. MPI предоставляет программисту единый механизм взаимодействия процессов внутри параллельно исполняемой задачи независимо от машинной архитектуры (однопроцессорные, многопроцессорные с общей или раздельной

памятью), взаимного расположения процессов (на одном физическом процессоре или на разных) и API операционной системы. Программа, использующая MPI, легко отлаживается и переносится на другие платформы, часто для этого достаточно простой перекомпиляции исходного текста программы [7].

В нашем эксперименте кластеры MOSIX были созданы в гетерогенной и виртуальной средах. В гетерогенной среде были установлены программы PVM и MOSIX и ней запускалось приложение, которое вычисляет процессорное время при умножении матриц. В виртуальной среде были установлены программы MPI и MOSIX и в ней запускалось приложение по определению времени задержки при синхронизации процессов MPI. В этом примере выполняется коммуникационный тест MPI (время задержки сообщений). MPI-0 отправляет 1-байтовое сообщение к MPI-1, затрачивая время на ожидание ответа между ними. После этого производится синхронизация для каждого повторения, среднее времени ожидания подсчитывается при завершении. Эти тесты были выполнены под управлением операционной среды MOSIX, с и без преимущественной схемы миграции процесса. Результаты этих тестов доказали достоинства использования преимущественной миграции процесса.

Сравнение процессорного времени при умножении матриц на PVM и MOSIX

Сначала выполняется приложение - умножения матриц на PVM без MOSIX, потом на MOSIX. Затем, та же процедура повторяется на PVM с MOSIX. В таблице 1 и на рис.1 приведены результаты по измерению процессорного времени при умножении матриц. Эти результаты наглядно доказывают преимущества использования приоритетных миграций процесса.

Таблица 1. Время вычислений (секунды)

Размерности матриц	Время вычислений (PVM без MOSIX)		Время вычислений (PVM с MOSIX)	
	6 процессоров	10 процессоров	6 процессоров	10 процессоров
400x400	0,326	0,376	0,322	0,347
600x600	1,082	1,066	1,065	1,053
800x800	1,805	1,626	1,864	1,628
1000x1000	3,032	2,378	2,8	2,670

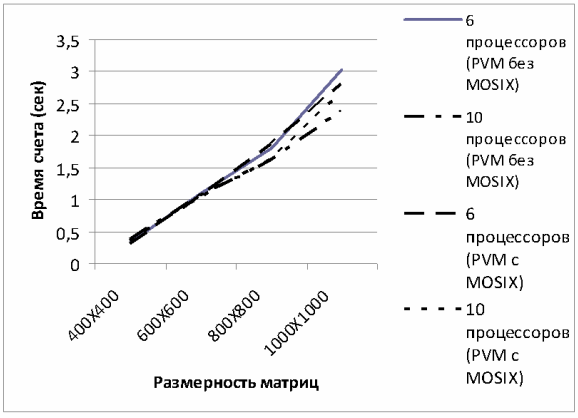


Рис. 1

Для сравнения времени задержки при выполнении тестового коммуникационного приложения на MPI и MOSIX, сначала было запущено приложение по определению времени задержки на MPI без MOSIX. И потом та же операция повторялась под MOSIX. В таблице 2 и на рис.2 показаны результаты по определению времени задержки на приложении MPI-latency. Эти результаты так же доказывают преимущества использования приоритетных миграций процесса.

Таблица 2. Время задержки (микросекунды)

Номер процесса	Время ожидания полного обхода		Время ожидания одностороннего движения	
	MPI	MOSIX	MPI	MOSIX
2	12152	11877	6079	5938

3	12643	11975	6321	5987
4	11875	11228	5937	5614
5	13296	11623	6648	5811
6	13160	11378	6580	5689

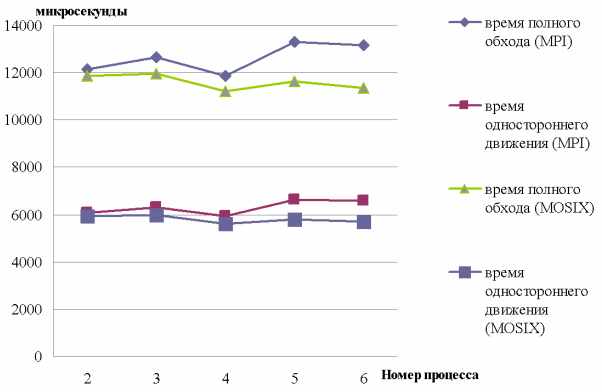


Рис.2

Отличительными особенностями выполнения приложений на MOSIX являются адаптивная политика распределения ресурсов, симметрия и гибкость конфигурации. Комбинированный эффект этих свойств подразумевает, что пользователь не должен знать текущего состояния ресурсов на различных узлах и даже их количества. Параллельные приложения могут выполняться, позволяя MOSIX назначать и переназначать процессы на эффективнейшие из возможных узлов, почти так же, как и в SMP. В отличие от таких пакетов, как MPI или PVM, фиксирующих процессы в конкретных узлах кластера, MOSIX обеспечивает их прозрачную динамическую миграцию. При этом MPI и PVM могут использоваться совместно с MOSIX. Зависимость производительности от скорости процессора и объема памяти на стартовом узле и для повышения производительности на сильно связанных задачах необходимо повышать скорость сети, например используя Gigabit Ethernet или Myrinet.

Литература

[1] Белогурский А.И., Васин Л.А., Вашкевич Н.П., Князьков В.С., Шашков Б.Д. Россия, Пенза, Пензенский государственный университет. Технологии построения и тестирования вычислительных кластеров. Труды V Международной научно-технической конференции "Новые информационные технологии и системы", изд-во Пенз. Гос. Университета, Пенза, 2002 г., с. 70-76.

[2] Н.В. Морев, ГОУ Владимирский государственный университет Кафедра вычислительной техники. Решение СЛАУ с использованием кластерной системы. http://kolia.pp.ru/programming/mpi/text.pdf

[3] A. Barak and A. Shiloh, A White Paper. The MOSIX Management System for Linux Clusters, Multi-Clusters, GPU Clusters and Clouds. http://www.mosix.org/pub/MOSIX_wp.pdf

[4] Д. Кузьмин, Ф. Казаков, Д. Привалихин, А. Легалов. На пути к переносимым параллельным программам. Журнал "Открытые системы", №5, 2003.

[5] Keren A. and Barak A., Opportunity Cost Algorithms for Reduction of I/O and Interprocess Communication Overhead in a Computing Cluster. IEEE TPDS, Vol. 14, No. 1, 2003.

[6] Direct File System Access (DFSA) http://support.math.cmu.edu/Parallel_Cluster/mosix/txt_whatis.html

[7] Аль-хулайди Абдулмаджид Ахмед Галеб, Распределённые вычисления (кластерные вычисления) с использованием пакета параллельного программирования MPI // Современные наукоемкие технологии. – 2010. – № 4 – С. 82-82.

GRIDCOM, GRID COMMANDER: ГРАФИЧЕСКАЯ ОБОЛОЧКА ДЛЯ РАБОТЫ С ЗАДАЧАМИ И ДАННЫМИ В ГРИДЕ

В.В. Галактионов
Лаборатория информационных технологий ОИЯИ, г. Дубна
galakt@jinr.ru

WLCG (Worldwide LHC Grid Computing) – программно-вычислительный комплекс распределенной архитектуры предназначен для сбора, хранения и обработки большого объема данных, генерируемых ускорителем **LHC** (Large Hadron Collider) [1].

В данной публикации описывается программный пакет **GridCom,** предназначенный для обеспечения автоматизации доступа к средствам распределенной системы WLCG (задачам и данным).

Все операции с задачами и данными выполняются пользователем в командном режиме **CLI** (Command Language Interface), представляющим собой совершенно недружественный интерфейс пользователя к WLCG. Поэтому, одной из **главных целей** данной работы было обеспечить более удобные средства для пользователя в форматах уже известных современных графических средств оперирования данными и задачами, в частности с использованием средств WEB-доступа к программному обеспечению WLCG.

1. Job Management. Управление задачами

Работа с задачами в WLCG выполняется в несколько этапов.

Подготовка задачи заключается в ее описании на языке **JDL** (Job Definition Language) в текстовом файле, как правило, с расширением имени **.jdl.** Это описание содержит основные параметры задачи:

- тип задачи,
- название исполнительного модуля,
- перечень файлов с программами, которые составят передаваемый в Грид пакет,
- перечень файлов с данными, которые будут передаваться в Грид в пакете (StdInput, InputSandbox),
- перечень файлов для результатов счета задачи (StdOutput, StdError, OutputSandbox),
- перечень Грид-файлов, необходимых задаче для обработки данных и размещения результатов обработки (InputData и OutputData),
- другие требования задачи для выбора подходящего для счета **CE** (Computing Element).

Запуск задачи и контроль ее состояния выполняется пользователем в режиме CLI на UI (User interface)-машине:

Запуск задачи:

glite-wms-job-submit –a <jdl-файл>

Задача передается в **WMS** (Workload Management System) и после успешного синтаксического и содержательного анализа jdl-файла пользователь получает сообщение типа:

Connecting to the service https://lcgwms01.jinr.ru:7443/glite_wms_wmproxy_server

с именем WMS-сервера, принявшего задачу, а также идентификатор задачи **jobID** типа:

https://lcglb11.jinr.ru:9000/HdG_Pv8WwtVoVo0HyBK0pA

Запрос **состояния** переданной задачи выполняется командой
glite-wms-job-status <jobID>.

При появлении статуса задачи типа **Done** (Done (Success) или Done (Failed)) можно затребовать на UI-машину результаты счета, описанные JDL-параметрами StdError и StdOut:

glite-wms-job-output <jobID> или
glite-wms-job-output –dir <dir_name> <jobID>

Результаты счета записываются на UI-машине в установленной UI-конфигуратором директории (первый вариант) либо в директории, указанной пользователем (второй вариант). После этого задача принимает состояние **Cleared**.

2. Data Management. Управление данными

В WLCG используются несколько типов операций с файлами:

- *Массовая пересылка* данных применяется специальными уполномоченными администраторами для перемещения больших объемов данных между основными хранилищами данных, например между Tier0 и Tier2. Эта задач выполняется сервисом **FTS** (File Transfer Service).
- *Пользовательский уровень* работы с файлами. Эти операции выполняются двумя группами команд: **lfc-*** - для работы с **LFC** (LCG File Catalogue)-каталогом и LCG Data Management tools (**lcg_util**) для непосредственных операций с файлами (заведение, копирование, удаление и др.
- *Специальный* (нижний уровень) работы с файлами: **GSIFTP** (edg-gridftp-* и globus-url-copy).

2.1. Операции с файлами пользовательского уровня

Upload, копирование файла из локальной UI-машины на SE в WLCG и включение его имени в LFN (Logical File Name)-формате в LFC-каталог. Обязательные параметры команды – название виртуальной организации (для правильного выбора LFC-каталога) и имя локального файла в UI-машине. Остальные атрибуты Грид-файла (**SE** (Storage Element) и LFN-имя файла) устанавливаются автоматически (по умолчанию) либо могут быть определены пользователем. В любых операциях с LFC-каталогом должно быть установлено значение переменной **LFC_HOST**.

Download, копирование файла из хранилища SE в локальную на UI-машине файловую систему.

3. Архитектура GridCom

Программный пакет GridCom состоит из двух частей: *клиентской* части **GridCom** и *исполнительной* - **Rex** (Runtime Executor):

Клиент GridCom не требует установки на конечных пользовательских компьютерах каких-либо специальных программ, выполняется в операционных системах Windows или UNIX стандартными сетевыми приложениями-браузерами, обеспечивает интерактивное взаимодействие с пользователем в графическом режиме с применением Java-технологий:

- Апплет-структура пакета обеспечивает WEB-интерфейс пользователя к исполнительной части. Этим обеспечивается популярная в Сети парадигма *тонкого* клиента: минимум нагрузки и инсталляций для машин потребителя.
- Две части клиентской программы **Job Management** (Управление задачами) и **Data Management** (Управление данными) выполняются в разных окнах и работают совершенно независимо. Программа формирует запросы пользователя в виде записей в базу данных *(семафоры* и данные), запоминает в очереди и периодически в режиме polling сканирует состояние запроса.
- Существует вариант GridCom, выполненный в виде Java-приложения.

Серверный компонент Rex выполнятся только в специальных вычислительных системах с программным обеспечением доступа к WLCG – UI-машинах. Таким, например, является LINUX-кластер ЦИВК ЛИТ ОИЯИ. Rex получает запросы пользователя и формирует из них наборы команд в формате CLI для работы с локальной файловой системой и с задачами и данными WLCG. Для эффективного функционирования серверная программа выполняет запросы, используя *потоковый* механизм распараллеливания процессов, группируя их по типам (работа с задачами, данными и др.).

База данных. В GridCom используется база данных общего пользования MySQL из стандартного программного обеспечения UNIX-кластера ЛИТ ОИЯИ. Для MySQL JDBC-драйверы содержатся в свободно распространяемом пакете **mysql-connector-java-3.1.12-bin.jar.**

4. Управление задачами в GridCom

Как уже упоминалось выше управление задачами в GridCom выполняется загружаемым через Интернет Java-апплетом (WEB-интерфейс) графическими изобразительными средствами и интерактивными запросами пользователя (Рис. 1). Все операции сводятся к выбору объекта (операция **Select**) в окне (мышью), минимальные действия с ним (редактирование) и выбора действия с ним в popup-меню. Выбранный пользователем тип действия над задачей трансформируется в команды **gLite**, которые, используя механизм **Runtime,** выполняются серверным компонентом **Rex** в операционной системе UI-машины. Основные операции с задачами:

- подготовка задачи,
- передача задачи в WMS,
- получение результатов счета задач.

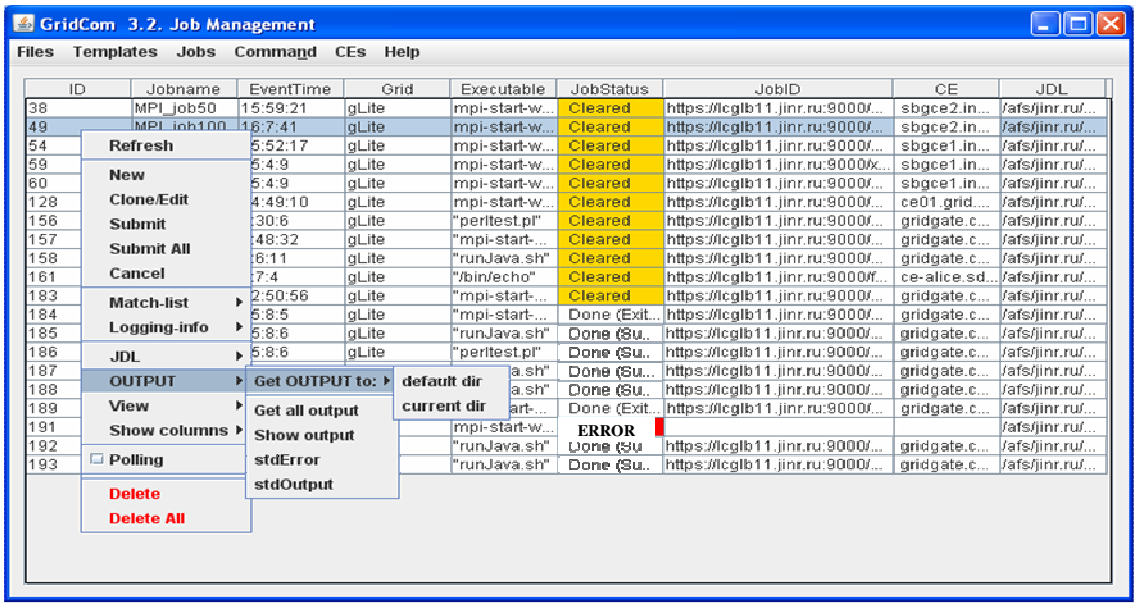


Рис. 1: Окно управления задачами и popup-меню

5. Управление данными в GridCom

Управление данными в GridCom выполняется в отдельном окне **“Data management”** (Рис. 2) независимо от окна управления задачами. Основные типы операций с данными в WLCG описаны выше. Как и в случае окна “Job Management”, в клиентской части GridCom графические средства используются для выбора объекта, типа операции над ним и формирования запроса в исполнительную часть – Lexor, в котором эти запросы формируются в систему gLite-команд и выполняются на UI-машине.

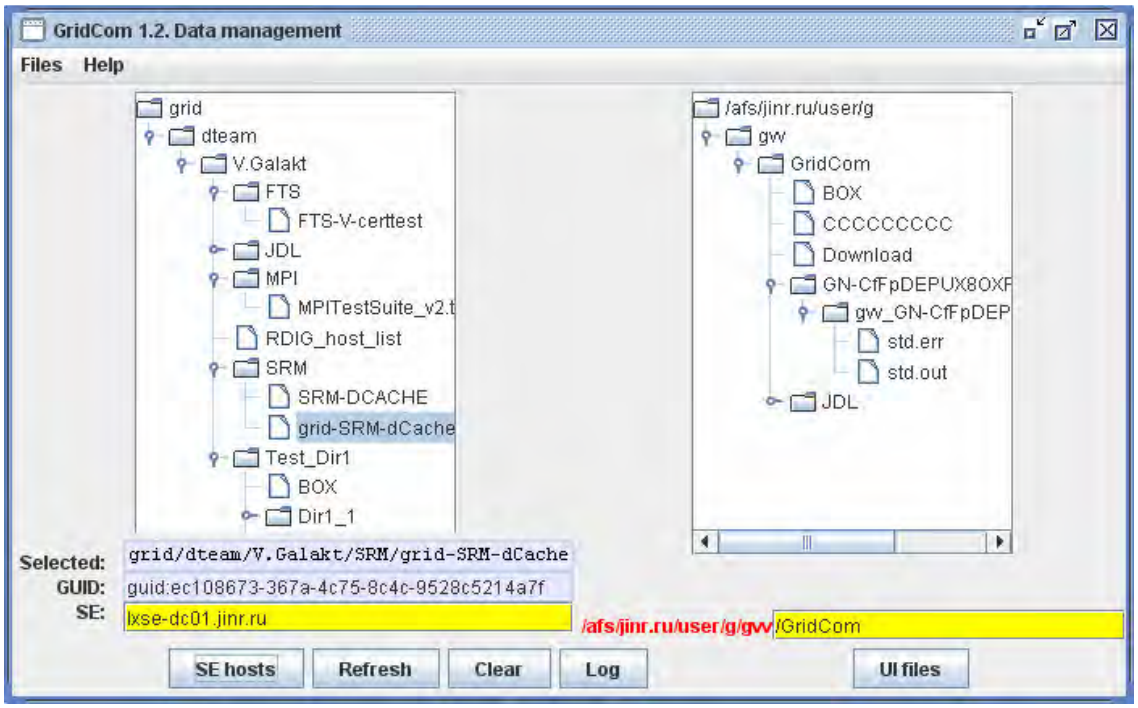


Рис. 2: Окно управления данными

Литература

[1] Фостер Я., Кессельман К., Ник Д., Тьюке С. Grid-службы для интеграции распределенных систем, “Открытые системы”, № 1, изд-во “Открытые системы”, Москва, 2003, с. 20.

[2] gLite 3.1 User Guide. https://edms.cern.ch/file/722398/1.3/gLite-3-UserGuide.html.

О ПРИМЕНЕНИИ ТЕХНОЛОГИИ CUDA ДЛЯ ОБРАБОТКИ ИЗОБРАЖЕНИЙ И РАСПОЗНАВАНИЯ ГРАФИЧЕСКИХ ОБРАЗОВ

И.М. Гостев

Национальный исследовательский институт «Высшая школа Экономики»
101000, Москва Россия
Лаборатория информационных технологий ОИЯИ, 141980, Дубна, Россия

Решение задач по обработке изображений и распознаванию графических образов обычно опирается на некоторую технологию, заключающую в себя последовательность некоторых операций. Очевидно, что на выполнение всех операций требуется время, которое зависит от их количества и трудоемкости, размеров входного изображения и скорости передачи информации между отдельными этапами обработки.

Используемая технология [1, 2] построена так, чтобы исключить зависимость результатов от качества изображений и количества цветов, числа и размеров распознаваемых объектов. И что самое важное, она обеспечивает инвариантность к аффинным преобразованиям при работе в реальном масштабе времени. Кроме того последовательность обработки изображений построена так, что требует для получения результата в каждой операции только один проход по изображению. Это дает возможность организовать конвейер методов, при котором каждый последующий метод начинает свою работу после получения от предыдущего метода некоторое количество строк, достаточное для его работы.

Тем не менее, современные требования к обработке изображений включают в себя не только работу со статическими изображениями, а и распознавание объектов на видео и потоковых изображениях. Такие условия налагают дополнительные требования к используемой технологии. При размерах изображений более 640x350 пикселей и 30 кадрах в секунду указанная технология уже не может обеспечить работу в реальном масштабе времени. Необходимо использовать некоторую распределенную среду, в которой каждая операция выполнялась бы на нескольких процессорах, число которых определялось бы её сложностью.

Такая технология была предложена в [3] и реализована. В ней каждая операция выполнялась на одном или нескольких отдельных узлах (процессорах). Каждый узел содержит свой буфер фиксированного размера, в котором сохраняются результаты операции и из которого поступают на следующий узел. Схема одного узла показана на рис.1. В такой схеме контроллер потока используется только для синхронизации отдельных процессов и не участвует в актах передачи данных. Размер буферов определяется на этапе планирования процесса и определяется вычислительной сложностью самой операции.

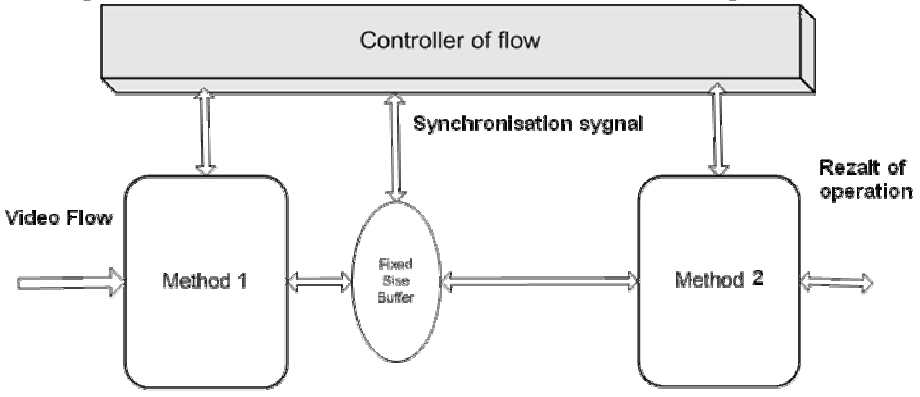


Рис. 1: Схема одного узла распределенной сети по обработке изображений и распознаванию образов

Исследования быстродействия разработанной системы показали, что она неплохо справляется с видео потоком до размеров 1024x760 пикселей. Далее работа в реальном времени становится неустойчивой. Некоторые усовершенствования архитектуры и механизмов передачи данных, проведенные на основе анализа задержек, показали, что самым слабым местом такой системы остается передача информации от одной операции к другой (от одного процессора к другому).

Рассмотрение различных технологий реализации системы и её моделирование в различных архитектурах показало, что:

1. Использование массивно-параллельных вычислительных систем приводит к непредсказуемым задержкам при передаче информации через высокопроизводительные коммутаторы с коммутацией пакетов или конкуренцией за доступ к массиву общей памяти (SMP). Этот эффект хорошо известен и объясняется тем, что при возрастании количества процессоров одновременно используемых в одной операции приводит к увеличению конкуренции за доступ к ресурсам и, следовательно, к возрастанию накладных расходов во времени.
2. Использование кластеров с разнесенными процессорами приводит к задержкам по транспортным и сетевым протоколам. Поскольку сам стандарт TCP/IP не гарантирует временные параметры доставки своих пакетов. Моделирование процессов с длиной пакета, определяемой сложностью и трудоемкостью метода обработки изображений показало, худшие результаты, чем при передаче пакетов фиксированной длины. Применение протоколов на основе ATM для передачи информации между узлами приводит к дополнительным накладным расходам на кодирование – декодирование, что тоже не дает выигрыша во времени.
3. В аналогичном процессе моделирования на SMP, ориентированном на использовании различного вида буферов (кэшей), в которых хранятся промежуточные результаты от каждой операции, показало, что оптимизация их размеров сводится на нет, из-за конкуренции процессов обмена информацией между ними.
4. Разработанные алгоритмы маршрутизации на основе теории графов для передачи информации от одного узла сети к другому с равномерными статическим и динамическим распределениями потоков несколько улучшают ситуацию, однако, из-за вышеуказанных причин, в различных архитектурах не позволяют эффективно использовать мощности распределенной вычислительной системы.

Все это заставило еще раз рассмотреть технологию обработки изображений и распознавания образов и разработать следующую схему, построенную на теории графов [4]. Для описания каждого из внутренних потоков была использована схема, в основе которой лежит тройка <S, L, P>, где S –начальная строка первого фрагмента, L – длина каждого фрагмента в строках, P – промежуток от конца предыдущего фрагмента до начала следующего. Таким образом, <S, L, P> задает последовательность фрагментов $\left[S+(L+P)*i,\ S+(L+P)*i+L\right]$, $i \in N_0$. Две схемы будут *смежными*, если поток, полученный слиянием двух потоков (состоящий из всех фрагментов первого и второго потока) так же может быть задан схемой потока (такая схема называется *объединенной*). Пример схемы со статическим распределением потоков, в котором количество процессоров на одну операцию фиксировано, приведен на рис. 2.

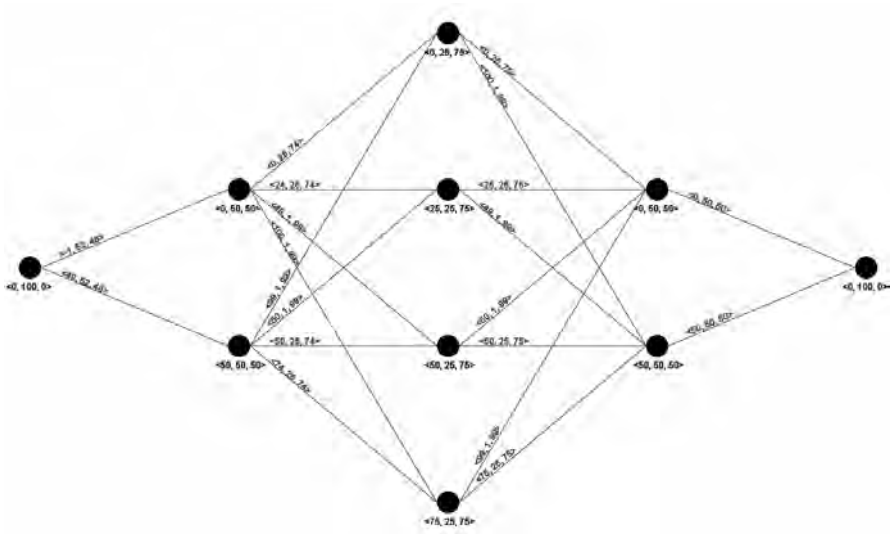


Рис. 2: Схема статического распределения видео потоков при обработке изображений и распознавании графических образов

Теперь с учетом разработанной схему и обнаруженных временных задержек можно сформулировать уточнённые требования к функционированию вычислительной сети в виде следующих правил:

1. Каждому элементу последовательности методов (операции) соответствует один или несколько узлов, являющихся экземплярами этого метода. Совокупность этих узлов образует ярус сети, соответствующий данному методу цепочки операций.
2. Все входные каналы данного узла, соединяют его только с узлами яруса, соответствующего предыдущему методу цепочки. Все выходные каналы узла соединяют его только с узлами яруса, соответствующего следующему методу в цепочке.
3. Первый ярус содержит единственный узел, соответствующий первому методу цепочки методов – методу загрузки (считывания) изображения и не имеет входных каналов.
4. Последний ярус содержит единственный узел, соответствующий последнему методу цепочки – методу сохранения (выдачи) результирующего изображения и не имеет исходящих каналов.

Реализация этих пунктов обязательно приводит к решению задачи по передаче пакетов данных между узлами некоторой сети. Причем одним из основных требований является контролируемость скорости обмена. В противном случае становится невозможно рассчитать и назначить необходимое число узлов на каждую операция для обеспечения режима реального времени при обработке видео потока.

Все эти исследования дали возможность сформулировать требования к системе по обработке изображений и распознаванию графических образов относительно используемых ею вычислительных ресурсов:

1. Обработка фрагментов изображения (видео поток при этом нарезается на небольшие фрагменты по 100-150 строк) должна производиться на отдельных процессорах с локальной памятью, независимой от других потоков – образуя *ярус* обработки.
2. Результаты отдельных операций от каждого узла в последовательности должен сохраняться в некоторой разделяемой памяти, из которой информация будет поступать на следующий ярус обработки.
3. Контроллер потока следит за тем, чтобы последующий ярус не смог считывать информацию из разделяемой памяти до состояния её готовности. Т.е. осуществлять

синхронизацию записи результатов отдельных процессов обработки в различные участки разделяемой памяти.

4. Возможность выполнения параллельных потоков обработки каждого яруса на большом (априори неопределенном) количестве процессоров.
5. Динамическое выделение и освобождение необходимого числа процессоров в ходе вычислительного процесса.
6. Отсутствие задержек на передачу информации между отдельными потоками в процессе и между процессами.

Анализ этих требований и заставил обратить внимание на технологию **CUDA** (Compute Unified Device Architecture) [5-7]. Архитектура CUDA по классификации Флинта соответствует SIMD компьютерам. Это подразумевает наличие в архитектуре одного процессора, параллельно обрабатывающего множество потоков данных как показано на рис. 3.

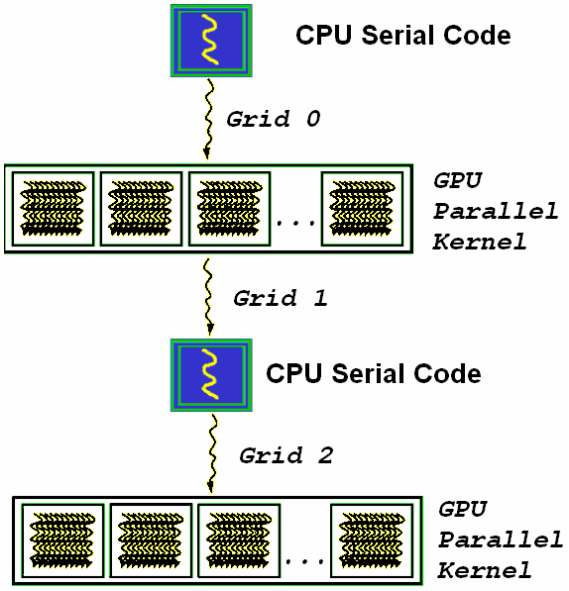


Рис. 3: Схема последовательности исполнения команд в архитектуре CUDA

Реально архитектура CUDA реализуется на базе видеоадаптера, в котором обработка информации осуществляется параллельно на множестве графических процессоров GPU. Эти процессоры в свою очередь сгруппированы в блоки — варпы (warp), состоящие и некоторого числа потоков (обычно 32). Данные обрабатываются только варпами. В варпе имеется специальная разделяемая память объемом 16 килобайт на каждый процессор, которую можно использовать при организации обмена между процессами яруса. Отметим, что каждый процесс варпа выполняются всегда на одном процессе, т.е. отсутствует перегрузка контента. В тоже время обмен через разделяемую память между варпами не разрешается. Каждый процессор имеет возможность обращаться к видеопамяти, однако быстродействие обмена здесь значительно ниже. Кроме того каждый процессор варпа имеет еще свой 8 килобайтный кэш для ускорения обмена и хранения временных данных.

Таким образом, применяя архитектуру CUDA, для решения задач по обработке видео потока автоматически получаем следующие:

1. Выполнение каждого яруса системы производится на отдельном варпе. При этом другие варпы могут выполнять операции в других ярусах. Начальная порция информации поступает на первый варп и обрабатывается нужным количеством процессоров. Затем через глобальную память передается на второй

варп, который обеспечивает выполнение другого яруса и так далее в конвейерном режиме.

2. Обмен данными между отдельными процессами яруса происходит независимо от всех других процессов в системе и, следовательно, конкуренция при передаче данных по единой шине практически отсутствует, поскольку в архитектуре CUDA используется множественный доступ к памяти за счет разбиения памяти на независимые банки. При этом каждый банк имеет свой адрес в цикле обработки, поэтому память поддерживает одновременно несколько параллельных потоков.
3. Обмен информацией между глобальной памятью видеоадаптера и разделяемой памятью варпа осуществляется автоматически, синхронно с выполнением процессов обработки данных в одном ярусе.
4. Архитектура CUDA автоматически осуществляет назначение нужного количества процессоров для выполнения операций на каждом ярусе (для обработки изображений их хватает с избытком).

Фактически эти свойства полностью удовлетворяют требованиям, предъявляемым к вычислительной системе. Для экспериментов был использован ноутбук Sony Vaio VPCF24M1R, имеющий следующие технические характеристики: Производитель видеоадаптера: NVIDIA; серия: GeForce GT 540M; графический процессор: GF108. Потоки: шейдеры 96, тактовая частота ядра: 627МГц, частота шейдеров: 1344 МГц , частота памяти: 1800 МГц , разрядность шины памяти: 128 Бит.

При использовании CUDA технологии вышеизложенный процесс можно записать в виде следующих этапов:

1. Покадровое копирование информации из глобальной памяти хоста в разделяемую память устройства (память видеоадаптера).
2. Разделение изображения на фрагменты и направление каждого фрагмента на один из процессоров первого варпа.
3. Параллельная обработка каждого фрагмента изображения на своём процессоре в варпе с сохранением результатов в разделяемой памяти и копирование общего результата в глобальную память. Этап выполняется для каждой операции в последовательности.

Необходимо отметить следующие положительные стороны использования CUDA технологии для обработки изображений и распознавания графических образов:

1. В каждом варпе могут выполняться до 256 потоков. Т.е. распараллеливание процесса на каждом этапе может быть очень большим, а это означает, что можно обрабатывать большие изображения.
2. Отсутствует необходимость принудительно назначать количество процессоров на один процесс.
3. Не требуется специальная синхронизация потоков.
4. Не нужно перезагружать локальные буферы памяти при последовательном переходе с одного процесса обработки к другому.

Все это позволило повысить суммарную производительность системы, по сравнению с вычислениями без использования CUDA технологии приметно в 50 раз.

Тем не менее, эта технология имеет ряд недостатков:

1. Для её реализации необходимы видеоадаптеры фирмы NVIDIA из серии GeForce версии не ниже восьмой.
2. Некоторое усложнение технологии программирования и необходимость дополнительного программного обеспечения.
3. При длине строки превышающей 16000 пикселей во входном цветном изображении возникают проблемы с недостатком локальной памяти.

Тем не менее, полученные результаты позволяют сделать заключение о том, что CUDA технология имеет много преимуществ при обработке больших изображений и видео потоков и

позволяет устранить большую часть проблем, возникающих в других технологиях и компьютерных архитектурах. Всё это позволяет считать эту технологию перспективной и продолжить исследования.

Литература

[1] Гостев И.М. О методах распознавания графических образов // Изв. РАН ТиСУ. – 2004. – № 1 – С.138-144.

[2] Гостев И.М. Об идентификации графических объектов по контурным фрагментам // Изв. РАН ТиСУ. – 2005. – № 1, – С. 144-151.

[3] Гостев И.М. About organization distributed calculation in system graphic pattern recognition // Тезисы IV Международной конференции «Распределенные вычисления и Грид-технологии в науке и образовании» Дубна, июнь 2010 г.

[4] Гостев И.М., Подгорбунский А.Г. О построении высокоскоростной системы по обработке изображений и распознаванию образов // Изв. вузов. Приборостроение №2. 2009.

[5] Сайт компании NVIDEA - www.nvidia.ru/

[6] Боресков А.В., Харламов А.А. Основы работы с технологией CUDA. М.: ДМК Пресс, 2010. 232 с.

[7] Сандерс Дж., Кэндрот Э. Технология CUDA в примерах: введение в программирование графических процессоров М.:ДМК Пресс, 2011. – 232 с.

МЕТОДЫ ВЫДЕЛЕНИЯ КЛАСТЕРОВ В БОЛЬШИХ СЕТЯХ ПЕРЕДАЧИ ДАННЫХ*

Я.Р. Гринберг[1], И.И. Курочкин[1], А.В. Корх[2]
[1]*Центр Грид-технологий и распределенных вычислений ИСА РАН, Россия, 117312, г. Москва, проспект 60-летия Октября, д.9*
kurochkin@isa.ru
[2]*Московский физико-технический институт, Россия, 141700, Московская область, г. Долгопрудный, Институтский переулок*
artkorkh@gmail.com

1. Введение

Данная работа является логическим продолжением работ [1 - 2], в которых решалась задача прокладки потоков в телекоммуникационных сетях при последовательно поступающих заявках. Тем не менее, скорость работы алгоритмов, предложенных для заполнения потоками сети от поступающих заявок, значительно замедлялась на сетях больших размеров (>100 узлов), из-за необходимости вычисления мультиразрезов графов. Для преодоления возникающих вычислительных сложностей было предложено использовать двухуровневую маршрутизацию [3]. Целью настоящей работы является решение задачи нахождения и выделения в телекоммуникационных сетях узлов с дополнительной функциональностью, для организации двухуровневой маршрутизации потоков в этих сетях. Для того, чтобы найти такие узлы, решаются задачи:

- Введения характеристик, по которым можно произвести выборку узлов сети с дополнительной функциональностью;
- объединения выделенных узлов с дополнительной функциональностью в связную подсеть;
- присоединения оставшихся узлов сети к узлам с дополнительной функциональностью, то есть проведения кластеризации узлов сети вокруг выделенных элементов.

В работе будет представлена такая характеристика узла, как его мера разгруженности, для выделения узлов с дополнительной функциональностью. Представленные меры разгруженностей будут зависеть, как от пропускных способностей дуг сети, так и от степени узла в сети. Выделение узлов с дополнительной функциональностью будет производиться с помощью метода агрегации сетевых данных, принцип работы которого также будет представлен. Помимо этого будут приведены экспериментальные результаты по выделению таких узлов.

2. Постановка задачи

В основе математической модели телекоммуникационной сети лежит связный граф $G = (V, E)$, где V - множество узлов графа, E - множество ребер графа, соединяющих узлы. Каждому ребру $e_{ij} \in E$, $\{i, j\} \in V$ графа G поставлено в соответствие неотрицательное число $c_{ij} \geq 0$ - пропускная способность ребра. Маршрутизация потоков в сети осуществляется через несколько выделенных узлов – суперпиров Sp, и для любой пары, $\{s_i, t_i\}$ каждый простой поток между ними проходит через один и более суперпиров.

* Поддержка ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы» (Госконтракт № 11.519.11.4012), РФФИ грант № 12-07-00415-а, программа фундаментальных исследований ПРАН № 14.

Для выделения суперпиров введем такую количественную характеристику узла, как разгруженность $U(p)$ – мера способности узла p принять, обработать и передать дальше поток. Предлагаемый в данной работе подход выделения суперпиров предполагает такое выделение узлов - суперпиров, что:

Величина разгруженности узла не меньше определенного порога $U(p) \geq U_{Limit}$, чтобы исключить попадание загруженных узлов в число суперпиров, и тем самым осуществлять маршрутизацию потоков через него.

Сумма разгруженностей суперпиров максимальна $\max \sum_{i \in Sp} U_i(p)$.

Для определения самой функции разгруженности $U(p)$ узлов рассматривались следующие варианты, учитывающие различные подходы к пониманию самой характеристики разгруженности:

1. $U(p) = \frac{\sum_i c_{pi} - f_{pi}}{\sum_i c_{pi}}$ (2.1). Числитель представляет собой сумму остаточных пропускных способностей ребер исходящих/входящих/всех из узла, а знаменатель сумму исходных пропускных способностей исходящих/входящих/всех ребер узла.

2. $U(p) = \exp\left(\lambda_1 * \frac{rescap(p)}{\max_p(rescap(p))}\right) + \exp\left(\lambda_2 * \frac{d(p)}{\max_p(d(p))}\right)$ (2.2). $rescap(p)$ - сумма остаточных пропускных способностей ребер инцидентных вершине, $d(p)$ - степень рассматриваемой вершины, обе эти величины нормированы на максимальные значения этих величин, встречающиеся в сети. λ_1 и λ_2 - нормирующие коэффициенты, определяют решающий вклад степени или остаточной пропускной способности дуг, инцидентных вершины.

3. $U(p) = \frac{\sum_i c_{pi} - f_{pi}}{\sum paths(p)}$ (2.3). Числитель представляет собой сумму остаточных пропускных способностей ребер исходящих/входящих/всех из узла, а знаменатель сумму потоков (выраженных через простые пути), проходящих через данную вершину.

4. $U(p) = \lambda_1 * \left(\frac{rescap(p)}{\max_p(rescap(p))}\right) + \lambda_2 * \left(\frac{cycle_3(p)}{\max_p(cycle_3(p))}\right)$ (2.4). Где $rescap(p)$ - сумма остаточных пропускных способностей ребер инцидентных вершине, $cycle_3(p)$ - число трехзвенных циклов, начинающихся и заканчивающихся в узле p. λ_1 и λ_2 - нормирующие коэффициенты.

Все указанные функции разгруженности были использованы в последующих численных экспериментах для определения наиболее подходящей для последующей реализации алгоритма двухуровневой маршрутизации.

3. Описание метода агрегации данных

Пусть каждый узел сети поддерживает соответствующий вектор данных: каждый вектор данных узла состоит из следующих параметров: $(TTL, n, \min U, \max U, d)$, TTL - время жизни вектора информации, n - параметр, определяющий размер сети, $\min U, \max U$ - расчет минимальной и максимальной загрузки узлов в сети, d - дополнительный вектор из b -

элементов для аппроксимации распределения загрузки в сети (наподобие столбцовой диаграммы). В самом начале узел создает вектор: $(TTL,1,U(p),U(p),I_p)$ (второй элемент вектора равен единице для одного узла, а для остальных его значение равно нулю), где I_p - начальная гистограмма распределения, определяемая $I_p(i)=\begin{cases}0 & \text{если } U(p)<\min U+i\times\lambda\\ 1 & \text{если } U(p)\geq\min U+i\times\lambda\end{cases}$, $0\leq i<b$, $\lambda=\dfrac{\max U-\min U}{b-1}$, b - системный параметр, такой что $b>1$. Далее для любых двух вершин p и q производится обмен и модификация их векторов. Причем модификация векторов данных производится следующим образом: $\left(\dfrac{TTL_p+TTL_q}{2}-1,\dfrac{n_p+n_q}{2},\min(\min U_p,\min U_q),\max(\max U_p,\max U_q),\dfrac{d_p+d_q}{2}\right)$. Наконец, в конце процесса агрегации, узел p рассчитывает системные характеристики: $N=\dfrac{1}{n_p}$, $\min U=\min U_p$, $\max U=\max U_p$ и для каждого $i\in N$, такого что $0\leq i<b$ производится $D=\dfrac{d_p(i)}{n_p}$, где D - диаграмма распределения количества узлов от разгруженности. Т.е мы получаем распределение загруженности узлов по сети.

После некоторого числа итераций, каждый узел будет обладать достаточно точной информацией по максимальной величине загруженности узла в сети, минимальной величине загруженности узлов в сети, числе узлов в сети и диаграмме распределения загруженностей узлов в сети. Таким образом, исходя из этих данных, узел может определить свое место на этой диаграмме распределений загруженности, и его разгруженность больше некоторого граничного значения, то он получает статус суперпира, о чем и уведомляет своих соседей.

Были проведены следующие численные эксперименты:

1. Нахождение общего числа суперпиров на каждом шаге при поступающей удовлетворенной заявке в сети, а также число связных подграфов, образуемых найденными суперпирами. Для всех сетей ставилось требование по нахождения суперпиров, составляющих треть (от 10 до 15) от общего числа узлов. (Первый эксперимент). Использовались сети с кластерной и стохастической топологиями, число узлов от 30 до 40.
2. Нахождение связного подграфа суперпиров, кластеризация узлов вокруг ближайших суперпиров, и анализ степени покрытия кластеризованными вершинами всей сети. (Второй эксперимент). Использовались сети с кластерной и стохастической топологиями, число узлов от 30 до 40.
3. Нахождение выделяемых суперпиров в зависимости от количества дуг в сети. (Третий эксперимент). Использовались сети со стохастической топологиями, число узлов равным 500.

4. Результаты экспериментов

В ходе проведения первого численного эксперимента, было установлено, что алгоритм последовательной маршрутизации потоков на выбранных сетях не играет существенной роли при определении числа суперпиров для одной и той же функции разгруженности. Поэтому представим результаты (среднее число найденных суперпиров) для простого и субоптимального минимально разрезного алгоритмов маршрутизации потоков. (Для термина «разгруженность» введем английский аналог «Utility», который будет использоваться на графиках и таблицах). На рис. 1 приведены типичные для всех экспериментов результаты по нахождению среднего числа суперпиров. На рис. 2 показаны величины разбросов числа суперпиров для сетей обоих топологий для всех функций разгруженности. Под величиной

разброса понимается разность между максимальным и минимальным числом суперпиров, получаемым для одной сети при использовании различных функций разгруженности узлов.

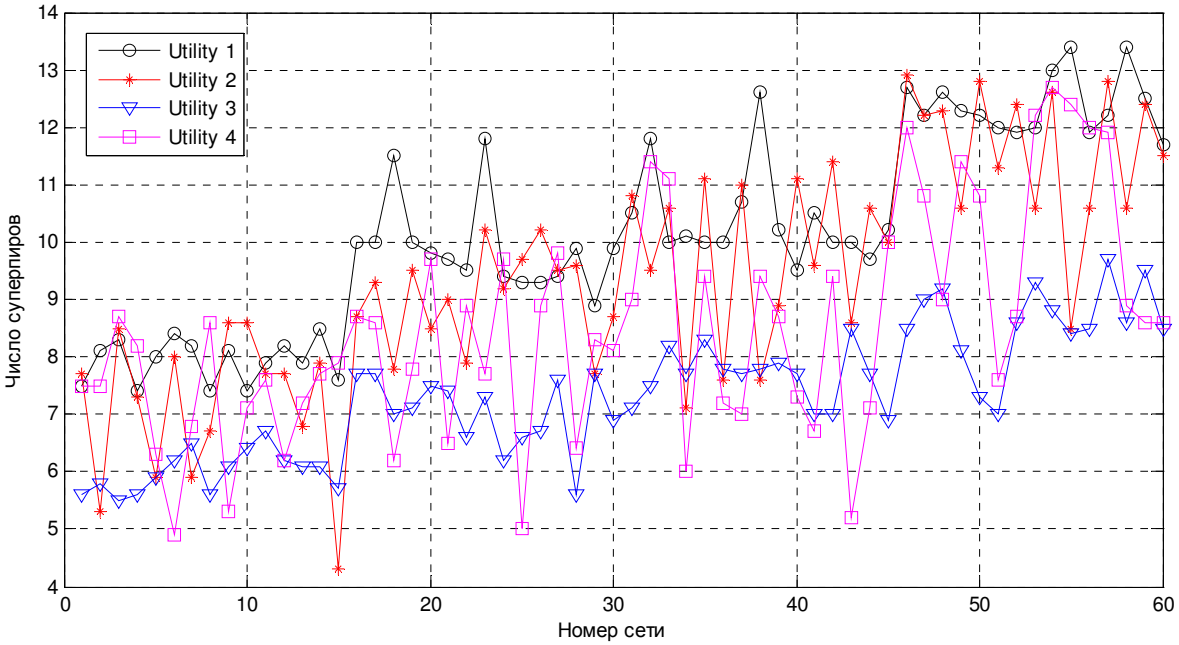


Рис. 1: Среднее число найденных суперпиров для сетей с кластерной топологией. Простой алгоритм заполнения.

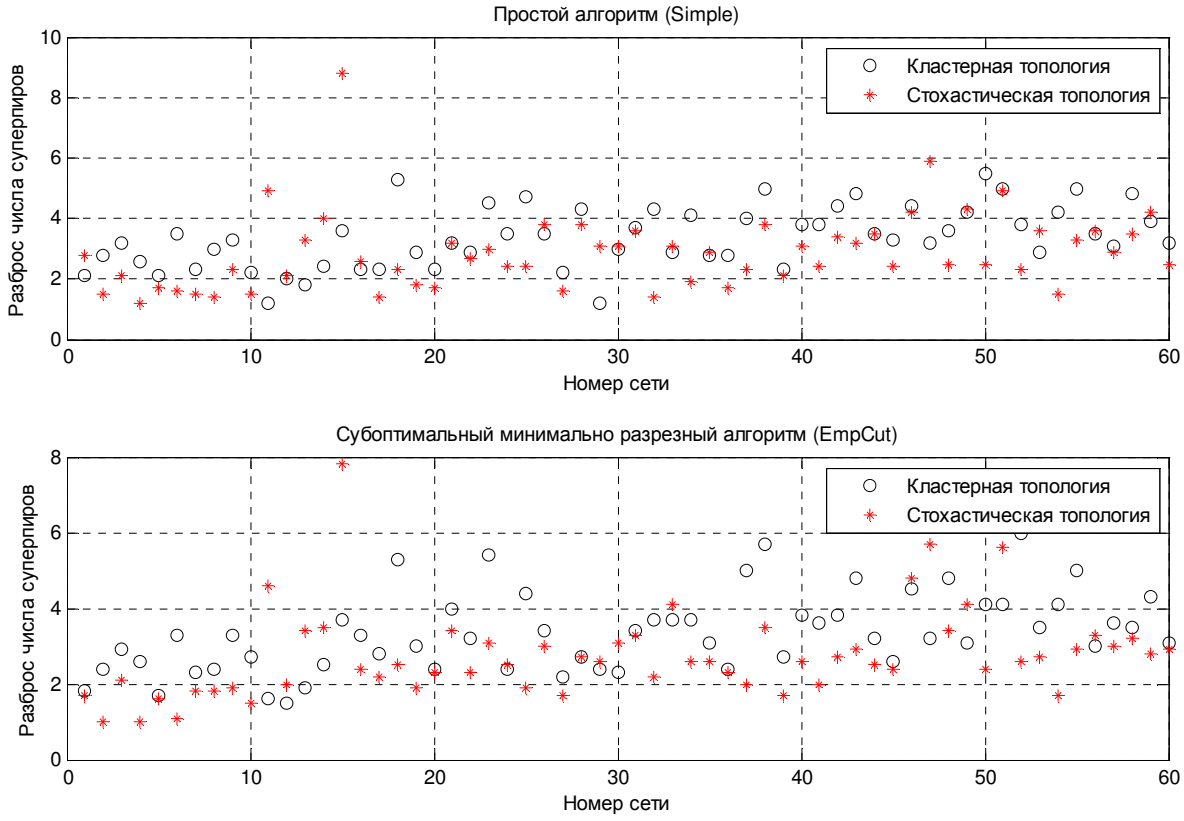


Рис. 2: Разброс числа суперпиров для сетей с кластерной и стохастической топологиями.

Таблица 1. Покрытие узлами кластеров в сетях с кластерной топологией

	Алгоритм	Простой	Дуговой	С.МинРаз	А.МинРаз	Гибридный	С.Дуговой
Разгруженность 1 (Utility 1)	Радиус 2	55%	11.7%	48.3%	50%	51.7%	31.7%
	Радиус 3	95%	90%	95%	95%	95%	95%
Разгруженность 2 (Utility 2)	Радиус 2	51.7%	53.3%	51.7%	51.7%	51.7%	51.7%
	Радиус 3	95%	95%	95%	95%	95%	95%
Разгруженность 3 (Utility 3)	Радиус 2	3.3%	1.7%	3.3%	3.3%	3.3%	3.3%
	Радиус 3	53.3%	55%	53.3%	55%	55%	56.7%
Разгруженность 4 (Utility 4)	Радиус 2	41.7%	40%	41.7%	41.7%	43.3%	41.7%
	Радиус 3	93.3%	93.3%	93.3%	93.3%	93.3%	93.3%

Таблица 2. Покрытие узлами кластеров в сетях со стохастической топологией

	Алгоритм	Простой	Дуговой	С.МинРаз	А.МинРаз	Гибридный	С.Дуговой
Разгруженность 1 (Utility 1)	Радиус 2	73.3%	66.7%	73.3%	75%	73.3%	63.3%
	Радиус 3	98.3%	98.3%	98.3%	98.3%	98.3%	98.3%
Разгруженность 2 (Utility 2)	Радиус 2	71.7%	76.7%	71.7%	71.7%	71.7%	73.3%
	Радиус 3	98.3%	98.3%	98.3%	98.3%	98.3%	98.3%
Разгруженность 3 (Utility 3)	Радиус 2	53.3%	53.3%	51.7%	51.7%	55%	53.3%
	Радиус 3	91.7%	93.3%	91.7%	88.3%	95%	91.7%
Разгруженность 4 (Utiltity 4)	Радиус 2	68.3%	65%	68.3%	68.3%	68.3%	66.7%
	Радиус 3	95%	95%	93.3%	95%	93.3%	93.3%

В третьем эксперименте оценивалась работа алгоритма на больших сетях со стохастической топологией (500 узлов). Плотность дуг колебалась от 70% до 30% дуг, присутствующих в полном графе. Вычислялось количество выделяемых суперпиров, образующих связную подсеть. Результаты моделирования показали, что число выделяемых суперпиров практически не коррелирует с плотностью дуг в графе, поэтому ниже будут приведены средние значения результатов.

Таблица 3. Среднее и дисперсия количества выделенных суперпиров, образующих связную подсеть на стохастических сетях.

	Доля суперпиров, %	Разгруженность 1 (Utility 1)	Разгруженность 2 (Utility 2)	Разгруженность 4 (Utility 4)
Среднее число	10	38.03	38.30	39.07
Дисперсия		6.48	6.17	6.02
Среднее число	30	125.79	124.78	125.53
Дисперсия		14.29	14.33	13.97

5. Выводы

Результаты первого эксперимента, полученные с использованием второй функции разгруженности узлов, учитывающей степень вершины и остаточные пропускные способности инцидентных дуг, дают результат со значительно меньшей дисперсией (малая дисперсия означает отсутствие частого изменения множества суперпиров и изменения маршрутизации потоков через них). Результаты первого эксперимента с использованием третьей функции разгруженности дают выделение малого числа суперпиров (3-4 суперпира на сеть), но также с малой дисперсией. Первые две функции разгруженности показали одинаковое число суперпиров, третья функция дала меньшее суперпиров, нежели требовалось. По результатам проведенных экспериментов показано, что в 63% сетей с кластерной топологией и 70% сетей

со стохастической топологией с помощью функций разгруженности (2) и (4) удалось получить связные подграфы суперпиров. Функции разгруженности (1), (2) и (4) выделяют требуемое число суперпиров, то есть треть от общего числа узлов в сети, одинаково на сетях с обоими типами топологий. Результаты в обеих таблицах показывают, что на сетях со стохастической топологией, доля полностью покрытых сетей кластерами узлов выше, чем на сетях с кластерной топологией.

Также было показано, что результат, а именно, число найденных суперпиров не зависит от плотности заполнения дугами графа сети, несмотря на то, что две метрики разгруженности зависят от такого показателя, как степень вершины.

Литература

[1] Я.Р. Гринберг, И.И. Курочкин: Исследование результатов математического моделирования последовательного заполнения сетей с кластерной топологией / Проблемы вычислений в распределенной среде: Труды ИСА РАН / Под ред. С.В. Емельянова, А.П. Афанасьева – Т.46 - М.: КРАСАНД, 2009, с.198-232.

[2] Я.Р. Гринберг, И.И. Курочкин: Математическое моделирование динамического последовательного заполнения сетей потоками связи / Проблемы вычислений в распределенной среде: Труды ИСА РАН / Под ред. С.В. Емельянова, А.П. Афанасьева – Т.46 - М.: КРАСАНД, 2009, с.233-258.

[3] Я.Р. Гринберг, И.И. Курочкин, А.В. Корх: Выделение кластеров в сетях с динамическим заполнением потоками связи / Труды II Всероссийской научной конференции молодых ученых с международным участием "Теория и практика системного анализа", 2012, т. 1, с.116-126.

АЛГОРИТМЫ УВЕЛИЧЕНИЯ СУММАРНОГО ТРАФИКА В СЕТЯХ ПЕРЕДАЧИ ДАННЫХ[1]

Я.Р. Гринберг

Центр Грид-технологий и распределенных вычислений ИСА РАН, Россия, 117312, г.Москва, проспект 60-летия Октября, д.9 , тел. (495) 718-96-31, greenjak@isa.ru

Предложен общий принцип для получения алгоритмов, увеличивающих суммарный трафик в сетях передачи данных в случае, когда заявки на организацию связи между абонентами поступают в сеть последовательно во времени. Он заключается в разделении ребер сети на классы дефицитности и выборе очередного маршрута на основании использования наименее дефицитных ребер. Описаны такие алгоритмы и приведены некоторые результаты ранее опубликованных численных экспериментов.

1. Введение

Настоящее исследование было инициировано следующей очевидной практической задачей. В телекоммуникационную сеть, состоящую из коммутационных узлов и линий связи между ними, объединяющую некоторое множество абонентов и имеющую ограничения на пропускные способности линий связи, последовательно во времени поступают заявки (требования) на передачу сообщений между парами абонентов. Как следует маршрутизировать эти сообщения, чтобы суммарное количество выполненных заявок было максимальным. Несмотря на кажущуюся простоту и очевидность такой постановки, в литературе нет работ, прямо посвященных решению этой проблемы.

В действующих протоколах маршрутизации сообщений в глобальных сетях передачи данных, таких, например, как RIP или OSPF, применяется принцип поиска пути минимальной стоимости. Стоимости (метрики) каждому каналу связи назначаются по-разному. Например, в RIP-протоколе каждой линии связи назначается одинаковая стоимость, таким образом, оптимальный путь в этом протоколе – это кратчайший по числу линий связи (хопов) путь. Такой алгоритм определения оптимального пути будем далее называть «простым» алгоритмом. OSPF-протокол допускает назначение разных метрик, например, на основании величин пропускной способности линий связи, (обратная пропорциональность), времени задержки, надежности и др. Ни одна из использующихся в настоящее время метрик не имеет своей прямой целью решение поставленной задачи, т.е. максимизацию суммарного потока. Настоящая работа представляет собой попытку предложить такие метрики.

Наиболее близкой к сформулированной задаче является так называемая многопродуктовая проблема, известная из теории потоков в сетях – см., например [1, 2]. Она формулируется как задача допустимости для организации потоков продуктов заданной интенсивности в сети с заданными ограничениями на пропускные способности линий связи. Правда, в такой классической постановке требования на интенсивность потоков продуктов между абонентами задаются сразу, одномоментно. Это отличие, тем не менее, оказывается весьма и весьма существенным. Алгоритмы маршрутизации, решающие задачу выбора очередного маршрута последовательно во времени, по мере поступления заявок, будем называть «последовательными» алгоритмами, в отличие от одномоментных или «синхронных» алгоритмов. Многопродуктовая задача допустимости имеет однозначное решение (т.е. «да» – возможен такой поток» или «нет» - невозможен»), в то время как выбор совокупности маршрутов, ее решающих, неоднозначен. Следует отметить, что специальных эффективных

[1] Поддержка ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы» (Госконтракт № 11.519.11.4012), РФФИ грант № 12-07-00415-а, программа фундаментальных исследований ПРАН № 14.

методов решения многопродуктовой проблемы не существует, но она может быть сформулирована как задача линейного программирования. Последовательные алгоритмы всегда «действуют» в рамках неопределенности, каковы именно будут следующие заявки, не говоря уже о всей совокупности требований в целом. Следует ожидать, что платой за это будет то, что они будут всегда «хуже» синхронного алгоритма. «Хуже» означает в данном случае следующее. Если в процессе действия какого-либо последовательного алгоритма случился «Отказ», т.е. невозможность удовлетворить очередную заявку, то решая задачу допустимости для всей совокупности заявок, включая «отказную», мы, скорее всего, получим положительный результат.

Отметим также наши работы [3, 4], посвященные этой проблеме. Можно сказать, что настоящее исследование является доосмыслением и, частично, переосмыслением представленных там идей. Тем не менее, модель заполнения сети осталась неизменной, а именно, она заключается в том, что в первоначально «пустую» сеть последовательно во времени поступают заявки на организацию каналов связи единичной пропускной способности между разными парами абонентов. Эти каналы связи имеют либо бесконечное время жизни («статический» режим), либо конечное время жизни, по окончании которого занятый ими ресурс освобождается («динамический» режим). Таким образом, на каждом такте задача нахождения пути решается в «новой» сети, которую будем называть остаточной. В ней пропускные способности ребер суть «остаточные» или «свободные» пропускные способности, т.е. те, которые не заняты другими, ранее проложенными путями. Отказ возникает тогда, когда в момент возникновения очередной заявки «остаточный ресурс» сети не обеспечивает ее связности, и никакой маршрут, связывающий данную пару абонентов, невозможен. Сравнение эффективности последовательных алгоритмов происходит следующим образом: экземпляры одной и той же «пустой» сети заполняются посредством различных последовательных алгоритмов при одном и том же потоке заявок и одинаковыми временами жизни путей (в динамическом режиме). Алгоритмы сравниваются по общему потоку до первого отказа и по количеству отказов.

2. Принцип экономии дефицитного ресурса

«Первоначальная» (или «пустая») ступенчатая сеть $\boldsymbol{G}^0(A^0,\ B^0,\ V^0,\ \mathcal{D}^0)$ представляет собой следующий математический объект.

1) $\boldsymbol{G}^0(A^0,\ B^0)$ – конечный, связный (неориентированный) граф, в котором A^0 – множество вершин, $|A^0|=N^0$, B^0 – множество ребер графа, которые мы будем называть сетевыми ребрами, $|B^0|=K^0$.

2) $C^0(B^0) \equiv\{c^0_1,\dots c^0_K\}$ - положительная действительная функция на множестве B^0, эта функция определяет первоначальные пропускные способности сетевых ребер.

3) V^0 – множество ребер графа, которые будем называть потоковыми ребрами или продуктами, $|V^0|=M^0$.

4) $\mathcal{D}^0$ – упорядоченное множество непересекающихся классов $\mathcal{D}^0{}_m$ сетевых ребер - классов дефицитности - $|\mathcal{D}^0|= D^0$, $\mathcal{D} \equiv \bigcup\limits_{m=1}^{D^0} \mathcal{D}^0{}_m$, $|\mathcal{D}^0{}_m|= K^0{}_m$, $\sum\limits_{m=1}^{D^0} K^0{}_m = K^0$, в котором классы расположены в порядке убывания дефицитности.

Вершины, инцидентные потоковым ребрам, будем называть полюсами, таким образом, каждый продукт – это пара полюсов. Вершины сети, не входящие в число полюсов, являются чисто коммутационными узлами сети.

«Первоначальная» (или «пустая») сеть $\boldsymbol{G}^0(A^0,\ B^0,\ V^0)$ – это сеть, обладающая всеми признаками ступенчатой сети, за исключением того, что в ней не определено множество классов дефицитности $\mathcal{D}^0$.

$\mathcal{R}_m$ - множества минимальных разрезов каждого продукта v_m, R_m – пропускные способности этих разрезов, равные, по теореме о максимальном потоке и минимальном разрезе, максимально возможным потокам соответствующих продуктов, для каждого продукта в отдельности.

Остаточная (по отношению к первоначальной) ступенчатая сеть $\boldsymbol{G}(G^0; A, B, V, \mathcal{D})$ и остаточная сеть $\boldsymbol{G}(G^0; A, B, V)$ определяются следующим образом.

1) $\boldsymbol{G}(G^0; A, B)$ - конечный (неориентированный) граф, в котором A – множество вершин, $A \subseteq A^0$ $|A| = N \le N^0$, B - множество ребер графа, $B \subseteq B^0$, $|B| = K \le K^0$; множества A и B определены ниже.

2) $C(B^0) \equiv \{c_1, \ldots c_{K^0}\}$ - неотрицательная действительная функция на множестве B^0, $c_i \le c_i^0$ $i=1,\ldots,K^0$.

3) $b_i \in B$, $\forall i\ c_i > 0$; $a_k \in A$ $\forall k$, a_k инцидентна $b \in B$.

4) $V \subseteq V^0$ $|V| = M \le M^0$; $v_m \in V$ $\forall m\ R_m > 0$.

5) $\mathcal{D}$ – упорядоченное множество непересекающихся классов $\mathcal{D}_m$ сетевых ребер множества B - классов дефицитности - $|\mathcal{D}| = D$, $\mathcal{D} \equiv \bigcup_{m=1}^{D} \mathcal{D}_m$, $|\mathcal{D}_m| = K_m$, $\sum_{m=1}^{D} K_m = K$, в котором классы расположены в порядке убывания дефицитности.

Для остаточных сетей, в т.ч. ступенчатых, те потоковые ребра v_m, для которых $R_m > 0$, т.е. $v_m \in V$, будем называть открытыми, остальные потоковые ребра $v_m \in V^0 \setminus V$ – закрытыми. Для первоначальных сетей, по определению, все потоковые ребра – открытые.

Пусть L – множество путей всех продуктов, $L \equiv \bigcup_{m=1}^{M} L_m$, где L_m - множество путей продукта v_m. Для каждого пути $\ell \in L$, рассматриваемого как множество ребер, образуем кортеж длины D, а именно, упорядоченную последовательность чисел $\ell_{(k)}$, $\ell \Rightarrow (\ell_{(1)}, \ldots \ell_{(D)})$, где $\ell_{(m)} = |\ell \cap D_m|$. Введем упорядоченность путей по следующему правилу:

$$\ell_1 \prec \ell_2 \Leftrightarrow (\exists i \le D\ \forall k<i\ \ell_{1(k)} = \ell_{2(k)} \wedge \ell_{1(i)} < \ell_{2(i)}),\ \ell_1, \ell_2 \in \mathcal{L}. \qquad (1)$$

Определение. Оптимальный путь продукта v_m в остаточной ступенчатой или пустой ступенчатой сети – это первый путь этого продукта в смысле упорядоченности (1).

Упорядоченность (1) – это лексикографическая упорядоченность. В контексте рассматриваемой задачи определенную выше оптимальность будем называть принципом экономии дефицитного ресурса.

Определение оптимального пути не изменится, если область поиска оптимального пути будет не множество $\mathcal{L}_m$, а любое множество путей.

Смысл введенных понятий становится совершенно понятен, если обратиться к описанной во введении модели заполнения сети. Каждый акт выбора очередного маршрута (кроме первого) представляет собой поиск оптимального пути в остаточной ступенчатой сети.

Оптимальные пути можно находить как пути минимальной стоимости, если соответствующим образом назначить стоимости ребрам сети – одинаковые стоимости для ребер, принадлежащих одному классу дефицитности.

Алгоритмы поиска пути минимальной стоимости, приводящие к оптимальным путям по принципу экономии дефицитного ресурса, будем называть контрастными алгоритмами. Вместе

с тем, если стоимости ребер представляют собой убывающую по мере убывания дефицитности классов последовательность положительных чисел, то соответствующие алгоритмы также в некоторой степени, «приближенно» будут удовлетворять принципу экономии дефицитного ресурса. Такие алгоритмы будем называть «мягкими» алгоритмами.

3. Алгоритмы последовательного заполнения

Пусть имеется первоначальная $\boldsymbol{G}^0(A^0, B^0, V^0)$ или остаточная $\boldsymbol{G}(G^0; A, B, V)$ сеть. Будем считать, что требования на организацию потоков продуктов подчиняются вероятностному закону, а именно, существуют M положительных чисел p_m, $m = 1, 2,\ldots, M$, таких, что $\sum_{m=1}^{M} p_m = 1$. Эти числа представляют собой вероятности появления требований на организацию каналов связи между полюсами, образующими потоковые ребра v_m. Мы предполагаем также, что удовлетворение заявки означает организацию потока единичной интенсивности.

Пусть $\overline{R} = \frac{1}{M}\sum_{m=1}^{M} R_m$ - среднее значение пропускных способностей минимальных разрезов, соответствующих открытым потоковым ребрам.

Опишем минимально-разрезный алгоритм определения классов дефицитности сетевых ребер и две его модификации.

Шаг 1. Будем считать множества $\mathcal{R}_m$ упорядоченными по признаку неубывания величин R_m.

Шаг 2. Введем величины вида

$$q_m = (1/M)(R_m / \overline{R}) - p_m \qquad (2)$$

и также расположим их в порядке неубывания. Обозначим классы равных величин q_m как Q_i, $i = 1, \ldots, M_1$, $M_1 \le M$, и будем также считать их упорядоченными по возрастанию величин q. Будем также обозначать $Q_i \equiv Q(q_i)$.

Шаг 3. Пусть сетевое ребро b входит в минимальные разрезы одного или нескольких продуктов. Обозначим через $q_i(b)$ то количество продуктов, которое входит в класс Q_i и в минимальных разрезах которых содержится ребро b. Образуем кортеж $T(b)$ длины $M_1 + 1$ $T(b) \equiv (q_1(b), q_2(b),\ldots, q_{M_1}(b), q_{M_1+1}(b)=0)$, где под классом Q_{M_1+1} мы понимаем класс ребер, не входящих ни в один минимальный разрез. (Таким образом, если $b \in \boldsymbol{Q}_{M_1+1}$, то $T(b) \equiv (0,\ldots,0, q_{M_1+1}(b)=1)$.

Шаг 4.1. Пусть $T(b_1) \equiv ({}_1q_i)$, $T(b_2) \equiv ({}_2q_i)$, $i = 1,\ldots, M_1 + 1$. Установим упорядоченность сетевых ребер по следующему правилу.

$$b_1 \prec b_2 \Leftrightarrow (\exists\, i \le M_1 + 1\ \forall k<i\ \ {}_1q_k = {}_2q_k \wedge {}_1q_i < {}_2q_i). \qquad (3)$$

Шаг 4.2. Пусть $d(b) = \sum_{i=1}^{M_1} q_i(b)$ и, если $d(b) > 0$, то редуцированный кортеж $T'(b)$ определим как $T'(b) \equiv (0, \ldots, 0, q_i(b)=d, 0, \ldots, 0)$, где i номер элемента нередуцированного кортежа ребра b, для которого $q_k(b) = 0, k < i$ и $q_i(b) > 0$. Установим упорядоченность сетевых ребер по правилу (3) для редуцированных кортежей $T'(b)$.

Шаг 4.3. Пусть $d(b) > 0$. Тогда для ребра b определим дважды редуцированный кортеж $T''(b)$ следующим образом: $T''(b) \equiv (0,\ldots,0,\ q_i(b)=1,0,\ldots,0)$, где i номер элемента кортежа ребра b, для которого $q_k(b) = 0, k < i$ и $q_i(b) > 0$.

Для ребер с $d(b) > 0$ установим упорядоченность сетевых ребер по правилу (3) для дважды редуцированных кортежей $T''(b)$.

Сделаем несколько комментариев.

1. Величины q_m представляют собой «меру относительной загруженности» сети потоком между данной парой полюсов при условии, что интенсивности потоков между всеми парами полюсов распределены в соответствии с величинами вероятностей p_m. В соответствии с введенными определениями

$$\sum_{m=1}^{M} q_m = (1/M)\sum_{m=1}^{M}(R_m/\bar{R}) - \sum_{m=1}^{M} p_m = 1-1 = 0$$

и, таким образом, либо все q_m равны нулю, либо среди этих величин есть положительные и отрицательные. Смысл введения этих величин - в ранжировании ребер сети не только по величинам пропускной способности минимальных разрезов, но и по относительной интенсивности заявок. Здесь неявно предполагается, что каждое ребро сети может «участвовать» в минимальных разрезах только одного продукта, что, конечно, для большинства сетей не выполняется. Тем не менее, мы будем исходить из того, что, пусть и не в полной мере, но эти величины сохраняют указанный смысл. Наличие отрицательных величин q_m не несет никакой дополнительной смысловой нагрузки, а является лишь следствием выбранной нормировки величин.

2. В общем случае последовательность чисел R_m, расположенных в порядке возрастания, совсем необязательно совпадает с последовательностью чисел q_m. Однако в том частном случае, когда распределение вероятностей возникновения заявок равномерно относительно всех потоковых ребер, т.е. $p_m = 1/M,\ m = 1,\ldots,\ M$, во введении величин q нет необходимости, и упорядочивать и агрегировать следует множество продуктов по величинам пропускных способностей минимальных разрезов R_m.

3. Если в остаточную сеть поступила заявка на организацию канала связи для закрытого потокового ребра, то фиксируется отказ, и процесс заполнения сети переходит к следующему шагу.

Если в качестве основы для упорядочивания ребер сети принять не принадлежность их к тому или иному минимальному разрезу, а значения их пропускной способности, то полученные алгоритмы будем называть реберными алгоритмами.

4. Результаты некоторых экспериментов

На Рис.1 приведены некоторые результаты применения последовательных алгоритмов для заполнения потоковых сетей потоками связи, взятые из работы [5]. На гистограммах представлены результаты заполнения серии из 131 сети потоками связи по четырем последовательным алгоритмам в «статическом» режиме, т.е. в случае, когда время жизни канала связи бесконечно. По оси абсцисс отложено относительное (в процентах) увеличение (или уменьшение) проведенного потока по всем продуктам посредством данного последовательного алгоритма по критерию «до первого отказа» по сравнению с простым алгоритмом. По оси ординат – количество сетей из данной серии, в которых это превышение имело место. Охарактеризуем в терминах настоящей работы применявшиеся в [5] алгоритмы. Алгоритм *Arc* – контрастный реберный алгоритм, алгоритм *Subopt* – мягкий минимально-разрезный алгоритм с функцией стоимостей ребер $W(x) = N - 1 + (D - x + 1)^4$, где x-номер класса дефицитности, алгоритм *Addopt* – эвристический алгоритм, полученный вне

рамок принципа экономии дефицитного ресурса, алгоритм *Hybrid* – контрастный минимально-разрезный редуцированный алгоритм по типу 4.3. Оставаясь на качественном уровне, можно сказать, что во многих случаях применение алгоритмов, полученных из принципа экономии дефицитного ресурса, приводит к лучшим результатам, чем при использовании простого алгоритма, причем для некоторых сетей это улучшение может составлять 30 – 50 процентов.

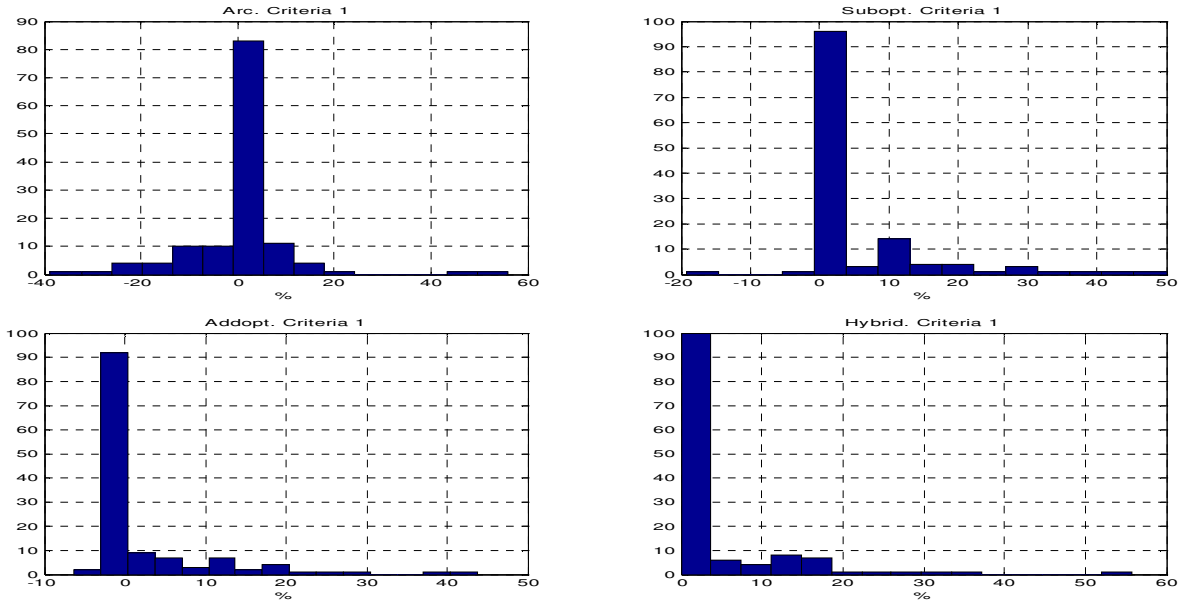


Рис.1

5. Благодарности

В течение всего времени работы я испытывал постоянную помощь и поддержку со стороны д.ф.-м.н. А.П.Афанасьева и к.ф.-м.н. А.М.Раппопорта, которым я приношу свою искреннюю благодарность.

Литература

[1] Л.Форд, Д.Фалкерсон, Потоки в сетях, М.,МИР,1966.

[2] Т.Ху, Целочисленное программирование и потоки в сетях, М.,МИР,1974.

[3] А.П. Афанасьев, Я.Р. Гринберг, И.И. Курочкин. «Равномерные» алгоритмы последовательного заполнения потоковой сети потоками продуктов. В кн. Проблемы вычислений в распределенной среде: Сборник трудов ИСА РАН. М.: ООО «КомКнига», стр. 118-140, 2005.

[4] А.П. Афанасьев, Я.Р. Гринберг, И.И. Курочкин. Сравнительный анализ двух последовательных алгоритмов заполнения сети потоками продуктов. В кн. Труды конференции САИТ-2005, т.2, стр. 136-140, 2005.

[5] Я.Р.Гринберг, И.И.Курочкин. Анализ результатов численного эксперимента по последовательному заполнению сетей со стохастической топологией// Проблемы вычислений в распределенной среде: распределенные приложения, коммуникационные системы, математические модели и оптимизация: Сборник трудов ИСА РАН / Под ред. А.П. Афанасьева – Т.25 - М.: КомКнига, с.99-128, 2006.

АРХИТЕКТУРА КОММУНИКАЦИОННОЙ СРЕДЫ СУПЕРКОМПЬЮТЕРОВ СЛЕДУЮЩЕГО ПОКОЛЕНИЯ И ТЕОРИЯ ПРОСТРАНСТВЕННО-ВЛОЖЕННЫХ СЛОЖНЫХ СЕТЕЙ[1]

А.П. Демичев[1,2], В.А. Ильин[1,2], А.П. Крюков[1,2], С.П. Поляков[2]

[1]*Национальный исследовательский центр "Курчатовский институт"*
Россия, 123182 Москва, пл. Академика Курчатова, д. 1

[2]*Научно-исследовательский институт ядерной физики имени Д.В. Скобельцына*
Россия, 119991, ГСП-1, Москва, Ленинские горы, д. 1, стр. 2

В работе предлагается подход к разработке коммуникационных сетей суперкомпьютеров следующего поколения. Рассмотрены алгоритмы построения сложных сетей со свойствами "малого мира", а именно, медленным (логарифмическим) ростом среднего расстояния между узлами при увеличении их числа. При этом сети, построенные на основе этих алгоритмов, имеют базовую структуру регулярной решетки с дополнительными перемычками между узлами, которые и обеспечивают свойства "малого мира". Поскольку вычислительные узлы располагаются в физическом трехмерном пространстве, и топологические аспекты сети коррелируют с пространственными аспектами, для анализа используются методы теории пространственно-вложенных сложных сетей.

We propose an approach to the interconnection network design for the next generation supercomputers. Algorithms of small-world complex networks construction with slow (logarithmic) growth of average distance between nodes are considered. The constructed networks have basic structure of the regular lattice with additional shortcuts which provide the small-world properties. Since the computing nodes are arranged in real three-dimensional space and topological aspects of the networks should correlate with spatial aspects, the methods of the theory of spatially embedded complex networks are used.

1 Введение

После того как был достигнут петафлопсный (10^{15} FLOPS) барьер производительности суперкомпьютеров, перед разработчиками высокопроизводительных вычислительных систем встал вопрос о принципах построения систем следующего поколения — с производительностью порядка экзафлопса (10^{18} FLOPS) [1]. Хотя появление реальных вычислительных систем такого уровня ожидается не ранее 2018–2020 года, подходы и принципы их построения начинают интенсивно разрабатываться уже сейчас, поскольку на пути к их построению предстоит решить ряд сложных научно-технических задач и выработать принципиально новые решения для их архитектуры и аппаратной реализации.

Одной из важнейших составляющих любого суперкомпьютера является коммуникационная сеть, которая в первую очередь определяет возможность увеличения числа вычислительных узлов, что необходимо для достижения желаемой производительности. Как ожидается (см., например, [2]), суперкомпьютеры экзафлопного уровня будут иметь порядка 100 000 и более вычислительных узлов. Таким образом, одной из важнейших задач, которую предстоит решить на пути к построению суперкомпьютеров следующего поколения, является разработка коммуникационных сетей с хорошими свойствами масштабируемости и возможностью эффективно обслуживать огромное число вычислительных узлов.

Тремя основными аспектами проектирования коммуникационных сетей, которые в наибольшей степени определяют их функциональные свойства, являются:

[1] Работа частично финансируется РФФИ, грант 12-07-00408-а.

- топология сети (network topology);
- метод управления потоками (flow control); иногда используют термин ``метод переключения (switching method)";
- алгоритм маршрутизации (routing algorithm).

В данной работе обсуждается, в основном, именно топология коммуникационной сети (в том смысле, в котором термин "топология" используется в теории сетей). Два других аспекта очень важны, но выходят за рамки текущего обсуждения. Выбор подходящей топологии жизненно важен для проектирования сети, поскольку маршрутизация и механизмы управления потоком в большой степени основаны на ее свойствах.

В идеальном случае коммуникационная сеть должна была бы быть полностью соединена (полный граф), чтобы позволить одновременную непосредственную связь между всеми парами узлов, достигая оптимальной пропускной способности и задержки. Этот подход может быть применен к системам с немногими узлами, но он не масштабируется на большие сети, так как число связей для каждого узла было бы равно числу всех узлов сети минус единица. Пропускная способность сети должна масштабироваться с ростом числа процессоров, что обеспечивается правильной комбинацией хорошего выбора топологии и алгоритмов маршрутизации.

Существуют два общих типа сетей:

- прямые сети (direct networks), в которых каждый узел является терминальным, действуя и как источник, и как приемник для сообщений, а также и как рутер для управления входящими сообщениями;
- непрямые сети (indirect networks) содержат "нетерминальные" узлы (рутеры), которые используются только для маршрутизации.

Непрямые сети имеют свои достоинства для ограниченного числа узлов, но плохо масштабируются. Поэтому мы будем рассматривать только прямые сети. Более точно, мы будем рассматривать обобщения регулярных решеток с топологией D-мерных торов. В литературе, посвященной сетям, для такой топологии часто используется термин "k-ary n-cube" [3] (n — размерность тора, в наших обозначениях $n = D$). При этом каждый узел, как в любой прямой сети, является рутером. Известно, что такие сети при большом числе узлов имеют преимущества по сравнению с другими архитектурами, например, гиперкубами высоких размерностей (см., например, [4]).

Важным аргументом в пользу использования регулярных решеток является тот факт, что на такую структуру коммуникационной сети естественным образом отображаются параллельные вычислительные задания, связанные с численным моделированием D-мерных объектов. В частности, коммуникационные сети со структурой трехмерной решетки оптимальны для моделирования трехмерных реальных объектов, а именно такого типа задачи, как предполагается, будут составлять значительную долю задач, решаемых на суперкомпьютерах экзафлопного уровня.

Однако при огромном числе узлов, характерном для компьютеров следующего поколения, архитектура регулярных решеток с топологией D-мерных торов имеет и существенные недостатки. В частности, решетки невысокой размерности имеют весьма большую среднюю длину пути между узлами, а решетки высокой размерности, сравнимой с логарифмом числа узлов, трудно реализовать технически из-за большой длины физических коммуникационных каналов. С другой стороны, известно, что наилучшими структурами вычислительных систем по различным критериям функционирования, например, производительности и надежности, при одинаковом числе вычислительных узлов и каналов связи являются структуры с минимальным средним расстоянием между узлами (см., например, [5]). Поэтому обычные сети с простой структурой регулярных решеток окажутся недостаточно эффективными для решения задач более общего типа, не связанных с триангуляцией трехмерных объектов.

В связи с этим представляется перспективным использовать для построения

коммуникативных сетей для экзафлопсных компьютеров сети со свойствами "малого мира" [6], одним из важнейших свойств которых является малое среднее расстояние между узлами и малый диаметр сети. Более точное выражение этого свойства заключается в следующем: для регулярной D-мерной решетки среднее расстояние d между узлами растет как степень числа узлов: $d \sim N^{1/D}$, а для сети со свойствами малого мира существенно медленнее: $d \sim \ln N$. Заметим, что в некоторых сложных сетях среднее расстояние зависит от числа узлов степенным образом, как и в случае регулярных решеток, однако показатель степени существенно меньше: $d \sim N^{\gamma}$, $\gamma \ll 1/D$. Зачастую исследования свойств сложных сетей осуществляются с помощью численного моделирования, и отличить такое степенное поведение от логарифмического весьма непросто. С точки зрения использования таких сетей в прикладных целях при достаточно малом показателе отличие тоже является несущественным. Поэтому можно считать, что такие сети также обладают свойствами малого мира (в широком смысле).

В классическом варианте [6] сложные сети со свойствами малого мира получаются на промежуточной стадии в процессе стохастической трансформации регулярных решеток в полностью случайные графы Эрдеша-Реньи [7], [8]. При этом структура регулярной решетки нарушается, что, как отмечалось выше, нежелательно для коммуникационных сетей экзафлопсных компьютеров. Поэтому в данной работе предлагается использовать модификацию способа построения сетей с малой средней длиной пути между узлами, при которой сохраняется базовая решеточная структура, но к ней определенным образом добавляются дополнительные связи, называемые перемычками, которые и обеспечивают свойства малого мира. Схематический вид таких сетей в одномерном и двумерном случаях представлен на рис.1. Для краткости мы в дальнейшем будем использовать для таких сетей термин "решеточные сети с перемычками" (РСП).

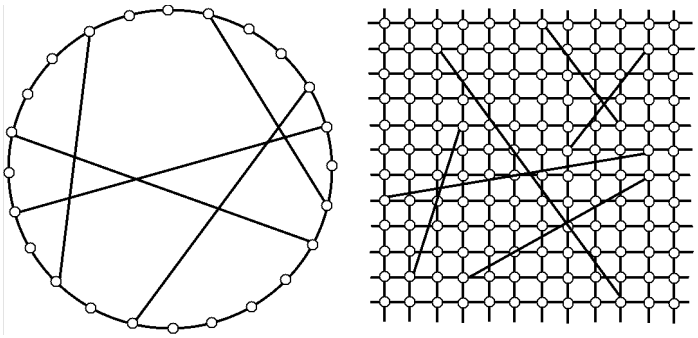


Рисунок 1: Схематический вид решеточных сетей с добавленными перемычками в одномерном (слева) и двумерном (справа) случаях

Необходимо отметить, что помимо малой средней длины пути между узлами, еще одним общим отличительным свойством сетей со свойствами малого мира является высокая степень кластеризации [6], [8]. Высокая кластеризация обеспечивает локальную устойчивость сети: существование локальных обходных путей при выходе из строя какого-либо узла сети. Однако в нашем случае такую локальную устойчивость (существование локальных обходных путей) для $D > 1$ обеспечивает решеточная основа, и поэтому мы не будем обсуждать это свойство малого мира в данной работе.

Длина пути (расстояние) между узлами понимается в сетевом смысле: как минимальное число ребер, по которым надо пройти, чтобы попасть из одного узла в другой. Соответственно среднее расстояние между узлами определяется как среднее по всем парам узлов данной сети. Однако для больших сетей, какой ожидается коммуникационная сеть экзафлопсных компьютеров, определенная таким образом длина пути между узлами может оказаться неадекватной характеристикой, поскольку для нахождения кратчайших маршрутов необходимо знать глобальную структуру сети. Соответственно маршрутизация сообщений, использующая кратчайшие пути, может оказаться слишком сложной и неэффективной, так как связана с

хранением и обработкой большого объема информации. Поэтому особую важность приобретают алгоритмы маршрутизации, основанные на локальной навигации [9], [10]. Задача локальной навигации в сетях ставится следующим образом: сообщение передается от узла к узлу по ребрам (связям); узел "знает" географическое положение (другими словами, положение в базовой решетке) всех узлов и своих *ближайших* сетевых соседей с учетом перемычек (может знать соседей на глубину больше единицы; но информация обо всех перемычках в сети не используется); необходимо доставить сообщение в узел назначения по возможно кратчайшему пути. В простейшем варианте эту задачу решает так называемый жадный алгоритм (англ. greedy algorithm; иногда называется также алгоритмом экономного продвижения): текущий узел пересылает сообщение тому из своих соседей, который географически (то есть в смысле координат на решетке) ближе всего к цели (узлу назначения).

Таким образом, в данной работе мы исследуем среднюю глобальную и среднюю навигационную длины пути между узлами сети, как важнейшие характеристики, определяющие коммуникационные свойства сети. Основной целью работы является разработка оптимального алгоритма построения сети с большим числом узлов, но малой средней глобальной или навигационной длиной пути между узлами. Общая идея состоит в добавлении к решеточной основе дополнительных перемычек по специальному алгоритму (или алгоритмам), так, чтобы оптимизировать соотношение "цены" и "качества" для получаемой таким образом сети. В качестве "цены" выступает удельная длина дополнительных перемычек (общая длина перемычек в единицах базовой решетки, деленная на число узлов сети), а "качество" — это глобальная или навигационная средняя длина пути между узлами.

2 Оптимизация параметров коммуникационной сети

Как уже упоминалось, оригинальный алгоритм получения сложной сети со свойствами малого мира [6] является стохастическим: на каждом шаге алгоритма ребра графа меняют свое положение с некоторой вероятностью. В результате многократного применения такого алгоритма возникает ансамбль графов с некоторым распределением их характеристик, в частности, с некоторым распределением средней длины пути между узлами экземпляра графа. Для многих реальных сетей стохастический процесс их образования оказывается внутренне присущим (так, это справедливо для сети Интернет; другие примеры стохастических сетей см., например, в [8]). Проектирование коммуникационной сети суперкомпьютера находится под контролем разработчика, и стохастичность не является внутренне присущим элементом этого процесса. Существует ряд работ [11] – [15], в которых предложены детерминистские алгоритмы построения сетей со свойствами малого мира. Предварительное исследование этих сетей показывает, что по соотношению "цены" и "качества" они проигрывают стохастическим сетям, особенно если показателем качества является навигационная длина. В любом случае свойства сетей, полученных с помощью детерминистских алгоритмов, должны быть сопоставлены с лучшими экземплярами сетей, которые можно получить с помощью стохастических алгоритмов и компьютерного моделирования. В данной работе мы ограничимся исследованием стохастических алгоритмов и разработкой методологии сравнения различных сетей. Кроме того, хотя часть полученных результатов справедлива для решеток произвольной размерности, в данной работе рассматривается простейший случай одномерной решетки с топологией окружности как на рис. 1 слева. Основными причинами для этого являются:

- удобство отработки различных алгоритмов и сравнения свойств получающихся сетей (сокращение времени численного моделирования);
- исследования свойств одномерных сетей могут представлять непосредственный прикладной интерес: например, при использовании так называемой DOR-маршрутизации (Dimension Ordered Routing; см., например, [16]), при котором сообщения двигаются вдоль каждого измерения независимо от других измерений, и это движение определяется свойствами одномерной (модифицированной) решетки.

Обобщения на решетки более высоких размерностей и результаты сравнения с сетями,

получаемыми с помощью детерминистских алгоритмов, будут представлены в последующих работах.

Стохастический алгоритм построения сетей малого мира с сохранением базовой решетки предложен в ряде работ, в частности, в работах [17] (см. также обзор [18] и ссылки в нем). В одномерном случае он формулируется следующим образом.

Алгоритм S1: (1) исходным объектом является одномерная решетка с L узлами и топологией окружности; (2) последовательно перебираются все узлы решетки и с вероятностью $0 < p \leq 1$ к каждому узлу i подсоединяют первый конец перемычки; (3) второй конец перемычки (то есть, узел решетки j, в который она входит) не может совпадать с соседями исходного узла в смысле базовой решетки и приводить к дублированию уже существующей перемычки, а в остальном выбирается случайно с вероятностью $P(r) \sim r^{-\alpha}$, которая является степенной функцией решеточного расстояния $r=r_{ij}$ между узлами.

Зависимость вероятности перемычки от расстояния между узлами отражает корреляцию между топологическими и пространственными свойствами сети [18].

При использовании алгоритма S1 построения РСП управляющими параметрами или, другими словами, параметрами, характеризующими ансамбль (аналог температуры для обычного канонического ансамбля в статистической физике), являются параметры L, p, α. Соответственно, эти параметры должны быть оптимизированы при построении сети на основе этого алгоритма.

При исследовании возможной архитектуры коммуникационной сети (которой внутренне не присуща стохастичность появления перемычек) может быть удобно использовать следующую модификацию алгоритма S1.

Алгоритм S1m: (1) исходным объектом является одномерная решетка с L узлами и топологией окружности (как в S1); (2) фиксируется число перемычек t, которые должны быть добавлены к решетке; (3) из всех L узлов решетки случайным образом выбираются t узлов, к которым подсоединяются первые концы перемычек; (4) второй конец каждой из t перемычек выбирается случайно и полностью идентично шагу 3 алгоритма S1.

При использовании алгоритма S1m построения РСП управляющими параметрами являются параметры L, t, α. Отличие от базового алгоритма S1 заключается в том, что в если в базовом алгоритме случайным является как расположение, так и число перемычек, в алгоритме S1m случайным является только положение перемычек. Другими словами, степень стохастичности алгоритма уменьшается за счет сужения вероятностного пространства событий (конкретных реализаций РСП). С другой стороны, объединение вероятностных пространств алгоритма при всех допустимых значениях t эквивалентно объединению пространств событий базового алгоритма при допустимых значениях параметра p. Поэтому поиск оптимальной РСП, построенной по алгоритмам S1 и S1m, при достаточно репрезентативной выборке должен приводить к одинаковым результатам.

Можно попытаться еще уменьшить степень стохастичности алгоритма построения РСП следующим образом.

Алгоритм S2: (1) исходным объектом является одномерная решетка с L узлами и топологией окружности (как в S1 и S1m); (2) фиксируется общее число перемычек t, которые должны быть добавлены к решетке (как в алгоритме S1m) и фиксируется число $c < t$ перемычек, которые будут добавлены специальным образом; (3) из всех L узлов решетки случайным образом выбираются t-c узлов, к которым подсоединяются первые концы перемычек; (4) второй конец каждой из t-c перемычек выбирается случайно и полностью идентично шагу 3 алгоритма S1; (5) c перемычек добавляются специальным образом; перемычки добавляются последовательно одна за другой, при добавлении каждой перемычки: составляется список узлов, к которым уже присоединены перемычки (как перемычки, построенные на предыдущих шагах 2 – 4, так и новые, уже построенные на этом шаге перемычки); из этого списка узлов случайным равновероятным образом выбирается один и к нему подсоединяется новая перемычка; второй конец дополнительных перемычек выбирается так же как на шаге 4 (при этом новая перемычка не должна совпадать с предыдущими и с

ребром базовой решетки).

При использовании алгоритма S2 построения РСП управляющими параметрами или, другими словами, параметрами, характеризующими ансамбль, являются параметры *L, t, c, α*. В алгоритме S2 не только число перемычек детерминировано, но положение их не вполне случайно, а подчиняется некоторым дополнительным правилам, причем степень этой регулярности определяется параметром *c* (при *c = 0* алгоритм S2 совпадает с S1m). Как показывает численное моделирование, среднее расстояние между узлами РСП S2 в широком диапазоне значений параметра *c* меньше, чем среднее расстояние в РСП, построенных по алгоритмам S1 и S1m.

Среднее сетевое расстояние *d* между узлами получается после двойного усреднения: по стохастическому процессу создания перемычек (другими словами, по статистическому ансамблю случайных графов, полученному в результате стохастического процесса образования перемычек) и по всем парам узлов графа. Характерное поведение средней длины в зависимости от параметров ансамбля из 1000 РСП, построенного с помощью алгоритма S1m, полученное нами с помощью численного моделирования для сети, состоящей из $L = 10000$ узлов, показано на рис. 2.

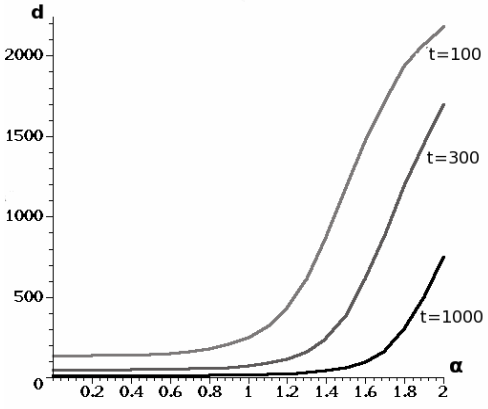


Рисунок 2: Средняя длина в зависимости от параметров ансамбля РСП (алгоритм S1m)

При $\alpha > 2$ средняя длина перемычек оказывается слишком малой и они не оказывают существенного влияния на свойства сети, так что свойства малого мира исчезают [17]. При наивном рассмотрении результатов, представленных на рис. 2 (для алгоритма S1 и параметра *p* зависимость аналогична), можно прийти к заключению, что для получения сети с наилучшей характеристикой расстояния между узлами надо просто выбирать наибольшее значение параметра *t* (или *p*) и наименьшее значение *α*. Однако, как отмечалось выше, каждое решение имеет свою "цену" и необходимо оптимизировать соотношение "цены" и "качества" для получаемой сети (иначе очевидным решением является просто полный граф, в котором все узлы соединены прямыми связями). Поскольку в данной работе в качестве "цены" выступает удельная длина дополнительных перемычек, необходимо рассматривать зависимость среднего расстояния от этой величины. Характерные результаты численного моделирования для такой зависимости приведены на рис. 3.

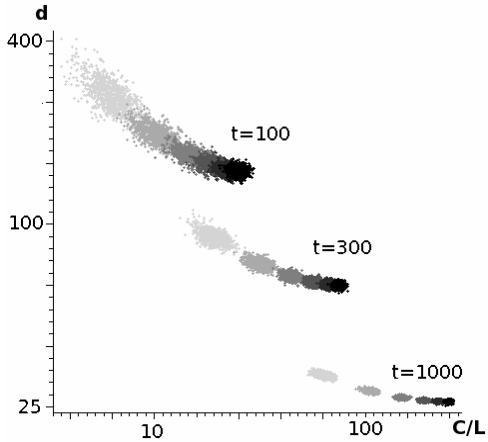


Рисунок 3: Зависимость распределений средних по экземплярам расстояний *d* между узлами от удельной длины перемычек

Каждая точка с соответствующей градацией серого цвета на рис. 3 соответствует экземпляру РСП в ансамбле со значениями α=0, 0.2, 04, 0.6, 0.8, 1.0 (чем больше значение α, тем светлее оттенок серого цвета) и *t* = 100, 300, 1000, ось ординат соответствует среднему расстоянию между узлами, а по оси абсцисс отложена реальная удельная длина перемычек *C/L* (*C* — общая длина перемычек в единицах базовой решетки; *L* — число узлов сети), полученная в результате численного моделирования для данного экземпляра ансамбля (по обеим осям использован логарифмический масштаб). Для алгоритма S1 получены аналогичные результаты, но с более широким, как и

ожидалось, распределением расстояний по ансамблю. Из представленных данных видно, что при одинаковой цене для уменьшения средней длины надо выбирать ансамбль с возможно большим значением t (или p), и, одновременно, с наибольшим возможным значением α (пока это возможно для данной удельной длины перемычек). Другими словами, большое количество не слишком длинных перемычек является более дешевым решением. Это контрастирует с результатами для простой зависимости средней длины от параметров p и α, которые при "наивном" рассмотрении могут привести к заключению, что надо просто выбирать наименьшее значение α.

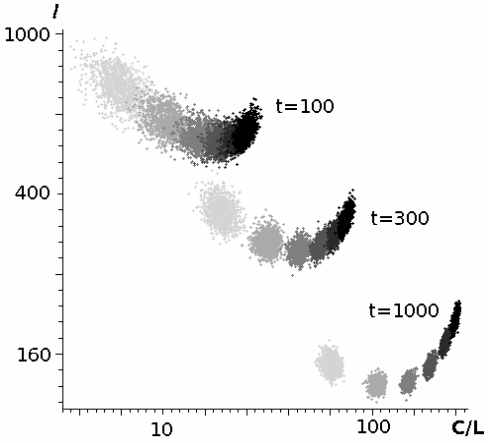


Рисунок 4: Зависимость распределений средних по экземплярам навигационных длин l от удельной длины перемычек

Как отмечено во Введении, с точки зрения оптимизации параметров стохастических РСП представляет интерес выяснение зависимости навигационной длины от параметров алгоритмов. На рис. 4 показаны распределения средней (по экземплярам) навигационной длины l в зависимости от удельной длины перемычек и параметра α при различных значениях t для алгоритма S1m. В случае алгоритма S1 поведение оказывается аналогичным при замене t на соответствующее значение параметра p. Видно, что при фиксированном числе перемычек t существует минимум l, достигаемый при различных значениях α, зависящих от t. Как и можно было ожидать, чем большее число перемычек добавляется к решетке, тем меньшая средняя длина получается в минимуме по α. Видно также, что для заданной "цены" можно подобрать параметры алгоритма S1m с наименьшей навигационной длиной, или, наоборот, для заданной навигационной длины можно подобрать параметры алгоритма S1m с наименьшей удельной длиной перемычек. Поэтому и в этом случае надо решать оптимизационную задачу "цена – качество".

Представленные детальные данные показывают, что использование только зависимостей качества от цены (средней длины от удельной длины перемычек) не позволяет однозначно выбрать оптимальную сеть, то есть, в случае алгоритмов S1 или S1m, выбрать оптимальные параметры p, t, α. Поэтому надо использовать методы так называемой многокритериальной оптимизации.

Достаточно часто в реальных ситуациях качество эксплуатации исследуемого объекта или системы оценивается не единственным критерием или показателем качества, а совокупностью таких критериев. Такая постановка задачи приводит к задаче оптимизации с векторной целевой функцией, которая должна трактоваться неким определенным образом. В нашем случае мы будем учитывать два показателя: среднюю длину пути между узлами (глобальную d или навигационную l), что выражает "качество" РСП, и удельную длину перемычек C/L, что отражает "цену" построения РСП. Очевидно, что эти две величины взаимосвязаны: увеличивая цену C/L можно улучшить качество (уменьшить d или l), и, наоборот, улучшение качества зачастую связано с увеличением цены. Существует ряд подходов и методов для решения задач такой многокритериальной оптимизации (см., например, [19]). Мы будем использовать один из простейших и наглядных методов, а именно, метод взвешенных сумм (в более общем контексте такой подход называется скаляризацией многокритериальной оптимизации). Для этого введем следующие скалярные целевые функции $G_w = w\,d + (1\text{-}w)\,C/L$ и $G'_w = w\,l + (1\text{-}w)\,C/L$. Минимизация этих целевых функций означает, что подобраны оптимальные значения параметров алгоритмов с точки зрения качества (малой длины пути между узлами) и цены (малой длины перемычек). При этом параметр $0 \le w \le 1$ характеризует относительную

значимость каждого из критериев (качество и цена). Другими словами, предлагаемый способ оптимизации предполагает, что для каждого значения параметра значимости критериев w должны быть подобраны значения параметров алгоритмов (например, p/t и α для S1/S1m; t, c и α для S2; размер сети L считаем заданным), которые минимизируют G_w или G'_w.

Для РСП, построенных по алгоритмам S1m с числом узлов L=10000, численно найдены параметры ансамблей t и α, при которых достигаются минимальные значения целевых функций при различных значениях значимости качества w. На рис. 5 эти данные представлены в случае G_w для параметра α, а на рис. 6 — для параметра t.

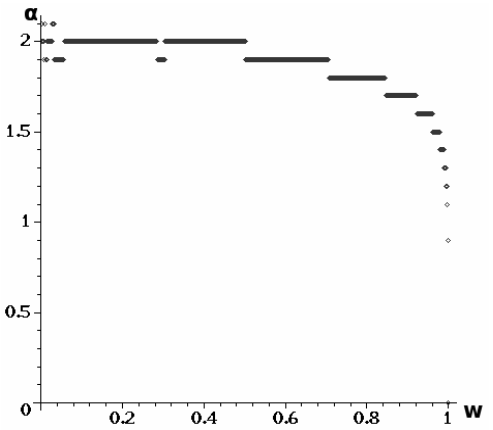


Рисунок 5: Зависимость оптимального значения α (минимизирующего Gw) от параметра значимости качества w

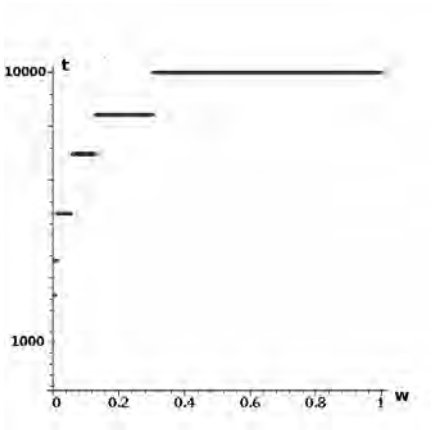


Рисунок 6: Зависимость оптимального значения t (минимизирующего G_w) от параметра значимости качества w

Из этих результатов следует, что когда значимость качества w превышает значение ~ 0.35, число перемычек должно быть максимально возможным (равным числу узлов сети); при меньших значениях этого параметра важность минимизации длины перемычек приводит к уменьшению числа перемычек (значения t). Напротив, значения параметра α близкие к 2 (то есть к критическому значению с точки зрения сохранения свойств малого мира [17]) является оптимальным при значениях параметра w меньших значения ~ 0.5, то есть когда доминирует цена.

Как отмечалось выше, при локальной навигации можно использовать информацию о сетевых соседях на глубину больше единицы. В частности, можно рассмотреть вариант локальной навигации, когда просматриваются не только ближайшие соседи, но и соседи соседей. При этом сообщение на следующем шаге пересылается в тот соседний узел, один из соседей которого ближе всего к узлу назначения в смысле решеточной метрики. Хотя при такой двухуровневой навигации на каждом шаге объем вычислений несколько увеличивается по сравнению с обычным жадным алгоритмом, но алгоритм остается локальным (не вычисляется весь путь до адресата, и объем не зависит от размеров системы). Поэтому этот алгоритм является хорошо масштабируемым и приемлем для сверхбольших коммуникационных сетей. Для такой навигации можно определить соответствующую целевую функцию $G''_w = w\, l^{(2)} + (1 - w)\, C/L$, где $l^{(2)}$ — навигационная длина при двухуровневой навигации.

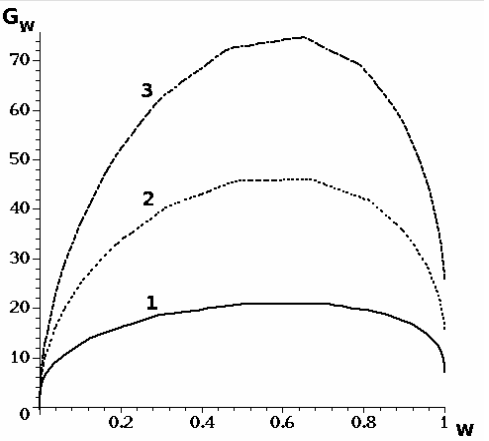


Рисунок 7: Зависимость минимальных значений целевых функций от параметра значимости w для алгоритма S1m: кривая 1 - min G_w; кривая 2 - min G''_w ; кривая 3 - min G'_w

На рис. 7 представлены результаты вычисления минимальных значений целевых функций G_w , G'_w и G''_w для сетей с числом узлов L = 10000, построенных по алгоритму

S1m (усреднение по ансамблям, состоящим из 100 – 1000 экземпляров). Каждая точка кривых на рис. 7 соответствует ансамблю сетей с параметрами t, α, обеспечивающими минимум G_w, G'_w или G''_w при данном значении параметра значимости критерия качества w. Видно, что во всем диапазоне w наименьшие значения имеет функция G_w ("бесконечная" глубина просмотра при навигации), минимальные значения G''_w (глубина просмотра — два) превышают G_w, а значения G'_w (единичная глубина просмотра) являются самими большими. Этот результат является интуитивно понятным: чем большая информация используется о структуре сети (большая глубина просмотра), тем меньше взвешенная сумма соответствующей средней длины пути и удельной длины перемычек.

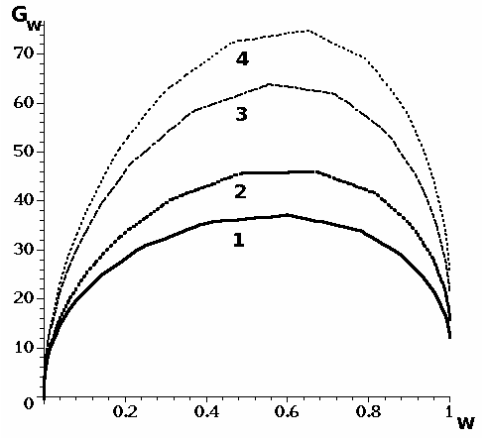


Рисунок 8: Зависимость минимальных значений целевых функций от параметра значимости w: 1 - min G"w для алгоритма S2; 2 - min G"w для алгоритма S1m; 3 - min G'w для алгоритма S2; 2 - min G'w для алгоритма S1m

Более важно, что целевые функции G_w, G'_w и G''_w позволяют сравнивать различные алгоритмы построения РСП, в том числе с различным набором параметров. В частности, результаты сравнения для алгоритмов S1m и S2 представлены на рис. 8.

Результаты сравнения минимальных значений целевых функций показывают, что при использовании маршрутизации сообщений, основанной как на локальной навигации с двухуровневой глубиной просмотра, так и на одноуровневом жадном алгоритме, для построения сетей более предпочтительным является алгоритм S2.

3 Заключение

В работе предложен подход к разработке коммуникационных сетей суперкомпьютеров следующего поколения, которые обладают свойствами "малого мира", а именно, медленным (логарифмическим) ростом среднего расстояния между узлами при увеличении их числа. При этом рассмотренные сети имеют базовую структуру регулярной решетки, что является важным для распараллеливания вычислительных заданий, связанных с численным моделированием D-мерных объектов. Эта базовая структура дополняется перемычками между узлами, которые и обеспечивают свойства "малого мира". Предложены методы оптимизации соотношения "цены" и "качества" для получаемой таким образом сети, причем в качестве "цены" выступает удельная длина дополнительных перемычек (общая длина перемычек в единицах базовой решетки, деленная на число узлов сети), а "качество" — это глобальная или навигационная средняя длина пути между узлами. Эти методы также позволяют количественно сравнивать сети, полученные с помощью различных алгоритмов их построения.

В последующих публикациях будут исследованы сети со свойствами малого мира, построенные на основе детерминистских алгоритмов, а также продолжено рассмотрение свойств предложенных в данной работе сетей, как с точки зрения других характеристик (в частности, нагрузки на узлы, устойчивости и т.п.), так и обобщения на более высокие размерности базовой решетки.

Литература

[1] Shainer G., Sparks B., Graham R. Toward Exascale computing, HPC Advisory Council, http://www.hpcadvisorycouncil.com/pdf/Toward_Exascale_computing.pdf;

Концепция создания экзафлопсного суперкомпьютера России: "Эксафлопные технологии. Концепция по развитию технологии высокопроизводительных вычислений на базе суперэвм эксафлопного класса (2012–2020 гг.)", http://filearchive.cnews.ru/doc/2012/03/esk_tex.pdf

[2] Report on Institute for Advanced Architectures and Algorithms Interconnection Networks Workshop 2008, Future Technologies Group Technical Report Series, Oak Ridge, Tennessee USA; http://www.csm.ornl.gov/workshops/IAA-IC-Workshop-08

[3] Dally W. J. , Towles B. P. Principles and Practices of Interconnection Networks.- Amsterdam: Elsevier Science, 2003.- 550 p.

[4] Dally W. J. Performance Analysis of k-ary n-cube Interconnection Networks, IEEE Transactions on Computers **39** (1990) 775.

[5] Kleinrock L. Communication Nets: Stochastic Message Flow and Design.- New York: McGraw-Hill, 1964.- 220 p.

[6] Watts D. J., Strogatz D. H. Collective dynamics of small-world networks, Nature 393 (1998) 440.

[7] Erdős P., Rényi A., On the evolution of random graphs, Publ. of the Math. Inst. of the Hungarian Academy of Sciences **5** (1960) 17.

[8] Albert R., Barabasi A.-L. Statistical mechanics of complex networks, Rev. Mod. Phys. **74** (2002) 47.

[9] Milgram S., The small world problem, Psychology Today, **2** (1967) 60.

[10] Kleinberg J. M. Navigation in the small world, Nature **406** (2000) 845.

[11] Zou Zhi-Yun etal., Regular Small-World Network, Chin. Phys. Lett. **26** (2009) 110502.

[12] Boettcher S. , Goncalves B., Azaret J., Geometry and Dynamics for Hierarchical Regular Networks, Journal of Physics A **41** (2008) 335003.

[13] Boettcher S., Goncalves B., Guclu H., Hierarchical Regular Small-World Networks, J. Phys. A **41** (2008) 252001.

[14] Comellas F., Ozona J., Peters J. G., Deterministic small-world communication networks, Information Processing Letters 76 (2000) 83;
Comellas F. , Mitjana M., Peters J.G., Broadcasting in Small-World Communication Networks, In: Proc. 9th Int. Coll. on Structural Information and Communication Complexity (2002), eds. C. Kaklamanis and L. Kirousis, pp. 73–85.

[15] Xiao W., Parhami B., Cayley graphs as models of deterministic small-world networks, Information Processing Letters **97** (2006) 115.

[16] Dally W.J. , Seitz C., Deadlock-free message routing in multiprocessor interconnection networks, IEEE Trans. Comput. **36** (5) (1987).

[17] Moukarzel C.F. , de Menezes M. A., Shortest paths on systems with power-law distributed long-range connections, Phys. Rev. E, 65 (2002) 056709;
Sen P., Chakrabarti B., Small-world phenomena and the statistics of linear polymer, J. Phys. A 34 (2001) 7749;
Petermann T., De Los Rios P., Spatial small-world networks: A wiring-cost perspective (2005) arXiv:cond-mat/0501420;
Petermann T., De Los Rios P., Physical realizability of small-world networks Phys. Rev. E **73** (2006) 026114.

[18] Barthelemy M., Spatial Networks, Phys. Reports 499 (2011) 1.

[19] Steuer, R.E. Multiple Criteria Optimization: Theory, Computations, and Application.- New York: John Wiley and Sons, 1986.- 330 p.

ПАРАЛЛЕЛЬНЫЕ ТЕХНОЛОГИИ В ЗАДАЧЕ МАКСИМИЗАЦИИ ПРАВДОПОДОБИЯ

А.В. Ермилов
Национальный исследовательский институт «Высшая школа Экономики», Россия, 101000, г. Москва, ул. Мясницкая, д.20.
alvalerm@mail.ru

В данной статье исследуется применение параллельных вычислений для получения оценок максимального правдоподобия параметров распределения Грам-Шарлье. Данное распределение применяется для описания AIM признаков, которые используются в качестве входных параметров в системах автоматического распознавания речи.

1 Введение

Исследовательские усилия в сфере речевых технологий привели к появлению большого числа коммерческих систем распознавания речи. Такие компании как Nuance, IBM, ScanSoft предлагают большой набор программных решений как для серверных, так и для десктопных приложений. Большинство этих систем построено на следующем принципе. Вектора признаков извлекаются из входящего сообщения и подаются на вход распознавателю, построенному на Скрытых Марковских Моделях (Hidden Markovs Models, HMM [1]). В качестве входных признаков обычно используются Мел-частотные Кепстральные Коэффициенты [2].

К сожалению, такой подход обладает следующим недостатком. В зависимости от длины речевого тракта (следует отметить, что длина речевого тракта зависит как от пола человека, так и от других физиологических параметров, например, роста, и может изменяться от 13 см у женщин до 18 см у взрослых мужчин), происходит сдвиг частот центральных формант. Разница в этих частотах может доходить до 25%. Из-за этого различия первоначально обученная модель может плохо распознавать сообщения нового диктора, то есть система становится дикторозависимой.

Один из способов решения этой проблемы – применение так называемой нормализации длины речевого тракта (Voice Tract Length Normalization, VTLN [3]), в ходе которой происходит преобразование исходного звукового сигнала таким образом, чтобы центральные форманты находились на одной частоте. Сложность этого подхода заключается в том, что необходимо предварительно оценивать параметры этого преобразования для каждого конкретного диктора, что не всегда представляется возможным, в случае, когда объем речевого материала недостаточно велик.

Другой способ – использование признаков, которые не меняются от диктора к диктору. В качестве таких признаков можно использовать признаки из Auditory Image Model (AIM)[4].

2 Auditory Image Model

Auditory Image Models были разработаны в лаборатории Роя Петерсона из Кембриджского университета с целью моделирования человеческой психоакустики. AIM – функциональная модель человеческой слуховой системы, которая принимает во внимание биологическую информацию. Модель состоит из трёх последовательных модулей:

1. Банк фильтров, состоящих из гамматоных фильтров, расположенных согласно ERB масштабу. Выходное значение этого банка фильтров примерно соответствует движению базальной мембраны улитки внутреннего уха.
2. Двумерный адаптивный пороговый механизм. На этом этапе сигнал, полученный на выходе банка фильтров, пропускается через однополупериодный выпрямитель и проходит через сжимающий логарифмический нелинейный канал. После этого применяется двумерный (по времени и частоте) адаптивный пороговый механизм. Временной порог включает в себя

краткосрочную память прошлой активности: если недавняя активность была низкой, то большая часть движения базальной мембраны пропускается через порог. Спектральный порог основан на взаимодействии близких частотных каналов: высокая активность в канале будет частично подавлять активность в менее активном канале. Выходной сигнал на этом этапе схож с нейронной моделью деятельности (Neural Activity Pattern - NAP) слухового нерва, который соединяет улитку внутреннего уха с ядрами ствола мозга. В дальнейшем мы работаем с NAP признаками.

3. Стробируемый интегратор. Стробируемый интегратор применяется к NAP представлению для синхронизации периодов между максимумами NAP.

Для получения признаков мы используем следующий подход. Активность в каждом канале NAP сглаживалась с помощью фильтра высоких частот с частотой среза 100 Гц, после этого NAP нарезался на фреймы длиной 10 мс и NAP профиль получался путем суммирования активности внутри фрейма. Далее полученный профиль нормализовался таким образом, чтобы с ним можно было работать как плотностью распределения. Для описания полученной плотности было использовано расширения Грам-Шарлье.

3 Расширение Грам – Шарлье

Если истинная плотность распределения случайной величины z неизвестно, то разумно представить её в виде:

$$g(z) = p_n(z)\varphi(z),$$

где $\varphi(z)$ – плотность стандартного нормального распределения, а $p_n(z)$ выбрана таким образом, чтобы $g(z)$ имела те же моменты, что и истинная плотность z. Такая аппроксимация носит название расширения Грам-Шарлье

Полиномы Эрмита образуют ортогональный базис относительно скалярного произведения, порожденного математическим ожиданием, взятым по плотности стандартного нормального распределения. Это свойство позволяет использовать многочлены Эрмита в функции $p_n(z)$:

$$p_n(z) = 1 + \sum_{i=1}^{n} c_i H_i(z) .$$

К сожалению, полученная функция не является в строгом смысле плотностью: для некоторых значений параметров функция может принимать отрицательные значения. Для решения этой проблемы предложено использовать положительную плотность [5]:

$$g(z) = \varphi(z)\left(1 + \sum_{i=1}^{n} c_i H_i(z)\right)^2 \frac{1}{k},$$

где $k = 1 + \sum_{i=1}^{n} c_i^2 i!$.

Подобная плотность удобна не только с точки теоретической точки зрения, но с практической – при оценке параметров методом максимального правдоподобия логарифмическая функция правдоподобия получается разделяемой и содержит логарифмы положительных выражений, что упрощает численную оптимизацию:

$$l = \ln \varphi(z) + \ln\left(1 + \sum_{i=1}^{n} c_i H_i(z)\right)^2 - \ln 1 + \sum_{i=1}^{n} c_i^2 i!$$

4 Эксперименты

В статье [6] было предложено моделировать NAP профиль с помощью смеси нормальных распределений. В данной статье мы предлагаем использовать более точное распределение для моделирования NAP профиля, а именно смесь 3 расширений Грам-Шарлье.

Для отыскания оценок весов смеси и параметров распределений было предложено использовать две модификации алгоритма симуляции отжига для решения задачи максимизации правдоподобия. В первой модификации мы стартуем из нескольких случайно определяемых начальных точек, а затем из полученных результатов мы выбираем тот, который даёт наибольшее значение функции правдоподобия. Во второй модификации мы случайно выбираем несколько точек, затем сравниваем значение функции правдоподобия и стартуем алгоритм симуляции отжига из лучшей точки. Мы сравниваем два данных подхода по времени выполнения на разном количестве процессоров и качеству подгонки распределения.

На рис.1 показана типичная картинка подгонки распределения смесью из 3 распределений расширений Грам-Шарлье. Мы видим, что степень подгонки распределения является почти идеальной.

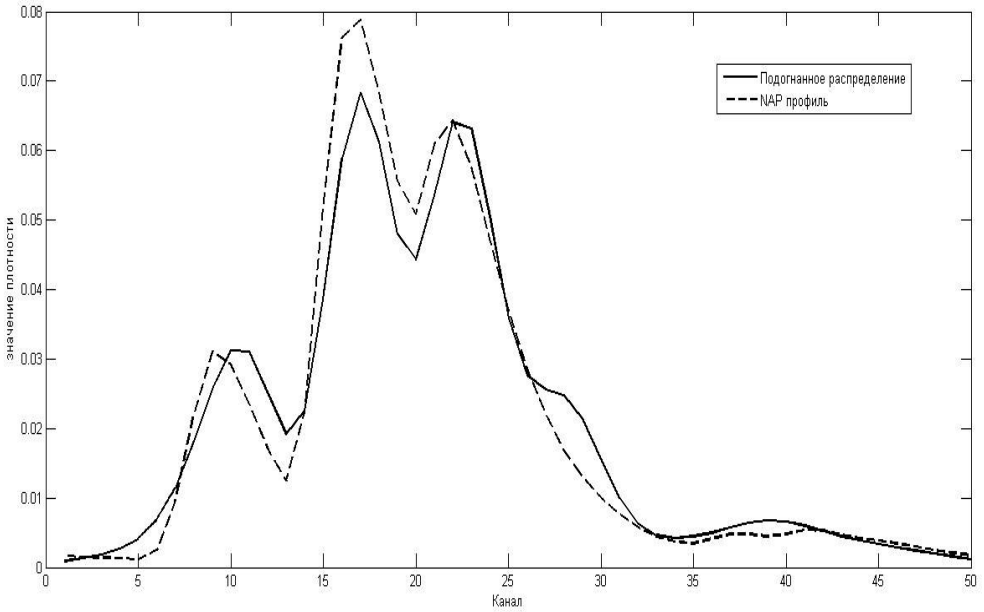


Рис.1: Подгонка NAP профиля смесью из 3 распределений расширений Грам-Шарлье

Таблица 1. Время выполнения алгоритмов

Количество процессоров	Время работы первого алгоритма, с	Время работы второго алгоритма, с
1	9756	15
3	4465	13
6	2463	11

В таблице 1 приведены результаты сравнения времени выполнения двух алгоритмов по времени выполнения на разном количестве процессоров. Как и следовало ожидать, первый алгоритм является более вычислительно сложным. Из таблицы видно, что время выполнения падает почти линейно с ростом количества процессоров. Таким образом, можно сделать вывод о том, что при реализации даже первого алгоритма на графическом ускорителе, система вполне может работать в реальном времени. Также интересно сравнить степень подгонки в первом случае и во втором случае.

Таблица 2. Степень подгонки распределения

Мера подгонки	Результат первого алгоритма	Результат второго алгоритма
Растояние Кульбака-Лейблера	0.117	0.140
Лог-правдоподобие	-32935	-34289

Из Таблицы 2 видно, что степень подгонки оказалось в первом случае лучше: расстояние Кульбака-Лейблера между исходным и подогнанным оказалось меньше и значение функции правдоподобия оказалась большим. Следует отметить, что преимущество можно назвать не слишком большим.

Заключение

В данной статье мы предложили параметрическое распределения профиля NAP, полученное по AIM модели. Параметры данного распределения могут использоваться в системах автоматического распознавания речи как дикторонезависимые признаки. Кроме этого мы предложили два алгоритма численного решения задачи максимизации правдоподобия с использованием параллельных вычислений. Мы сравнили работу этих алгоритмов по скорости и качеству подгонки.

В качестве дальнейшей работы можно предложить реализацию алгоритмов на GPU. Кроме того, интересно построить тест качества подгонки распределений, полученных с помощью предложенных алгоритмов.

С практической точки зрения представляется интересным сравнить качество работы системы автоматического распознавания речи, построенной на признаках из предложенной модели, и классической системы на кепстральных коэффициентах.

Литература

[1] Rabiner, L. (1989): A tutorial on hidden markov models and selected applications in speech recognition'. *Transactions on Information Theory,* 77(2), 257–284.

[2] Sahidullah, Md., Saha, Goutam: Design, analysis and experimental evaluation of block based transformation in MFCC computation for speaker recognition. *Speech Communication,* 54 (4): 543–565.

[3] T. Kamm, G. Andreou, and J. Cohen: Vocal Tract Normalization in Speech Recognition: Compensating for Systematic Speaker Variability. *Proc. of the 15th Annual Speech Research Symposium,* 161-167.

[4] M. E. Munich and Q. Lin: Auditory Image Model features for Automatic Speech Recognition. *9th European Conference on Speech Communication and Technology (Interspeech'2005 - Eurospeech)*, Lisbon, Portugal, September 4–8, 2005.

[5] Ñiguez, T. and Perote, J.: Forecasting the Density of Asset Returns. *STICERD Working Paper EM/2004/479, London School of Economics.*

[6] Jessica J. M. Monaghan, Christian Feldbauer, Thomas C. Walters and Roy D. Patterson: Low-Dimensional, Auditory Feature Vectors that Improve VTL Normalization in Automatic Speech Recognition, *Journal of the Acoustical Society of America*, 123, 3066.

РЕШЕНИЕ ПАРАЛЛЕЛЬНЫХ ЗАДАЧ ЗА РАМКАМИ КЛАССА EP В РАСПРЕДЕЛЕННЫХ ВЫЧИСЛИТЕЛЬНЫХ СРЕДАХ[1]

Ю.А. Жолудев

Факультет Вычислительной математики и кибернетики Московского Государственного Университета имени М.В. Ломоносова
119991 ГСП-1 Москва, Ленинские горы, МГУ имени М.В. Ломоносова
diemenator@gmail.com

В данной работе представлен новый подход к метакомпьютингу, позволяющий решать параллельные задачи с произвольным коммуникационным профилем в распределенных вычислительных средах.

Целью данной работы является создание инструментария для построения распределенных вычислительных сред, позволяющих решать параллельные задачи, в которых возможны обмены данных между любыми параллельными работающими на различных узлах процессами.

Введение

Существующие распределенные вычислительные среды можно подразделить на несколько групп, основываясь на способе организации вычислений и области применения.

Первая группа – это традиционные GRID-системы, строящиеся с помощью инструментария типа gLite или Globus Toolkit [1], которые объединяют суперкомпьютеры в единую инфраструктуру, позволяющую ставить на решение параллельные задачи, которые могут быть решены на одном из нескольких включенных в инфраструктуру суперкомпьютеров, учитывая при этом требования задач к установленным прикладным библиотекам, аппаратному обеспечению и т.д. Развертывание подобных систем обычно представляет собой трудоемкий процесс, включающий в себя не только установку и настройку интерфейсных компонент менеджеров очередей на вычислительных кластерах, но и тонкую настройку всей инфраструктуры, включая определение политик безопасности, распределения задач по ресурсам и т.д.

Вторая группа – это традиционные Desktop GRID системы, такие как BOINC[2] и X-Com [3]. В таких системах требуется наличие центрального (серверного) компонента, отвечающего за управление ходом решения задач, включение и отключение в расчет новых узлов. На клиентском уровне в таких системах находятся вычислительные узлы, которые получают от серверной части вычислительные порции исходной задачи, обрабатывают их и отправляют обратно результаты вычислений. С помощью таких систем можно объединять в вычислительную среду неоднородные ресурсы, причем это могут быть компьютеры совершенно любого типа: от домашних персональных компьютеров и офисных рабочих станций, до узлов вычислительных кластеров. Клиент-серверная организация вычислений накладывает определенные ограничения на класс задач, решаемых в таких системах: задача должна быть разбита на множество последовательных подзадач, при этом между подзадачами не предусматривается возможность обмена данными, т.е. исходная задача должна быть из класса EP (Embarassinly Parallel). Кроме того, наличие центрального серверного компонента подразумевает высокие требования к производительности и надежности сервера и каналов связи с ним, т.к. отказ в работе центрального компонента будет означать для таких систем полную остановку решения задачи.

Третья группа – это P2P Desktop GRID системы.

[1] Работа выполняется при поддержке гранта Президента Российской Федерации для молодых российских ученых-кандидатов наук МК-5104.2011.9.

Системы этой группы с помощью технологий на базе peer-to-peer (p2p) [4] позволяют организовывать распределенные вычисления без участия центральных компонент, так как все узлы, участвующие в подобных системах, одновременно играют роли сервера и клиента.

Организация распределенных вычислений в P2P Desktop GRID

Технологии peer-to-peer сетей позволяют строить на основе существующих физических сетей виртуальные сети, передача данных и адресация может быть однородна и независима от физической топологии и поддерживаемых протоколов передачи данных.

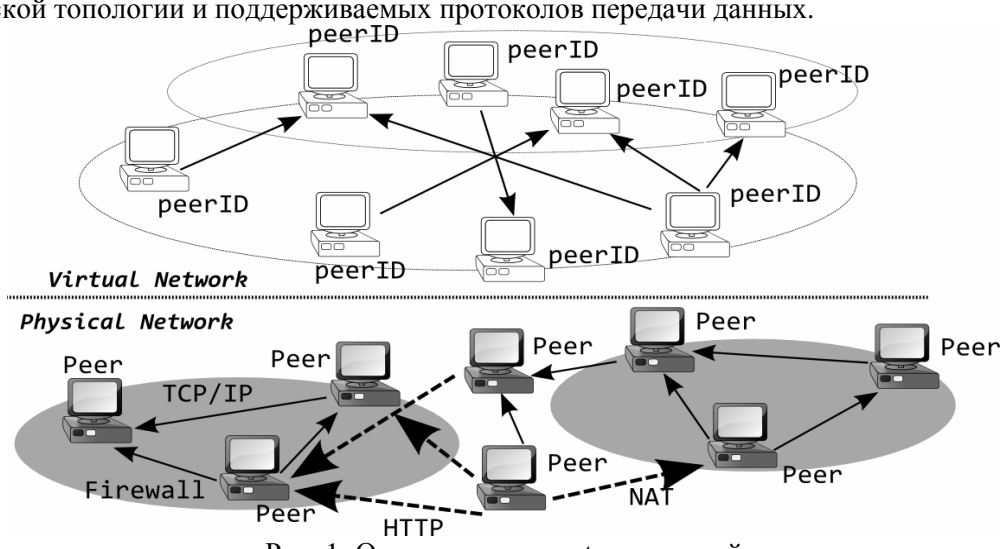


Рис. 1: Организация peer-to-peer сетей

Основными достоинствами вычислительных сред на основе P2P сетей являются:

- децентрализованность,
- сбалансированная нагрузка на каналы связи,
- поддержка коммуникации между любыми узлами сети,
- независимость от топологии физической сети и присутствующих в ней барьеров,
- надежность (отключение нескольких физических узлов или каналов связи не приведет к нарушениям в работе сети).

При этом имеют место недостатки, связанные со способом построения и ограничениями физических сетей, лежащих в основе виртуальной, а именно:

- высокая латентность при передаче данных (частично обуславливается количеством узлов-посредников, передающих данные через различные административные барьеры),
- сложность в управлении (подключении и отключении узлов),
- накладные расходы на поддержку работы сети.

Существует множество распределенных вычислительных систем на основе p2p-технологий, которые реализуют различные стратегии организации распределенных вычислений:

- распределенный динамичный пул задач (Cohesion Orbweb [5]),
- распределенные системы постановки множества последовательных задач (bag of tasks) для решения на любых участвующих в системе вычислительных узлах (Jalapeno, JNGI, OurGird, Triana, Xeerkat) [6],
- распределенная вычислительная среда для решения параллельных задач с поддержкой передачи сообщений (P2P-MPI [7]).

Вариант Cohesion Orbweb организовывает вычисления таким образом, что любой узел среды может пополнять пул последовательных задач, либо брать из пула задачу, на основе которой порождать и решать новые подзадачи, которые немедленно становятся доступны всем остальным узлам вычислительной среды. Ограничения Cohesion Orbweb связаны со сложностью развертывания системы (необходимо устанавливать на всех узлах XMPP-сервер

OpenFire), и зависимости от физической топологии сети (используемые технологии передачи сообщений на основе протокола XMPP не поддерживают работу в сетях с искусственными административными барьерами).

Системы из второй группы позволяют разделять вместе с большой группой владельцев компьютеров свои вычислительные ресурсы, давая возможность решать собственные задачи на чужих компьютерах и принимать участие в решении сторонних задач. Часто в таких системах применяется рейтинговая система, позволяющая определить приоритетность решения задач одного пользователя на основе количестве решенных с помощью его ресурсов чужих задач, аналогично рейтинговой системе, применяющейся в файлообменных сетях. Подобные системы, как правило, не предназначены для решения больших задач, но могут быть использованы, если исходную задачу заранее разбить на фиксированное множество исполняемых подзадач, что, потребует вхождения прикладной задачи в класс EP.

Вариант P2P-MPI представляет собой реализацию MPJ (подмножество MPI 1.1 для Java), позволяющую ставить на счет полноценные параллельные задачи в модели программирования MPI с фиксированным набором процессов в рамках одной локальной сети, реализуя дублирующее выполнение параллельных процессов задачи с целью обеспечения устойчивости к сбоям каналов связи и/или вычислительных узлов. Peer-to-peer технологии здесь используются в рудиментарном варианте, не обеспечивая никакой поддержки работы среды в сетях с наличием искусственных административных барьеров (типа Firewall, NAT и т.д.). При этом поддерживаются только Java-программы, а прикладные библиотеки нужно ставить на узлах заблаговременно.

В данной работе также предлагается подход организации распределенных вычисления, основанный на использовании технологий p2p.

Требования и модель программирования

Определим общие требования к разрабатываемой системе и решаемым с её помощью задачам:

- Распределенность:
  - задача может быть поставлена на решение из любого узла среды,
  - в среде может параллельно решаться более одной задачи,
  - исходные данные задачи и результаты вычислений, хранящиеся в файлах, должны быть доступны с любого узла среды,
  - независимость от топологии физической сети.
- Динамичность:
  - задача должна быть решена независимо от количества изначально доступных узлов,
  - любая задача может быть решена на одном узле независимо от количества процессов,
  - решение задачи не должно прерываться при изменении состава вычислительных узлов:
    - процессы задачи могут мигрировать с узла на узел по достижении контрольных точек,
    - для каждого процесса может быть несколько работающих на разных узлах экземпляров;
- независимость от программно-аппаратной платформы,
- легкость в развертывании,
- поддержка работы с внешними прикладными библиотеками.

Исходя из требований и ограничений, связанных с чертами распределенных вычислительных сред, модель программирования для разрабатываемой системы определяется следующим образом:

- Прикладная задача – набор взаимодействующих параллельных процессов (Job).

- Процессы могут общаться друг с другом посредством сообщений и разделять между собой данные посредством файлов.
- Исполняемый код процесса разбивается контрольными точками на независимые от других процессов фрагменты.
- Коммуникационные операции выполняются в контрольных определенных точках процесса.
- Процессы не имеют никаких сведения о низлежащей топологии физической сети и адресуют друг друга по номерам (rank).

Пример взаимодействия узлов при таких требованиях и модели программирования показан на диаграмме последовательности (рис. 2).

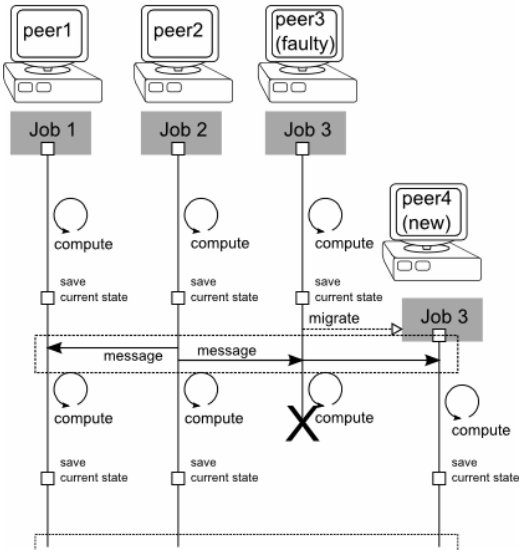


Рис. 2: Пример взаимодействия при динамичном составе узлов

Реализация

Разрабатываемая в рамках данной работы система X-P2P[8] строится на основе peer-to-peer технологии JXTA[9], которая представляет собой набор протоколов для поддержки работы peer-to-peer сети, и определяющая такие объекты как peer'ы (узлы сети с уникальными идентификаторами), группы узлов (механизм абстракции определенного множества узлов в группу по определяемых приложением требованиям), advertisement (публикации - временные объекты, доступные всем участникам сети или группе узлов, содержащие метаданные существующих в сети объектов), pipe'ы (каналы - объекты, отвечающие за передачу данных между узлами) и content (файлы или какие-либо другие разделяемые в группе узлов данные).

В данной работе используется Java-реализация протоколов JXTA JXTA SE 2.6, как наиболее стабильная и эффективно поддерживающаяся библиотека работы с peer-to-peer сетями.

На рис. 3 представлена в общем виде архитектура приложения, отвечающего за включение в распределенную вычислительную среду, работающего на вычислительном узле. Где процесс job – Java-процесс прикладной задачи, связывающийся с другими процессами посредством инфраструктурного компонента и сохраняющий в контрольных точках своё состояние и данные задачи в локальной файловой системе, которые в дальнейшем становятся доступны другим узлам сети посредством инфраструктурного компонента (верхний уровень архитектуры), отвечающего за включение в среду, поиск и адресацию подключенных к среде узлов.

При этом прикладная задача представляет собой:

1. Набор файлов библиотек и исходных данных задачи.

2. Исполняемый код загрузки и инициализации внешних библиотек и программных модулей.
3. Код порождения исходного набора процессов.

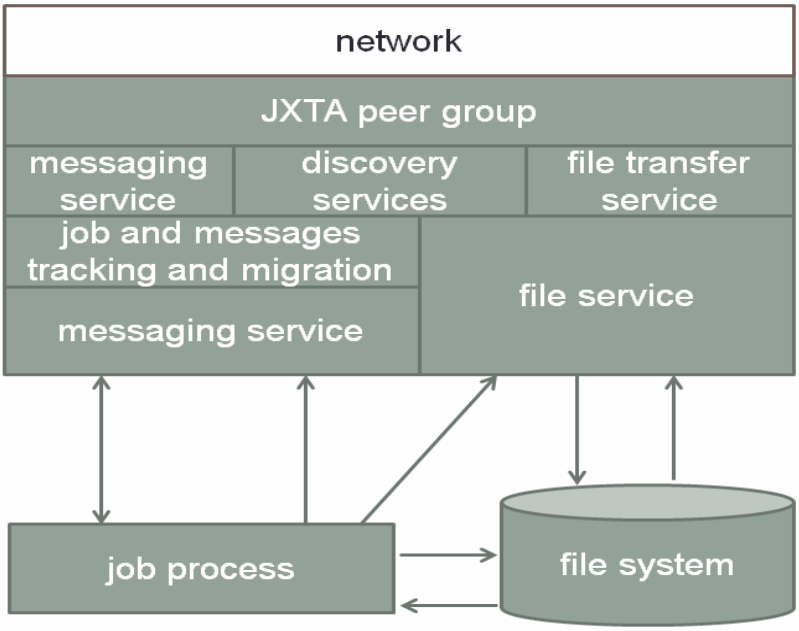


Рис. 3: Архитектура приложения узла X-P2P

При запуске задачи:

1. На начальном узле создается группа узлов и публикуется описание задачи вместе с идентификатором группы.
2. Начальный узел присоединяется к группе узлов текущей задачи.
3. Выполняется инициализация библиотек и порождение процессов.
4. Состояние порожденных процессов сохраняется в виде файлов и публикуется в группе.
5. Вход в основной цикл работы с процессами задачи.

При включении нового узла в расчет:

1. Узел запрашивает описание задачи и присоединяется к соответствующей группе узлов.
2. Загружаются файлы задачи и инициализируются необходимые библиотеки.
3. Вход в основной цикл работы с процессами задачи.

Основной цикл работы узла:

1. Получить (локально или из сети) копию свободного заблокированного процесса задачи.
2. Получить все посланные ранее процессу сообщения.
3. Продолжить выполнение процесса до контрольной точки.
4. Сохранить состояние процесса и приостановить его выполнение.
5. Опубликовать данные о новом состоянии процесса.

Заключение

В данной работе был представлен подход и предложены основные идеи для организации распределенных вычислений, отвечающий требованиям надежности и динамичности и позволяющий решать полноценные параллельные задачи за рамками класса EP на любых доступных вычислительных ресурсах с помощью технологий peer-to-peer. На основе данного подхода разрабатывается система метакомпьютинга X-P2P, которая в перспективе не только позволит решать параллельные задачи с произвольным коммуникационным профилем, но

сможет объединить большое количество вычислительных ресурсов в одну масштабную распределенную вычислительную среду, в которой смогут принять участие все желающие.

Литература

[1] The Globus Alliance. http://www.glous.org/allicance

[2] Open-source software for volunteer computing and grid computing. https://boinc.berkeley.edu/

[3] 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 157-164.

[4] R. Steinmetz and K. Wehrle. Peer-to-Peer Networking and Computing. Informatik-Spectrum, 27(1). Springer. 2004.

[5] S. Schulz, W. Blochinger, and M. Poths. Orbweb - A Network Substrate for Peer-to-Peer Grid Computing Platforms based on Open Standards. Journal of Grid Computing, 8(1):77-107, Springer, 2010.

[6] Marco Ferrante - The JXTA way to Grid: a dead end? Dottorato in Infromatica, XXII ciclo, 2008.

[7] A Peer-to-Peer Framework for Message Passing Parallel Programs, Stéphane Genaud and Choopan Rattanapoka, in Parallel Programming, Models and Applications in Grid and P2P Systems , vol. 17, pages 118--147, Ed. Fatos Xhafa, Advances in Parallel Computing,IOS Press, ISBN:978-1-60750-004-9, June 2009.

[8] X-P2P computing platform http://code.google.com/p/xp2p/

[9] Gabriel Antoniu, Mathieu Jan and David Noblet. Enabling the P2P JXTA Platform for High-Performance Networking Grid Infrastructures. In Proc. of the first Intl. Conf. on High Performance Computing and Communications (HPCC'05), (3726):429-439, Springer-Verlag, Sorrento, Italy, September 2005.

ИССЛЕДОВАНИЕ ОСОБЕННОСТЕЙ ПРОБЛЕМЫ ИНТЕРОПЕРАБЕЛЬНОСТИ В GRID-ТЕХНОЛОГИИ И ТЕХНОЛОГИИ ОБЛАЧНЫХ ВЫЧИСЛЕНИЙ[1]

Е.Е. Журавлёв[1], В.Н. Корниенко[2], А.Я. Олейников[2]
[1] *ФИАН им. П.Н. Лебедева РАН, Россия*
euzhurav@pluton.lpi.troitsk.ru
[2] *ИРЭ им. В.А. Котельникова РАН, Москва, Россия*
kvn@cplire.ru, olein@bk.ru

Введение

Настоящий доклад представляет развитие результатов наших работ, доложенных на предыдущей конференции GRID2010 [1]. Тогда мы рассматривали только вопросы стандартизации и «проблему интероперабельности» применительно к GRID-вычислениям. Однако, как известно, за истекшие два года весьма активно развивались облачные вычисления («облака»), притом практическая работа опережала создание научных основ. В связи с этим тематика облачных вычислений в нашей стране была включена в тематику Российского фонда фундаментальных исследований и Программу Президиума РАН №14. Стало совершенно очевидным, что поскольку облака в общем случае представляют собой, как и GRID-системы, гетерогенную среду, то и для облаков должна возникнуть проблема взаимодействия разнородных элементов, получившая название проблемы интероперабельности. Возникает естественный вопрос: есть ли разница в способах решения этой проблемы в GRID-системах и в облаках и если есть, то в чем она заключается? Для этого следует понять разницу в сущностях GRID-технологий и технологий облачных вычислений. Задача эта весьма непростая, хотя бы потому, что в обоих случаях нет общепринятых однозначных определений. Вопросам сопоставления этих технологий посвящен целый ряд статей [2, 3]. В настоящем докладе кратко описывается разработанный нами единый подход к обеспечению интероперабельности для систем широкого класса, оформленный в виде ГОСТ Р, и на основе этого подхода и обзора зарубежного и отечественного опыта делается попытка выявить общие черты и особенности при обеспечении интероперабельности для GRID-технологии и технологии облачных вычислений.

Единый подход к обеспечению интероперабельности систем широкого класса

Интероперабельность, согласно определению, данному в ISO/IEC 24765:2009, Systems and Software Engineering – Vocabulary есть «способность двух или более систем или компонентов к обмену информацией и к использованию информации, полученной в результате обмена». Свойство интероперабельности, наряду со свойством переносимости, составляет одно и важнейших свойств открытых систем, достигается за счет использования методов функциональной стандартизации (построение профилей стандартов), и авторами получен значительный опыт в этой области [4]. В настоящее время всё большее внимание уделяется именно вопросам обеспечения интероперабельности для информационных систем (ИС) различного масштаба (от наносистем до сверхсложных систем – Systems of Systems) и ИС различного назначения, в том числе для науки и образования. Возрастание интереса к проблеме интероперабельности обусловлено тем, что если раньше занимались вопросами т.н. технической интероперабельности, то в последнее время возрос интерес к интероперабельности более высоких уровней (семантической, организационной), требующей

[1] Работа выполнена при поддержке РФФИ (проект 12-07-00261а) и Программы Президиума РАН № 14).

своих стандартов [5]. В [6] авторами опубликован обзор по современному состоянию проблемы интероперабельности с рассмотрением основных ее аспектов, начиная от вопросов терминологии и заканчивая положением о необходимости разработки единого подхода к обеспечению интероперабельности ИС возможно более широкого класса. Такой подход был нами разработан и оформлен в виде проекта ГОСТ Р [7], который должен войти в действие в 2012 г. Несмотря на то, что в названии этого стандарта фигурируют промышленные предприятия, он имеет гораздо более широкое значение. На Рис.1 показана блок-схема, отражающая основные этапы достижения интероперабельности.

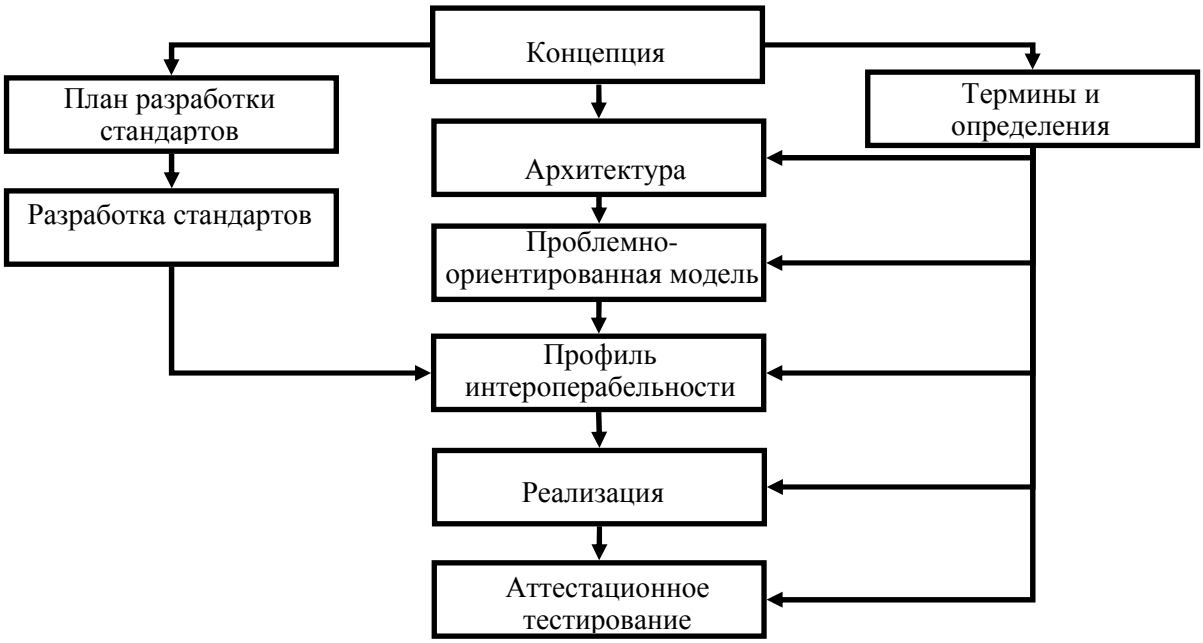


Рис.1: Основные этапы достижения интероперабельности

Основная технологическая цепочка включает этапы, начиная от построения концепции и заканчивая этапом тестирования с постепенной детализацией. Авторы намерены применить описанный подход к достижению интероперабельности в области GRID-систем и в области облаков, что и позволит выявить общие места и особенности. До настоящего времени работы по интероперабельности в GRID-системах в облаках носили фрагментарный характер

Сравнение GRID-систем и облаков

Всякая GRID-система есть некая система, образованная с помощью интеграции, виртуализации и управления сервисами и ресурсами в распределённой, гетерогенной среде. Система поддерживает объединение пользователей и ресурсов (виртуальных организаций) параллельно традиционным административным и организационным областям (доменам – реальным организациям). Ключевые требования к успешной реализации и управления GRID - системой включают стандартизацию интерфейсов входящих компонентов и использование стандартизованной информационной модели и моделей данных [8].

Облачное вычисление - это модель сетевого доступа, удобного и разрешённого из любого места по запросу к разделяемому пулу, который можно сконфигурировать из вычислительных ресурсов (сети, серверы, хранилища данных, приложения и сервисы) и который может быть быстро собран и предоставлен для работы при минимальных организационных усилиях при организации взаимодействии с провайдером сервиса.

Ключевые требования к успешной реализации и управления облачными вычислениями, также как и GRID – систем, включают стандартизацию интерфейсов входящих компонентов и использование стандартизованной информационной модели и моделей данных. Требование стандартизации интерфейсов и использование стандартной информационной модели является прямым следствием обеспечения условий функционирования ИС, а именно соответствия требованиям открытости: переносимости приложений, интероперабельности и масштабирования [4].

Судя по определению облачных вычислений, данному NIST [9], главной проблемой при реализации облачных вычислений является проблема организации процесса предоставления услуг клиенту с обеспечением режима наибольшего благоприятствования. Основную трудность, которую видят сторонники и инициаторы облачных вычислений, составляет взаимодействие между провайдерами, т.е. обеспечение интероперабельности на организационном уровне.

Проблема взаимодействия между провайдерами в GRID-системах решается на этапе создания виртуальной организации, представляющую собой интеграцию пользователей и владельцев — провайдеров услуг будущей GRID-системы.

Можно отметить, что общими в той и другой технологиях являются также виды интероперабельности (Рис.2) (см. модель интероперабельности на (Рис.1.).

Организационная (прагматическая)	8. Экономическая/регламентная политика (Политические и экономические цели, декларируемые в политике и регламентах) 7. Цели бизнеса (Стратегические и тактические цели распределённые между участникам бизнеса) 6. Процедуры бизнеса (Установление соответствия процедур действиям бизнес - процесса)
Информационная (семантическая)	5. Контекст бизнеса (Существенные знания о бизнесе, определяющие семантику в документообороте) 4. Семантические условия (Условия концепций помещённых в послании о структуре данных)
Техническая (синтаксическая)	3. Понимание структуры данных в послании при обмене между системами 2. Сетевая интероперабельность (на множестве сетей) 1. Физические и логические соединения

Рис.2: Виды интероперабельности

Общим для обеих сред служит также то, что GRID-технологии и облачные вычисления используют транспортную среду сети Интернет. Однако, GRID - вычисления опираются на использование Web – служб, в то время как облачные вычисления предполагают использовать сетевую файловую систему (NFS — Network File System) [2]. В частности, в облаках используется протокол RPC-вызов удалённой процедуры (Рис.3), который позволяет использовать файлы, содержащие процедуры управления внешним компьютером и его периферией. Тем самым, профили Grid-систем и облаков будут отличаться уже на сетевом уровне (см. Рис.2).

На рис.4 приведена т.н. комбинированная концептуальная справочная диаграмма NIST (National Institute Standards and Technology USA) технологии облачных вычислений.

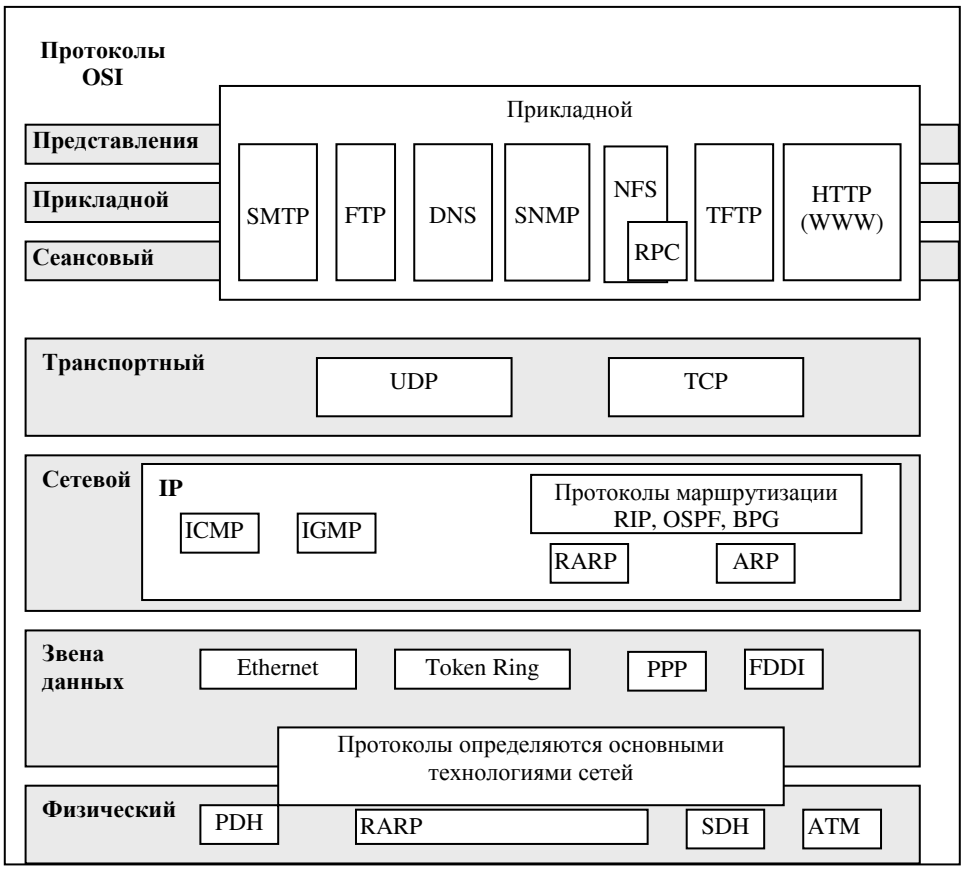


Рис.3: Протоколы RPC и HTTP в стеке протоколов Интернет (http://ru.wikipedia.org/wiki/TCP/IP)

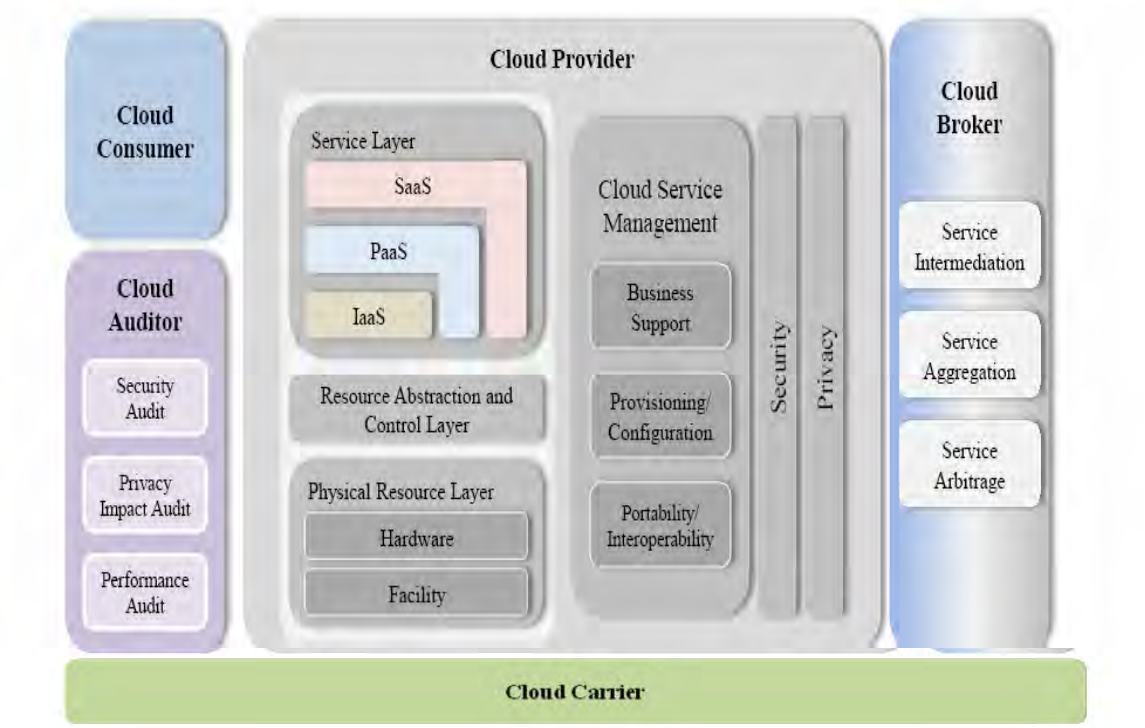


Рис.4: Комбинированная концептуальная справочная диаграмма
(The Combined Conceptual Reference Diagram) [9]

На рис.5 показана связь различных решений, связанных с системами. Системы образованы службами, с помощью которых обеспечивается доступ к ресурсам. Интерфейсы обеспечивают системам доступ и применение. Прикладные программные интерфейсы (API) в свою очередь обеспечивает приложениям доступ к интерфейсам.

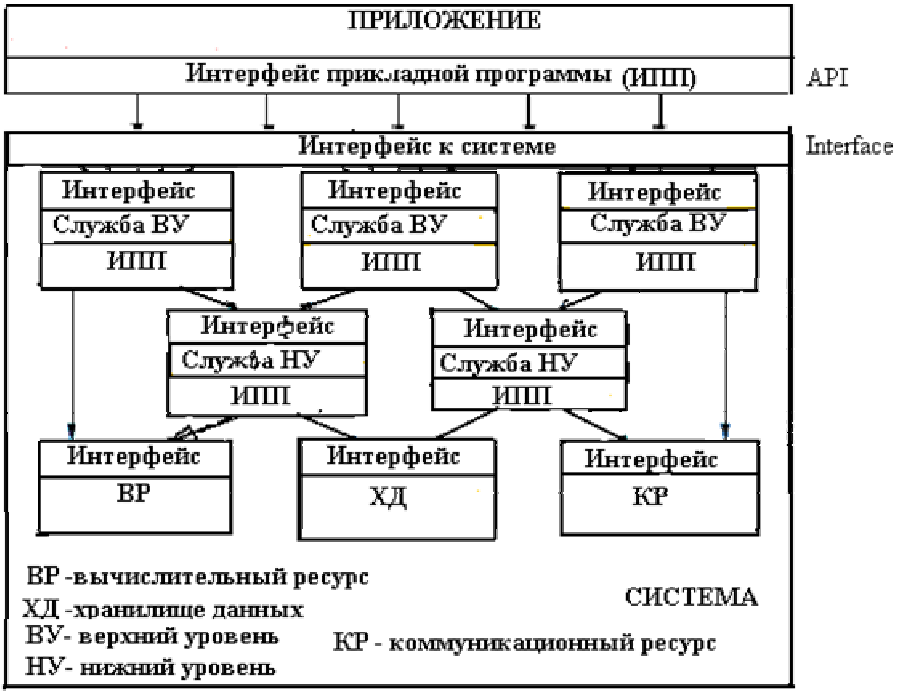


Рис.5: Иллюстрация идеи применения прикладных программных интерфейсов (в частности API) для осуществления облачных вычислений [10]

Под службой подразумевается некий узел, который предоставляет возможности ресурса или выполнение некоторых действий на ресурсе. Службы в свою очередь могут быть службами низкого уровня, которые реализованы преимущественно на физических ресурсах, и службами высокого уровня, реализованными в виде виртуальных ресурсов (т.е. на других службах). Службы обнаруживают свои возможности с помощью интерфейсов служб.

Система представляет собой упорядоченное множество служб и ресурсов, которые образуют некое единое целое (см. Рис.6). Замысел системы по своей природе иерархический, то есть могут существовать системы из систем. Системы высокого уровня являются системами, построенными на применении других систем (подсистем – систем низкого уровня) путем их агрегирования.

Рис.6: Смысловое упорядочивание абстрактного представления узлов рис.4 [10]

Смысловая сложность узлов уменьшается в направлении верхних уровней; удобство применения узлов с точки зрения конечного пользователя растёт по мере уменьшения смысловой сложности.

В настоящее время по проблеме обеспечения интероперабельности в области GRID-технологий международная организация Open Grid Forum (http://www.gridforum.org/) опубликовала почти 200 документов, из них по облачным вычислениям пока только 5 (GFD 150, GFD 162, GFD 183, GFD 184, GFD 185), последние 4 из которых подготовлены рабочей группой Open Cloud Computing Interface.

За рубежом проблемой обеспечения интероперабельности в области облачных вычислений занимается также организация Cloud Computing Interoperability Forum (http://www cloudforum.org) и IEEE (http://standards.ieee.org/news/2011/cloud.html). В частности, обсуждается предложение в рамках OSGI – Open Service Gateway Initiative использовать в качестве языка обеспечения интероперабельности облачных вычислений Java [11].

В 2010 году стартовал проект StratusLab имеющий цель исследование влияния парадигмы облачного вычисления на сборку сервисов вычислительной GRID-системы [11].

В связи с большой гетерогенностью среды разделяемой инфраструктуры, в проекте StratusLab важную роль играет интероперабельность. В настоящее время основой решения видится обеспечение интерфейсов IaaS, доступ к устройствам виртуальной машины и безопасность. Особенно важным в StratusLab видится обеспечение интероперабельности интерфейса между GRID - middleware и сервисом администрирования облака. В таком случае вопросы авторизации и мониторинга стали первостепенными для исследования.

Ядром дистрибутива StratusLab является OpenNebula, уже принятая OGF OCCI в качестве стандарта, а вопросы авторизации и безопасности принято обеспечивать сертификатом X509. Управление виртуальной организацией и авторизация пользователя осуществляется сервисом VOMS (Virtual Organization Management Service).

Возможности инфраструктуры GRID показаны на рис.7.

Рис.7: Возможности инфраструктуры GRID[12]

Рабочая группа OGSA обозначает логический средний уровень рис.7 термином «сервисы». Интерфейсы этих сервисов демонстрируют индивидуальное и коллективное состояние ресурсов, принадлежащих данным сервисам, и взаимодействие этих сервисов между собой по принципам сервис-ориентированной архитектуры (SOA). На рис.8 приведена структура сервисов, представленных группой OGSA. Здесь цилиндрами обозначены индивидуальные сервисы. Эти сервисы создавались по стандартам Web сервисов с

семантиками, дополнениями, расширениями и модификациями, релевантными Грид-структурам.

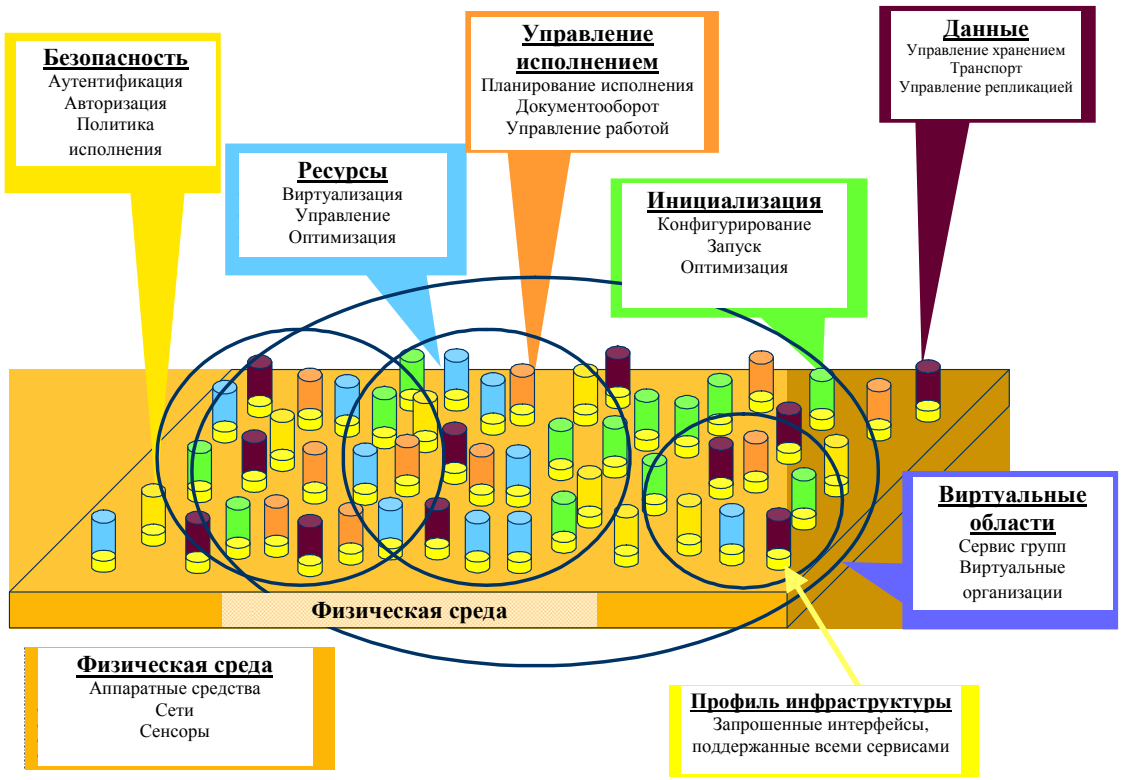


Рис.8 Структура сервисов, представленная группой OGSA [12]

В соответствии с Рис. 1 нами предложена модель интероперабельности открытой GRID-среды (см. Рис. 9).

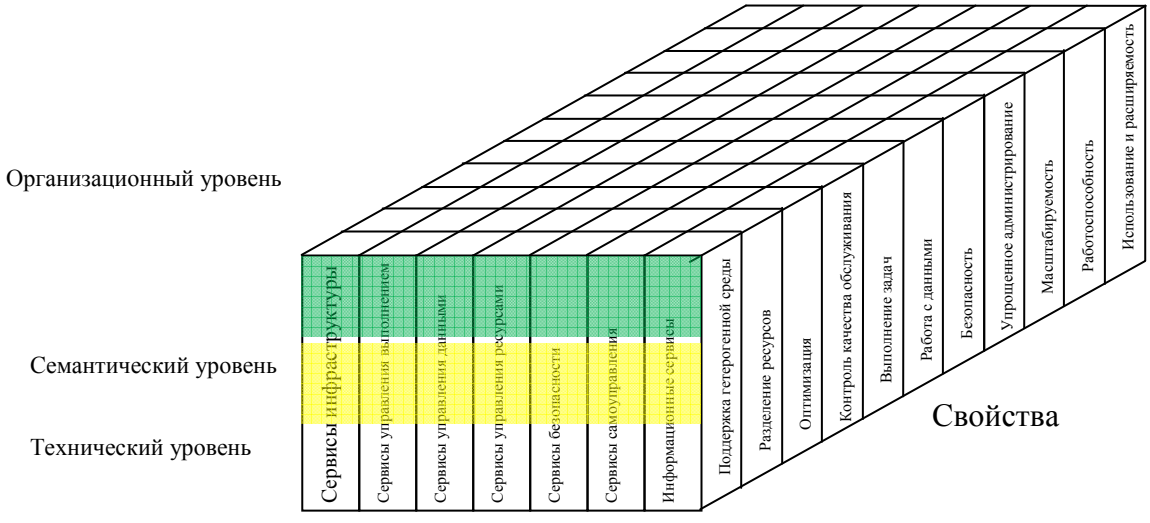


Рис. 9: Модель интероперабельности открытой GRID-среды

Эта модель будет зафиксирована в ГОСТ Р, разработка которого должна быть завершена авторами в 2013 г.

Следующим этапом согласно Рис. 1 должна стать разработка профиля интероперабельности. Он должен состоять из многих частей и построен на базе документов OGF, приведенных в Таблице 1.

Таблица 1

№№ п.п.	Наименование (перевод на русский)	Документ OGF
1	Вызов удалённого процесса и интерфейса прикладных программ для приложений конечного пользователя	A GridRPC Model and API for End-User Applications GFD.52 Rec.
2	Архитектура сервисов открытых GRID. Базовый профиль: концептуальная основа ресурсов WEB сервисов (версия 1.0)	OGSAWSRF Basic Profile.01 GFD.72-P- Rec
3	Спецификация байтного ввода/вывода, версия 1.0	ByteIO Specification 1.0 GFD. 87 P - Rec.
4	Спецификация сервиса пространства имен ресурсов	Resource Namespace Service Specification GFD. 101 P -Rec.
5	Базовый сервис исполнения	Basic Execution Service V. 1.0 GFD.108 Rec.
6	Спецификация прикладного программного интерфейса для управления распределённым ресурсом	Distributed Resource Management Application API Specification 2.0 GFD.194 Rec. (был 133)
7	Базовый профиль безопасности, версия 2.0	OGSA® Basic Security Profile 2.0 GFD.138 P- Rec.
8	Спецификация языка описания представления задач, версия 1.0	Job Submission Description Language (JSDL) Specification, Version 1.0 GFD.136 Rec.

Кроме того, согласно Рис.1 должен быть создан «План разработки стандартов» на основе документа GFD-I.123 Defining the Grid: A Roadmap for OGSA® Standards Version 1.1, а также глоссарий на основе документа GFD-I.120 Open Grid Services Architecture Glossary of Terms Version 1.6.

Следует отметить, что разработанный авторами ГОСТ Р «Спецификация языка описания представления задач, версия 1.0» [1] вступит в действие с января 2013 г. Этот стандарт по нашему мнению, может быть использован и для облачных вычислений.

Заключение

Т.о., можно сделать следующие выводы и предложения:

1. Проблема интероперабельности существует как в GRID-системах, так и в облаках и имеет общие черты и свои особенности.
2 Решение проблемы интероперабельности на основе использования ИТ-стандартов в обоих случаях может быть достигнуто на основе единого подхода, зафиксированного в

разработанном авторами ГОСТ Р «Информационные технологии. Системы промышленной автоматизации и их интеграция. Интероперабельность. Общие положения», и авторы работают в этом направлении.

3 В первом приближении можно утверждать, что профиль интероперабельности для GRID-систем и облаков должен быть один и тот же, и первым из стандартов, входящих в профиль, может использоваться разработанный авторами ГОСТ Р «Спецификация языка описания представления задач, версия 1.0»

Литература

[1] Журавлев Е.Е., Корниенко В.Н., Олейников А.Я. Вопросы стандартизации и обеспечения интероперабельности в GRID-системах. //Распределенные вычисления и Грид-технологии в науке и образовании: Труды 4-й междунар. конф. (Дубна, 28 июня – 3 июля, 2010 г.).- Дубна: ОИЯИ, Д-11-2010-140, 2010.- с. 364-272 ISBN 978-5-9530-0269-1

[2] Ian Foster, Yong Zhao, Ioan Raicu, Shiyong Lu Cloud Cpmputing and Grid Computing 360-Degree Compared. URL: http://arxiv.org/ftp/arxiv/papers/0901/0901.0131.pdf

[3] Judith Myerson Cloud computing versus grid computing URL: http://www.ibm.com/developerworks/web/library/wa-cloudgrid/

[4] Технология открытых систем. / Под редакцией А.Я. Олейникова. – М.: Янус-К, 2004. – 288 с., илл.

[5] State of the art on Semantic IS Standardization Interoperability & Quality. [Электронный ресурс] Erwin Folmer, Jack Verhoosel. University of Twente 163 pp. URL: https://noiv.nl/files/2011/03/SOTA.pdf

[6] Гуляев Ю.В., Журавлев Е.Е., Олейников А.Я. Методология стандартизации для обеспечения интероперабельности информационных систем широкого класса. Аналитический обзор // Журнал радиоэлектроники. 3 (2012), URL: jre.cplire.ru/jre/Mar/12/2/text /pdf.

[7] ГОСТ Р «Информационные технологии. Системы промышленной автоматизации и их интеграция. Интероперабельность. Общие положения»

[8] GFD.120 Open Grid Services Architecture® Glossary of Terms Version 1.6. INFO J. Treadwell 2007-12-12 Architecture OGSA-WG

[9] National Institute of Standards and Technology Special Publication 500-291 NIST Cloud Computing Standards Roadmap

[10] GFD 150 – I Документ OGF

[11] Peter Kriens, OSGi Service Platform, Service Compendium www.sienainitiative.eu, info@sienainitiative.eu

[12] GFD-I.080 Open Grid Services Archtecture http://forge.gridforum.org/projects/ogsa-wg

ВОПРОСЫ ИНТЕРОПЕРАБЕЛЬНОСТИ В ОБЛАЧНЫХ ВЫЧИСЛЕНИЯХ[1]

С.В. Иванов
Российский Новый Университет (РосНОУ)
105005, Москва, ул. Радио, 22

Введение

Среда облачных вычислений Cloud Computing, состоящая из разнородных программно-аппаратных платформ, заведомо представляет собой гетерогенную среду, в которой неизбежно возникает проблема взаимодействия входящих в нее систем, получившая название «проблемы интероперабельности» [1]. Проблема интероперабельности имеет ряд фундаментальных и прикладных аспектов и, согласно мировой практики, должна решаться на основе использования согласованных наборов стандартов информационных технологий **– профилей.**

В данной работе рассматриваются существующие подходы к решению проблемы интероперабельности в среде облачных вычислений, реализуемые за рубежом. К сожалению, в России, где облачные вычисления также активно развиваются и применяются, до настоящего времени стратегия обеспечения интероперабельности и стандартизации отсутствует. В заключение авторы на основе международного и собственного опыта дают рекомендации о первоочередных шагах в этом направлении.

Облачные вычисления

На данный момент существует множество трактовок термина «облачные вычисления». Наиболее адекватным, по нашему мнению, является определение NIST (The NIST Definition of Cloud Computing) [2]:

> Облачные вычисления - модель предоставления повсеместного и удобного сетевого доступа по мере необходимости к общему пулу конфигурируемых вычислительных ресурсов (например, сетей, серверов, систем хранения, приложений и сервисов), которые могут быть быстро предоставлены и освобождены с минимальными усилиями по управлению и необходимостью взаимодействия с провайдером услуг (сервис-провайдером).

Облачные вычисления основаны на традиционных технологиях, но до середины 2000-х годов сфера применения этих технологий оставалась ограниченной, а потенциал – нераскрытым. В настоящее время завершается ранний этап развития облачных технологий, который характеризуется новаторскими экспериментами и неустойчивостью бизнес-моделей [3].

Интероперабельность. Стандартизация облаков

Одной из фундаментальных особенностей развития и применения современных ИКТ выступает формирование гетерогенной ИКТ-среды. В такой среде возникает проблема взаимодействия разнородных компонентов (систем), получившая название «проблема интероперабельности».

Согласно определению Международной организации по стандартизации: «Интероперабельность» – способность двух или более систем или элементов к обмену информацией и к **использованию** информации, полученной в результате обмена. (ISO/IEC FCD24765-Systems and Software Engineering-Vocabulary). Способность к использованию означает интероперабельность на более высоких уровнях, чем технический.

Основным инструментом решения проблемы интероперабельности служит планомерное и последовательное использование принципов открытых систем, в основе которых лежит

[1] Работа выполнена при поддержке РФФИ (проект 12-07-00261а) и Программы Президиума РАН № 14.

использование методов функциональной стандартизации и согласованных наборов ИКТ-стандартов – профилей [4].

Проблема интероперабельности присутствует и в облачных вычислениях. Приложения, разрабатываемые на одной облачной платформе, часто оказываются несовместимыми с другой. Зачастую приложению требуются строго определенные настройки сервера, файловой системы и сети, которые другие облачные провайдеры предоставить не могут либо запрашивают за это слишком высокую цену

Решение заключается в разработке открытых стандартов облачных приложений и сервисов.

Стандартизация позволяет не только договориться об общей терминологии, но и определить те технологии, использование которых обязательно для создания совместимых решений.

Целый ряд организаций ведёт разработку облачных стандартов см. Таблицу 1 [5].

Таблица 1

Организация	**Область деятельности**
ISO/IEC JTC 1/SC 27 www.iso.org	Стандарты в сфере облачной безопасности
Open Grid Forum http://www.gridforum.org/	Разработка стандартов для сетей и создания сетевых сообществ
Cloud Security Alliance http://cloudsecurityalliance.org	Стандарты в сфере облачной безопасности
Cloud Standards Customer Council www.cloudstandardscustomercouncil.org	Разработка облачных стандартов, отражающих интересы пользователей облачных вычислений
Distributed Management Task Force (DTMF) www.dmtf.org/standards/cloud	Стандарты управления корпоративными и облачными вычислительными ресурсами
IEEE http://standards.ieee.org	Стандарты в области интероперабельности и практического внедрения облачных систем
National Institute of Standards and Technology (NIST) www.nist.gow/itl/cloud	Определение облачных вычислений; требования к использованию облачных вычислений в госсекторе США
OASIS www.oasis-open.org	Актуализация стандартов WS*, SAML, XACML и KMIP в связи с распространением облачных вычислений
Open Cloud Consortium (OCC) http://opencloudconsortium.org/	Разработка стандартов в сфере облачных вычислений и их совместимости
Storage Networking Industry Assotiation (SNIA) www.snia.org/cloud	Спецификация по управлению облачными системами Cloud Data Management Interface (CDMI)
Рабочая группа по облачным вычислениям в составе Open Group www.opengroup.org/	Стандартизированные модели, позволяющие избежать зависимости от поставщика

Хотелось бы отметить ряд инициатив по разработке стандартов, направленных на обеспечение интероперабельности в области облачных вычислений. Это 2 стандарта Open Grid Forum:

- GFD-P-R183 «Open Cloud Computing Interface - CORE»;
- GFD-P-R184 «Open Cloud Computing Interface - Infrastructure»;

и 2 стандарта Института IEEE:

- IEEE P2301, Draft Guide for Cloud Portability and Interoperability Profiles;
- IEEE P2302, Draft Standard for Intercloud Interoperability and Federation;

Первые два входят в число 200 документов OGF, дают высокоуровневое определение Протокола и API. Вторые два будут содержать перечни стандартов и спецификаций, необходимых для создания совместимых облачных систем, а также базовые сведения и рекомендации по обеспечению интероперабельности и переносимости в «облаках».

Что же касается России, то известно, что Минкомсвязь РФ реализует Федеральную программу «Информационное общество». В рамках подпрограммы «Информационное государство» разрабатывается «Национальная платформа для распределенной обработки данных». Также известно, что был организован консорциум для разработки стандартов [6], однако, насколько нам известно, конкретных результатов пока не получено.

Единый подход к обеспечению интероперабельности

На основании обобщения собственного опыта [4] и мирового опыта (NIST - http://www.nist.gov/index.html, NEHTA - http://www.nehta.gov.au/, OGSA - http://www.globus.org/ogsa/) в [7] предложен единый подход к обеспечению интероперабельности информационных систем разных классов, в том числе облачных вычислений. Его можно представить в виде ряда последовательных этапов см. рис.1.

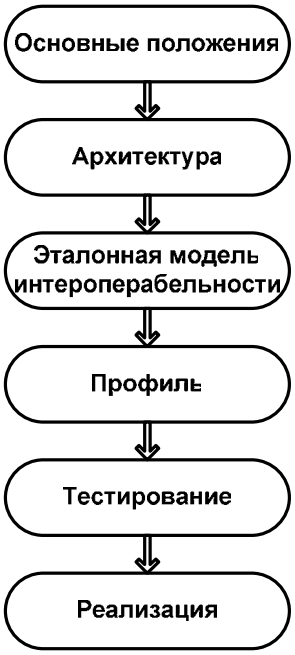


Рис.1: Единый подход к обеспечению интероперабельности систем широкого класса

Рассмотрим содержание и смысл указанных этапов.

Основные положения (англ. Framework) – содержат основные положения по достижению интероперабельности. Термин **Framework,** вообще говоря, имеет много значений: начиная от буквального смысла (каркас, рамка), широкого смысла (концептуальная основа, контекст, основные принципы, описание основных проблем предметной области и обобщенные правила для их решения и узкого смысла, понимаемого программистами – структура программной системы; программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.

Архитектура (англ. Arhitecture) – однозначное определение элементов системы и связей между ними и с окружающей средой.

Эталонная модель интероперабельности (англ. Interoperability Reference model) – представляет собой развитие известной эталонной 7-уровневой модели взаимосвязи открытых систем (ГОСТ Р 7498-1-99).

Профиль интероперабельности (англ. Interoperability Profile) – согласованный набор стандартов, структурированной в терминах эталонной модели интероперабельности.

Тестирование (англ. Testing) – оценка соответствия реализации стандартам, указанным в профиле интероперабельности

Реализация (англ. Solution) – программно-аппаратная реализация конкретной интероперабельной системы в соответствии с профилем интероперабельности.

Одним из ключевых этапов общего цикла обеспечения интероперабельности служит построение эталонной модели интероперабельности. На рис.2 приведена многоуровневая эталонная модель интероперабельности. [8] Каждому уровню модели соответствуют свои стандарты. Для конкретных решений эталонная модель может уточняться. Следует отметить, что до настоящего времени эталонная модель интероперабельности не зафиксирована ни в одном стандарте, как это сделано с 7-уровневой моделью.

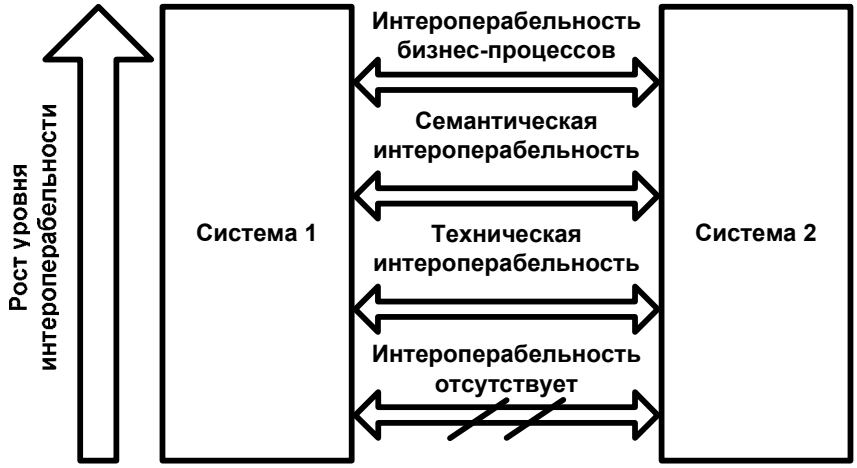


Рис.2: Многоуровневая эталонная модель интероперабельности

В настоящее время описанный подход в модернизированном виде зафиксирован в разработанном в Институте радиотехники и электроники ГОСТ Р, который должен быть введен в действие в 2013 г.

Состояние работ по интероперабельности

Рассмотрим состояние работ по интероперабельности в области облачных вычислений согласно приведенным выше этапам общего подхода.

Известен проект, находящийся в стадии разработке: A Cloud Interoperability **Framework** And Platform For User-Centric, Semantically-Enhanced Service-Oriented Applications Design, Deployment And Distributed Execution (Cloud4soa) [9]. Данный проект фокусируется на:

1. решении вопросов семантической интероперабельности, которые существуют в современных платформах облаков;
2. внедрении подхода, ориентированного на пользователя.

Доступен целый ряд документов, разработанных разными авторитетными организациями (DMTF - The Distributed Management Task Force, NIST - National Institute of Standards and Technology, etc.) в которых описывается **архитектура** облаков см. рис. 3. Можно убедиться, что единого описания архитектуры нет, что и естественно, поскольку и единого определения облаков тоже, как известно.

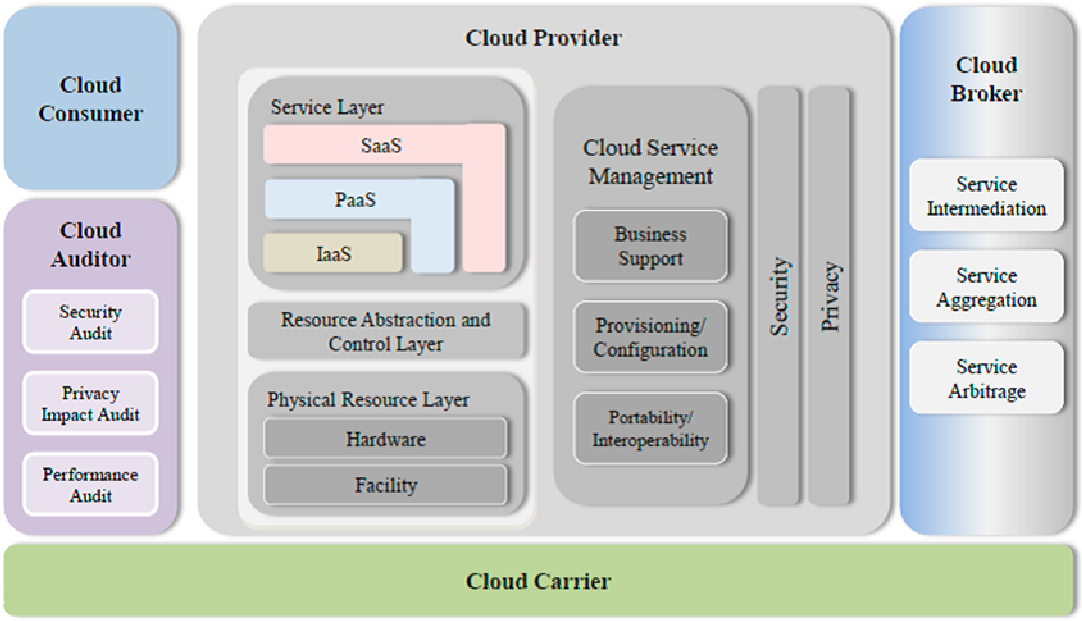


Рис. 3: Эталонная модель облачной архитектуры NIST

Что касается **моделей**, то единой модели, как и архитектуры не имеется. В разработке моделей также участвует множество известных организаций.

Известен документ, который в котором даются рекомендации по **профилям** переносимости и интероперабельности: P2301 - Guide for Cloud Portability and Interoperability Profiles (CPIP). Данный документ является проектом IEEE.

Заключение

Обобщая вышесказанное, можно сделать следующие выводы:

- облачные вычисления – сугубо гетерогенная среда, в которой возникает проблема взаимодействия входящих в нее систем, известная как «проблема интероперабельности»;
- основываясь на едином подходе, предложенном нами и зафиксированном в ГОСТ Р, необходимо:
    - создать и утвердить Общие положения (Концепцию) по обеспечению интероперабельности в облачных вычислениях;
    - выбрать архитектуру и модель;
    - в терминах выбранной модели построить профиль стандартов;
- необходимо также создать План разработки национальных стандартов для облачных вычислений и Глоссарий по облакам.

Литература

[1] Ю.В.Гуляев, Е.Е.Журавлев, А.Я.Олейников. Методология стандартизации для обеспечения интероперабельности информационных систем широкого класса? // http://jre.cplire.ru: Журнал радио электроники. URL http://jre.cplire.ru/alt/mar12/2/text.pdf (дата обращения 29.08.2012).

[2] Определение Облачных Вычислений (Драфт). // http://cloud.sorlik.ru/index.html: Основы облачных вычислений URJ: http://cloud.sorlik.ru/definition.html (дата обращения 1.07.2012)

[3] Облачные сервисы. Взгляд из России**.** Под ред. Е. Гребнева. М.: CNews, 2011. 282 с.

[4] Технология открытых систем. Под ред. А.Я. Олейникова. М.: Янус-К, 2004. 288 с.

[5] Куда плывут облака? Современные тренды. // http://cloud.cnews.ru : Облачные сервисы URL; http://cloud.cnews.ru/reviews/index.shtml?2011/04/26/438141_4 (дата обращения 28.07.2012)
[6] Результаты работы по внедрению облачных решений. // http://minsvyaz.ru/: Минкомсвязь России URL: http://minsvyaz.ru/ru/news/index.php?id_4=43384 (дата обращения 1.09.2012).
[7] А.Я. Олейников, Е.И. Разинкин. Особенности подхода к обеспечению интероперабельности в области электронной коммерции. Журнал Информационные технологии и вычислительные системы 2012 №3
[8] Kamenshchikov A.A., Kornienko V.N., Oleynikov A.Ya., Zhuravlev E.E. Standardization in E-Science in the Russian Federation Proceedings 16th EURAS Annual Standardization Conference 'Standards for Development' - / Edited by Vladislav Fomin, Kai Jakobs / EURAS 2011, p.p 197-206
[9] European commission: CORDIS // http://cordis.europa.eu/ Projects URL; http://cordis.europa.eu/search/index.cfm?fuseaction=proj.document&PJ_RCN=11532965 (дата обращения 10.07.2012)

ПОИСК АССОЦИАТИВНЫХ ПРАВИЛ В БОЛЬШИХ МАССИВАХ ДАННЫХ С ПОМОЩЬЮ ГЕТЕРОГЕННОЙ ГРИД НА БАЗЕ BOINC[1]

Е.Е. Ивашко, А.С. Головин
Институт прикладных математических исследований Карельского научного центра РАН
Россия, 185910 г. Петрозаводск, ул. Пушкинская, д. 11 ИПМИ
{ivashko,golovin}@krc.karelia.ru

Введение

При принятии решений в различных сферах человеческой деятельности, как правило, необходимо выполнить анализ данных для получения новой информации. С развитием вычислительной техники все большую роль приобретает автоматизированный анализ больших объемов данных. Для целей интеллектуальной обработки данных были разработаны специальные алгоритмы и подходы, объединенные термином Data Mining.

Согласно определению [1], Data mining — это процесс обнаружения в сырых данных (1) ранее неизвестных, (2) нетривиальных, (3) практически полезных и (4) доступных интерпретации знаний, необходимых для принятия решений в различных сферах человеческой деятельности.

В сферу применения Data Mining входят все области, в которых собираются большие объемы данных для получения из них полезных знаний. Одним из направлений Data Mining является поиск ассоциативных правил, отражающих нетривиальные взаимосвязи между наборами данных.

Рост объемов собираемой информации и развитие средств и методов ее надежного хранения привели к повышению актуальности разработки новых методов и алгоритмов анализа больших и сверхбольших наборов данных. Так, например, в работе «The Fourth Paradigm: Data-Intensive Scientific Discovery» [2] автором высказано утверждение, что выявление закономерностей в больших массивах данных становится основным инструментом для исследования и получения новых знаний в передовых областях науки. Действительно, стремительный рост объемов данных, предназначенных для обработки, характеризует не только IT-компании (например, Google [3]) и научную сферу (см., например, [4]), но и широкий спектр организаций в самых различных областях ([5]). В современной науке и технике возникло отдельное направление, связанное с анализом больших и сверхбольших наборов данных, Big Data [6]. Анализ таких объемов данных требует привлечения технологий и средств реализации высокопроизводительных вычислений.

BOINC-грид

Как правило, для обработки больших массивов данных используются суперкомпьютеры или вычислительные кластеры. Для достижения большей производительности вычислительные кластеры объединяются высокоскоростными каналами связи в специализированные грид-системы. Однако с развитием сети Интернет появился и другой подход в построении грид-систем, позволяющий объединить значительное число источников сравнительно небольших вычислительных ресурсов для решения вычислительноемких задач и задач обработки данных. В большинстве случаев такие системы построены на использовании свободных вычислительных ресурсов частных лиц и организаций, добровольно присоединяющихся к этим системам (volunteer computing). Однако существуют и примеры построения подобных частных (в масштабах организации или группы организаций) распределенных систем (см., например, [7]). Наиболее эффективно использование таких распределенных систем для проведения серий независимых вычислительных экспериментов (см., например, [8]).

[1]Работа поддержана грантом РФФИ 12-07-31147 мол_а

BOINC (Berkeley Open Infrastructure for Network Computing) — это открытая программная платформа для организации грид-систем и систем распределенных вычислений, разработанная в университете Беркли [9]. Это программное обеспечение (ПО) стало основой для большого числа мировых научных проектов [10, 11]. Платформа BOINC отличается простотой в установке, настройке и администрировании, а также обладает хорошими возможностями по масштабируемости, простоте подключения вычислительных узлов, использованию дополнительного ПО, интеграции с другими грид-системами и др.

Платформа BOINC имеет архитектуру «клиент-сервер», при этом клиентская часть может работать на компьютерах с различными аппаратными и программными характеристиками. Ключевым объектом системы является проект — автономная сущность, в рамках которой производятся распределенные вычисления. BOINC-сервер поддерживает одновременную работу большого числа независимых проектов; каждый вычислительный узел может одновременно производить вычисления для нескольких BOINC-проектов. Проект однозначно идентифицируется своим URL-адресом. BOINC предоставляет возможность гибкой настройки клиентской части, регулируя максимальный размер загружаемых файлов, время выполнения рабочих заданий, загрузку CPU или GPU, используемый объем оперативной памяти и дискового пространства.

Серверная часть BOINC основана на последовательном выполнении ряда служб, наиболее важные из которых — это служба планирования, выполняющая распределение заданий между вычислительными узлами, и служба освоения, обрабатывающая промежуточные результаты, полученные от вычислителей.

Ассоциативные правила

Одним из наиболее популярных методов Data Mining обнаружения знаний являются различные методы поиска ассоциативных правил. Ассоциативные правила позволяют описывать закономерности между связанными событиями.

Пусть $I = \{i_1, i_2, \ldots, i_n\}$ — это набор из n различных предметов. D — набор транзакций различной длины над I. Каждая транзакция T из D содержит набор предметов $i_1, i_2, \ldots, i_k$ из I. Ассоциативным правилом называется импликация $X \Rightarrow Y$, где $X \subset T$, $Y \subset T$ и $X \cap Y = \varnothing$. X называется условием, а Y — следствием правила. Каждый предметный набор имеет меру статистической значимости, называемую поддержкой (support). Поддержкой определенного набора элементов называется количество транзакций, содержащих этот набор. Набор элементов является часто встречающимся, если его поддержка превышает заданные порог, который называется минимальной поддержкой (minsupp). Правило $X \Rightarrow Y$ имеет поддержку s, если $s\%$ транзакций из D содержат это правило. Достоверность (confidence) правила показывает какова (статистическая) вероятность того, что из X следует Y. Т.е. правило $X \Rightarrow Y$ справедливо с достоверностью c, если $c\%$ транзакций из D, содержащих X, также содержат и Y. Достоверность определяется как отношение $support(X \cup Y)/support(X)$.

Задача поиска ассоциативных правил заключается в нахождении всех правил, чьи поддержка и достоверность, больше чем некоторые заданные пользователем порог минимальной поддержки и достоверности соответственно.

Впервые задача поиска ассоциативных правил была предложена для нахождения типичных шаблонов покупок, совершаемых в супермаркетах, поэтому иногда ее еще называют анализом рыночной корзины.

На Рис. 1 представлена схема реализации алгоритма Partition, предназначенного для поиска ассоциативных правил. Этот алгоритм был адаптирован для выполнения в гетерогенной грид на базе BOINC. Выполнение алгоритма состоит из трех этапов, два из которых выполняются параллельно на вычислительных узлах грид-сети. На завершающем этапе происходит объединение промежуточных результатов.

Рассмотрим работу алгоритма Partition в BOINC подробнее:

- Программа генерации заданий (разработанная специально для проекта) создает подзадачи и необходимые для их расчета входные файлы. Указанная программа получает на вход исходный файл с транзакционной базой данных и ряд параметров:

значение минимальных поддержки и достоверности, а также параметр, отвечающий за разбиение базы данных на части. Последний параметр может ограничить размер каждой части в байтах или задать количество этих частей. По окончании работы программа сохраняет рабочие задания в базе данных проекта.

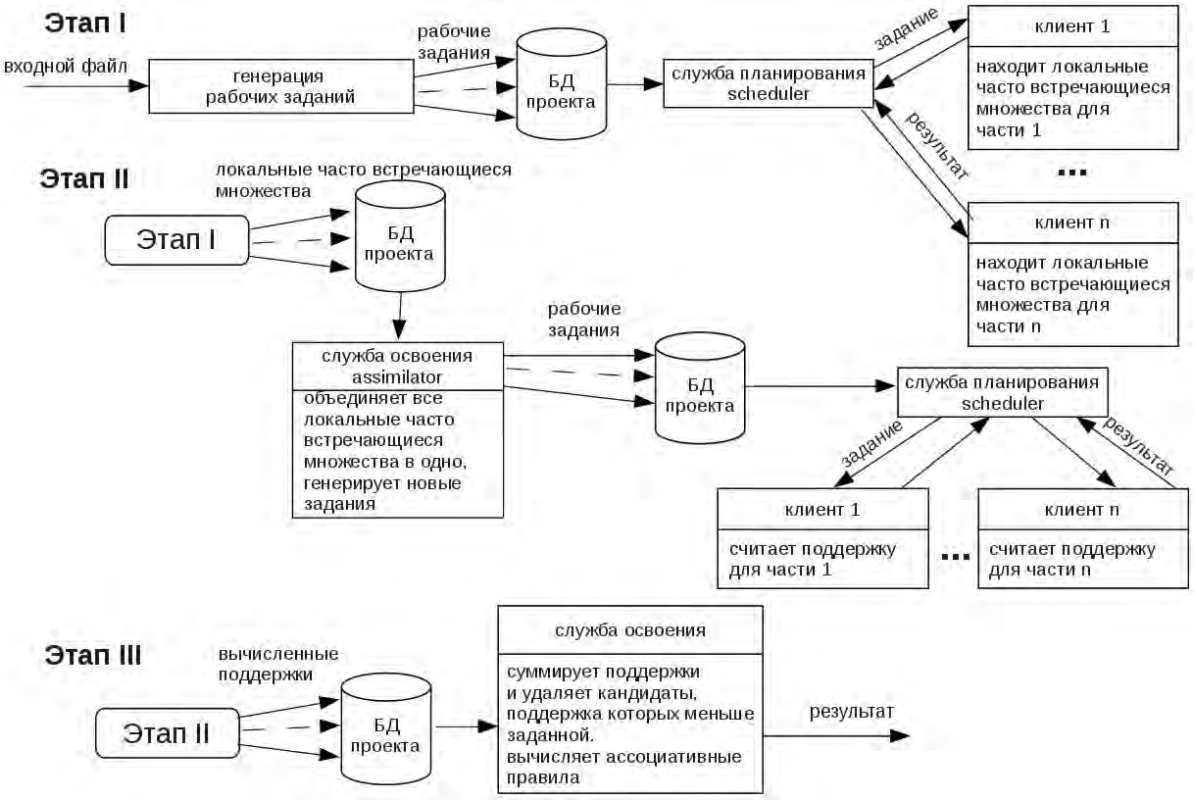


Рис. 1: Схема реализации алгоритма Partition в грид на базе BOINC

- BOINC создает для каждого из подзаданий один или несколько одинаковых экземпляров (в зависимости от настроек проекта).
- Планировщик BOINC распределяет подзадания различным клиентским программам.
- Каждый BOINC-клиент загружает с сервера входные файлы, являющиеся частями исходной транзакционной базы данных. Далее клиент запускает приложение, которое на первом этапе вычисляет локальные часто встречающиеся наборы для загруженной части, а на втором — поддержку глобальных кандидатов для своей части транзакционной базы данных.
- После расчета подзадания BOINC-клиент загружает выходные файлы на сервер.
- Клиентская программа отчитывается о выполнении подзадания (возможно, после небольшой задержки, необходимой для снижения нагрузки на программу-планировщик сервера).
- Служба проверки результатов проверяет выходные файлы и определяет наличие канонического результата.
- Когда найдено каноническое решение, служба освоения (разработанная специально для проекта) обрабатывает результаты, например, помещая их в отдельную базу данных или отсылая на электронную почту. В ходе выполнения общей программы поиска ассоциативных правил служба освоения запускается 2 раза. После выполнения первого этапа служба освоения формирует из полученных локальных часто встречающихся наборов множество всех глобальных кандидатов. Кроме того служба освоения, формирует новую порцию рабочих заданий и сохраняет их в базе данных проекта. После выполнения второго этапа служба освоения запускается

повторно. На этот раз данная служба суммирует полученные поддержки для каждого кандидата, удаляет те, чьи поддержки меньше заданного минимального порога и вычисляет ассоциативные правила.

- Когда все экземпляры подзадания завершены, служба удаления файлов удаляет ненужные больше входные и выходные файлы, а также очищает базу данных от информации о каждом подзадании и его экземплярах.

Еще раз обратим внимание на то, что некоторые службы являются стандартными и не зависят от конкретного проекта и его реализации. Однако другие службы необходимо разрабатывать отдельно для каждого проекта. В рамках данной работы, кроме приложения для клиента BOINC, был разработаны генератор рабочих заданий и служба освоения.

Результаты экспериментов

Для оценки производительности разработанного ПО был проведен ряд экспериментов. Вычисления проводились с использованием грид-сегмента ЦКП КарНЦ РАН «Центр высокопроизводительной обработки данных» [12]. На момент проведения экспериментов в состав грид-сегмента входили 84 вычислительных узла с различными аппаратными и программными характеристиками, а также разными настройками, связанными с организацией вычислений. Суммарная пиковая производительность грид составила 1,04 TFLOPS. Для проведения экспериментов использовалось от 1 до 32 вычислительных узлов.

В качестве исходных данных при верификации разработанного комплекса программ использовались тестовые наборы Frequent Itemset Mining Dataset Repository [13]. Характеристики наборов данных представлены в табл. 1.

Таблица. 1. Характеристики используемых наборов данных

	Файл	**Количество транзакций**	**Средняя длина транзакции**	**Минимальная поддержка**
I	T10I4D100K.dat	100000	10	1%
II	T25I20D100K.dat	100000	25	1,5%
III	T40I10D100K.dat	100000	40	5%

Результаты проведенных экспериментов показали, что время поиска ассоциативных правил на используемых наборах данных достигает минимального значения при использовании 28-30 вычислительных узлов с ускорением в 6-9 раз (см. Рис. 2).

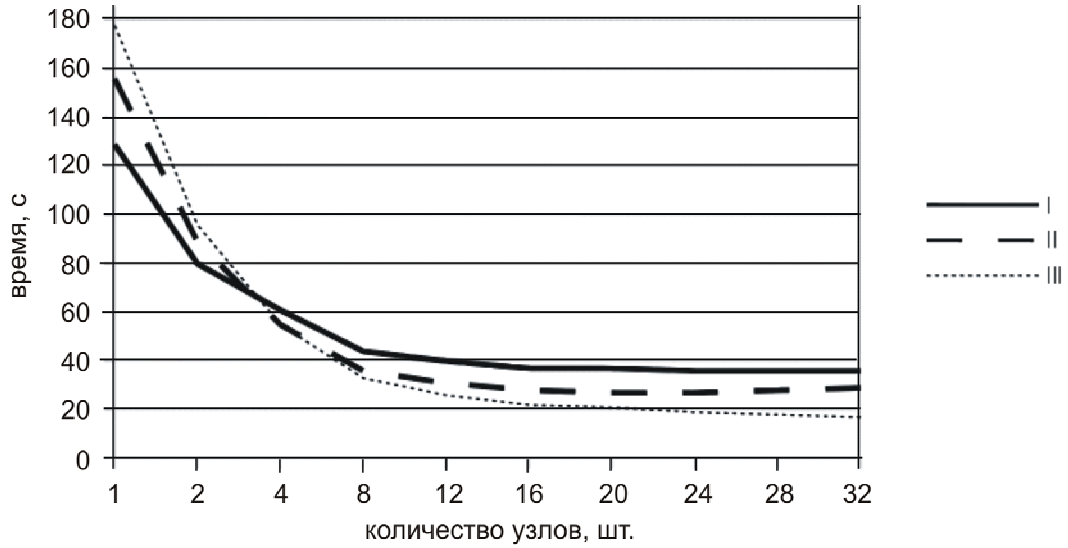


Рис. 2: Время выполнения анализа данных в зависимости от числа используемых вычислительных узлов (верификация)

На основе разработанной программы были проведены вычислительные эксперименты по оценке производительности поиска ассоциативных правил в базах данных размером в 100 Гб. Результаты экспериментов представлены на Рис. 3. Из рисунка видно, что ускорение анализа больших наборов данных на 32 узлах составило порядка 6 раз.

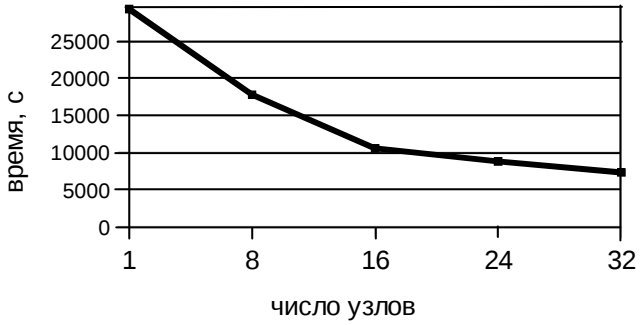


Рис. 3: Время выполнения анализа данных в зависимости от числа используемых вычислительных узлов (эксперименты)

Заключение

В статье представлены результаты исследований, связанных с реализацией на BOINC-грид алгоритмов поиска ассоциативных правил в больших наборах исходных данных. Описана реализация алгоритма, предназначенного для работы в распределенной среде, представлены результаты верификации реализованного комплекса программ и результаты экспериментов по оценке производительности разработанного ПО на тестовых базах данных.

Литература

[1] Дюк В., Самойленко А. Data Mining: учебный курс (+CD) СПб: Изд. Питер, 2001. 368 с.

[2] The Fourth Paradigm: Data-Intensive Scientific Discovery, 2009, URL: http://research.microsoft.com/en-us/collaboration/fourthparadigm

[3] Обзор технологий, google.ru, URL: http://www.google.ru/intl/ru/about/corporate/company/tech.html

[4] Loek Essers: CERN pushes storage limits as it probes secrets of universe, URL: http://news.idg.no/cw/art.cfm?id=FF726AD5-1A64-6A71-CE987454D9028BDF

[5] Yevgeniy Sverdlik. Making Way for Big Data // DatacenterDynamics Focus, April/May 2012, Volume 3, Issue 21.

[6] Л. Черняк. Большие Данные — новая теория и практика// *Открытые системы. СУБД.* — М.: Открытые системы, 2011. — № 10. — ISSN 1028-7493.

[7] Прорывная технология машинного перевода и вокруг нее. PC WEEK, №9, 12 апреля 2011 г.

[8] Е.Е. Ивашко, Н.Н. Никитина. Организация квантовохимических расчетов с использованием пакета Firefly в гетерогенной грид-системе на базе BOINC // Вычислительные методы и программирование, Том 13, 2012 г., с. 8 — 12.

[9] BOINC: Программное обеспечение с открытым исходным кодом для организации добровольных распределённых вычислений и распределённых вычислений в сети. URL: http://boinc.berkeley.edu/index.php

[10] Проект добровольных вычислений Climateprediction.net. URL: http://climateprediction.net

[11] Проект добровольных вычислений SETI@home. URL: http://setiathome.berkeley.edu

[12] Центр высокопроизводительной обработки данных ЦКП КарНЦ РАН / Институт прикладных математических исследований Карельского научного центра РАН. URL: http://cluster.krc.karelia.ru

[13] Frequent Itemset Mining Dataset Repository, URL: http://fimi.ua.ac.be/

[14] Foster I. The Grid: Blueprint for a New Computing Infrastructure. — Morgan Kaufmann Publishers. — ISBN ISBN 1-55860-475-8.

ГРИДННС: СОСТОЯНИЕ И ПЕРСПЕКТИВЫ[1]

В.А. Ильин[1,3], В.В. Кореньков[2], А.П. Крюков[†1,3]
[1]*Московский государственный университет имени М.В. Ломоносова,*
[2]*Лаборатория информационных технологий ОИЯИ, Дубна*
[3]*НИЦ «Курчатовский институт»*

Проект грид-инфраструктуры Национальной нанотехнологической сети (ГридННС) выполнялся с 2008 по 2011 года. Основная задача проекта — это создать промежуточное ПО и развернуть инфраструктуру, которая объединит суперкомпьютерные центры российских научно-исследовательских институтов и производственных центров, входящих в Национальной нанотехнологической сети (ННС). Что позволит предоставить участникам ННС унифицированный удаленный доступ к суперкомпьютерным ресурсам и повысит эффективность научных разработок в области нанонаук, материаловедения и других инновационных областях.

В настоящей работе представлен текущий статус проекта ГридННС и его перспективы.

1 Введение

Грид, как технология глобальных распределенных высокопроизводительны вычислений была предложена Я.Фостером и К.Кессельманом [1] в конце 1990-х годов. В ее основе лежат представление о ресурсах как сервисах, которые согласованным образом предоставляются пользователям виртуальных организаций [2].

По мере развития грида, базовая концепция осталась прежней, однако технологическая основа претерпела существенные изменения. Так если в первых реализациях грид-сервисов использовались нестандартные протоколы и форматы обмена данными, то современные гриды основаны на веб-сервисах.

В 2006 году был принят стек стандартов WSRF [3] предназначенный для построения гридов. Это был важный шаг к их стандартизации на основе веб-сервисных технологий. В качестве образцовой реализации этого стека стандартов можно указать GlobusTookit версии 4 [4].

WS-* стек стандартов — это набор стандартов, предназначенных для построения универсальных открытых гридов. Однако опыт эксплуатации гридов, построенных на основе WSRF, показал, что WS-* стек очень тяжелый в реализации и использовании и даже GT4 не является его полной и корректной реализацией. К тому же скорость работы промежуточного ПО (ППО) на базе GT4 желает оставлять лучшего.

В 2000 году Р.Филдинг предложил другой способ построения веб-сервисов на основе модели Representational state transfer (REST) [5]. В этой модели все представляется в виде ресурса. Однако в отличии от WSRF, операции с этими ресурсами строго ограничены и стандартизованы [6]. Большим достоинством RESTful-сервисов является то, что в них протокол HTTP используется в своем прямом назначении как протокол обмена сообщениями, в то время как WSRF-сервисы используют его только в качестве транспортного протокола, а в качестве протокола обмена сообщениями используется SOAP. Эти свойства делают использование архитектурного стиля REST очень привлекательным для разработки ППО грида.

В 2000-х в мире уже было создано несколько больших гридов. В первую очередь — это европейский проект EDG/EGEE/EGI [7] и интегрированный с ним проект грида для обработки и анализа данных с Большого адронного коллайдера — WLCG [8]. Данная инфраструктура развивается с 2003 года и в настоящее время включает по всему миру более 140 сайтов, в которых сосредоточено 250 тысяч ЦПУ и 150 Пбайт дисков. Основной тип задач, на который рассчитан данный грид — это обработка огромного потока относительно небольших задач,

[1] При поддержке РФФИ (грант 11-07-00434-а) и гранта Президента РФ НШ-3920.2012.2
† E-mail: kryukov@theory.sinp.msu.ru

предназначенных для работы с большим объемом данных. WLCG продемонстрировала высокую эффективность для решения такого рода проблем.

Важным случаем высокопроизводительных вычислений являются суперкомпьютерные вычисления. Суперкомпьютеры (СК) - дорогие вычислительные установки и вопрос их эффективного использования является ключевым. Для этого к каждому суперкомпьютеру организован удаленный доступ пользователей. Как правило такой доступ осуществляется по протоколу ssh. Таким образом вокруг каждого СК формируется сообщество пользователей, а управление учетными записями и распределение компьютерных ресурсов осуществляется администраторами СК. Это приводит к тому, что наблюдается значительная неравномерность загрузки СК, а использование свободных ресурсов на других вычислительных установках оказывается невозможным.

Проект ГридННС был призван решить эти проблемы путем объединения СК участников ННС в единую грид-инфраструктуру. Важным достоинством такого подходя является унификация процесса запуска задач пользователей на любом СК, входящем в грид независимо от используемой на СК системы управления заданиями, особенностями установки прикладных пакетов и так далее. Кроме того, грид обеспечивает аутентификацию и авторизацию пользователей в рамках модели SSO (Single Sign-On), что существенно снижает нагрузку на администраторов СК по управлению учетными записями пользователей.

2 Проект ГридННС

Проект ГридННС начал выполняться в рамках федеральной целевой программы «Развитие инфраструктуры наноиндустрии в Российской Федерации на 2008-2010 годы». Цель проекта — обеспечение ученым и инженерам, объединенных в небольшие коллективы, доступа к суперкомпьютерным ресурсам организаций-участников ННС. Таким образом, в самом начале проекта было понятно, что решения, примененные для EGI/WLCG не могут быть применены впрямую и встал вопрос о разработки собственного ППО. Основными задачами проекта стало разработка ППО ГридННС и развертывание инфраструктуры на его основе.

Учитывая опыт участия в европейских грид-проектах и предметную область участников ННС было принято решение по-возможности отказаться от использования WSRF сервисов и разрабатывать ключевые сервисы используя архитектурный подход REST. Однако, для ускорения подготовки дистрибутива было принято решение использовать в качестве грид-шлюза к СК ресурсам ППО на базе GT4, а в качестве системы хранения данных — gridFTP-серверы, использующие gsiFTP протокол. Система безопасности была построена на основе инфраструктуры открытых ключей (PKI) и цифровых сертификатов X.509. При этом в ГридННС признаются как собственный сертификаты ГридННС, так и сертификаты РДИГ — российской части проекта EGI/WLCG.

Общая структура ГридННС изображена на рисунке 1.

К слою общих сервисов относятся:

- сервис регистрации, который предназначен для регистрации всех сервисов подключенных к инфраструктуре;
- удостоверяющий центр, предназначенный для выдачи сертификатов;
- сервис мониторинга [10], предназначенный для контроля текущего состояния инфраструктуры;
- сервис управления виртуальными организациями;
- сервис распределения нагрузки;
- информационный сервис;
- сервис учета, предназначенный для сбора информации о потребленных ресурсах.

Часть сервисов реализована как RESTful-сервисы. В первую очередь это относится к сервису распределения нагрузки — Pilot [11] и информационной системе [12], которая обеспечивает другие сервисы информацией о текущем состоянии ресурсов, их нагрузке, конфигурации.

Запуск заданий пользователь может выполнять как с использованием интерфейса

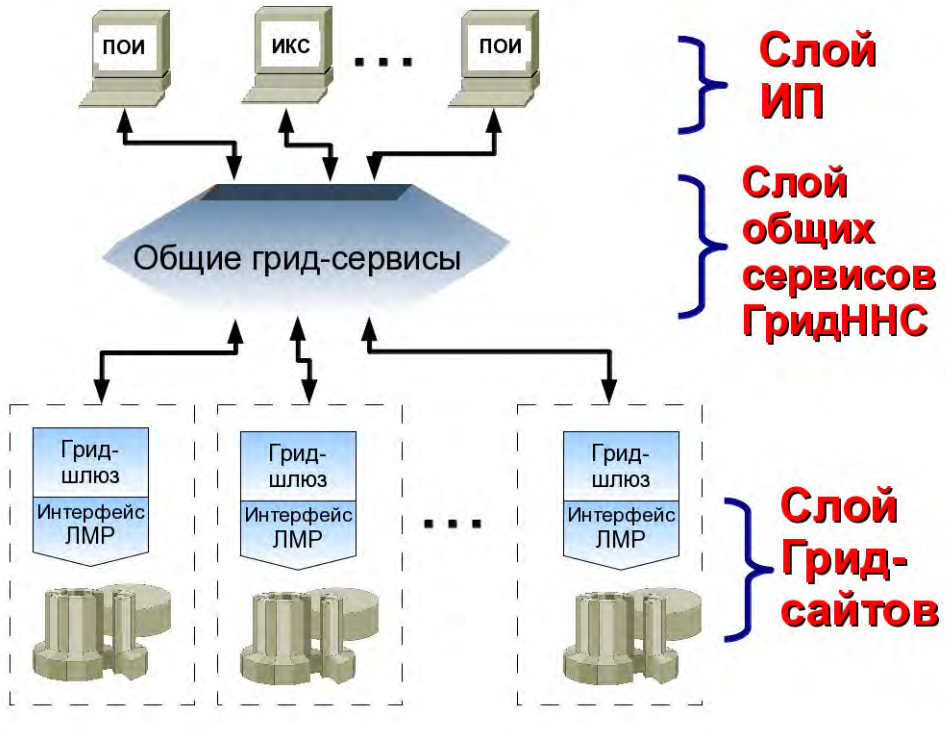


Рис. 1: Общая структура ГридННС

командной строки, так и через веб-интерфейс. В последнем случае пользователю предоставляется графические средства создания композитных заданий. Сами задания представляют собой набор описаний задач, логически связанных между собой и образующих ациклический направленный граф, на языке JSION [13].

Хотя в ГридННС пользователь может запустить произвольную задачу, дополнительно ему предоставляется широкий набор пакетов прикладных программ (ППП) из области моделирования наноматериалов, аэро- и гидродинамики. Список таких пакетов представлен в таблице 1. Вокруг каждого пакета образовано сообщество пользователей, организованных в виртуальную организацию.

Таблица 1: Список пакетов прикладных программ установленных на СК центрах ГридННС

FDTD-II	GAMESS
Firefly	MOLPRO
FlowVision	NAMD
ABAQUS	Gromacs
TecPlot	OpenMX
ABINIT	LAMMPS
VASP	OpenFOAM
GAUSSIAN	ANSYS
GaussView5	

Важным компонентами ГридННС являются набор проблемно-ориентированных интерфейсов для этих ППП. Большинство из них построено на базе пользовательского веб-интерфейса и являются модулями расширения (plug-in) для него [14, 15]. На рисунке 2 показан

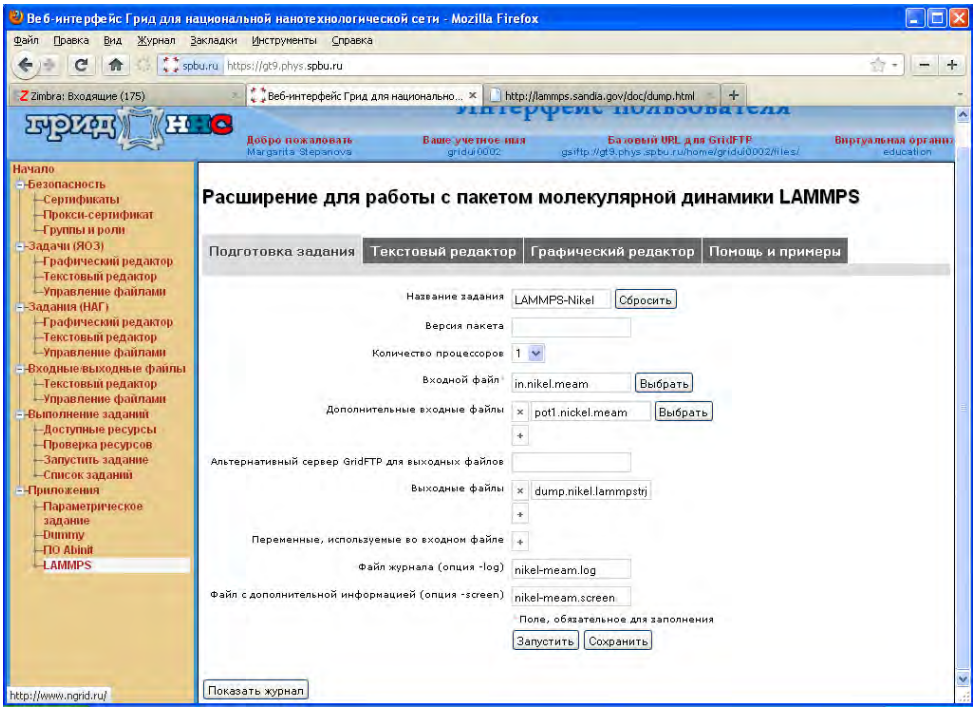


Рис. 2: Пример проблемно-ориентированного интерфейса для пакета LAMMPS

пример такого проблемно-ориентированного интерфейса для пакета LAMMPS.

В настоящее время к инфраструктуре ГридННС подключено более 10 СК, с том числе такие крупные СК установки как СК НИЦ «Курчатовского института» и «Чебышев» Московского государственного университета. Общее доступное число ядер ЦПУ около 16 000.

3 Заключение

Проект ГридННС является первым российским грид-проектом, в рамках которого была создана полнофункциональная грид-инфраструктура. В нем, в первые в мире, были разработаны грид-сервисы на основе архитектурного стиля REST. Данный подход показал свою эффективность и удобство в использовании.

Несмотря на то, что финансирование проекта по линии ФЦП прекращено, вокруг ГридННС возникло сообщество разработчиков и пользователей, которое продолжает совершенствовать ППО и изыскивает возможность расширения инфраструктуры.

В настоящее время завершается тестирование нового грид-шлюза, построенного на основе архитектурного стиля REST. С вводом этого компонента будет полностью завершен переход на RESTful-сервисы.

Другим важным направлением работ является расширение типов обрабатываемых параллельных задач на современных СК, в которых все больше используются многоядерные и графические процессоры. Обеспечение эффективного запуска таких гибридных задач является актуальной проблемой, в том числе и грид-технологии.

Рис. 3: СК центры, объединенные в ГридННС на конец 2011г.

Литература

[1] Foster, I. and Kesselman, C. The Globus Project: A Status Report. // In Proc. Heterogeneous Computing Workshop, IEEE Press, 1998, 4-18.

[2] The Open Grid Services Architecture, Version 1.5. GFD-I.080, 24 July 2006.

[3] OASIS. Web Services Resource Framework (WSRF) – Primer v1.2, Committee Draft 02 - 23 May 2006. URL: http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf

[4] I. Foster, Globus toolkit version 4: Software for service-oriented systems, Vol. LNCS 3779 of IFIP International Conference on Network and Parallel Computing, Springer-Verlag, 2006, pp. 2—13.

[5] R.T.Fielding: Architectural styles and the design of network-based software architectures // Dotoral dissertation, University of California, Irvine, 2000.

[6] А.Демичев, А.Крюков и Л.Шамардин: Принципы построения грид с использованием Restful-веб-сервисов.// Программные продукты и системы, №4, 2009, г.Тверь.

[7] Europien Grid Infrastructure // URL: http://www.egi.eu/

[8] Worldwide LHC Computing Grid // URL: http://wlcg.web.cern.ch/

[9] Грид для Национальной нанотехнологической сети // URL: http://ngrid.ru/ngrid, 2008.

[10] S.D. Belov, T.M. Goloskokova, V.V. Korenkov et. al., Monitoring, accounting and registration services for russian grid network // ibid pp. 30-33.

[11] А.Демичев, В.Ильин, А.Крюков и Л.Шамардин: Реализация программного интерфейса грид-сервиса Pilot на основе архитектурного стиля REST.// Вычисленные методы и программирование, с.62-65, т.11, 2010.

[12] Информационная система ГридННС // Программные продукты и системы, № 1, 2012, стр. 6 – 10.

[13] K.Zyp: A JSON Media Type for Describing the Structure and Meaning of JSON Documents. // Techniocal report, IETF Network Workong Group, draft-zyp-json-schema-02, March 2010.

[14] A.P. Gulin, A.K. Kiryanov, N.V. Klopov, S.B. Oleshko, Y.F. Ryabov, On approaches to building problem-oriented web-interfaces for application software suites in GridNNN // ibid pp. 147-149.

[15] N.A. Kutovskiy, R.N. Semenov, I.I. Lensky, Problem-oriented web-interfaces for Russian grid network // // ibid pp. 186-188.

СИСТЕМА МАССОВОЙ ИНТЕГРАЦИИ БАЗ ДАННЫХ: ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ И СПОСОБ РЕАЛИЗАЦИИ[1]

В.Н. Коваленко, Е.И. Коваленко, А.Ю. Куликов
Институт прикладной математики им. М.В.Келдыша РАН
125047, Москва, Россия

Введение

В статье [1] представлена постановка задачи массовой интеграции – объединения большого числа автономных баз данных (БД) в информационную инфраструктуру, доступ к которой обеспечивает промежуточное ПО – платформа интеграции. Один из возможных вариантов такой платформы – система MQ-DAI (Massive Queries - Data Access and Integration) – разрабатывается в ИПМ им. М.В. Келдыша РАН.

Назначение системы – реализация операций поиска и извлечения информации, которые эквивалентны поддерживаемым традиционными системами управления базами данных (СУБД), но для условий, когда информация распределена по многочисленным БД. Массовые операции поиска дают возможность получать данные из произвольного числа БД в одном запросе. Система MQ-DAI ориентирована на интеграцию БД реляционного типа, однако, не накладывается ограничений на тип СУБД, схемы интегрируемых БД – достаточно наличия драйвера удалённого доступа, удовлетворяющего спецификации JDBC [2].

Известно много интеграционных подходов, но массовая интеграция десятков и сотен БД ставит новые задачи. В данной работе описываются функциональные возможности системы MQ-DAI и способы их реализации в условиях массовой интеграции.

Требования к платформе массовой интеграции

Постановка рассматриваемой задачи заключается в следующем. Имеется большое динамическое множество реляционных БД, тип которых и схемы могут различаться. БД ведутся различными организациями, которые наполняют их информацией. Платформа интеграции должна:

- представлять множество БД в виде единого информационного пространства, что избавит конечного пользователя (или приложение) от необходимости знать адреса конкретных БД и специфику схем этих БД;
- позволять выполнять поисковые запросы, одновременно выбирающие данные из произвольного подмножества интегрированных БД;
- позволять организациям-владельцам БД в любой момент выполнять операции подключения базы данных к платформе интеграции и их отключения. Подключение БД не должно требовать установки дополнительного программного обеспечения или модификации БД;
- обеспечивать контроль доступа пользователей к интегрированным БД и содержащейся в них информации.

В соответствии с этими требованиями система MQ-DAI содержит функции интеграции данных, интерпретации и выполнения массовых запросов, управления инфраструктурой обработки запросов, контроля доступа к данным инфраструктуры. Эти функции, описываются в следующих разделах.

Общей характеристикой MQ-DAI является то, что в ней использован подход виртуальной интеграции. Виртуальная интеграция отличается от физической тем, что данные интегрируемых баз не перемещаются в общее хранилище, но, тем не менее, система интеграции обеспечивает

[1] Работа выполнена при поддержке гранта РФФИ 11-07-00147, программы фундаментальных исследований Президиума РАН, гранта Президента РФ для ведущих научных школ НШ-8129.2010.9

доступ ко всей их совокупности. Преимущества этого подхода для условий массовой интеграции мы видим в следующем.

- В случае физической интеграции актуальность информации в центральном хранилище должна обеспечиваться частым опросом БД для определения произведенных изменений. В этом смысле, виртуальная интеграция предпочтительнее, так как результат запроса всегда возвращает актуальную информацию из исходной БД.
- Часто необходимо определять, из какой БД получена та или иная часть результата запроса, поэтому при проектировании схемы хранилища, необходимы дополнительные поля, определяющие связь информации с БД-источником.
- Для организации, владеющей БД, копирование данных в центральное хранилище (физическая интеграция) и предоставление доступа к данным (виртуальная интеграция) существенно различаются с правовой точки зрения.

Интеграция данных и язык массовых запросов

Цель интеграции данных – образование единого по всем включенным в инфраструктуру БД, так что операции доступа к данным не зависят от их расположения и способа представления. Для интеграции данных в MQ-DAI используется развитый в многочисленных исследованиях подход, основанный на глобальной схеме, которая унифицирует представления семантически эквивалентных данных в интегрируемых БД. Унификация осуществляется путём задания отображений элементов глобальной схемы (таблиц) на схемы источников – реальных БД (метод GAV – Global as View) [3].

Предложенное в [1] развитие метода GAV решает проблему определения глобальной схемы в форме, не зависящей от конкретного состава БД. Для этого полное пространство данных представляется в виде объединения разделов. Каждый раздел содержит данные из одной БД, которые отображаются в некоторую таблицу глобальной схемы. Отображения разделов задаются независимо друг от друга.

Язык массовых поисковых запросов представляет собой расширение языка SQL-92 (оператор SELECT), причём запросы формулируются в терминах глобальной схемы. Благодаря этому есть возможность получения данных из всех БД инфраструктуры. Кроме того язык позволяет выделять пространство поиска запроса как определённоё подмножество БД. Важность варьирования пространства поиска можно пояснить тем, что хотя виртуальная интеграция потенциально позволяет интегрировать БД в масштабах страны, должна сохраняться возможность получать данные, например, по конкретному региону или городу. Отличие массового оператора SELECT от стандартного состоит в том, что глобальные таблицы адресуются в форме: Имя_Группы.Имя_Таблицы, и состав группы задаёт область поиска.

Отбор БД в группу производится на основе расширяемого набора метаатрибутов, содержательно характеризующих БД. Метаатрибутами могут быть, например, название местоположение, и тип организации-владельца БД. Состав группы задаётся логическим условием на значения метаатрибутов (например, регион = «Москва»). MQ-DAI позволяет формировать группы как перед выполнением запроса, так и непосредственно в нём самом:

SELECT avg(MSK.person.salary) FROM MSK.person WHERE MSK.region=”Москва”

Здесь MSK – имя группы, person – имя таблицы, region – имя метаатрибута. Метаатрибуты можно использовать аналогично обычным полям таблиц: выбирать их значения, производить по ним операцию JOIN и т.д.

Поскольку используется виртуальная интеграция, на физическом уровне нет глобальных таблиц и единого информационного пространства, а есть отдельные БД и их схемы. Поэтому MQ-DAI выполняет интерпретацию массовых запросов, чтобы сформировать запрос в терминах интегрируемых БД. Приведем основные этапы алгоритма интерпретации.

1. Синтаксический анализ массового запроса.
2. Выделение имен групп и определение их состава.
3. Выделение обращений к глобальным таблицам групп.

4. Для каждой БД, помимо значений метаатрибутов, задано множество отображений таблиц глобальной схемы на таблицы этой БД (согласно развитию метода GAV). Поэтому для любой группы G, состоящей из БД {DB1, …DBn}, обращение к глобальной таблице G.table можно представить в виде объединения соответствующих отображений: G.table = S1 U … U Sn, где Sk – представляет собой SELECT-запрос к базе данных DBk, который выбирает из неё информацию, соответствующую глобальной таблице table.
5. Все обращения к глобальным таблицам групп меняются на объединение соответствующих отображений баз данных, входящих в состав групп.
6. Полученный запрос сформулирован в терминах интегрируемых СУБД.

Алгоритм интерпретации реализован на языке Java с использованием синтаксического анализатора ANTLR[4].

Обрабатывающая инфраструктура и выполнение запросов

MQ-DAI представляет собой распределённую систему, которая состоит из центрального сервера (ЦС) и нескольких серверов выполнения запросов (СВЗ) (рис. 1).

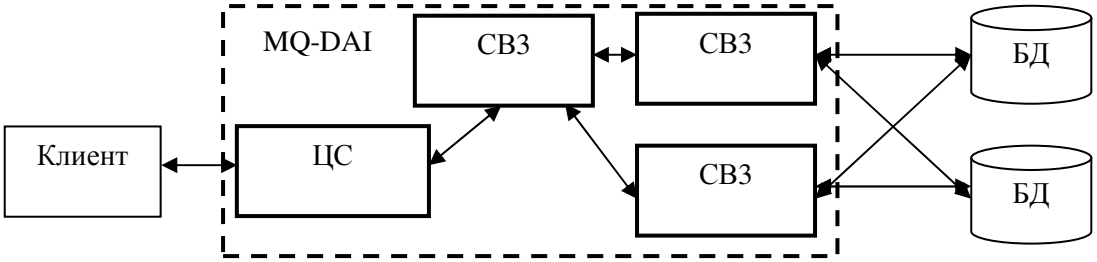


Рисунок 1: Состав и взаимодействие компонентов MQ-DAI

ЦС принимает массовые запросы от клиентов и интерпретирует их описанным выше образом. Далее запросы оптимизируются и разбиваются на частичные подзапросы, после чего каждый подзапрос отправляется на выполнение назначенному ему СВЗ. Назначение того или иного СВЗ осуществляется исходя из его текущей загрузки.

СВЗ выполняет частичные запросы, обращаясь к БД по протоколу JDBC, и передаёт результаты вышестоящему СВЗ или ЦС. Кроме того, СВЗ выполняет операции над данными из нескольких БД, например, объединение содержимого таблиц двух БД. Все СВЗ являются равноправными и какая-либо иерархия отсутствует. Дерево СВЗ появляется в результате построения плана выполнения запроса, и определяется потоками данных между СВЗ.

В случае, если СВЗ не может получить информацию из некоторой БД (выключена, сетевые проблемы и т.п.), ответом от этой БД считается пустое множество строк, а пользователю, отправившему запрос, передаётся диагностика о том, что результат не полный.

Управление инфраструктурой

Управление информационной инфраструктурой должно быть максимально децентрализовано. Для этого в составе MQ-DAI реализованы функции дистанционного выполнения операций подключения/отключения БД и СВЗ. Управление БД возлагается на их администраторов, а управление СВЗ – на администратора инфраструктуры. Выполнение операций управления не приостанавливает работу системы интеграции и не прерывает обработку запросов.

При подключении, БД не изменяется, и на неё не ставится дополнительное программное обеспечение. С помощью операции подключения системе интеграции передаётся: сетевой адрес доступа к БД, выбранные для неё значения метаатрибутов, а также множество отображений таблиц глобальной схемы на таблицы подключаемой БД.

При выполнении операции подключения нового СВЗ он автоматически конфигурируется, получая доступ к БД инфраструктуры. При изменении состава БД на всех СВЗ запускается процесс реконфигурирования. Противоположная операция – отключение СВЗ осуществляется не сразу, а после того как завершатся все запросы, которые используют данный сервер.

Помимо операций по управлению серверами, администратору инфраструктуры доступны операции по изменению глобальной схемы. Необходимо учитывать, что изменение имеющихся глобальных таблиц может привести к тому, что правила интеграции БД станут некорректными. Операции по добавлению и удалению глобальных таблиц с этой точки зрения безопасны.

Вся информация, описывающая состояние инфраструктуры,MQ-DAI хранится в Реестре, который реализован с использованием СУБД MySQL.

Контроль доступа

Безопасность данных обеспечивается в MQ-DAI двумя механизмами: аутентификацией и авторизацией. Механизм аутентификации основан на архитектуре открытых ключей PKI (Public Key Infrastructure) [5], которая широко используется во многих вычислительных гридах. Каждый зарегистрированный пользователь, имеющий право выполнять запросы к информационной инфраструктуре, имеет закрытый ключ, который используется для шифрования или подписи сообщений, и открытый ключ, который позволяет расшифровать или проверить подлинность сообщения. Открытый ключ и уникальное имя пользователя (ИД) хранятся в пользовательском сертификате.

Для целей авторизации сертификат расширен ещё тремя атрибутами, характеризующими пользователя: сфера деятельности (СФД), специальность (СПЕЦ) и роль (РОЛЬ). Политика доступа описывается набором правил вида:

ПИД, СФД, СПЕЦ, РОЛЬ ⇒ (БД…) ((Таблица (Столбец…) (Строка)) …)

В левой части правила (до символа ⇒) перечисляются значения атрибутов, для которых оно применимо. Правая часть определяет доступные данные, причём обеспечивается детализация прав с точностью до столбцов и отдельных строк таблиц глобальной схемы. Список баз данных (БД…) задается аналогично группе поискового запроса – условием на метаатрибуты (например, region=”Москва”). Элемент (Столбец…) перечисляет список доступных столбцов некоторой таблицы глобальной схемы. Элемент (Строка) представляет собой вычисляемое ограничение, которое определяет доступные строки. Расстановка скобок соотносит столбцы и строки соответствующим глобальным таблицам.

Состав атрибутов пользователя выбран таким образом, чтобы уменьшить количество правил, составляющих политику доступа, и систематизировать их путём факторизации. Факторизация заключается в том, что вводятся частичные правила, в которых права доступа определяются на неполном наборе атрибутов. Возможность факторизации обусловлена тем, что мы связываем атрибуты с различными уровнями детализации прав доступа, которые могут определяться независимо. Так атрибут СФД предназначается для спецификации списка доступных БД. Пара атрибутов СПЕЦ, РОЛЬ определяет набор доступных глобальных таблиц, их столбцов и строк. В этой паре атрибут СПЕЦ задаёт максимальный набор таблиц и их элементов, который доступен пользователям, работающим по тематике некоторой специальности, а атрибут РОЛЬ даёт возможность сократить список доступных для неё таблиц, столбцов и строк. Сказанное выражается в частичных правилах:

СФД ⇒ (БД…)

СПЕЦ, РОЛЬ ⇒ ((Таблица (Столбец…) (Строка))…)

Частичные правила служат для порождения полных правил. В процессе авторизации комбинируются частичные правила, левые части которых соответствуют значениям атрибутов пользователя, а их композиция даёт набор атрибутов полного правила.

Контроль доступа осуществляется по множеству явно заданных или порождённых полных правил на этапе интерпретации запроса:

- все недоступные БД удаляются из состава групп, заданных в запросе;

- если в запросе есть обращение к глобальной таблице или ее полю, к которому нет доступа, выполнение запроса прекращается, а пользователю возвращается информирующее сообщение;
- контроль доступа к строкам таблиц реализуется подстановкой ограничивающего условия в блок WHERE запроса.

При выполнении запросов и их частей в БД, они авторизуются в фиксированную учётную запись, которая предоставляется администраторами БД при подключении к инфраструктуре.

Программная архитектура системы MQ-DAI

Система MQ-DAI содержит интерфейсы двух типов: пользовательский и программный. Пользовательский интерфейс позволяет:

- конечным пользователям – составлять и выполнять поисковые запросы, получая данные из инфраструктуры;
- администраторам БД – дистанционно включать/отключать свои базы в/из информационной инфраструктуры;
- администраторам информационной инфраструктуры – дистанционно включать/отключать серверы, обрабатывающие запросы, модифицировать глобальную схему данных.

Программный интерфейс отражает инструментальность системы: все функции реализованы в виде библиотеки классов языка Java. В этом качестве MQ-DAI служит средством разработки приложений, работающих с информационной инфраструктурой.

Имеющаяся версия системы MQ-DAI опирается на комплекс OGSA-DAI/DQP [6]. Первая составляющая этого комплекса – OGSA-DAI, являющаяся реализацией стандартов информационного грида [7], поддерживает выполнение запросов к нескольким БД. Язык таких запросов имеет процедурный характер: операции с отдельными БД задаются явно. Вторая составляющая – OGSA-DQP представляет собой надстройку над OGSA-DAI, в которой реализована поддержка декларативных распределённых запросов на расширенном языке SQL-92, однако в этом расширении БД адресуются явным образом. MQ-DAI в свою очередь развивает возможности OGSA-DQP в направлениях:

- поддержки массовых запросов и адресации с помощью групп БД;
- интеграции данных на основе глобальной схемы;
- динамического формирования и управления инфраструктурой;
- балансировки нагрузки между СВЗ;
- контроля доступа к данным.

Программная архитектура серверной части MQ-DAI представлена на рисунке 2.

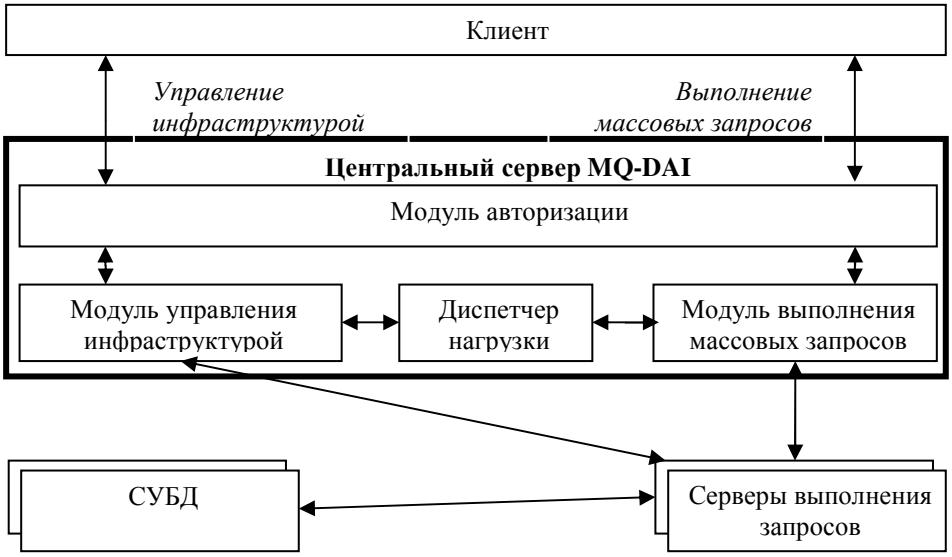


Рисунок 2: Программная архитектура серверной части MQ-DAI

СВЗ реализуется комплексом OGSA-DAI без каких-либо изменений. Интерес представляет ЦС, на который устанавливается расширенная версия OGSA-DQP, дополненная 4 модулями системы MQ-DAI. Модуль авторизации на основании правил доступа ставит в соответствие каждому пользователю множество доступных ему ресурсов. Диспетчер нагрузки хранит текущие значения загруженности СВЗ и распределяет запросы между ними. Модуль выполнения запросов: преобразует массовые запросы в форму OGSA-DQP, выносит авторизационное решение, оптимизирует и передает СВЗ частичные подзапросы, сформулированные на языке OGSA-DQP. Модуль управления инфраструктурой осуществляет выполнение операций, описанных в соответствующем разделе.

Заключение

Разработанная версия системы MQ-DAI реализует описанные функциональные возможности: поддержку массовых поисковых запросов, интеграцию данных, управление инфраструктурой, контроль доступа к источникам и содержащимся в них данным. Проведённые на модельной инфраструктуре эксперименты показали, что время выполнения массовых запросов практически такое же, как и при их моделировании средствами базового комплекса OGSA-DAI/DQP. Для массового запроса к 100 БД оно составляет 800 - 1000 Мс при выборке 1000 строк суммарно.

MQ-DAI опробована на практике: она стала основой для разработки системы доступа к распределённым архивам медицинских изображений – PACS-серверам. Разработка выполнена в сотрудничестве с Институтом нейрохирургии имени академика Н.Н.Бурденко. Перспективы развития системы MQ-DAI мы связываем в первую очередь с улучшением технических характеристик (устойчивости, быстродействия), развитием средств управления контроля доступа, а также с опробованием системы в различных практических приложениях.

Литература

[1] В.Н. Коваленко, А.Ю. Куликов. Интеграция данных и язык запросов в масштабных информационных инфраструктурах. Программные продукты и системы, № 3, 2012, с. 124-130.

[2] http://java.cnam.fr/iagl/biblio/spec/jdbc-3_0-fr-spec.pdf

[3] Lenzerini M. Data Integration: A Theoretical Perspective. PODS 2002, pp. 233–246. http://www.dis.uniroma1.it/~lenzerin/homepagine/talks/TutorialPODS02.pdf.

[4] ANTLR, ANother Tool for Language Recognition. http://www.antlr.org/

[5] Wei Jie, Junaid Arshad and Pascal Ekin. Authentication and authorization infrastructure for Grids—issues, technologies, trends and experiences. The Journal of Supercomputing, Volume 52, Number 1, 2010, 82-96.

[6] OGSA-DAI, Open Grid Service Architecture – Data Access and Integration www.ogsadai.org.uk/

[7] M. Antonioletti, M. Atkinson, A. Krause, S. Laws, S. Malaika, N. Paton, D. Pearson, G. Riccardi. Web Services Data Access and Integration - The Core WS-DAI Specification, http://ogf.org/documents/GFD.74.pdf

МОДЕЛИРОВАНИЕ ГРИД СИСТЕМЫ OFF-LINE ОБРАБОТКИ ДАННЫХ ДЛЯ ЭКСПЕРИМЕНТА NICA[1]

В.В Кореньков, А.В. Нечаевский, В.В. Трофимов
Лаборатория информационных технологий,
Объединенный институт ядерных исследований, 141980, Дубна, Россия

В работе обоснована необходимость создания имитационной модели грид системы хранения и обработки данных ускорительного комплекса NICA. На данном этапе работ в качестве платформы для создания модели выбрана GridSim. Для моделирования предложен ряд задач. В работе приведены результаты работы модели, а также сформулированы параметры оценки эффективности модели. Представлены интерфейсы для работы пользователя и графического отображения результатов.

Введение

В настоящее время в Объединённом институте ядерных исследований создаётся ускорительный комплекс NICA. Комплекс NICA представляет собой ускоритель тяжёлых ионов NICA и установку MPD, объединяющую детекторы для изучения ядерной материи в горячем и плотном состоянии, которое возникает при столкновении ускоренных тяжёлых ионов. MPD является источником информации с интенсивностью потока десятки петабайт в год.

Ожидаемая интенсивность потока информации настолько велика, что массивы данных характеризуются как сверхбольшие. Для обработки таких потоков информации используются распределённые системы коллективного пользования, построенные на грид технологиях.

Для оптимизации структуры будущего комплекса обработки информации необходимо определить его основные параметры, структуру и проверить предлагаемые технические решения с помощью моделирования.

Система обработки информации ускорительного комплекса NICA

Хранение и использование экспериментальных данных в современных экспериментах физики высоких энергий является актуальной проблемой. Объем получаемых и обрабатываемых данных исключает возможность хранения и использования информации не только на одном кластере, но и в пределах одной организации, поэтому на первый план выходит создание распределённой системы хранения и обработки данных для эксперимента.

В случае комплекса NICA поток данных имеет следующие параметры:

- высокая скорость набора событий (до 6 КГц),
- в центральном столкновении Au-Au при энергиях NICA образуется до 1000 заряженных частиц,
- размер файла с первоначальной моделируемой информацией с детекторов для 100000 событий занимает сейчас порядка 5 ТБ.

Схема получения и обработки данных представлена на рисунке 1. Данные, идущие от персональных компьютеров поддетекторов установки MPD (Multi Purpose Detector), накапливаются специально предназначенными для сборки событий программами EB (Event Builder) в компьютерной ферме в онлайн режиме и записываются на диск в офлайн режиме после формирования события через специально предназначенную для этой цели 10 Гб/с

[1] Работа выполнена в рамках ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы» (Гос.контракт №07.514.12.4006).

волоконно-оптическую линию связи. Каждая EB записывает один «рабочий файл» каждую минуту сбора данных.

События, отобранные после триггера высокого уровня, записываются в RAW файлы (скорость записи - один файл в 1 минуту сбора данных) и затем полностью восстанавливаются.

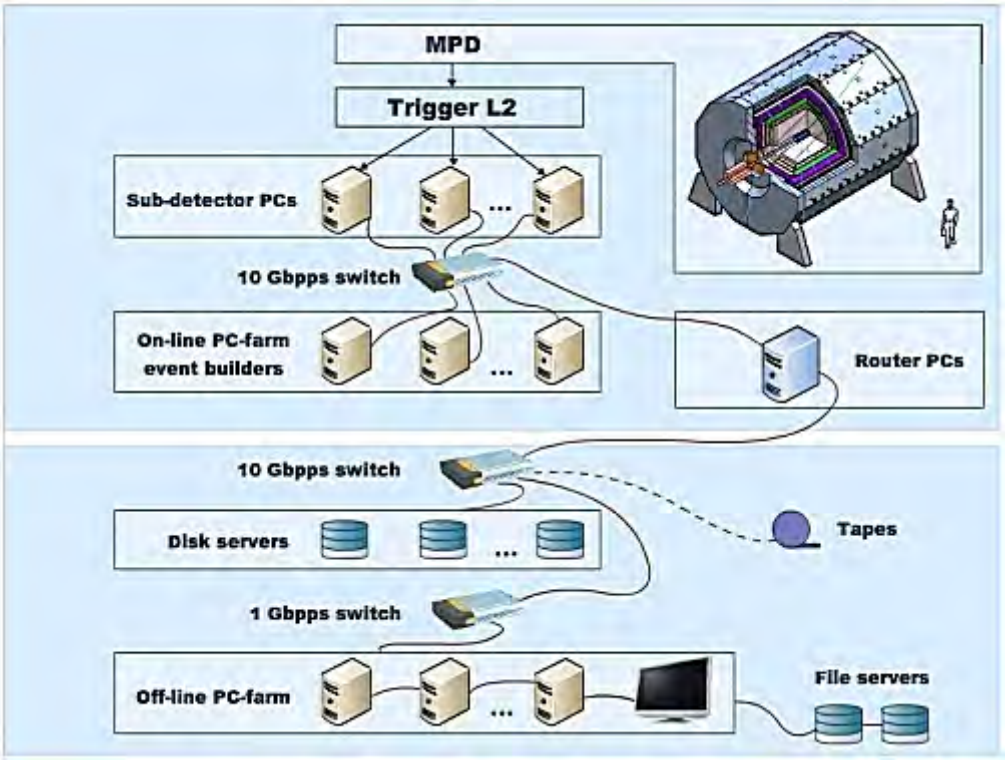


Рисунок 1: Схема обработки физических данных ускорительного комплекса NICA

Прогнозируемое количество обрабатываемых событий при этом приблизительно 19 миллиардов. Принимая скорость передачи данных от датчиков 4.7 GB/s, общий объем исходных данных может быть оценен в 30 PB ежегодно, или 8.4 PB после обработки. Эти оценки основаны на особенностях DAQ, предыдущем опыте и подобных оценках, выполненных для эксперимента ALICE [1].

В качестве системы обработки физической информации ускорительного комплекса NICA планируется грид-структура. Существующие решения по созданию распределенных систем для сбора, передачи и обработки сверхбольших объемов информации базируются на общих принципах построения грид-инфрасруктур. Так компьютерная инфраструктура для эксперимента ALICE представляет собой иерархическую структуру с компьютерными центрами класса Tier 0/1/2. Для хранения и обработки данных с эксперимента PANDA предполагается использование инфраструктуры, построенной на принципах грид по образцу и подобию ALICE.

При создании распределенной системы требуется принять решения по архитектуре инфраструктуры, количеству ресурсных центров, объему требуемых ресурсов. Кроме того, необходимо обеспечить достаточную пропускную способность, решить проблемы сохранности данных (устойчивость к повреждениям и удалениям) на протяжении всего жизненного цикла проекта, обеспечить распределение ресурсов между различными группами пользователей, выбрать алгоритмы обработки и запуска задач и многое другое. Для решения этих вопросов, а также обоснования решений, требуется создание имитационной модели обработки данных, удовлетворяющей всем требованиям эксперимента.

Актуальность темы обуславливается тем, что на основе этой модели в дальнейшем может быть сформулировано конкретное техническое задание на разработку грид-инфраструктуры.

Исходя из вышеизложенного, для проектирования грид структуры центров обработки и параметров системы off-line обработки данных ускорительного комплекса NICA необходимо создать имитационную модель. Согласно планируемой процедуре использования, модель должна включать в себя интерфейс пользователя, собственно ядро моделирования, которое обрабатывает описания структуры обработки и потока заданий и определяет параметры прохождения заданий, систему визуализации результатов моделирования. Ядро моделирования будет включать имитаторы ограниченного набора функций грид, которые наиболее существенно влияют на прохождение заданий.

Платформа моделирования GridSim

Изучив предлагаемый на сегодняшний день инструментарий моделирования грид систем [2], мы решили разрабатывать систему моделирования off-line обработки данных ускорительного комплекса NICA на базе платформы GridSim.

Проект GridSim [3] разрабатывается группой исследователей в лаборатории по изучению облачных и распределенных вычислений отдела информатики и компьютерных вычислений в университете Мельбурна, Австралия.

GridSim это набор библиотек, предназначенных для построения модели грид-системы. Она в свою очередь построена на стандартной библиотеке java SimJava, с помощью которой можно моделировать поток дискретных событий во времени. Приложение создаётся расширением классов GridSim и объединением их в программу, которая моделирует обработку потока заданий грид-структурой, обладающей определёнными ресурсами и с заданной дисциплиной их резервирования и использования. Основные принципы, на которых построено описание ресурсов и их использование, следующие:

а. моделирование гетерогенных типов ресурсов;
б. возможность моделирования приложений с различными параллельными прикладными моделями;
в. отсутствие ограничений на количество задач, которые могут быть отправлены на определенный ресурс;
г. возможность задания пропускной способности сети между ресурсами;
д. поддержка моделирования статистических и динамических планировщиков заданий;
е. возможность регистрации статистики всех или выбранных операций.

Таким образом, платформа позволяет пользователям моделировать работу грид-системы с возможностью симулирования характеристик ресурсов и вычислительных сетей при различных конфигурациях. С помощью GridSim можно проводить воспроизводимые эксперименты, которые сложно реализовать в настоящем окружении динамических грид-систем. В [4] показано, что платформа GridSim недостаточно эффективна для моделирования больших систем, но в нашем случае количество центров обработки не превысит 20.

Задачи моделирования

В рамках выполняемой работы мы рассматриваем ряд задач, которые можно моделировать.

Первая задача подразумевает моделирование распределения данных на “нулевом” уровне (Tier0) обработки данных ускорительного комплекса NICA. Зная характеристики событий и предполагаемый объем данных, необходимо ответить на следующие вопросы: сколько понадобится устройств для записи/чтения данных; что произойдет, если пользователь запросит файл с ленты; как будет работать при этом вся система и т.п.

На данный момент предполагается, что распределённая обработка и анализ моделированных данных с установки МПД о событиях столкновения тяжелых ионов на коллайдере NICA будет возможна на вычислительных ресурсах следующих научных центров:

а) Объединенный институт ядерных исследований, Дубна, Россия.

б) Институт ISS (Institute of Space Sciences), Румыния, город Бухарест.

в) Кейптаунский университет (Университет Кейптауна) — одно из ведущих высших учебных заведений в ЮАР, расположенное в городе Кейптаун.

г) Санкт-Петербургский государственный университет – высшее учебное заведение города Санкт-Петербург, входящий в группу национальных исследовательских университетов России.

В дальнейшем список будет существенно расширяться как за счёт российских, так и международных партнёров. Из этого следует вторая задача – разработка инструмента для моделирования грид системы. На основании концепции дизайн-проекта ускорительного комплекса NICA [5] можно построить модель, отражающую общие принципы построения систем в грид архитектуре, с возможно более широкой возможностью вариации параметров и возможностью дальнейшего их уточнения.

Структура модели

Имитационная модель моделирует прохождение набора заданий заданными пользователем параметрами, через грид структуру с заданной пользователем топологией и параметрами центров обработки. Модель позволяет получить оценку временных параметров обработки потока заданий при заданной пользователем дисциплине распределения ресурсов между заданиями и структурой очередей к центрам обработки.

Моделирование даёт ответы на вопросы:

а) какие вычислительные ресурсы требуются для обработки данных, чтобы получить результат в заданное время;

б) как должны быть связаны между собой центры обработки;

в) какое должно быть разделение функций между центрами;

г) какая стратегия запуска заданий должна применяться;

д) сколько памяти необходимо выделить для хранения информации.

Модель рассчитывает 8 параметров, определяемых как процент, или абсолютное значение:

а) средняя загрузка сети по дням [%],

б) количество активных /ожидающих заданий,

в) количество запрошенных и используемых ЦПУ,

г) использование грид-структуры по часам [%] в день,

д) использование ресурсов хранения данных [%],

е) процент отказавших ЦПУ по дням [%],

ж) объем переданных данных в час,

з) использование кластеров [%].

Данный набор параметров достаточен для оценки эффективности топологии структуры, оценки её технического оснащения, эффективности алгоритмов распределения задач по узлам обработки.

С технической точки зрения система моделирования состоит из трех функциональных компонентов:

а) веб-сервер,

б) веб-клиент,

г) комплекс моделирования грид структуры.

Входные данные для имитации хранятся в каталоге файлов, и изменяются через веб интерфейс (рисунок 2). Это даёт пользователю возможность описывать и менять моделируемую грид-структуру и параметры её загрузки заданиями, хранить варианты структур, выбирать вариант структуры.

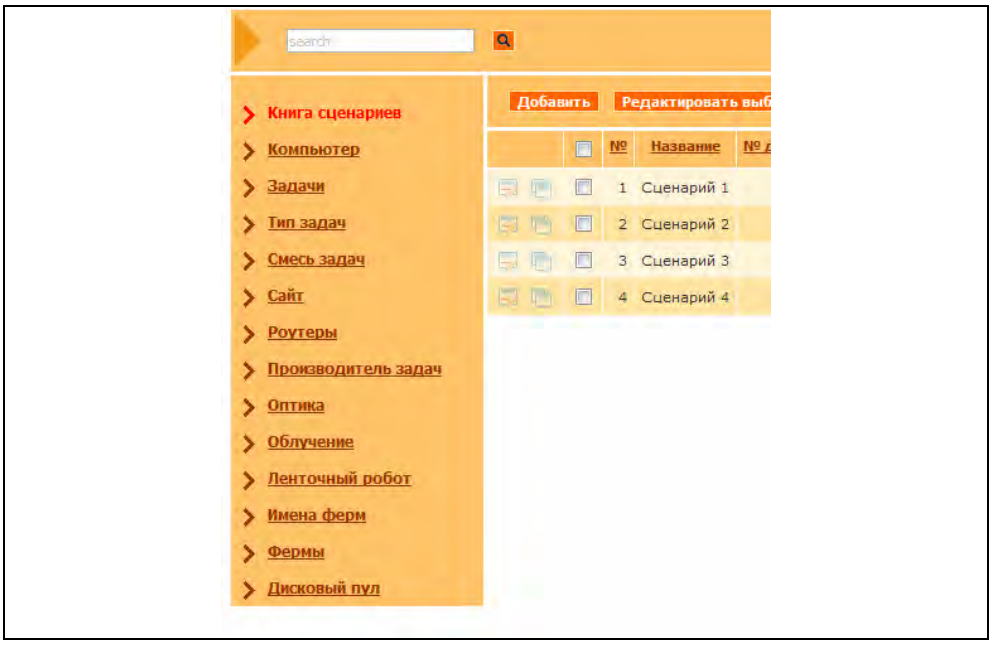


Рисунок 2: Веб интерфейс системы моделирования

Графическое представление результатов моделирования (рисунок 3) ускоряет процесс анализа и принятия решения в ходе технического проектирования системы обработки информации.

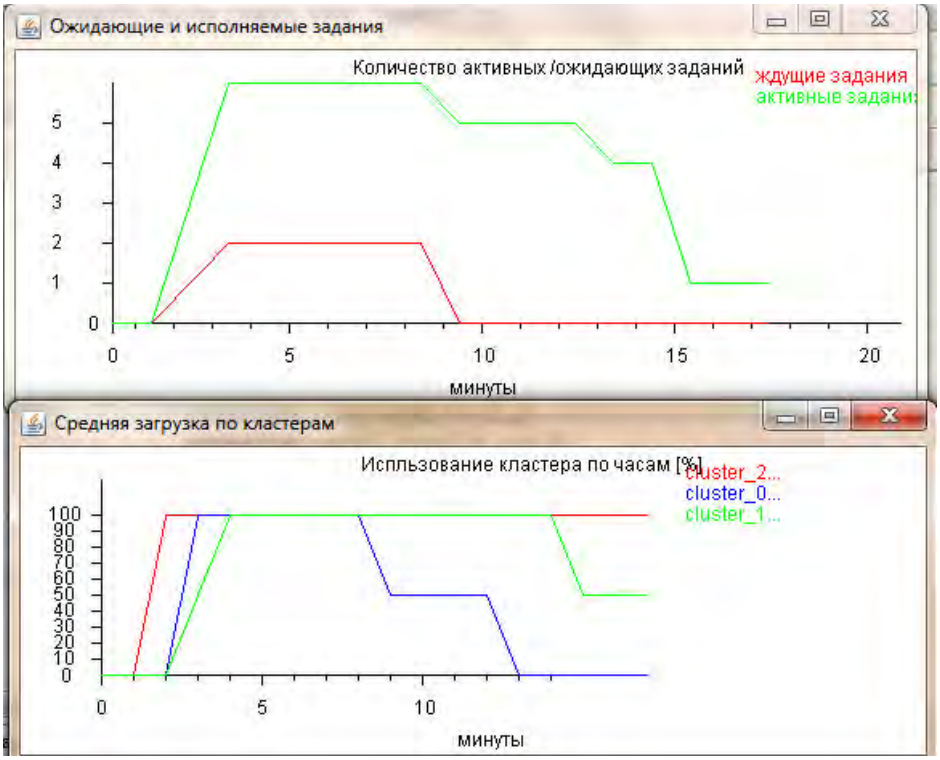


Рисунок 3: Графическое представление результатов моделирования

На данном этапе выполнены следующие работы: создан веб-интерфейс редактирования модели с одним тестовым сценарием работы грид, выделены ключевые параметры оценки модели, созданы средства визуализации результатов, имитационная модель прошла отладку и верификацию.

Заключение

Созданная в рамках данного проекта система моделирования, позволит оценивать несколько вариантов архитектуры (параметров) системы обработки информации изменяя только входные данные, а не программу. Методика применения системы моделирования позволит определить параметры системы обработки информации ускорительного комплекса NICA на этапе технического проектирования. Хранение нескольких сценариев обработки информации, структур и параметров системы обработки, позволит сравнивать различные технические решения и выбирать оптимальное в соответствии с критериями, которыми руководствуется проектировщик.

Дальнейшее развитие системы моделирования предполагает:

- разработку пользовательского интерфейса,
- отладку запуска имитационной модели в клиент-серверной архитектуре,
- разработку набора сценариев работы грид системы,
- возможность добавления пользователем своих сценариев и грид архитектур.

Литература

[1] ALICE Collaboration (P. Cortese et al.), ALICE Technical Design Reort of the Computing. – CERN/LHCC 2005-018, ALICE TDR 12, 2005.

[2] Нечаевский А.В., Кореньков В.В. Пакеты моделирования DataGrid// Системный анализ в науке и образовании.- Электрон. журн.- 2009.- Вып. 1.- URL:http://sanse.ru

[3] http://www.gridbus.org/gridsim/

[4] Aida, K., Takefusa, A., Nakada, H., Matsuoka, S., Sekiguchi, S., Nagashima, U. Performance Evaluation Model for Scheduling in a Global Computing System. //The International Journal of High Performance Computing Applications, Vol. 14, No. 3. Sage Publications, USA (2000).

[5] Sissakian, A., Sorin, A. Многоцелевой Детектор – MPD для изучения столкновений тяжелых ионов на ускорителе NICA (Концептуальный дизайн-проект), версия 1.4. [Электронный ресурс] – 2011. – Режим доступа: http://nica.jinr.ru/files/CDR_MPD/MPD_CDR_ru.pdf

СОЗДАНИЕ ОБЛАЧНОЙ ПЛАТФОРМЫ УРОВНЯ Tier3 В ГРИД-ИНФРАСТРУКТУРЕ ЭКСПЕРИМЕНТОВ НА LHC ДЛЯ РАЗРАБОТКИ ПРИЛОЖЕНИЙ РАДИОЛОКАЦИОННОГО КОСМИЧЕСКОГО МОНИТОРИНГА

В.В. Кореньков, В.М. Котов, Н.А. Русакович, А.В. Яковлев
Объединенный институт ядерных исследований, 141980, Дубна, Россия

В статье рассмотрены технологические решения интеграции программного обеспечения удаленного мониторинга эксперимента ATLAS LHC и инструментального программного комплекса NEST ESA, а также создания на их основе облачной платформы уровня Tier 3 общей системы грид-обработки. Представлены результаты исследований архитектуры и алгоритмов управления интегрированным комплексом. Предлагаемые решения основаны на программном обеспечении с открытым кодом и общих стандартах, включая промежуточное программное обеспечение и инструментальные программные комплексы NEXT , BEAM, BEST Европейского космического агентства (ESA).

Введение

Сбор, предварительная обработка и анализ данных, полученных при дистанционном зондировании Земли (ДЗЗ) с помощью космических радиолокаторов с синтезированной апертурой (РСА), связаны с накоплением и обработкой информации, объемы которой в сотни тысяч раз превышают средний уровень потребностей и достигают десятков Tb в год.

Европейское Космическое Агентство (European Space Agency (ESA)) начиная с 90-х годов прошлого столетия поставляет пользователям данные космических аппаратов, имеющих РСА (Envisat, Radarsat, TerraSar, Cosmo-Skymed), и уже собран большой архив радарных данных [1].

В настоящее время и в России в рамках Федеральной космической программы ведется разработка космических аппаратов («Метеор-М №3», «Аркон-2М»), оснащенных многофункциональной РСА с активной фазированной антенной решеткой, характеристики которой соответствуют характеристикам современных РСА [2].

Традиционные подходы к разработке инфраструктуры и обработки таких объемов данных (получивших название «Большие данные») не годятся, необходимы новые подходы к обработке и анализу данных для современных РСА, в том числе и реализация интерактивного взаимодействия в режиме удаленного доступа для многоуровневой, географически распределенной вычислительной системе обработки данных больших объемов.

Аналогичная проблема стояла и в области экспериментальной физики высоких энергий еще в 90-е годы прошлого столетия при создании системы сбора и обработки данных экспериментов на Большом адронном коллайдере (БАК) [3]. На сегодня не существует технологии, обеспечивающей необходимую функциональность и эффективность обработки таких потоков и объемов экспериментальных данных в одной системе. Поэтому система сбора и обработки эксперимента ATLAS на БАК была разделена на два уровня: online - сбор и предварительная обработка и offline - полная обработка и анализ [4].

Опыт разработки и реализации системы сбора и обработки данных экспериментов на БАК может быть использован при создании систем обработки данных в системах ДЗЗ, и в частности для данных, полученных с помощью космических радиолокаторов с синтезированной апертурой.

Концепция

Современная архитектура наземной географически распределенной системы обработки данных SAR (Synthetic Aperture Radar), предлагаемая Европейским космическим агентством, имеет уровень предварительной обработки входных данных перед выполнением полного цикла обработки информации с SAR в пакетном режиме.

Необходимость первичной обработки радиолокационных данных в интерактивном режиме определяется особенностью форматов и структурой данных SAR. Кроме того, структура радарных данных позволяет проводить эффективную предварительную обработку на уровне первичной обработки радиолокационных изображений, оперируя изображением в целом как образом, для быстрого опознания и анализа объектов и их характеристик.

Радарные данные имеют целый ряд особенностей: сложность обработки из-за геометрических искажений, а также непростая интерпретация изображений. Состав функций предварительной обработки включает следующие возможности обработки данных: фокусировка, корегистрация, удаление спекл-шумов, извлечение характеристик (включая когерентность), геокодирование, радиометрическую калибровку и нормализацию, составление мозаики и классификации.

ESA в октябре 2007 г. заключило договор с канадской фирмой Array Systems Computing на разработку инструментального программного обеспечения и с 2010 г. поставляет комплект программного обеспечения с открытым исходным кодом NEST (Next ESA SAR Toolbox), включающего в себя функциональность всех предыдущих версий инструментального программного обеспечения (ПО) BEST, BEAM и др. [5].

По условиям технического задания ESA инструментарий NEST предназначен для помощи в подготовке элементов системы обработки данных SAR, но не является SAR процессором или готовой системой обработки SAR данных в полном технологическом цикле. Вместе с тем, статус пакета NEST как программы с открытым исходным кодом позволяет использовать его в составе облачной платформы для разработки приложений радиолокационного космического мониторинга. Кроме того, начиная с 2002 г. в ESA успешно реализуется проект G-POD (ESA Grid Processing on Demand) [6]. G-POD является вычислительной Грид-инфраструктурой, предназначенной для разработки новых приложений по обработки данных ДЗЗ. G-POD обеспечивает необходимую гибкость для создания новых приложений по обработке пространственных данных с быстрым доступом к вычислительным ресурсам, необходимым для разработки и инструментальному ПО.

Web-портал G-POD представляет собой гибкую, безопасную, и распределенную платформу, в которой пользователь может легко управлять своими задачами. В ходе всех стадий от создания приложений для новой задачи и до публикации результатов, через стадии выбора данных и мониторинга выполнения задач, разработчик (пользователь) имеет дружественный и интуитивно понятный интерфейс.

Архитектура G-POD представлена на рис. 1.

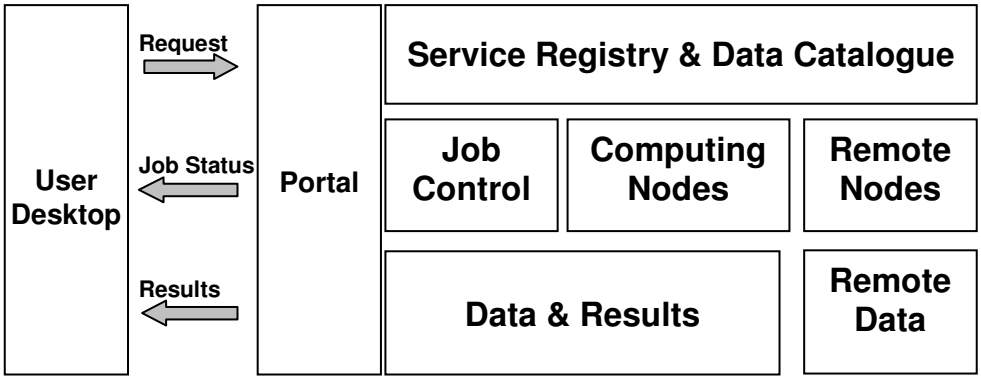


Рис. 1: Архитектура G-POD

Идеология и архитектура G-POD предполагает использование ресурсов локального вычислительного кластера, а при необходимости использование облачных вычислительных ресурсов.

Принимая во внимание свободно распространяемое инструментальное ПО NEST ESA для обработки радиолокационных данных космического мониторинга, а также учитывая опыт и результат разработки и реализации системы сбора и обработки данных экспериментов на БАК, предлагается разработать платформу уровня Tier 3 (PaaS&IaaS) для разработки, тестирования, и поддержки приложений по обработке данных с космических локаторов РСА, реализующей полный технологический цикл обработки радарных данных. В качестве основы для разработки прототипа подобной PaaS предлагается использовать ПО системы сбора и обработки данных эксперимента ATLAS.

В соответствии с концепцией «Открытой инновационной лаборатории» эксперимента ATLAS-LAB (ATLAB), на рабочем совещании ЦЕРН-ОИЯИ «Brainstorming workshop on applications from ATLAS using EU-funding for R&D-upgrades» в г. Дубне 24.10.2010 г. при обсуждении доклада ОИЯИ «Real Time remote access system for ATLAS» было поддержано предложение о возможности прикладного использования ПО ATLAS TDAQ в области космического мониторинга, проводимого при поддержки ESA совместно с ЦЕРН.

Структура программного обеспечения ATLAS TDAQ и системы HLT, а также структура инструментального ПО NEST ESA использует объектно-ориентированный подход проектирования и соответствует стандарту PSS05 ESA. Объединение NEST и сервисов системы сбора и обработки данных экспериментов на БАК в единую платформу, как основу платформы уровня Tier 3, обеспечит интеграцию NEST в общую систему Грид-обработки данных экспериментов БАК, а значит и возможность отладки в последующем и offline режима обработки данных космического мониторинга в географически распределенной вычислительной системе Грид-обработки с участием ESA и ЦЕРН.

Следует отметить, что кроме инструментария NEST, отражающего специфику обработки радарных данных, предлагаемая платформа Tier 3 будет обеспечивать также доступ к сервисам сбора и обработки данных, необходимых для функционирования NEST в составе системы удаленного доступа реального времени (СУДРВ).

Модель инструментальной платформы

Модель инструментальной платформы имеет клиент-серверную архитектуру. Серверное ПО развернуто на локальном вычислительном кластере уровня Tier 3, который интегрирован в общую систему Грид-обработки экспериментов на БАК.

В качестве клиента предполагается использовать ПО NEST. Данный вариант предполагает сохранение ПО NEST в качестве самостоятельного программного обеспечения, способного реализовывать все возможности, заложенные в него при разработке. При этом ПО NEST получит дополнительный инструментарий, который позволит проводить обработку SAR данных на вычислительной ферме.

В качестве базового ядра предлагаемой системы предполагается использовать набор компонентов из Системы сбора и обработки данных в реальном времени эксперимента ATLAS LHC (ATLAS TDAQ Online Software и High Level Trigger), кроме того предполагается разработать новые компоненты, необходимые для реализации функционала инструментальной платформы NEST. Компоненты должны будут использовать существующие возможности сервис-ориентированной архитектуры TDAQ и High Level Trigger. В качестве алгоритмов обработки SAR данных используются операторы из пакета Graph Processing Tools (GPF Operator) ПО NEST, необходимые для предварительной обработки.

Вычислительный кластер интегрирован в Грид-инфраструктуру экспериментов на БАК, поэтому предполагается обмен данными и результатами обработки между локальным хранилищем данных и удаленными Грид-хранилищами.

Общая модель инструментальной платформы представлена на рис.2.

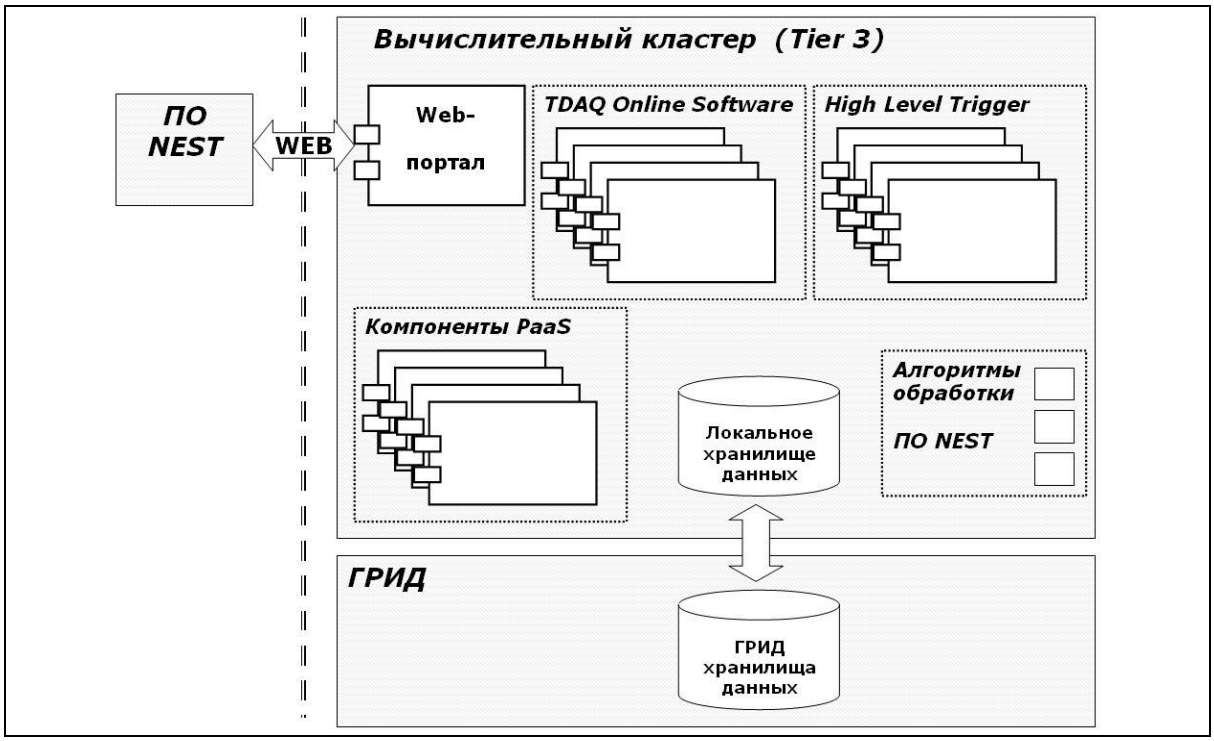


Рис. 2: Модель инструментальной платформы

Общая архитектура инструментальной платформы приведена на рис. 3.

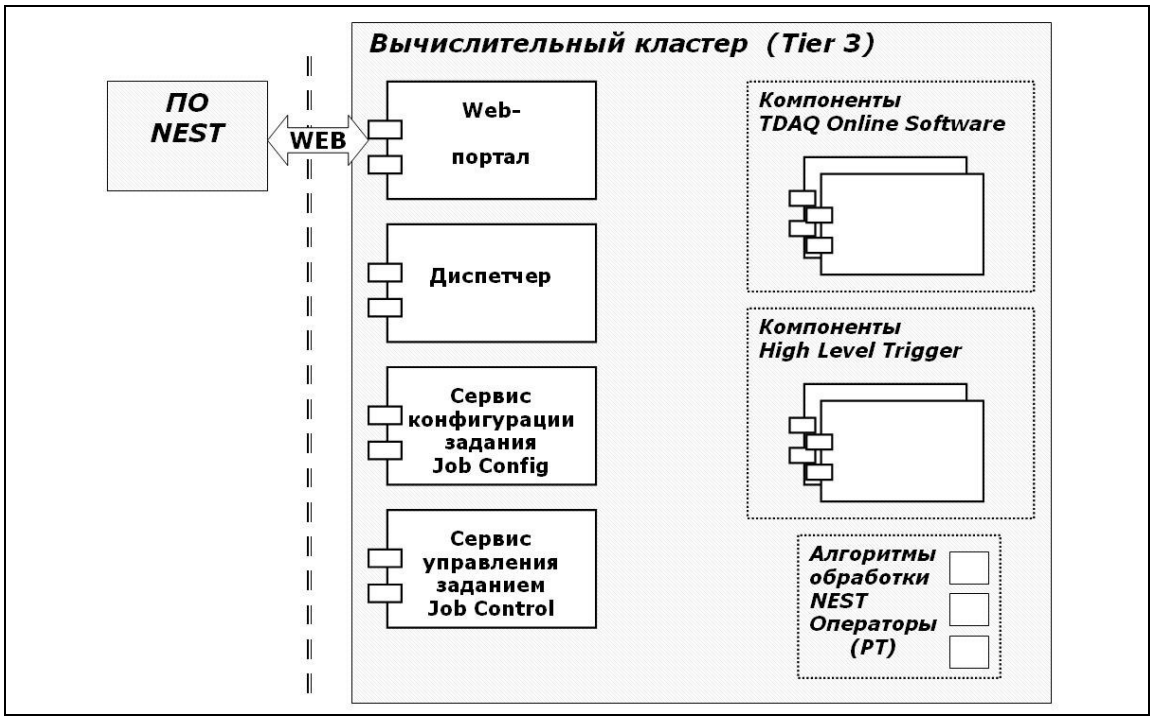


Рис. 3: Архитектура инструментальной платформы.

Клиентское ПО

ПО NEST будет адаптировано для взаимодействия с удаленной вычислительной фермой. Клиентское ПО позволит пользователю составлять задание на удаленную обработку SAR данных. Для этого предполагается разработать интерфейс на основе Graph Builder User

Interface из компонента Graph Processing Tools (GPT) NEST. Используя данный интерфейс, пользователь, взаимодействуя с серверной частью, получит возможность задавать источники данных, выбирать из набора алгоритмов NEST алгоритмы обработки данных, параметры, последовательность шагов обработки и т.п. После чего составленное задание будет принято к выполнению на удаленной ферме. Кроме того, клиентское ПО должно предоставлять пользователю возможность мониторинга процессов удаленной обработки, и, при необходимости, возможность вмешиваться в процесс обработки.

Web-портал

Web-портал является для удаленного пользователя точкой доступа к ресурсам вычислительной фермы. Через web-портал пользователь взаимодействует с остальными компонентами серверного ПО, составляет задания, наблюдает за ходом обработки, получает результат обработки задания. Также на web-портале осуществляется авторизация и аутентификация пользователя. Для этого используются соответствующие сервисы системы сбора и обработки данных экспериментов на БАК.

Сервис конфигурации задания (Job Config)

Сервис Job Config отвечает за составление задания на обработку SAR данных. Для этого используются соответствующие компоненты из подсистемы Config ПО TDAQ ATLAS. Для корректного описания задания на обработку SAR данных используется понятие TDAQ-partition.

TDAQ-partition – в рамках TDAQ ATLAS это подмножество компонентов системы сбора и обработки данных (TDAQ–system), выделенное для выполнения определенной задачи сбора и обработки данных. TDAQ-partition в виде иерархической структуры содержит описание поддетекторов ATLAS, источников данных, аппаратных и программных ресурсов, алгоритмов обработки, параметров и т.п. TDAQ-partition представляет собой полный и законченный набор компонентов TDAQ, полный в том смысле, что данное сочетание компонентов позволяет создать систему, способную полноценно выполнять конкретную задачу сбора и обработки данных.

В рамках системы обработки SAR данных используется понятие Конфигурации задания, которое в целом повторяет структуру TDAQ-partition. В Конфигурации задания должны быть определены источники данных, алгоритмы обработки, параметры, последовательность шагов обработки и т.п., при этом осуществляется проверка задания на полноту и непротиворечивость. Удаленный пользователь, при помощи клиентского интерфейса составляет требуемую ему Конфигурацию задания на обработку SAR данных, и это задание сохраняется на сервере в виде TDAQ-partition. Одно задание на обработку SAR данных соответствует одному partition в терминологии TDAQ ATLAS.

Сервис управления заданием (Job Control)

Сервис Job Control отвечает за выполнение задания на обработку SAR данных. Для этого используются соответствующие компоненты из подсистемы Control ПО TDAQ ATLAS. Job Control запускает сеанс обработки данных в соответствии с полученным заданием (TDAQ-partition). При этом каждое конкретное задание соответствует отдельному TDAQ-partition.

Кроме этого Job Control осуществляет взаимодействие между сервисами системы обработки SAR данных. В случае, если при выполнении сеанса обработки требуется вмешательство пользователя, сервис Job Control переадресовывает запрос на такое вмешательство пользователю, а получив команды от пользователя, сервис Job Control переадресовывает их соответствующим сервисам.

Диспетчер

Диспетчер является компонентом, который отвечает за управление всеми заданиями (partition) в целом. Для системы обработки SAR данных предполагается наличие большого количества пользователей, при этом каждый пользователь может оперировать большим количеством заданий (partition). И соответственно, требуется механизм для управления таким количеством заданий. Сервис Job Control, созданный в соответствии с архитектурой сервисов подсистемы Control TDAQ, отвечает за выполнение одного конкретного задания (partition).

Каждое вновь создаваемое задание регистрируется в Диспетчере и ставится в очередь заданий. Именно через очередь Диспетчера пользователь получает доступ к своим заданиям и имеет возможность ими оперировать - редактировать Конфигурацию задания, запускать задание на выполнение и останавливать сеанс обработки. Состояние задания в очереди, на разных стадиях выполнения отражается в статусном регистре.

Алгоритмы обработки данных (операторы)

В качестве алгоритмов обработки SAR данных используются операторы из пакета Graph Processing Tools (GPF Operator) ПО NEST. Для того чтобы операторы NEST корректно работали с сервисами TDAQ ATLAS, они должны быть представлены в виде программных ресурсов (SW resource) TDAQ. Для этого каждый GPF Operator необходимо обернуть в программную оболочку. В этом случае сервисы TDAQ будут вызывать стандартные операторы в рамках TDAQ-partition – в терминологии TDAQ - Process Tasks (PT), а фактически будут выполняться операторы из пакета GPF NEST.

Заключение

Предлагаемая концепция обеспечивает применение современных методов при разработке приложений для сервис-ориентированных архитектур распределенных вычислительных систем и предоставляет возможность интеграции приложений в уже существующие ИТ-системы. Разрабатываемая система представляет уникальные возможности по использованию её как программную платформу Tier 3 для разработки системы моделирования в реальных условиях работы вычислительного комплекса в распределённой грид-среде. Принципиально новым в таком подходе является использование в процессе моделирования реальных интерфейсов и сервисов для взаимодействия с моделируемыми ресурсами, и это означает, что разработчики распределённой системы обработки данных получают возможность интеграции разрабатываемых ими приложений в реальную грид-среду без необходимости внесения изменений сразу в исходный код.

Успешное выполнение данной работы было бы невозможно без поддержки и помощи: Livio Mapelli CERN Serguei Kolos, CERN, Petersburg Nuclear Physics Institute (PNPI); Igor Soloviev, CERN, Petersburg Nuclear Physics Institute (PNPI), и сотрудников ATLAS DAQ Group CERN Physics Department.

Литература

[1] Кантемиров Ю. И. Обзор современных радарных данных ДЗЗ // Журнал "Геоматика", 2021 г. N1 – с. 18.

[2] Костюк Е.А., Веремчук Ю.А., Денисов П.В. Перспективные технологии обработки космической радиолокационной информации в НКПОР Оператора КС ДЗЗ // V Международная конференции «Космическая съемка — на пике высоких технологий».

[3] ATLAS High-Level Trigger, Data Acquisition and Controls. Technical Design Report (ATLAS TDR-016); http://atlas-proj-hltdaqdcs-tdr.web.cern.ch/atlas-proj-hltdaqdcs-tdr/tdr-v1-r4/PDF/TDR.pdf

[4] Mapelli L. Spanning from Data Acquisition to GRID - Today and a view of tomorrow. // XXIII International Symposium on Nuclear Electronics & Computing NEC'2011.

[5] Software Architecture Document (SAD) for the Next ESA SAR Toolbox (NEST) (ARR-NEST-RS07-016); http://www.array.ca/nest/Software_Architecture_Document_v2.0.pdf

[6] Service Level Agreement for ESA GRID Processing on Demand (G-Pod); http://wiki.services.eoportal.org/tiki-download_wiki_attachment.php?attId=1483&page=GPOD+Wiki&download=y

ПОИСК РЕШЕНИЯ ВАРИАЦИОННОЙ ЗАДАЧИ В ВИДЕ МИНИМАЛЬНОГО ПУТИ НА ГРАФЕ

Д.Т. Лотарев
Институт системного анализа РАН, Москва, Россия
dimlot@mail.ru

Задача о прокладке коммуникаций – медных или оптоволоконных кабелей, которые необходимы для соединения локальных, региональных и других компьютерных сетей, рассматривается как вариационная задача. Для решения задачи строятся модели – модель участка территории, на котором строится коммуникация, модель трасы коммуникации, модель самой задачи. Моделью территории является цифровая модель местности, которая имеет вид специального графа, моделью трассы является путь на графе, моделью задачи – вариационная задача, которая сведена к задаче о кратчайшем пути на графе.

1. Введение

Рассматривается следующая задача. На некотором участке земной территории задано положение двух пунктов. В одном размещен источник потока, в другом – сток (потребитель). Поток может быть материальный, энергетический или информационный. Необходимо построить коммуникацию, по которой потечет поток. В зависимости от вида потока коммуникация может быть автомобильной дорогой, трубопроводом, линией электропередачи или волоконно-оптической линией связи (ВОЛС), и т.д. Мы будем полагать, что это ВОЛС. В этой коммуникации затраты на транспортировку потока малы по сравнению с затратами на ее строительство и ими можно пренебречь. Строительство коммуникации – процесс дорогостоящий. Территория неоднородна относительно удельных строительных затрат, и стоимость строительства зависит от конфигурации коммуникации. Поэтому задача состоит в отыскании трассы коммуникации, т.е. той линии, вдоль которой размещается коммуникация. Трасса должна быть такой, что стоимость строительства соответствующей коммуникации минимальна.

Земная территория отличается существенной неоднородностью - леса, болота, горы, сельскохозяйственные угодья, жилые и хозяйственные постройки. Каждый из элементов такой неоднородности определяет свои условия и стоимость прокладки.

Чтобы определить оптимальное (в том или смысле) размещение коммуникации на территории, нужно построить модель участка территории, модель коммуникации, поставить и решить задачу размещения на этих моделях.

2. Постановка задачи

Модель участка территории представляет собой область Q на координатной плоскости (xOy). Между географическими координатами участка местности и координатами системы (xOy) установлено взаимно однозначное соответствие. В области Q заданы точки, соответствующие размещению источника, стока и функция удельных строительных затрат $f(x,y)$. Эта функция отражает значения удельных затрат на строительство коммуникации в точках территории. Удельные строительные затраты в точке территории – это затраты на строительство отрезка коммуникации единичной длины в окрестности этой точки. Территория на участке может быть однородной (если функция удельных строительных затрат является константой) или неоднородной (если ее значение меняется от точки к точке).

Моделью коммуникации является кривая, соединяющая в область Q ту пару точек с координатами (x_1, y_1), (x_2, y_2), которые являются образами объектов, соединяемых на территории.

Задача состоит в отыскании на плоскости кривой, которой соответствует минимальное значение критерия оптимальности. За критерий оптимальности принимается сумма затрат на строительство всей коммуникации протяженностью от одного пункта до другого.

Если удельные строительные затраты можно представить некоторой функцией *f(x,y)*, то задачу о размещении коммуникации можно сформулировать в виде вариационной задачи.

$$J[y(x)] = \int_{x_1}^{x_2} f(x,y)\sqrt{1+y'^2}\,dx, \tag{1}$$

$$y(x_1) = y_1, \quad y(x_2) = y_2, \tag{2}$$

$$f(x,y) \geq 0. \tag{3}$$

Эту задачу можно решить аналитически только в исключительных случаях [1], когда функция *f(x,y)* "достаточно хорошая". Примером такой хорошей функции является константа, *f(x,y)* = const. В некоторых случаях функция *f(x,y)* является кусочно-постоянной. Территорию холмистого участка можно описать суммой функций, подобных функции распределения вероятностей Гаусса

$$f(x,y) = c_0 + \sum_{i=1}^{L} c_i \exp\left(-\frac{(x-a_i)^2}{\sigma_{xi}} - \frac{(y-b_i)^2}{\sigma_{yi}}\right). \tag{4}$$

Холмистую территорию можно также представить в виде прямых круговых конусов, радиусы оснований которых равны R_i, а проекции вершин лежат в точках с координатами (a_i, b_i). В этом случае функция удельных строительных затрат имеет вид

$$c(x,y) = c_0 + \sum_{i=1}^{L} \frac{c_i}{R_i}\left[R_i - ((x-a_i)^2 + (y-b_i)^2)^{1/2}\right]. \tag{5}$$

Несмотря на кажущуюся простоту функций (4) и (5), задачу (1) – (3) не удается решить аналитически для функции удельных строительных затрат вида (4) или (5).

Чтобы найти аналитическое выражение для трассы нужно решить вариационную задачу, например, (5). Для этого нужно поступить стандартным образом – для подынтегрального выражения в (5) написать уравнение Эйлера и решить дифференциальное уравнение второго порядка. Однако, как правило, дифференциальные уравнения второго порядка интегрируются в конечном виде лишь в исключительных случаях. В простейшем случае одного прямого кругового конуса, расположенного в начале координат, уравнение Эйлера имеет вид

$$\left[c_0 + c_1 - \frac{c_1}{R_1}(x^2+y^2)^{1/2}\right]\frac{y''}{1+y'^2} - \frac{c_1}{R_1(x^2+y^2)^{1/2}}(xy'-y) = 0.$$

Найти аналитическое решение этого уравнения не удается и для решения задачи приходится применять численные методы. Хорошо известны метод Эйлера, метод Ритца, метод Канторовича.

3. Численный способ решения

Предлагаемый численный способ решения данной задачи, отличный от указанных, состоит, по существу, в переборе кривых *y(x),* которые в области *Q* проходят через точки (x_1, y_1), (x_2, y_2), и в выборе из них той кривой, которой соответствуют минимальные строительные затраты.

Кривая *y(x)* ищется в виде ломаной. Для этого на область *Q* накладывается квадратная сетка, на сетке строится специальный граф, и ломаная ищется в виде пути на этом графе.

Схема всей этой процедуры следующая. Область *Q* с заданной в ней функцией *f(x,y)* представляется в виде цифровой модели местности (ЦММ) [2]. ЦММ в общепринятом смысле обычно описывает рельеф, инженерно-геологические, технико-экономические и другие показатели территории. В нашей задаче ЦММ отражает только функцию *f(x,y).*

Строится ЦММ следующим образом.

На область *Q* накладывается квадратная сетка $\tilde{Q}$ шага $h > 0$ с множеством горизонталей $I_g = \{i \mid i = 1,2,...,n_g\}$ и множеством вертикалей $I_v = \{j \mid j = 1,2,...,n_v\}$. Узлы сетки составляют

множество $V = \{v_{ij} \mid i = 1,2,...,n_g, j = 1,2,...,n_v\}$ и x_{ij}, y_{ij} - координаты узла $v_{ij} \in V$, $x_{ij} = hj$, $y_{ij} = hi$. Каждому узлу сетки $v_{ij} \in V$ поставлено в соответствие число $f(v_{ij})$, равное значению функции $f(x,y)$ в той точке области Q, в которой находится этот узел. Открытую прямоугольную окрестность узла $v_{ij} \in V$, которая состоит из точек $(x,y) \in Q$, удовлетворяющих условию $|x - x_{ij}| < h/2$, $|y - y_{ij}| < h/2$ называют h-окрестностью узла (рис.1).

Вместо функции $f(x,y)$ в области Q рассматривается кусочно-постоянная функция $\tilde{f}(x, y)$, имеющая значение $f(v_{ij})$ во всей h-окрестности каждого узла $v_{ij} \in V$. Строится граф $H(V, E(Z))$, у которого множество узлов составляют узлы сетки $\tilde{Q}$, а множество ребер $E(Z)$ зависит от целочисленного параметра Z, $0 < Z \le max\{n_v, n_g\}$, и определяется следующим правилом, задающим смежность узлов. Множество узлов S, смежных с узлом $v_{ij} \in V$, определяется следующим образом:

$S(v_{ij}) = \{v_{i+k, j+l}: v_{i+k, j+l} \in V, 0 \le |k| \le Z, 0 \le |l| \le Z, k$ и l *взаимно простые*$\}$.

Число Z определяет локальную степень узла v_{ij}. В табл. 1 показана локальная степень P узла графа $H(V, E(Z))$ при различных Z. На рис.1 показаны узлы графа, смежные с узлом v_{ij} при значении $Z = 4$, и ребра графа, связывающие смежные узлы. Для некоторых узлов показаны h-окрестности. Разные цвета соответствуют своим значениям функции $f(x,y)$.

Таблица 1

Z	1	2	3	4	5	6	7	8	9	10	11
P	8	16	32	48	80	96	144	176	224	256	336

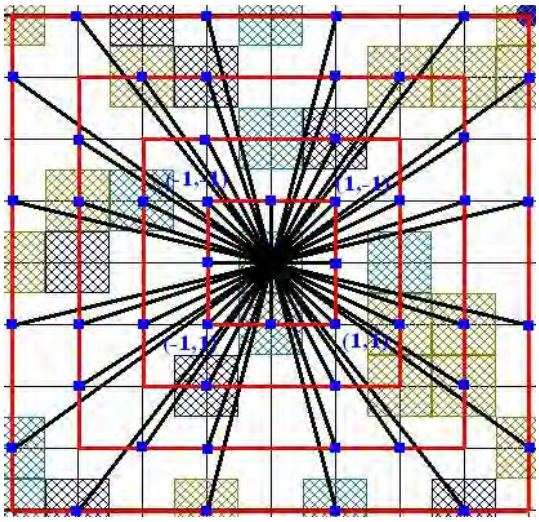


Рис.1: Множество узлов, смежных с узлом v_{ij}

Ребро $(v_{ij}, v_{i+k, j+l}) \in E$, соединяя в области Q узлы $v_{ij} \in V$ и $v_{i+k, j+l} \in V$, пересекает h-окрестности нескольких узлов v_{i_n, j_n}, $n = 1,2,..., N$. Если a_{i_n, j_n} - длина того отрезка ребра, который лежит в окрестности узла, то вес $c((v_{ij}, v_{i+k, j+l}))$ ребра $(v_{ij}, v_{i+k, j+l}) \in E$ есть

$$c((v_{ij}, v_{i+k, j+l})) = \sum_{n=1}^{N} a_{i_n, j_n} f(v_{i_n, j_n}).$$

Решение задачи ищется в виде минимального пути на графе $H(V, E(Z))$. Путь ищется по алгоритму Дейкстра [3] с некоторой адаптацией к данной задаче. Основным элементом адаптации является то, что признаком временной пометки для узла является не просто какое-либо число, а номер, под которым этот узел записан в списке временно помеченных узлов. Это исключает поиск при сравнении стоимостных частей старой и новой пометок узла, и повышает быстродействие алгоритма.

4. Результаты решения

Алгоритм успешно работает на графе с числом узлов $256^2 = 65536$ и числом ребер $256^3 = 16777216$ (при Z = 10).

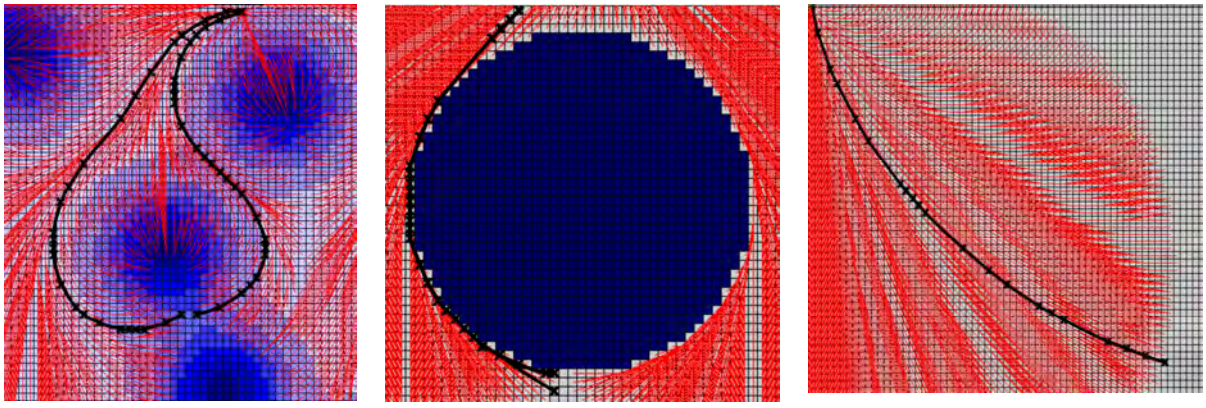
Найдены оценки точности алгоритма. Для оценки рассматривается однородный участок территории. Для такого участка $f(x,y) = const$, и решением вариационной задачи (1) – (3) является прямолинейный отрезок. Для прямолинейного отрезка длины d_0, длина d_1 аппроксимирующего пути на графе $H(V,E(Z))$, есть $d_1 \le d_0\delta_1$, а расстояние d_2 от точек пути до отрезка есть $d_2 \le d_0\delta_2$, где $\delta_1 = \sqrt{\left(\sqrt{1+Z^2} - Z\right)^2 + 1}$, $\delta_2 = \sqrt{1+Z^2} - Z$.

Превышение D длины пути над длиной отрезка в % показано в таблице 2.

Таблица 2

Z	1	2	3	4	5	6	7	8	9	10
D	8.2	2.7	1.3	0.75	0.49	0.34	0.25	0.19	0.15	0.12

Алгоритм Дейкстра, адаптированный к задаче, написан на языке Delphy. Посредством этого алгоритма в среде программирования Delphi 7 решено несколько задач: задача поиска трассы коммуникации на холмистой территории; задача построения трассы на равнинном участке, на котором размещен объект цилиндрической формы; задача о брахистохроне. Программа написана студентом МФТИ Шабановым А.



a b c

Рис. 2. Три вида трасс: a – на холмистой территории, b – на равнинном участке с запретной круговой областью, c – брахистохрона

Заключение

Задача о прокладке коммуникации на местности представлена как вариационная задача. Разработан прямой численный метод ее приближенного решения, основанный на методах теории графов. Основная идея метода состоит в том, что вначале непрерывная модель территории преобразуется в цифровую модель местности, которая затем преобразуется в специальный граф. Путь минимальной стоимости, связывающий пару заданных узлов графа, принимается за решение рассматриваемой задачи. Для практического применения метода необходимо строить цифровую модель местности для того участка территории, на котором строятся коммуникации.

Литература

[1] Эльсгольц Л. Дифференциальные уравнения и вариационное исчисление – УРСС Э. Москва, 1998.

[2] Лотарев Д.Т. Цифровая модель местности для задачи размещения коммуникаций // АиТ 1999. № 12. С.41-49.

[3] Дейкстра Э. Дисциплина программирования. — М.: Мир, 1978.

ЭФФЕКТИВНЫЙ МЕТОД ПЛАНИРОВАНИЯ РЕСУРСОВ В ГЕТЕРОГЕННЫХ РАСПРЕДЕЛЕННЫХ СИСТЕМАХ И ЕГО РЕАЛИЗАЦИЯ В MAUI

С.В. Минухин[1], С.В. Баранник[2], С.В. Знахур[1], Р.И. Зубатюк[3]

[1] *Харьковский национальный экономический университет, пр. Ленина. 9-а, Харьков, Украина*

[2] *Институт сцинтилляционных материалов НАН Украины, пр. Ленина, 60, Харьков, Украина*

[3] *НТК «Институт монокристаллов» НАН Украины, пр. Ленина, 60, Харьков, Украина*

Введение

Планировщики современных гетерогенных распределенных систем (Грид-систем) являются ключевыми компонентами в их сложной информационно-коммуникационной архитектуре. Архитектура этих систем и роль планировщиков (брокеров) ресурсов достаточно подробно рассмотрена в работах [1–3]. В настоящее время проблемой является выбор политики, метода и алгоритмов планирования, который обеспечивал бы высокую эффективность планирования в условиях изменения характеристик поступающих в распределенную систему заданий, изменения количества и производительности ресурсов. Используемые в современных планировщиках алгоритмы не обеспечивают достаточную эффективность планирования (необходимую производительность функционирования системы) в случае высокой интенсивности поступающих заданий и динамического отключения/подключения ресурсов, что приводит к запаздыванию выполнения заданий, простою ресурсов и, как следствие, их неэффективному использованию.

Целью данной работы является моделирование и сравнительный анализ работы планировщиков FCFS и предлагаемого эвристического алгоритма минимального покрытия MC (minimal cover) для планирования ресурсов в гетерогенной распределенной системе при изменении характеристик потока поступающих заданий – интенсивности и сложности (workloads) – и реализация MC в планировщике MAUI.

Основными задачами исследования являются следующие:

1. Проанализировать влияние интенсивности поступающих в систему заданий на результаты работы планировщика ресурсов системы при использовании алгоритмов FCFS и MC.
2. Проанализировать влияние уровня сложности (трудоемкости) поступающих в систему заданий на результаты работы планировщика ресурсов системы при использовании алгоритмов FCFS и MC.
3. Оценить влияние изменения количества вычислительных ресурсов на работу планировщика при использовании разных алгоритмов планирования.
4. Адаптировать предлагаемый эвристический алгоритм для его подключения в планировщик MAUI.

Процедура планирования загрузки ресурсов разработана с учетом работы механизма Workload Management System (WMS), а именно, службы распределения ресурсов (Resource Broker, RB). Основной задачей WMS является поиск оптимального ресурса для выполнения конкретного задания на основании описания задания на языке JDL и информации о доступных на момент планирования ресурсах. Наличие соответствующих ресурсов для задания зависит не только от состояния ресурсов, но и от политики их использования, которой следуют администраторы ресурса или администраторы виртуальной организации. В предлагаемой процедуре планирования используется пакетный режим (batch mode): осуществляется пакетное планирование заданий (bag of tasks, BoT) на доступные ресурсы; при этом задания могут находиться в планировщике (пакете заданий на ресурс) до тех пор, пока какой-либо ресурс не становится доступным, после чего на освободившейся ресурс направляется спланированное на него задание. Следует отметить, что службы, выполняющие мониторинг заданий, перезапуск

заданий, мониторинг состояния ресурсов реализованы в самой системе и не являются компонентами планировщика.

Метод и модель планирования ресурсов

Особенностью предлагаемой процедуры планирования является наличие пула заданий и пакета заданий для каждого ресурса, имеющих определенные размеры. Пул представляет собой стек для временного хранения и обработки поступающих в систему заданий глобальной очереди. Размер пула определяется в зависимости от максимальной интенсивности поступающих заданий: если размер пула меньше количества поступающих заданий, то будет создана дополнительная очередь уже на входе самой системы. Задания из пула выгружаются через определенные промежутки времени, определяемые периодичностью планирования (scheduling interval), в блок планирования, в котором реализован алгоритм планирования, на основе задачи о наименьшем покрытии, предложенный и исследованный в [4]. Время планирования (время работы алгоритма решения задачи о наименьшем покрытии) зависит от размера пула – количества поступивших на блок планирования заданий – и количества доступных и свободных на момент планирования ресурсов. Результатом планирования является назначение заданий, которые размещаются в пакеты на выделенные для них ресурсы, и по мере их освобождения посылаются на решение. Организация пакета заданий позволяет обеспечить постоянную очередь перед каждым ресурсом и, таким образом, максимизировать его загрузку (коэффициент использования).

Математическая постановка задачи планирования представляет собой задачу линейного булевого программирования [4]:

$$L_t = \sum_{j=1}^{n} x_j(t_k) \to \min \tag{1}$$

при ограничениях

$$\sum_{j=1}^{n} \beta_{ij} x_j(t_k) \geq 1, \quad i=\overline{1,m}; \tag{2}$$

$$\beta_{ij} \in \{0,1\}; \; x_j(t_k) \in \{0,1\} \; ,$$

где m – количество заданий, подлежащих планированию; n – количество ресурсов исследуемой системы, доступных и свободных на момент планирования; $t_k \in [T_0, T_N]$.

Планирование осуществляется на интервале времени $[T_0, T_N]$, где T_0 – время начала планирования; T_N – время окончания планирования заданий глобальной очереди.

Задачу (1), (2) можно рассматривать как задачу определения минимального числа столбцов в булевой матрице B, покрывающего единицами все строки данной матрицы, элементы которой в контексте решения задачи планирования интерпретируются следующим образом: столбцам соответствуют свободные на момент планирования ресурсы распределенной системы, а строкам – задания, подлежащие планированию, которые должны быть решены на этих ресурсах (табл. 1).

Данный подход базируется на следующих положениях:

1. Система планирования организована в виде двухуровневой структуры, на первом уровне которой формируется пакет заданий (пул), подлежащих планированию, к ним применяется метод решения задачи (1), (2). Далее спланированные задания назначаются на доступные и свободные на момент планирования ресурсы и решаются на них под управлением локального планировщика.

2. Метод планирования на каждом шаге планирования максимально загружает минимальное количество свободных и доступных на момент планирования ресурсов.

3. Метод (алгоритм) решения задачи (1), (2) должен иметь малую временную сложность его реализации для минимизации времени, отводимого на процесс планирования заданий.

4. Система планирования использует пакетную технологию: задания, организованные в виде пакета заданий (пула), выбираются из глобальной очереди, по мере их планирования на

ресурсы помещаются в пакет заданий на назначенный (выбранный) ресурс (ресурсы) и далее передаются на решение на этот ресурс (ресурсы). В случае, если пакет заданий на ресурс полностью заполнен, т.е., отсутствует свободное место для вновь спланированного задания на этот ресурс, ему назначается другой, также входящий в покрытие, свободный ресурс; в противном случае задание возвращается в пул с высшим приоритетом и опять планируется. Таким образом, предлагаемая процедура является процедурой динамического планирования и не резервирует ресурсы как принято в системах, использующих алгоритм backfill.

5. На моменты планирования t_k задания m являются независимыми и n ресурсов в системе являются доступными и свободными.

Таблица 1

Матрица соответствия заданий ресурсам

	R_{1*}	R_2	R_{3*}	R_4	R_{5*}	R_6	R_7	R_8
$З_1$	1							
$З_2$		1	1	1				
$З_3$	1					1		
$З_4$			1					1
$З_5$		1	1	1				
$З_6$	1						1	

В качестве метрик производительности работы Грид-системы используются следующие: среднее время выполнения задания (время ответа), время выполнения всех заданий глобальной очереди и коэффициент использования ресурсов.

Среднее время выполнения одного задания определяется по формуле:

$$t_{среднее\_время\_выполнения\_задания} = \frac{1}{N}\sum_{i=1}^{N} t_{i\,выполнения,} \qquad (3)$$

где $t_{i\,выполнения,}$ – суммарное время нахождения в пуле, время планирования, время нахождения в пакете (обслуживания), время передачи по коммуникационной среде, время решения на ресурсе i–ого задания; N – количество заданий глобальной очереди.

Время выполнения всех заданий очереди T_N рассчитывается по формуле:

$$T_N = T_{посл} - T_{перв}, \qquad (4)$$

где $T_{перв}$ – время поступления первого задания в очередь; $T_{посл}$ – времени завершения выполнения последнего задания очереди.

Коэффициент использования ресурса R_j определяется по формуле:

$$K_{использ} = \frac{T_{R_i}}{T_N}, \qquad (5)$$

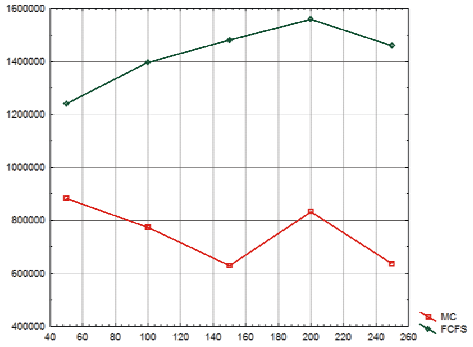
где T_{Ri} – время выполнения всех заданий из глобальной очереди на ресурсе R_j..

Результаты и анализ вычислительных экспериментов

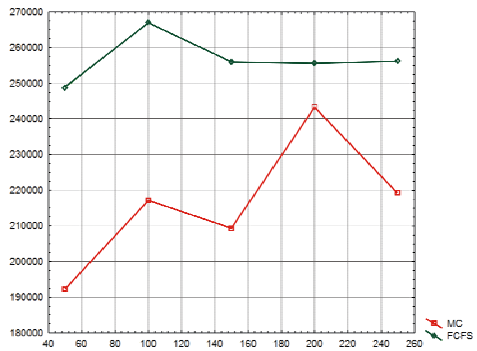
Для проведения вычислительных экспериментов используется имитационная модель системы планирования Грид-системы [5]. В качестве единицы времени планирования и расчетов в программе используется внутреннее время имитационной модели 1 такт, которое соответствует времени решения одного задания, имеющего сложность 1000 MI на ресурсе, производительность которого 1000 MIPS.

План эксперимента предполагает исследование процедуры планирования для заданий высокой вычислительной сложности, которая изменяется по нормальному закону со средним значением 30000 тактов, что соответствует времени решения до 8 часов, и среднеквадратическим отклонением 10000 тактов. Количество ресурсов изменяется в интервале от 5 до 30, средняя производительность ресурса – 100 тактов.

В результате моделирования оценивается эффективность эвристического алгоритма MC по отношению к наиболее распространенному алгоритму FCFS в условиях изменения интенсивности поступления заданий в систему, а также количества доступных ресурсов. На рис. 1, 2 показано изменение времени выполнения всех заданий очереди в зависимости от изменения интенсивности заданий для методов MC и FCFS для 5 и 30 ресурсов. Для 5 ресурсов время выполнения заданий высокой сложности не зависит от периодичности планирования и интенсивности заданий, так как для каждого ресурса формируется очередь заданий, ресурсы освобождаются медленно и дальнейшее планирование осуществляется «на перспективу». Вместе с тем, предлагаемый алгоритм MC имеет значительное преимущество по отношению к FCFS за счет возможности оптимальной упаковки ресурсов. Как следует из графика на рис. 1а, выигрыш во времени выполнения заданий составляет до 45 %. Увеличение количества ресурсов до 30 приводит к уменьшению времени выполнения заданий, однако увеличение интенсивности позволяет линейно увеличить время выполнения заданий (рис. 1б). Аналогичный результат показывают графики на рис. 2. При линейном изменении интенсивности заданий для 5 ресурсов алгоритм MC показывает высокое (предельное) значение коэффициента использования (0,95), что свидетельствует о наличии очереди перед каждым ресурсом. В отличие от MC, алгоритм FCFS позволяет загрузить только 50 % ресурсов (коэффициента использования равен 0,5). Увеличение количества ресурсов до 30 приводит к снижению коэффициента использования у метода MC, и его эффективность приближается к эффективности FCFS. Таким образом, преимущество алгоритма MC для заданий высокой сложности может быть получено в случае ограниченного количества свободных ресурсов или большой интенсивности потока заданий.



а) для 5 ресурсов



б) для 30 ресурсов

Рис. 1: Сравнительный анализ времени выполнения для планировщиков MC и FCFS для разных интенсивностей заданий с высокой вычислительной сложностью

Проведенные эксперименты показали, что при постоянной интенсивности заданий с увеличением количества ресурсов количество поступающих заданий становится меньше количества свободных ресурсов, что приводит к снижению производительности системы (коэффициента использования) как для алгоритма MC, так и для FCFS (рис. 3).

Алгоритм MC и его адаптацию к существующей системе планирования на кластерах реализовано на базе планировщика MAUI, который имеет открытый программный код на языке C.

Структурно адаптация включает в себя процедуры:

1. Настройка конфигурации MAUI для работы с несколькими очередями и определение политик для работы с разными типами пользовательских заданий.

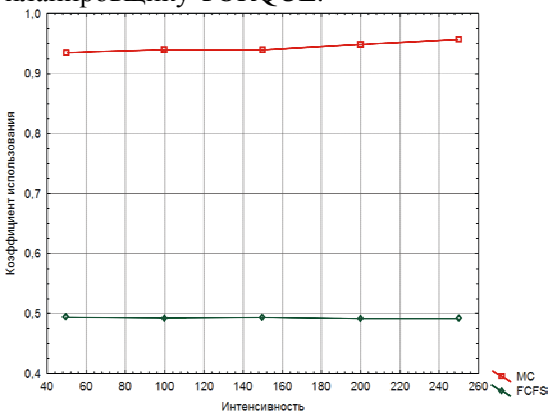
2. Создание пула заданий для алгоритма MC (создание массива с соответствующими типами описания заданий).

3. Заполнение пула теми заданиями, которые определены политиками и результатом работы MAUI.

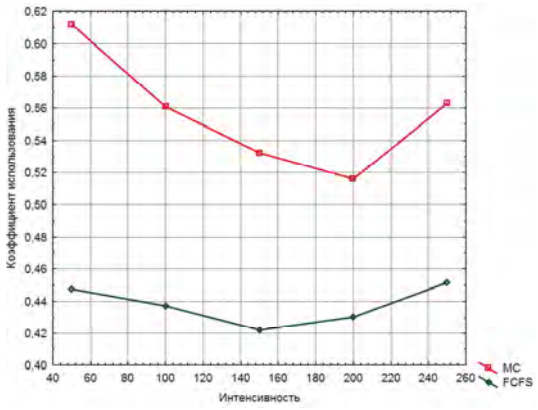
4. Мониторинг свободных ресурсов и формирование массива соответствия заданий-ресурсов.

5. Запуск собственного алгоритма планирования MC.

6. Назначение заданий на ресурсы и передача указателей (назначений) локальному планировщику TORQUE.



а) для 5 ресурсов



б) для 30 ресурсов

Рис. 2: Сравнительный анализ значений коэффициента использования для планировщиков MC и FCFS для разных интенсивностей заданий с высокой вычислительной сложностью

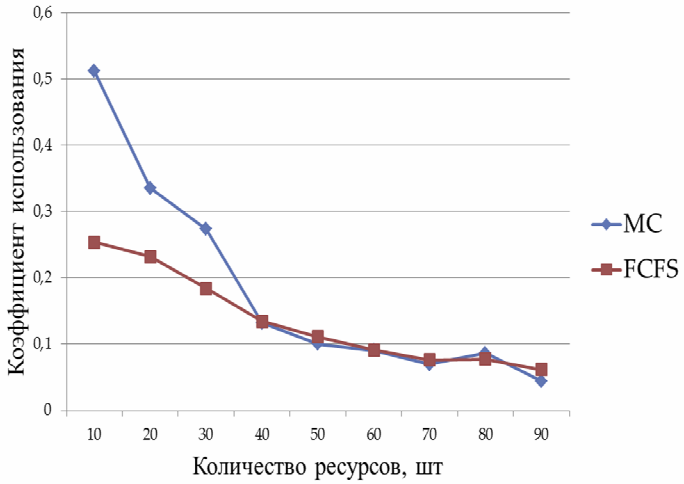


Рис. 3: Зависимость коэффициента использования от количества ресурсов для MC и FCFS

Функции заполнения пула пакетом заданий и вызов алгоритма реализуется в файле «*\maui-3-*-*\src\moab\MSched.c». Для включения и адаптации алгоритма MC в данном файле дополнительно созданы функции создания пула и вызов алгоритма планирования MC.

```
#include "../../contrib/sched/MC_Alg.c" Подключение файла с алгоритмом MC
int MLocalQueueScheduleIJobs(
  int    *Q,
  mpar_t *P)
  {
  mjob_t *J;
  int     jindex;
  if ((Q == NULL) || (P == NULL))
    {
   return(FAILURE);
```

```
  }

 /* NOTE:  insert call to scheduling algorithm here */
 for (jindex = 0;Q[jindex] != -1;jindex++)
  {
  J = MJob[Q[jindex]];
```

/* Вызываем функцию формирования пакета. В параметры передаем переменную работы(массив работ, элементы которого задачи в очереди) и вторым параметром передаем поле ***ARes*** к которому получаем доступ из набора параметров (mpar_t *P)*/

MJobMass(J,P->ARes); /* Call MC Algorithm */

```
    DBG(7,fSCHED) DPrint("INFO:     checking job '%s'\n",
   J->Name);
  }  /* END for (jindex) */

 return(SUCCESS);
 }  /* END MLocalQueueScheduleIJobs() */
```

Выводы

1. Предложенный метод пакетного планирования предпочтительней использовать в случае высокой интенсивности заданий входного потока, превышающей количество свободных ресурсов. При этом на ресурсы формируются очереди, разрешение которых более эффективно осуществляется алгоритмом МС. Для заданий высокой сложности преимущество метода МС нивелируется вследствие значительного превышения времени решения заданий на ресурсах по отношению к времени планирования и ожидания в глобальной очереди и в очереди на ресурсы.
2. Эффективность предложенной процедуры планирования МС зависит от настроек параметров пула, блока планирования, пакетов заданий на ресурсы, количества ресурсов.
3. Предложенный метод позволяет достичь лучших показателей производительности работы системы, а именно, уменьшения времени выполнения всех заданий и увеличения среднего коэффициента использования ресурсов за счет максимальной загрузки ресурсов.
4. Предложенный метод является универсальным: его можно использовать в двухуровневой распределенной системе (Грид-системе) в качестве планировщика первого (глобального) уровня (метапланировщика), так и на уровне гетерогенного кластера Грид-системы.
5. Реализацию алгоритма МС возможно осуществить на базе планировщика MAUI, эффективность которой зависит от предварительных настроек конфигурации MAUI (количества очередей, политик, количества ресурсов).

Литература

[1] Grid Resource Management: State of the Art and Future Trends. // J. Nabrzyski, J. M. Schopf, J. Weglarz (Eds). – Kluwer Academic Publisher, 2004. – 575 p.

[2] Hesham El-Rewini; Hesham H Ali; Theodore G Lewis. Task Scheduling in Parallel and Distributed Systems. – PTR Prentice Hall, 1994. – 304 p.

[3] J. Blythe, S Jain, E. Deelman, Y. Gil, K. Vahi and A Mandal K. Kennedy. Task Scheduling Strategies for Workflow-based Applications in Grids. // In Proc. of International Symposium on Cluster Computing and Grid (CCGrid'05), pp.759–767, Cardiff, UK, May 2005.

[4] Листровой С.В. Метод решения задач о минимальном вершинном покрытии в произвольном графе и задачи о наименьшем покрытии / С.В. Листровой, С.В. Минухин. // Электронное моделирование. – 2012. – Т. 34. – №1. – С. 29 – 43.

[5] Минухин С.В. Имитационная модель и ее программная реализация планирования ресурсов Грид-системы. / С.В. Минухин, С.В. Знахур. // Проблеми програмування. – 2012. – №2–3. Спеціальний випуск. – С. 133–143.

КОНСОЛИДАЦИЯ ЭЛЕКТРОННЫХ БИБЛИОТЕЧНЫХ И ИНТЕРНЕТ-РЕСУРСОВ ДЛЯ ОБРАЗОВАТЕЛЬНЫХ И НАУЧНЫХ ЦЕЛЕЙ НА ОСНОВЕ GRID-ТЕХНОЛОГИЙ

Б.В. Олейников, А.И. Шалабай
ФГАОУ ВПО «Сибирский федеральный университет»,
Россия, 660041, Красноярск, пр. Свободный, 79
oleynik48@mail.ru, andrsh@gmail.com

В настоящее время Интернет является основным местом для размещения публичной информации. По данным аналитического агентства IDC количество информации в Интернет растет в геометрической прогрессии и удваивается каждые 18 месяцев [1]. Так, если в 2008 году в Интернете размещалось 486 эксабайт данных, то в 2012 году ожидается увеличение этого показателя до 2502 эксабайт (2,5 йотабайта). Для сравнения, количество книг в библиотеках мира в середине 90-х годов XX века оценивалось в 20 миллиардов [2]. Предположив, что средний прирост библиотечных фондов составляет не более 2% в год [3, 4]), а оцифрованная книга занимает в среднем 7-8 мегабайт, получаем, что количество информации, находящейся в библиотеках мира, в 2008 году оценочно составляло 0,06% от объема информации в Интернет, а в 2012 году, с учетом прогноза IDC, этот показатель может уменьшиться до 0,012%.

Обеспечивая легкость, оперативность, во многих случаях гипертекстовую связность размещения информации, Интернет вместе с тем несет негативные моменты, связанные, прежде всего, с отсутствием гарантий долговременной сохранности Интернет ресурсов и невозможностью объединения с традиционными хранилищами информации (библиотеки, архивы и пр.). Как следствие, это приводит к невозможности общепринятого (библиотечного) учета ценных документов, являющихся фиксатором человеческого знания, а также консолидированного тематического поиска по всем источникам, осуществляемого по принципу одного окна. Ситуацию усугубляет постоянное дублирование одних и тех же публикаций на различных ресурсах, при этом часто копирование информации происходит без указания первоисточника.

Для частичного решения указанных проблем многие ведущие организации и компании разрабатывают соответствующие программы и предпринимают определенные усилия, направленные на массовое архивирование электронных копий документов, предполагая и ресурсы Интернет [5]. Это позволяет сохранить созданные человечеством информационные ресурсы, представленные в цифровом виде. В последние годы, с целью оперативного сохранения цифрового наследия, широкое распространение получает идея непосредственного архивирование информации, размещенной в сети Интернет. Некоторые страны, а также крупнейшие национальные библиотеки разрабатывают проекты, призванные обеспечить сохранность данных, находящихся в конкретных национальных доменных зонах. Наиболее известные из этих проектов – американский NDIIPP (LCWA/MINERVA), австралийский Pandora, британский UK WebArchive, австрийский Web@rchive и другие. Архивированием всей сети Интернет занимается некоммерческая организация InternetArchive, которая предоставляет и бесплатный сервис для просмотра архива. В рамках своей деятельности организация InternetArchive достаточно мощно взаимодействует с библиотекой Конгресса США, которая архивирует сайты с 2000 г.

В январе 2012 году было опубликовано руководство Международной федерации библиотечных ассоциаций и институтов (IFLA) [6], в котором указана необходимость консолидации цифровой литературы (в том числе находящейся в открытом Интернет-доступе) в традиционных библиотеках, а также важность разработки отдельной политики для отбора хранимых электронных ресурсов. В этом руководстве не дается рекомендаций по разработке подобной политики, подчеркивается лишь важность и сложность отбора наиболее ценной в

научном плане информации для долговременного и надежного хранения.

В России в рамках концепции Национальной электронной библиотеки (подготовлена РГБ и РНБ в 2004 г.) [7] также предусмотрена работа по сбору электронных изданий, свободно размещенных в Интернет (п. 5.2.2.), однако развернутых программ реализации этой деятельности, подобных зарубежным, пока нет. Последним по времени значимым проектом является новый федеральный проект Минобрнауки РФ [8], основной целью которого является создание информационной системы доступа к электронным каталогам библиотек сферы образования и науки в рамках единого интернет-ресурса.

В качестве одного из подходов, обеспечивающих распределенную долговременную тематическую сохранность открытых Интернет ресурсов, можно рассматривать, предложенное одним из авторов, создание личных полнотекстовых коллекций, базирующихся на использовании распространенных современных полнотекстовых библиотечных систем [9]. При наличии библиографических описаний (записей) электронных ресурсов, личные полнотекстовые коллекции с успехом могли бы использоваться при формировании электронных фондов традиционных библиотек, получающих дополнительные возможности при минимальных затратах.

Для составлении библиографических записей (включающих библиографические описания) в настоящее время разработано несколько форматов, используемых в автоматизированных библиотечных информационных системах (АБИС). Это форматы семейства MARC (в России используется локализация формата UNIMARC - RusMARC [10]), формат Dublin Core – DCMES, формат MODS (считающийся подмножеством MARC21), а также новая схема описания RDA, основанная на модели FRBR, и др. [11]. Каждый из них имеет свои особенности, начиная от широты охвата описания ресурса, определяемое количеством заполняемых полей и кончая требованиями к их учету при заполнении. Предусматривается возможность конвертации между этими форматами (включая и автоматическую в случае от широкого к узкому описанию ресурса).

В библиотеке Конгресса США сайты описываются в основном с использованием формата MODS, а для генерального каталога еще и в стандарте MARC, так что веб-коллекции описаны в общем каталоге вместе с другими материалами [12].

Со стороны поисковых служб Интернета предпринимаются усилия к интеграции традиционного библиотечного содержания и служб в своих поисковых машинах. В частности, можно указать два проекта Google: Google LibraryThing и Google BookSearch.

Учитывая изначально цифровое представление Интернет-ресурсов для их каталогизации целесообразно разработать подход, заключающийся в автоматическом составлении и публичном размещении библиографических записей цифровых документов в наиболее распространенном формате (для России это формат RusMARC). При этом многие поля RusMARC могут заполняться в автоматическом режиме (URL, протокол доступа, размер файла и др.), либо полуавтоматическом (автор, название, тематика).

В дальнейшем, основываясь на межкоммуникативных стандартах и протоколах (ISO 2709, Z39.50 и др.) такими библиографическими записями можно будет обмениваться с любыми библиотеками, деятельность которых базируется на использовании различных АБИС.

Очевидно, что в силу огромного количества цифровой литературы, ни одна организация не способна осуществить полную ее каталогизацию. Поэтому предлагается использовать идеи Грид для распределения человеческих и вычислительных ресурсов и создать библиотечную систему, служащую для каталогизации Интернет-ресурсов. Учитывая, что в настоящее время образовательные и научные ресурсы носят комплексный характер и наиболее многопланово представлены в сети Интернет (общие и специализированные сайты и web-порталы, электронные библиотеки, сайты свободного доступа, базы данных, ftp-ресурсы и др.), а также традиционно представлены и в классических библиотеках, в первую очередь рассматривается взаимодействие с образовательными и научными ресурсами.

Концепция разрабатываемой распределенной библиотечной системы образовательных и научных ресурсов строится на основании следующих положений:

1) Полные тексты документов хранятся в узлах Грид-сети, являющихся территориально распределенными клиентскими рабочими местами, имеющими выход в Интернет. Каждый узел Грид-сети направлен на сосредоточение определенной тематической информации (тематик может быть и несколько), часть из которой может быть задействована другими узлами. В этом случае, наиболее актуальная информация дублируется на локальных ресурсах множества компьютеров и, следовательно, даже с отключением части узлов, остается доступной в сети. Данный подход позволяет экономить серверные ресурсы и обеспечивать более высокую надежность сохранности ресурсов и скорость обмена информацией.

2) На узлах Грид-сети обеспечивается поиск Интернет-документов, порождение требуемых библиографических записей, индексирование документов, а также их резервное долговременное хранение в виде полных текстов. Таким образом, узлы Грид-сети являются резервными держателями представляющих интерес документов. Затем полученные библиографические записи, их держатели, а также список ссылок на полные тексты Интернет-документов направляются в централизованное хранилище (центральный узел распределенной библиотечной системы). Использование такого хранилища позволяет оперативно осуществлять поиск требуемых документов в образовательной сети без непосредственного поиска в Интернет и последовательного опроса узлов. Дополнительно появляются возможности централизованной каталогизации ценных документов сети Интернет (частичное решение проблемы выборочного архивирования Интернет), а также обмена библиографической информацией с уже существующими системами по протоколу Z39.50. Кроме того, вся описательная информация дублируется непосредственно на ресурсах узла, поэтому даже при недоступности централизованного хранилища Грид-сеть может продолжать функционировать.

3) Узлам сети присваиваются определенные полномочия, например добавление новой информации определенных тематик (размещение информации в хранилище), её модификация, возможность блокировки определенных узлов с изъятием полученных от них данных и т.д. Также целесообразно для каждого узла разделять полномочия для документов разной области знаний.

Составление библиографической записи найденного узлом Грид-сети документа, по возможности, должно осуществляться автоматически. В частности, в формате RusMARC предполагается, автоматическое заполнение обязательных для электронного документа полей 001, 100, 101, 200, 230, 300, а также полей 102, 135, 675 (676,679,680,686 686 – для других классификаторов кроме УДК), 700, 856, необходимых для организации поиска документов по основным критериям. В случае, когда невозможно автоматически определить требуемые для библиографического описания данные, необходимо участие человека-сотрудника узла. Так, для текстового документа, размещенного в Интернет, в большинстве случаев возможно автоматически установить название документа, дату его создания (размещения), а в некоторых случаях автора и тематику по одному из библиотечных классификаторов (например, УДК). В случае же невозможности автоматического определения части этих характеристик документа (большей частью это относится к библиотечным классификаторам) эта задача ложится на сотрудника узла Грид-сети, размещающего документ в сети. При консолидации библиографической информации о документах (независимо от их типа и места описания) необходимо обеспечить уникальность идентификаторов записей - поля 001. Никаких ограничений на форму этого поля не устанавливается, и для бумажных книг в качестве него может использоваться любой международный стандартный номер, в частности, в большинстве случаев туда заносится код ISBN (для цифровых документов может заноситься код DOI).

При передаче цифровых документов в Грид-сети необходим дополнительный идентификатор описании электронного документа – контрольная сумма (MD5), отвечающий за целостность документа. Это обеспечит уникальность идентификатора, предоставит возможность объединения нескольких описаний (полученных в разных местах) одного документа и позволит получать полные тексты документов одновременно из нескольких мест их хранения (подобный подход широко используется в сетях p2p). Для размещения контрольной суммы может быть использовано одно из полей дополнительного блока 9 – блока

локального использования формата RusMARC.

Корректность определения всей библиографической информации проверяется модераторами соответствующего раздела области знаний при получении библиографической записи от узла Грид-сети, после чего запись помещается в хранилище.

В тематический справочник, кроме библиотечного классификатора, возможно добавление разделов, посвященных различным конкурсам и грантам, правилам приема для абитуриентов и т.д., а также списка образовательных учреждений, входящих в проект, что позволит конкретным учреждениям размещать различные положения и нормативные документы, а при поиске отделять эту информацию от учебных материалов.

4) Взаимодействие узлов сети по части доступа к документам может осуществляться в соответствии с технологиями, принятыми в файлобменных сетях..

5) Доступ к полнотекстовым документам может осуществляться различными способами: возможно отображение списка всех полнотекстовых документов каждого конкретного узла, формирование каталога с документами определенной тематики, полнотекстовый поиск по всему хранилищу или его части.

6) Целесообразным является предоставить возможность узлам прикреплять к документам отсканированные копии авторских согласий на размещение материалов, которые будут доступны всем желающим для ознакомления.

7) Каждый узел может самостоятельно группировать документы из различных источников по определенным признакам. Например, для размещения учебно-методического комплекса документации (УМКД) в системе достаточно добавить группу, содержащую рабочую учебную программу, методические указания и рекомендации. Список учебной литературы возможно сформировать из уже присутствующих в системе полнотекстовых документов и присоединить его к созданной группе. Таким образом, УМКД будет размещен не на одном узле, а рассредоточен по всей системе, в то же время это не будет видно обычному пользователю, который, выбрав УМКД для ознакомления, получит весь список доступных для скачивания материалов, независимо от того где они физически находятся.

Грид-сеть, реализующая предложенную концепцию, имеет общепринятую архитектуру [13], каждый уровень которой несет соответствующую функциональную нагрузку.

На базовом уровне (fabric level) размещаются полнотекстовые ресурсы и библиографические записи с помощью стандартных инструментов файловой системы, представленная дополнительная информация сосредотачивается в специально спроектированной реляционной базе данных, в которой объединяется информация со всех зарегистрированных узлов.

Уровень связи (connectivity layer) может быть реализован стандартными средствами:

1) Для идентификации каждого узла используется его IP адрес или доменное имя.
2) Служба авторизации определяет набор привилегий, необходимых как для добавления новой информации в систему, так и для доступа к уже размещенным полнотекстовым источникам (узлы-поставщики могут ограничивать доступ к некоторым данным).
3) Связь между узлами обеспечивается посредством протокола TCP/IP.

Ресурсный уровень (resource layer) библиотечной Грид-сети обеспечивает мониторинг и учет использования ресурсов, формирует список доступных в данный момент узлов и для каждого из них ведет статистику скачанных цифровых документов. Определяет политику использования ресурса: набор правил доступа к каждому полнотекстовому источнику, ограничение на максимальное число подключений за единицу времени. Для каждого узла формирует список хранимых полнотекстовых источников и предоставляет его по требованию других узлов.

Коллективный уровень (collective layer) объединяет следующие службы и протоколы:

1) Служба обмена метаданными передает файлы в формате RusMARC и xml между узлами и централизованным хранилищем. Первый используется для хранения библиографических записей, а второй содержит дополнительную информацию о

полнотекстовом источнике, в том числе данные о узле-поставщике и полный список ключевых слов. Необходимость централизованного хранилища обусловлена потребностью дальнейшего быстрого поиска в системе, а также важностью существования единого каталога описаний литературы для объединения его с каталогами традиционных библиотек. Подобная схема не нарушает принципов Грид, т.к., во-первых, вычислительная, ресурсная и сетевая нагрузка на централизованное хранилище незначительна, а поэтому не требует распределения: объем передаваемых данных незначителен, поиск документов осуществляется с помощью простейших (не ресурсоемких) запросов к реляционной базе данных. Во-вторых, недоступность хранилища в определенный момент времени влечет за собой только прекращение работы единой поисковой системы, а не всей Грид-сети, т.к. все полнотекстовые источники и соответствующие им описания хранятся в том числе и на узлах-поставщиках.

2) Служба резервирования данных отслеживает появление новых документов в Грид-сети, копирует их на определенное число произвольных узлов (определяемое общей нагрузкой на сеть). Данная служба управляется из двух источников: максимальное число автоматически создаваемых резервных копий определяется администратором системы, в то же время узел может запретить копировать предоставленную им литературу, являющуюся объектом имущественного права и запретить копирование сторонних ресурсов в его хранилище.
3) Служба выделения ресурсов предоставляет адрес для скачивания полнотекстовых источников узлам-потребителям в соответствии с набором правил, определенным каждым узлом-поставщиком (максимальное количество скачиваний в единицу времени, приоритет скачивания с конкретного узла - в случае предоставления данных по HTTP-протоколу подобная возможность позволяет передавать дополнительную информацию, например рекламного характера). Для распределения вычислительных ресурсов не требуется наличие специальной службы, т.к. каждый узел самостоятельно определяет для себя список задач.
4) Протоколы доступа к полнотекстовым источникам определяются узлами Грид-сети. В частности, могут поддерживаться NetBios, HTTP, FTP.

На прикладном уровне (application layer) Грид-сети работают два модуля, реализующие функции двух типов узлов:

1) Узел-поставщик обрабатывает данные, полученные из активного окна браузера, сервисами проверки доменных имен (Whois), свойств файла и т.д., с помощью которых формирует библиографическую запись в автоматическом режиме (включая, по возможности, и определение тематического кода документа – например, УДК). Определяет контрольную сумму полнотекстового источника и индексирует его. Пользователь при необходимости вносит дополнительные данные. Сформированная информация отправляется в централизованное хранилище. Кроме того, при наличии определенных полномочий модуль предоставляет функции управления Грид-сетью в целом, либо модерирования размещенной литературы.
2) Узел-потребитель реализует сервисы поиска, коммуникаций с централизованным хранилищем (для получения библиографических записей) и узлами поставщиками (для установления отношений по получению полнотекстовых источников).

Таким образом, узлы-поставщики являются членами виртуальной организации – единой распределенной библиотеки, в которой аккумулируется литература из множества источников, а узлы-потребители представляют собой посетителей этой библиотеки.

С внедрением предложенной библиотечной Grid-сети и объединением баз данных описательной информации с классическими библиотеками появится возможность одновременного её поиска в интернет (в частности, на web-порталах, открытых сайтах, в специальных web-архивах и др.), в классических и электронных библиотеках. При этом будет возможен гибкий доступ к данным с помощью отбора документов практически по всем

критериям, принятым в классических библиотеках, тем самым будет обеспечена возможность консолидации требуемых ресурсов, независимо от места их размещения, и возможность доступа к ним с позиций одного окна.

Данную разработку можно рассматривать как своеобразное дополнение к проекту [8], но в отличие от него, в настоящей работе основной акцент ставится на то, как обеспечить массовое представление исходных цифровых ресурсов, находящихся, по сути дела, где угодно, для обеспечения последующего эффективного доступа к ним с позиций единого «интернет-окна».

Литература

[1] Digital Information Growth Outpaces Projections, Despite Down Economy. // EMC. URL: http://www.emc.com/about/news/press/2009/20090518-01.htm (дата обращения 29.08.2012).

[2] Сколько в мире библиотек // Международная ассоциация русскоязычных писателей. URL: http://www.rulit.org/read/7#5 (дата обращения 22.08.2012).

[3] Библиотеки Новгородской области в 2010 году // КБД Центральные библиотеки субъектов Российской Федерации. URL: http://www.nlr.ru/nlr/div/nmo/zb/part/search.php?id=2561&r=2 (дата обращения 12.08.2012).

[4] The Library of Congress. URL: http://www.loc.gov/about/generalinfo.html (дата обращения 08.08.2012).

[5] Бизнес-аналитика IBM в системах интеллектуального архивирования данных // Электронный журнал BYTEmag URL: http://www.bytemag.ru/articles/detail.php?ID=16118 (дата обращения 14.02.2012).

[6] Sharon Johnson. Key Issues for e-Resource Collection Development: A Guide for Libraries // IFLA Acquisition & Collection Development Section, 2012. URL: http://www.ifla.org/publications/key-issues-for-e-resource-collection-development-a-guide-for-libraries (дата обращения 18.08.2012).

[7] Национальная электронная библиотека: Концепция. URL: http://www.elibconsult.ru/page.jsp?pk=node_12785007639801, www.nlr.ru:8101/e-resn/concept.doc (дата обращения 13.03.2012).

[8] Шрайберг Я.Л. Информационная система доступа к электронным каталогам библиотек сферы образования и науки в рамках единого интернет-ресурса: новый федеральный проект Минобрнауки РФ. / Всероссийская научно-практическая конференция «Фонды библиотек в цифровую эпоху: традиционные и электронные ресурсы, комплектование, использование», 19-23 марта, г.Санкт-Петербург. URL: http://www.nlr.ru/tus/20120319/prog.html (дата обращения 08.06.2012).

[9] Олейников Б.В. Создание эффективного инструмента формирования личных полнотекстовых коллекций для научной и образовательной деятельности // Электронные библиотеки: перспективные методы и технологии, электронные коллекции: Труды Восьмой Всероссийской научной конференции (RCDL-2006). Суздаль, 17-19 октября 2006. / Ярославский государственный университет им. П.Г.Демидова. - 2006. - С.176-183.

[10] Национальная служба развития системы форматов RusMarc. URL: http://rusmarc.ru (дата обращения 28.08.2012).

[11] Жлобинская О.Н. MARC-форматы в современной информационной среде. - РНБ// http://www.rusmarc.ru/publish/MARC_now.pdf

[12] Браккер Н.В., Куйбышев Л.А. Сбор и архивирование сетевых ресурсов. Опыт национальных библиотек зарубежных стран.// Научная конференция «Электронные ресурсы в информационно-телекоммуникационной среде: законодательные основы комплектования, хранения и доступа» - М.: РГБ, URL: www.minervaplus.ru/.../Harvesting_Preservation_Net_Resources.doc (дата обращения 28.05.2012).

[13] Кирьянов А.К., Рябов Ю.Ф. Введение в технологию Грид. //http://window.edu.ru/resource/689/49689/files/Metodichka-grid.pdf (дата обращения 02.03.2012).

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ С ОТКРЫТЫМ ИСХОДНЫМ КОДОМ ДЛЯ ПОСТРОЕНИЯ И УПРАВЛЕНИЯ ОБЛАЧНЫМИ СРЕДАМИ НА РАСПРЕДЕЛЕННЫХ ГЕТЕРОГЕННЫХ ИНФРАСТРУКТУРАХ

А.В. Пярн

Московский государственный университет имени М.В. Ломоносова,
Факультет вычислительной математики и кибернетики

Цель работы

Целью работы является изучение категории программного обеспечения с открытым исходным кодом, используемого для управления облачными инфраструктурами, позволяющее объединить отдельные виртуализованные сервера архитектуры x86 и x86-64 и системы хранения данных в единый ресурсный пул, используя встроенные функциональные возможности гипервизоров, организовать возможность предоставления виртуальных серверов по модели IaaS.

Целью исследования является сопоставление архитектур и функциональных возможностей наиболее перспективных программных продуктов с открытым исходным кодом и их применимость для создания виртуальных образовательных полигонов.

Основными аспектами исследования являются:

1) реализация вычислительной подсистемы, взаимодействие с гипервизорами;
2) реализация подсистемы хранения данных для централизованного хранения и управления образами виртуальных машин, шаблонами настроек виртуальных машин;
3) реализация миграции виртуальных машин между узлами кластера без останова запущенных сервисов;
4) реализация сетевой подсистемы;
5) реализация графического интерфейса пользователей;
6) общая архитектура программного обеспечения, целесообразность использования тех или иных решений.

Основными рассматриваемыми в работе решениями являются OpenNebula и OpenStack.

OpenNebula [1] - приложение, разработанное в Мадридском университете Universidad Complutense de Madrid, на сегодняшний день успешно используется в крупных компаниях и научно- исследовательских институтах, среди которых China Mobile, CERN, Космическое агентство Европы. Поддерживает интеграцию с облачными сервисами Amazon, поддержку протокола OCCI и EC2 Query API.

OpenStack [2] - комплекс проектов свободного программного обеспечения, основной вклад в который внесли компании Rackspace Hosting и NASA, объявившие об открытии кода проектов платформы Nebula (IaaS - NASA) и платформы Rackspace Cloud Files (Rackspace Hosting). Может быть использовано для создания вычислительных облаков и облачных хранилищ, как публичных, так и частных.

Результаты работы

Программное обеспечение (ПО) было установлено и протестировано на оборудовании в следующей конфигурации: использовался модульный сервер DEPO Storm 5302A1 с четырьмя вычислительными модулями DEPO Storm 3300JZ. Каждый вычислительный модуль компоновался двумя шестиядерными процессорами серии Intel Xeon 5600, поддерживающими режим аппаратной виртуализации Intel VT, 24 Гб оперативной памяти, 3 жесткими дисками (3.5 дюйма), двухканальным Gigabit Ethernet контроллером Intel® 82574L.

Первый вычислительный модуль использовался в качестве управляющего сервера OpenNebula, а так же в качестве системы хранения данных для образов, шаблонов, настроек виртуальных машин (в качестве системы хранения данных использовался запущенный nfs-сервер). Используемая ОС – Ubuntu Server 10 [4], дистрибутив управляющего ПО – OpenNebula 3.6 (далее – OpenNebula).

Второй и третий вычислительный модули использовались в качестве рабочих серверов (узлов) [5, 6] для запуска виртуальных машин под управлением ПО OpenNebula, на них были установлены и настроены гипервизоры QEMU-KVM [3], работающие на базе ОС Ubuntu Server 12.04 LTS.

Четвертый модуль, работающий под управлением ОС Ubuntu Server 12.04 LTS [4], был использован для тестирования ПО OpenStack Essex (2012.1) (далее – OpenStack). Он совмещал в себе ПО управления и рабочий узел виртуализации, т.к. данная конфигурация для ПО OpenStack является допустимой в тестовых средах и инсталлируется базовым установочным скриптом [8].

В качестве гипервизора на всех рабочих узлах использовался модуль QEMU-KVM [5, 6], собранный в составе ядра ОС Ubuntu Server 12.04 LTS.

В системе OpenNebula дополнительное ПО на рабочие узлы не устанавливалось, управление гипервизором осуществлялось посредством библиотеки libvirt [7], входящей в состав стандартных дистрибутивов Linux, за счет установления беспарольной rsa-ssh сессии между управляющим сервером OpenNebula и рабочими узлами.

Сводные результаты исследования представлены в разделе «Сравнение платформ OpenNebula и OpenStack».

Сравнение платформ OpenNebula и OpenStack
1. Общие аспекты

	История проекта	**Размер и активность сообщества**	**Примечания**
OpenStack	OpenStack изначально проект хостинговой компании RackSpace и NASA, открывших часть исходных кодов. От NASA – вычислительная часть Nova, от RackSpace – объектное хранилище Swift. Критика OpenStack обычно касается (1) непрозрачность проекта, (2) изначально сложная структура и плохая логическая связность входящих в состав проекта компонент, низкое качество кода, сложность многоузловой установки.	Сообщество быстро растущее. Разработку официально поддерживают многие международные компании производители вычислительного оборудования и ПО, но четкого разграничения ответственности и вклад компаний в разработку не ясен.	Обращений в техническую поддержку не было, установка ПО в простой одноузловой конфигурации прошла без инцидентов.

OpenNebula	Исследовательский проект, стартовавший в 2005 году, с 2008 года начинается период активного роста и продвижение проекта, открыты исходные коды. Четкая логическая структура ПО. Стабильные и запланированные релизы, продуманная миграция на новые версии ПО, прозрачность установки и настройки.	Поддержка на уровне сообщества у OpenNebula на текущий момент больше чем, поддержка у OpenStack. Международных компаний, официально заявляющих о поддержке проекта, существенно меньше, чем OpenStack, но проект внедрен и используется во многих европейских научно-исследовательских центрах, в т.ч. ЦЕРНе.	При обращении в техническую поддержку отвечают непосредственно разработчики ПО. Ответы точные, оперативные, время реакции обычно не более 24 часов.

2. Архитектурные аспекты

ПО	**Сравнение**
OpenStack	OpenStack – комплекс проектов в рамках единой платформы, состоит из нескольких различных по назначению логических частей: Nova — вычислительный модуль, сетевой сервис, контроллер вычислительных ресурсов, Swift — объектное хранилище, Glance — сервис управления образами виртуальных машин, Keystone — сервис идентификации, Horizon - web-портал управления. Каждый модуль требует установки и настройки специальных компонент и необходимых для их работы системных пакетов ОС. Коммуникации между компонентами осуществляются по протоколу AMPQ через выделенный брокер соединений – вычислительный контроллер. Для обеспечения работы вычислительного модуля, необходимо на каждом рабочем сервере запускать сетевые и вычислительные агенты, которые взаимодействуют с управляющим контроллером. Системные настройки хранятся в SQL базе данных (MySQL, PostgreSQL). Система может быть легко децентрализована за счет разнесения сервисов по различным физическим серверам. **1) Реализация вычислительной подсистемы, взаимодействие с гипервизорами** Вычислительный модуль `nova-compute` устанавливается на каждый рабочий сервер, управляет работой гипервизоров и виртуальных машин посредством локального выполнения системных команд, поддерживаемых гипервизором. Вычислительный модуль взаимодействует с вычислительным контроллером (`nova-api`), сервисом аутентификации (`keystone`), сетевым сервисом (`nova-network`), диспетчером задач (`nova-scheduler`) и другими сервисами OpenStack. Управление блочными устройствам и их подключением к виртуальным машинам в релизе Essex так же осуществляется данным модулем (пакет `nova-volume`). Реализация блочных устройств осуществляется посредством функциональности системного ПО Linux LVM, либо подключением внешних блочных систем хранения данных iSCSI.

	2) Реализация подсистемы хранения данных для централизованного хранения и управления образами виртуальных машин, шаблонами настроек виртуальных машин Управление хранением и использованием образов виртуальных машин осуществляется модулем glance, который по умолчанию работает в связке с объектным хранилищем Swift. В качестве хранилища для образов, помимо Swift, можно использовать обычную файловую систему. Объектное хранилище Swift (Object Store) позволяет преобразовать серверы в масштабируемое хранилище данных со встроенными функциями по обеспечению отказоустойчивости. Система автоматически делает несколько избыточных реплик данных между серверами и в случае сбоя одного из серверов, целостность данных не нарушается. Не является файловой системой и плохо работает с OLTP данными, предназначено для долгосрочного хранения больших объектов (образы виртуальных машин, мультимедиа-контент), аналог сервиса Amazon S3 [9]. **3) Реализация миграции виртуальных машин между узлами кластера без останова запущенных сервисов** Миграция работающих виртуальных машин осуществляется исключительно посредством функциональных возможностей гипервизора, миграция возможна только между узлами с установленными гипервизорами одного типа. **4) Реализация сетевой подсистемы** Сетевые настройки реализованы посредством использования встроенных механизмов ОС Linux по управлению сетью – создание мостов, vLAN-ов. Управляет сетью модуль `nova-network`. **5) Реализация графического интерфейса пользователей** Графический интерфейс пользователя и администратора реализован посредством модульного web-сервера, написанного на языке Python с использованием фреймворка Django. Является графическим интерфейсом ко всем основным сервисам OpenStack.
OpenNebula	Архитектуру ПО OpenNebula можно представить в виде трех основных слоев: функциональных драйверов, монолитного ядра системы и специализированных утилит. Драйверы ответственны за работу с системными компонентами и ОС – гипервизорами, файловыми и сетевыми сервисами, виртуальными машинами, легко настраиваются администратором под свои функциональные задачи. Ядро (системные процессы на управляющем сервере) управляет всеми компонентами системы - виртуальными машинами, системами хранения данных, виртуальными сетями, осуществляет балансировку нагрузки и диспетчеризацию запросов и команд. Утилиты обеспечивают дополнительный функционал в работе системы. **1) Реализация вычислительной подсистемы, взаимодействие с гипервизорами** Работа с гипервизорами осуществляется посредством установления беспарольной rsa-ssh сессии с рабочими узлами, установка агентов не требуется. **2) Реализация подсистемы хранения данных для централизованного хранения и управления образами виртуальных машин, шаблонами настроек виртуальных машин** Поддерживаются все виды сторонних систем хранения данных (СХД) – блочные, файловые, локальные, распределенные, дополнительных

	ограничений на тип системы не накладывается. За взаимодействие отвечают драйверы, которые настраиваются под используемый тип СХД. Отсутствуют сервис автоматического управления подключаемых к виртуальным машинам блочных устройств, необходимо настраивать вручную. Нет собственной системы хранения данных. **3) реализация миграции виртуальных машин между узлами кластера без останова запущенных сервисов** Миграция работающих виртуальных машин осуществляется исключительно посредством функциональных возможностей гипервизора, миграция возможна только между узлами с установленными гипервизорами одного типа. **4) реализация сетевой подсистемы** Сетевые настройки реализованы посредством использования встроенных механизмов ОС Linux по управлению сетью – создание мостов, vLAN-ов. Возможно использование распределенного виртуального коммутатора. **5) реализация графического интерфейса пользователей** Графический интерфейс пользователя и администратора реализован посредством модульного web-сервера, написанного на языке Ruby с использованием фреймворка Sinatra.

3. Установка системы

	Управляющий сервер	**Рабочие серверы (узлы)**	**Примечания**
OpenStack	Используется официальный дистрибутив Ubuntu Server, запускался автоматический установочный скрипт. Установка в одноузловой конфигурации, совмещающей роль управляющего сервера и рабочих серверов.		Для простых тестовых конфигураций, установка простая, выполняется без инцидентов и с минимальным участием администратора. Для производственной установки (многоузловой) инсталляция существенно сложнее. Отсутствует алгоритм централизованного перезапуска системы, необходимо перезапускать все установленные компоненты по отдельности, в т.ч. на рабочих узлах.
OpenNebula	Установка из исходников по документации, т.к. автоматическая установка пакета из репозитория проходит некорректно. Первичная настройка системы: генерация ключей для беспарольного доступа к рабочим узлам, настройка административного эккаунта и группы, настройка NFS сервера, настройка сетевых служб.	Установка ПО на рабочие серверы не требуется, требуется настройка административного эккаунта и группы, подключения разделяемых файловых директорий для системы хранения данных NFS, настройка сетевых служб.	Часть зависимостей автоматически могут не разрешаться, приходится доустанавливать вручную. Нет автоматизированного скрипта полной установки. Хорошая документация, в которой зафиксированы все основные сложности при установке и базовой настройке. Системные сервисы централизованного останова и запуска управляющего сервера.

Выводы

Обе рассмотренные системы, не смотря на заявления сообществ-разработчиков, пока еще не являются законченными решениями для построения систем по предоставлению облачных сервисов IaaS, в том виде и с той функциональностью, как это реализовано, например, на самой распространённой коммерческой закрытой платформе Amazon Web Services (AWS) [10]. Решения по своим функциональным возможностям, возможно, в силу своей универсальности, уступают проприетарным аналогам и требуют существенной доработки под конкретные задачи и архитектуры. Рассмотренное ПО является удобной управляющей прослойкой между отдельными системными компонентами – сетевыми сервисами, системами хранения данных, файловыми сервисами, гипервизорами, оставляя реализацию данных компонент на разработчике решения, в то время как качество и надежность облачного решения зависит преимущественно от последних. Преимуществами данных решений является наличие модульного web-портала управления виртуальными средами, хорошая масштабируемость при росте нагрузок, единые интерфейсы управления. Не смотря на имеющиеся ограничения, рассмотренное ПО, с учетом возможностей проектных доработок, обеспечивает необходимый инструментарий для создания систем виртуальных образовательных полигонов, выступая надежным и развивающимся фреймворком для построения такого рода систем в короткие сроки.

Рассмотрение систем выявило следующие недостатки и отсутствие необходимых функциональных возможностей, которые должны являться темами отдельных исследований:

1) отсутствие сервисов автоматического перезапуска виртуальных машин при сбое рабочего сервера;
2) отсутствие или неполную реализацию высоконадежных распределенных, в том числе блочных систем хранения данных;
3) отсутствие специализированных систем хранения данных, файловых систем, оптимизированных под специфику виртуализированных сред;
4) слабую степень проработки специализированных средств управления сетевыми настройками в масштабируемых виртуальных инфраструктурах;
5) отсутствие собственных гипервизоров в составе систем и соответственно отсутствие механизма оптимизации гипервизора под задачи системы.

Литература

[1] http://opennebula.org/ - официальный сайт проекта OpenNebula.
[2] http://www.openstack.org/ - официальный сайт проекта OpenStack.
[3] http://www.linux-kvm.org/ - официальный сайт проекта KVM.
[4] http://help.ubuntu.ru/ - официальный сайт проекта Ubuntu.
[5] http://www.ibm.com/developerworks/ru/library/l-linux-kvm/ - статья, «Архитектура и преимущества KVM».
[6] http://www.ibm.com/developerworks/ru/library/l-virt/ - статья, «Виртуализация в GNU/Linux».
[7] http://libvirt.org/index.html - официальный сайт проекта libvirt.
[8] http://devstack.org/
[9] http://aws.amazon.com/s3/
[10] http://aws.amazon.com/

ЭФФЕКТИВНЫЙ МОНИТОРИНГ КОММУНИКАЦИЙ НА ОСНОВЕ ВНЕШНЕЙ АППРОКСИМАЦИИ ГРАФА[1]

А.М. Раппопорт

Центр Грид технологий и распределенных вычислений, Институт системного анализа РАН, Россия, 117312, Москва, проспект 60-летия Октября д.9, ram43@mail.ru

При организации мониторинга систем коммуникаций естественно возникает проблема построения кратчайшего замкнутого маршрута, содержащего все элементы. Один из подходов к ее исследованию дает решение задачи о китайском почтальоне (см. [1-3]). В тоже время в эйлеровом графе всегда имеется замкнутый обход всех ребер без повторений. Поскольку все степени в нем четные [4], естественно попытаться в произвольном графе изменить нечетные степени на четные, затратив на это минимальное число операций. Такому подходу и посвящен настоящий доклад. В нем для произвольного связного графа ищется включающий его ближайший в метрике Хемминга эйлеров граф, т.е. отличающийся лишь наличием новых ребер. (Следует отметить, что добавление новых ребер не обязательно влечет появление дополнительных коммуникаций, поскольку исходная конфигурация может являться частью уже имеющейся структуры.)

Пусть $G = G(V,E) = G(V_1,V_2,E)$ - конечный, связный (неориентированный) граф, где $V_1,(V_2)$ - подмножества вершин четной (нечетной) степени, $|V_1| = n_1, |V_2| = n_2$, $G_1 = G_1(V_1,E_1), G_2 = G_2(V_2,E_2)$ - подграфы графа G на соответствующих подмножествах вершин. Ищется эйлеров граф $G' = G'(V,E')$, $E \subset E'$, минимизирующий хеммингово расстояние от исходного графа, т.е $\rho(G,G') = |E\Delta E'| \to \min$.

Один из способов аксиоматического введения этого расстояния предложен в [5]. Как уже отмечалось, «эйлеровость» графа гарантируется четностью степеней всех его вершин. Поэтому минимизации перестроений структуры графа для преобразования к искомому виду можно добиться изменением всех нечетных степеней на единицу. Это замечание и четность числа таких вершин [4] означают, что таких изменений не может быть меньше $n_2/2$, когда для каждой пары вершин из V_2 добавляется или удаляется одно ребро. В работе показано, что всякий конечный граф можно представить в виде некоторого многодольного (полнодольного) графа с максимальными независимыми подмножествами вершин (максимальными полными подграфами). На основе многодольного представления предложена последовательная процедура, которая при помощи операций добавления новых ребер приводит за $n_2/2$ шагов к эйлерову графу, если в результате преобразований остается пустой граф. Сформулированы условия, которым должен удовлетворять подграф на вершинах с нечетными степенями, гарантирующие минимальность числа используемых операций в независимости от выбора многодольного представления. К этому классу, в частности, относятся неполные двудольные графы с нечетными долями и двудольные графы с четными долями.

[1] Поддержка ФЦП «Исследования и разработки по приоритетным направлениям развития научно-технологического комплекса России на 2007-2013 годы» (Госконтракт № 1.519.11.4012) и РФФИ, грант № 12-07-00415-а.

1. Многодольное представление графа

В этом разделе строится представление произвольного конечного графа $H = H(A,U)$, $|A| = p$, в котором множество всех вершин A образует разбиение на максимальные независимые подмножества, а ребра соединяют вершины из разных подмножеств.

Определение 1. Совокупность подграфов $H_i = H_i(A_i,U_i), i = \overline{1,k}$ графа $H = H(A,U)$ образуют его многодольное представление, если

$$A = \bigcup_{i=1}^{k} A_i,\ A_i \bigcap A_j = \varnothing,\ U = \bigcup_{i,j=1}^{k} U_{ij},\ U_i = \varnothing,\ i, j = \overline{1,k},\ i \neq j,\ 1 \le k \le p, \qquad (1)$$

A_i, A_j - максимальные независимые подмножества вершин (см. [4]).

Среди подмножеств вершин A_i можно выделить одноэлементные, для этого положим

$$k = k_1 + k_2, k_1, k_2 \ge 0, |A_i| \ge 2, i = \overline{1,k_1}, |A_i| = 1, i = \overline{k_1+1,k}, \bigcup_{i,j=k_1+1}^{k} U_{ij} = U_0, i \neq j. \qquad (2)$$

Подграф $H_0 = H_0(A_0,U_0)$, если имеется (т.е. $k_2 > 0$), является полным графом с $k_2 = k - k_1$ вершинами, т.к. в противном случае нашлось бы независимое подмножество, содержащее не менее двух несмежных вершин из A_0. Из максимальности подмножеств A_i также следует, что любые два подграфа H_i, H_j содержат не менее двух смежных вершин $x \in A_i, y \in A_j$. Таким образом, справедливо:

Утверждение 1. Всякий граф $H = H(A,U)$ является некоторым многодольным графом вида (1), (2), причем любые два подграфа H_i, H_j, $i, j = \overline{1,k}$, $i \neq j$ содержат хотя бы две смежные вершины, а подграф $H_0 = H_0(A_0,U_0)$ - полный.

Для получения многодольного представления можно использовать следующую процедуру порождения максимальных независимых подмножеств. На 1-ом шаге в A_1 включается любая пара несмежных вершин. Если такой нет, то все независимые подмножества одноэлементные, $k_1 = 0, k = k_2$, т.е. $H = K_p$ - полный граф. Если A_1 уже содержит t вершин, то на $t+1$ - ом шаге добавляется любая вершина, несмежная с каждой из включенных. Если такой новой вершины нет, то A_1 построено. Для построения следующего возможного максимального независимого подмножества рассматривается подграф графа H, не содержащий вершины из $A \setminus A_1$ и инцидентные им ребра. В нем повторяются операции, аналогичные используемым при получении подмножества A_1.

Очевидно, что такая схема образования независимых подмножеств A_i может приводить к неоднозначному результату и зависит от выбора очередной добавляемой вершины. Следует отметить, что многодольное представление графа дает некоторый способ его декомпозиции и может оказаться полезным при решении задач на основе распределенных вычислений. Переход к дополнительному графу позволяет получить представление графа через максимальные полные подграфы.

Определение 2. Совокупность подграфов $H_i = H_i(A_i,U_i),\ i = \overline{1,k}$ графа $H = H(A,U)$ образуют его полнодольное представление, если $A = \bigcup_{i=1}^{k} A_i$, $A_i \bigcap A_j = \varnothing$ и H_i - максимальные полные подграфы, $i, j = \overline{1,k},\ 1 \le k \le p$. (3)

Среди подмножеств вершин A_i также можно выделить одноэлементные, т.е.

$$k = k_1 + k_2,\ k_1, k_2 \geq 0,\ |A_i| \geq 2,\ i = \overline{1,k_1},\ |A_i| = 1,\ i = \overline{k_1+1,k},\ \bigcup_{i=k_1+1}^{k} A_i = A_0 . \quad (4)$$

Подграф $H_0 = H_0(A_0, U_0)$, если имеется (т.е. $k_2 > 0$), теперь является пустым графом с $k_2 = k - k_1$ вершинами, и из утверждения 1 получаем:

Следствие 1.1. Всякий граф $H = H(A,U)$ является некоторым полнодольным графом вида (3), (4), причем любые два подграфа H_i, H_j, $i, j = \overline{1,k}$, $i \neq j$ содержат хотя бы две несмежные вершины, а подграф $H_0 = H_0(A_0, U_0)$ ребер не имеет.

Используя полнодольное представление возможно построение процедуры преобразования в эйлеров граф на основе операции удаления ребер.

2. Метод единичного изменения степеней вершин

Будем называть граф с нечетными степенями вершин не*четным*. Нетрудно показать, что его степени можно изменить на единицу при помощи операций двух типов (добавления и удаления ребер). Для этого воспользуемся многодольным представлением (1), (2) нечетного графа $H = H(A,U)$ с четным числом вершин в виде многодольного графа (1), (2) с максимальными независимыми подмножествами вершин A_i, $i = \overline{1,k}$. Поскольку $|A| = p$ - четное, то число нечетных подмножеств A_i, $|A_i| = p_i$, включая единичные, если они имеются, также четное.

В каждом из подмножеств вершин A_i строим максимальное множество ребер без общих вершин. В подмножестве с четным числом вершин таким способом будут покрыты, т.е. на единицу увеличены степени $p_i/2$ вершин, а с нечетным $(p_i - 1)/2$ вершин. Поскольку число «нечетных» подмножеств было четным, подмножество непокрытых вершин A' также «четно». Переходим к новому графу H' с четным числом вершин A', являющемуся подграфом исходного графа H и не содержащим покрытые вершины с инцидентными им ребрами. Для него также имеется многодольное представление вида (1), (2) и описанная процедура может быть повторена.

В результате конечного числа шагов возможны два варианта: либо все вершины исходного графа будут покрыты новыми ребрами, либо оставшийся граф является полным с четным числом вершин. В первом случае используемая процедура увеличивает все степени вершин на единицу путем добавления $p/2$ ребер. Во втором случае ограничиться лишь операцией добавления ребер не удается, но в полном «четном» графе имеется паросочетание, содержащее половину ребер. Их удаление уменьшает степени вершин этого полного графа на единицу.

Предложенный способ перестроения исходного графа приводит к следующему:

Утверждение 2. Минимальное число операций добавления или удаления ребер для изменения всех степеней вершин графа $H = H(A,U)$ с четным числом вершин p на единицу равно $p/2$, причем, а) если в результате произведенной процедуры его перестроения и добавления ребер образовался полный граф с $2r$ вершинами, то число добавленных ребер равно $(p - 2r)/2$, а удаленных - r; б) в противном случае число добавленных ребер равно $p/2$, а удаленных - 0.

Заметим, что единичное изменение степеней вершин графа $H = H(A,U)$ с четным числом вершин за $p/2$ операций также реализуется исходя из его полнодольного представления (3), (4).

Утверждение 2 позволяет сформулировать условие для перестроения графа G в эйлеров за счет добавления новых ребер.

Утверждение 3. Исходный связный граф $G = G(V, E)$ на основе многодольного представления его подграфа на вершинах нечетной степени $G_2 = G_2(V_2, E_2)$ преобразуется в граф $G' = G'(V, E')$, содержащий эйлеров цикл, при помощи $n_2/2$ операций добавления ребер, если в результате последней конфигурацией оказался пустой граф.

3. Достаточные условия построения эйлеровой аппроксимации графа операцией добавления ребер

Если для преобразования графа в эйлеров используется только операция введения новых ребер, то вопрос о связности нового графа не возникает. Тогда при увеличении степеней всех вершин на единицу подграфа G_2 с «нечетными» вершинами аппроксимирующий граф содержит эйлеров цикл. Поэтому целесообразно отдельно описать конфигурации графов, когда такое перестроение возможно. В этом разделе формулируются некоторые условия на подграф G_2 и графы, получающиеся в процессе перестройки. Для этого снова рассматривается граф H с четным числом вершин и ищутся условия, когда добавление $p/2$ ребер приводит к единичному увеличению степеней всех вершин . При этом считаем, что в многодольном представлении вида (1), (2) этого графа (или графов, возникающих в процессе добавления ребер,) все подмножества $A_i, i = \overline{1,k}$ содержат нечетное число вершин, поскольку, как следует из утверждения 2 в «четных» подмножествах соединение p вершин $p/2$ ребрами увеличивает их степени на единицу. Понятно тогда, что число «нечетных» подмножеств, равное k, четно.

Утверждение 4. Если в многодольном представлении (1), (2) графа $H = H(A,U)$, где $|A| = p$ - четное, среди нечетных подмножеств A_i, $i = \overline{1,k}$ имеются $A_i, i = \overline{1,l}$, $l \le k - l$, в каждом из которых найдется нечетное число таких вершин x_s, что $(x_s, y_s) \notin U$, где $y_s \in A_s$, $s = \overline{l+1,k}$, тогда добавление $p/2$ ребер увеличивает все степени на единицу.

Доказательство. Поскольку все подмножества A_i, A_s содержат нечетное число вершин, а p -четное, то для получения требуемого в каждом A_i, $i - \overline{1,l}$ число вершин вида x_s должно быть нечетным .Тогда каждую вершину x_s, $s = \overline{l+1,k}$ можно соединить ребром с некоторой вершиной y_s, $s = \overline{l+1,k}$, а оставшиеся вершины (их четное число) в каждом подмножестве A_i, A_s также можно попарно соединить. Таким образом, в результате общее число добавленных ребер равно $p/2$.

Утверждение 4 дает возможность выделить некоторые частные случаи при определенных соотношениях между параметрами k, l, допускающие эффективную проверку соответствующих условий. В частности, если удается построить разбиение на 2 группы равномощных подмножеств, то его формулировка значительно упрощается.

Следствие 4.1. Если в многодольном представлении (1), (2) графа $H = H(A,U)$, где $|A| = p$ - четное, возможно разбиение нечетных подмножеств A_i, $i = \overline{1,k}$ на 2 группы:

$\{A_1, \ldots, A_{k/2}\}$, $\{A_{k/2+1}, \ldots, A_k\}$, что $(x_i, y_i) \notin U$, где $x_i \in A_i, i = \overline{1,k/2}$, $y_i \in A_i, i = \overline{k/2+1,k}$, тогда добавление $p/2$ ребер увеличивает все степени на единицу.

Формулировка следствия 4.1. иллюстрируется на рисунке 1, где пунктиром обозначены отсутствующие ребра, а имеющиеся ребра и остальные вершины не показаны.

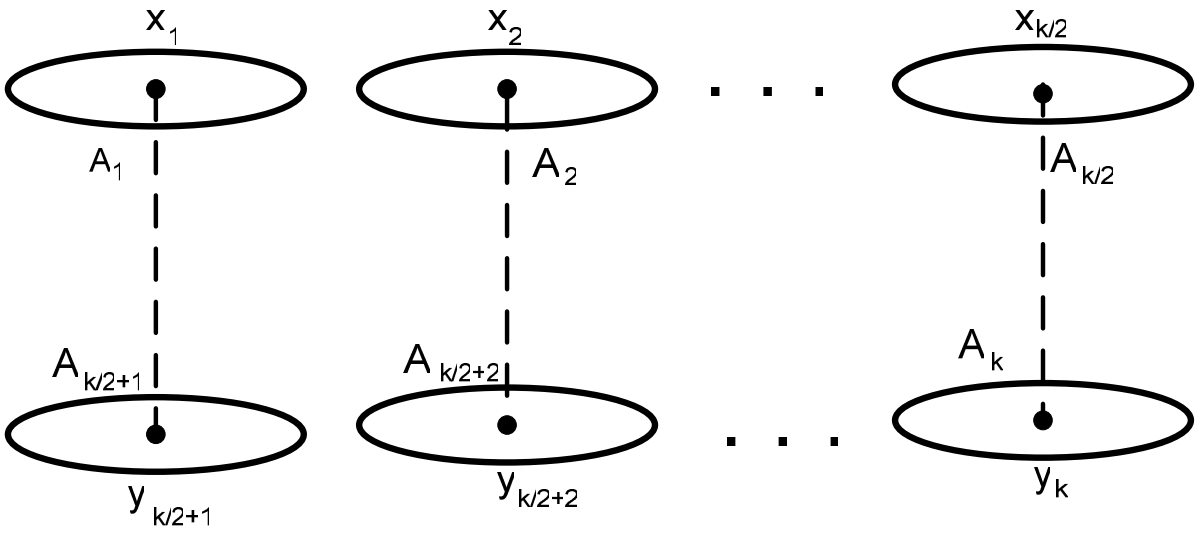


Рис.1

В случае, когда $l = 1$, приходим к следующему результату.

Следствие 4.2. Если в многодольном представлении (1), (2) графа $H = H(A,U)$, где $|A| = p$ - четное, среди нечетных подмножеств A_i, $i = \overline{1,k}$ имеется подмножество A_1, в котором найдется нечетное число таких вершин x_i, $i = \overline{2,k}$, что $(x_i, y_i) \notin U$, где $y_i \in A_i$, $i = \overline{2,k}$, тогда добавление $p/2$ ребер увеличивает все степени на единицу.

Утверждение 4 имеет простую интерпретацию, когда в многодольном представлении графа $H = H(A,U)$ имеется всего 2 подмножества.

Следствие 4.3. Если граф $H = H(A,U)$, где $|A| = p$ - четное, является двудольным с четными долями либо неполным двудольным с нечетными долями, тогда добавление $p/2$ ребер увеличивает все степени на единицу.

Для доказательства в условии следствия 4.1. достаточно положить $k = 2$.

Деревья входят в класс двудольных графов, поэтому справедливо:

Следствие 4.4. Если граф $H = H(A,U)$, где $|A| = p$ - четное, является лесом или деревом, но не звездой, тогда добавление $p/2$ ребер увеличивает все степени на единицу.

В графе H на рис 2а удается добиться единичного изменения степеней вершин лишь применением операции добавления ребер, поскольку он удовлетворяет всем следствиям. В его 3- дольном представлении 2 «нечетных» подмножества A_1 и A_2 (рис 2б). Подграф графа H на этих вершинах изображен на рис.3а. Для него, в частности, выполнено следствие 4.1: вершина 1 не смежна с вершинами 6 и 8 из A_2. Поэтому граф H преобразуется в граф H' добавлением ребер: {(3, 4), (5, 7), (1, 8), (2, 6)} (рис.3б).

Все приведенные здесь утверждения естественно применимы к подграфу $G_2 = G_2(V_2, E_2)$ $|V_2| = n_2$ с четным числом вершин исходного графа $G = G(V,E)$, для которого ищется эйлерова аппроксимация. Поэтому они гарантируют его преобразование в эйлеров граф $G' = G'(V,E')$ при помощи минимального числа $n_2/2$ операций добавления новых ребер.

Утверждение 5. Если для подграфа G_2 графа $G = G(V_1, V_2, E)$ из n_2 вершин нечетной степени выполняется любое из следствий 4.1- 4.4 , тогда добавление в нем $n_2/2$ ребер преобразует граф G в эйлеров.

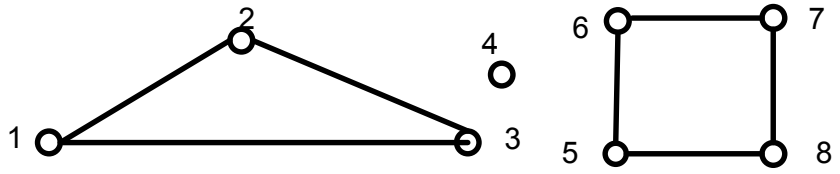


Рис.2а

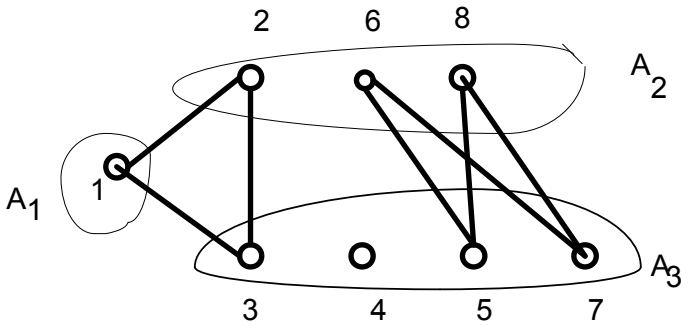


Рис.2б

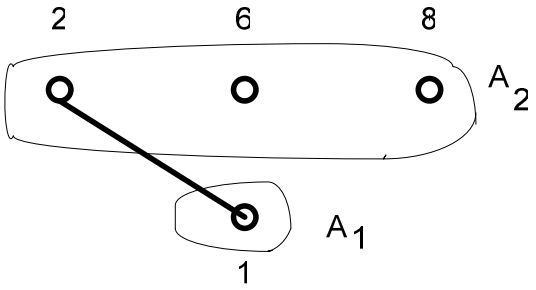


Рис.3а

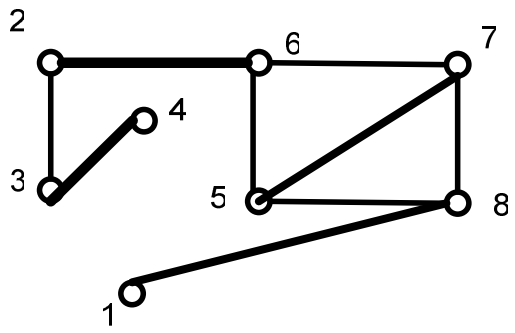


Рис.3б

В целом, полученные в работе результаты позволяют переходить от произвольного графа к эйлерову за счет добавления минимального количества новых ребер, что позволяет осуществлять эффективный мониторинг соответствующих коммуникационных систем.

Литература

[1] Фляйшнер Г. Эйлеровы графы и смежные вопросы. М.: Мир, 2002, 335 с.

[2] Fleischner H. Eulerian graphs and related topics. , part 1, v. 2, Amsterdam: Elsevier Science publishers B.V., 1991, 337 p.

[3] Кристофидес Н. Теория графов (алгоритмический подход). М.: Мир, 1978, 432 с.

[4] Оре О. Теория графов. 2-е изд. М.: Наука, 1980, 336 с.

[5] Раппопорт А.М. Измерение расстояний между взвешенными графами структуризованных экспертных суждений. «Многокритериальный выбор при решении слабоструктуризованных проблем». Сб. трудов. Вып.5, М.: ВНИИСИ, 1978, с.с. 97-108.

ПАРАЛЛЕЛЬНЫЕ ЗАДАНИЯ В ГРИД-СРЕДЕ

М.М. Степанова, О.Л. Стесик

Санкт-Петербургский Государственный Университет,
физический факультет, кафедра вычислительной физики
mstep@mms.nw.ru, stes@mms.nw.ru

На сегодняшний день любая грид-инфраструктура позволяет проводить параллельные расчеты, однако, в каждой из них есть свои ограничения по запуску параллельных заданий по сравнению с обычным кластером. Достаточно трудно обеспечить максимально простой и универсальный механизм, который был бы эффективен для оптимального резервирования ресурсов и обработки разных типов задач в гетерогенной среде.

При работе с очередью на кластере можно точно указать параметры ресурсов и запуска задачи. Грид позволяет запускать параллельные задачи, но, как правило, накладывает значительные ограничения на такую параметризацию. Процедура выполнения заданий в грид включает следующие стадии:

· определение характеристик задания в описании;
· поиск и выбор подходящего ресурса;
· резервирование ресурса для задания;
· выполнение на конкретном ресурсе.

Можно выделить следующие основные особенности в реализациях грид-систем, которыми определяются присущие всем гридам ограничения по сравнению с кластером:

· возможности формата описания заданий;
· полнота публикуемой информации и гибкость алгоритма поиска ресурса;
· различие в типах и конфигурации LRMS на сайте;
· реализация адаптера GRAM-LRMS;
· специфичность требований к установке окружения и запуску для разных типов задач.

Одним из ключевых моментов является правильное выделение и резервирование на сайте вычислительных ресурсов (в первую очередь, подмножества cpu) в соответствие с описанием требований задачи.

В данной работе представлены результаты сравнения и тестирования возможностей запуска характерных типов параллельных заданий, предоставляемые рядом популярных грид-проектов. Все тестируемые шлюзы (за исключением gLite) имеют полностью идентичную конфигурацию вычислительного ресурса. Все работают с одним и тем же однородным кластером, состоящим из многоядерных вычислительных узлов под управлением Torque 3.0.2. Важный момент – включение при сборке Torque механизма cpuset [1], который резервирует ресурсы (подмножество процессоров и памяти узла) под монопольное использование задачей на время выполнения. Кластер имеет две реализации MPI (MPICH2 и OpenMPI), требующие разного окружения для компиляции и запуска приложений.

Полигон. Конфигурация полигона, на котором выполнялась работа, включает сервера грид-шлюзов и кластер из четырех 8-ядерных smp-узлов с выделенным torque-сервером. Всем узлам доступен общий NFS-раздел.

· Шлюз GTK 5.2.1, установка из Globus-репозитория
· Шлюз ARC11.05, установка из NorduGrid-репозитория
· Шлюз UNICORE 6.3.1, установка из дистрибутива UNICORE
· Шлюз ГридННС 1.0, установка из ГридННС-репозитория

Для тестирования gLite выполнена установка сайта из EMI2-репозитория

· CreamCE + MPI_CE + TORQUE_server (v.2.5.7) + TORQUE_utils
· 2 узла WN + MPI_WN+TORQUE-client +набор ПО для параллельной среды.

Тестовые параллельные задачи

Описание всех параллельных заданий, которые использовались для тестирования, приводятся в Табл.1. Второй столбец содержит директивы резервирования ресурсов на кластере и команду запуска задачи. Задача, оформленная пользователем и прошедшая все стадии грид, для корректного выполнения должна в итоге запускаться на конечном ресурсе именно таким образом. Если язык описания заданий или промежуточные звенья не позволяют точно задать и обеспечить эти условия запуска, то на многих сложных задачах возможно очень сильное снижение производительности.

В Табл.2 представлены результаты прохождения тестовых задач для всех систем, участвовавших в тестировании - Globus, ARC, gLite(CremCE), ГридННС и UNICORE. Результаты, помеченные (*), следует понимать как условно-успешные, т.е. для корректного запуска задач соответствующего типа сайт нуждается в дополнительной специальной настройке, или пользователю должна быть известны некоторые особенности конфигурации, которые в общем случае не публикуются в информационной системе.

Таблица 1. Тестовые задачи с параметрами запуска на кластере

Описание	Резервирование ресурсов (cpu) на PBS и параметры запуска
MPI	
M1. Dense mode: 1 mpi-процесс на одно ядро с произвольным распределением по узлам	#PBS - l nodes=N:ppn=K, где N*K=M; $MPIEXEC -n M ./mpi_task
M2. Sparse mode: 1 mpi-процесс на один узел с полным резервированием выделенных узлов	#PBS - l nodes=N:ppn=K, где K=SMPSize; $MPIEXEC -pernode ./mpi_task
M3. Fixed mode: M mpi-процессов на каждый из N узлов	#PBS - l nodes=N:ppn=K, где K >= M; $MPIEXEC -npernode M ./mpi_task
OpenMP	
O1. Резервирование под задачу одного узла полностью	#PBS -l nodes=1:ppn=K, где K=SMPSize; ./omp_task
O2. Резервирование под задачу подмножества из K ядер одного узла	#PBS -l nodes=1:ppn=K, где K<SMPSize; ./omp_task
Hybrid MPI+OpenMP: простой вариант гибридной задачи, где каждый mpi-процесс является многопоточным	
H1. Запуск аналогичен M2	
H2. Запуск аналогичен M3, но с дополнительным требованием	#PBS -l nodes=N:ppn=K, export OMP_NUM_THREADS=L, где M*L=K; $MPIEXEC -npernode M ./hyb_task

Таблица 2. Результаты прохождения тестовых задач

	MPI-M1	MPI-M2	MPI-M3	OMP-O1	OMP-O2	HYB-H1	HYB-H2
GTK5.2.1	Да	Нет	Нет	Да*	Да*	Нет	Нет
ARC	Да*	Да*	Да*	Да	Да*	Да*	Да*
UNICORE	Да	Да	Да	Да	Да	Да	Да*
ГридННС	Да	Нет	Нет	Да*	Да*	Нет	Нет
gLite	Да	Да	Да*	Да	Да*	Да*	Да*

*/ Требуется специальная настройка сайта и/или знание конфигурации пользователем.

Особенности реализации грид-систем

Globus. ПО GlobusToolKit берется за основу многих грид-проектов, поэтому для полноты мы рассматриваем и тестируем его последнюю версию (5.2.1).

В описании задания на языке RSL [1] за резервирование ресурсов (cpu) отвечает два атрибута - count и host_count. Параметры запуска определяются типом задачи job_type и параметром count. Поддерживаются три типа job_type:

- single - для любого count>0 стартует 1 процесс;
- multiple - запускается count независимых экземпляров процессов;
- mpi - для старта используется определенный в конфигурации метод ($MPIEXEC -n count) , по умолчанию по одному процессу на count нодах (# PBS -l nodes=count). Следует отметить, что для эффективного использования smp-узлов в globus-pbs.conf должно быть установлено значение cpu_per_node и #cluster="1". В этом случае можно задавать host_count для резервирования ресурса и count для указания числа mpi-процессов.

Программное окружение можно явно задать атрибутами environment и library_path, что предполагает знание пользователем конфигурации сайта. Интегрированного механизма установки окружения нет - потенциально возможна установка ПО SoftEnv, однако не реализован соответствующий атрибут.

Компиляция последовательного или OpenMP кода может быть выполнена непосредственно перед запуском, для компиляции MPI-программы требуется запуск отдельного задания.

Средства описания задачи не предоставляют возможность передать дополнительный аргумент $MPIEXEC, что приводит к проблеме запуска MPI и гибридных задач даже с простыми требованиями к геометрии распределения ресурсов.

Перечисленные особенности существенно сужают круг задач, которые могут быть успешно выполнены, до M1,O1,O2 из Табл.1.

ГридННС. В данном проекте реализация грид-шлюза базируется на GlobusToolKit, и, как следствие, наследует ряд его недостатков. Так же следует отметить, что в ГридННС конечный пользователь может отправить задание на сайт только через брокер (Pilot). Это с одной стороны, позволило реализовать хорошую поддержку групповых задач, а с другой накладывает ограничения на возможные типы заданий.

Для описания заданий используется JSON[3]. В настоящее время резервирование ресурсов выполняется на основе единственного параметра count, но также поддерживается недокументированное расширение "extensions":{"nodes": "N:ppn=K"}. Параметры запуска определяются типом задачи (single или mpi), который автоматически присваивается брокером Pilot по значению count (если count>1 то mpi, если count=1 или не задан, то single); job_type не может быть явно указан пользователем в описании задачи.

Для установки окружения используется модифицированный вариант SoftEnv. Это набор сценариев, которые выполняются на узлах кластера перед запуском задания, в случае, если это указано в секции requirements в описании задания.

Компиляция программы описывается в виде отдельной задачи в составе группы, ее выполнение порождает запуск второй (основной) задачи.

ГридННС, также как и Globus, не имеет штатных способов передачи аргументов для $MPIEXEC. Аналогичны и результаты тестирования (см. Табл.2), причем прохождение тестов O1 и O2 достигается только с использованием "extensions": { "nodes": "1:ppn=K"}.

UNICORE. Описание заданий выполняется на JSDL 1.0 [4] с расширениями SPMD extension [5] и UNICORE-specific extension [6]. Это наиболее полный и гибкий вариант спецификации из всех представленных в данной работе грид-проектов.

Для резервирование ресурсов (cpu) предназначено три атрибута: CPUs, CPUsPerNode, Nodes. Параметры запуска задачи задаются Execution environment[6], никакие дополнительные скрипты не нужны. С версии UNICORE 6.4 добавились SPMD-расширения NumberOfProcesses,

ProcessesPerHost и ThreadsPerProcess, которые допускают еще более гибкую параметризацию запуска.

Необходимые программные окружения (Execution environment) создаются администратором сайта в Incarnation Database (IDB) и интерпретируются при запросе сервисом XNJS. Окружения доступны клиенту через свойства ресурса. Кроме того, пользователь может добавлять параметры окружения атрибутом Environment и через pre/post команды.

Для компиляции предназначен атрибут User precommand, в котором можно задать все необходимые команды.

Среда UNICORE имеет наиболее продуманную и оптимальную спецификацию описания заданий и ее полную поддержку на шлюзе. Характерная особенность – четкое разделение функциональности, а именно, параметры в Resources полностью определяют резервирование, а через ExecutionEnv можно очень гибко настроить параметры и опции запуска задачи. Это позволяет обеспечить поддержку корректного запуска практически любых параллельных задач. На всех тестах получен положительный результат (см. Табл.2.).

gLite. В качестве языка описания заданий используется JDL[7]. На грид-шлюзе (CreamCE), начиная с версии glite-ce-cream v.1.13, для резервирования ресурсов введен новый расширенный набор атрибутов: CPUNumber (полное число запрашиваемых cpu), SMPGranularity (минимальное число ядер на узел), HostNumber (полное число запрашиваемых узлов), WholeNodes (полное резервирование узла, т.е все ядра). Отметим, что в gLite-инфраструктуре CreamCE и брокер WMS поддерживают разные атрибуты и разные типы заданий. Простой тип (Normal) поддерживают оба сервиса, составные типы (Parametric, DAG, Collection) только WMS.

Для запуска параллельных заданий предназначен набор скриптов MPI-Start [8] - пользователь должен передать скрипт с установленными параметрами и вызовом MPI-Start.

Компиляцию программы можно выполнить из пользовательского скрипта перед запуском или используя отдельный скрипт, заданный атрибутом Prologue.

Атрибутом Requirements в описании задания устанавливается требование на поддержку на сайте нужной среды окружения; атрибуты Environment, Prologue позволяют задать pre/post сценарии.

Итерфейс MPI-Start в сочетании с новым набором атрибутов, по сути, дает унифицированную связку с любым LRMS. Теоретически можно запустить все, но требуется очень аккуратная настройка как сайта, так и параметров запуска пользователем. На нашем полигоне все тесты прошли успешно (см. Табл.2) .Однако, тестирование в рамках VO dteam более десятка сайтов инфраструктуры EGI, имеющих MPI-окружение, показало, что большинство из них поддерживает запуск и прохождение тестовых заданий, но ни один не обеспечивает: 1) адекватное выделение и резервирование ресурсов; 2) правильный запуск заданий, которые сложнее, чем M1 и O1.

ARC. В проекте NorduGrid ARC описание заданий выполняется на языке xRSL [9]. Для резервирования cpu в описании имеется единственный атрибут count, который в зависимости от настройки сайта может интерпретироваться как число узлов или число ядер. Более гибкий вариант резервирования на сайте может быть доступен путем настройки runTimeEnvironment (RTE). RunTimeEnvironment – это механизм установки среды на основе shell-скриптов. Три вызова (до создания PBS submit-скрипта, перед запуском задачи и после завершения) позволяют очень гибко настроить исполняющую среду для любых приложений. Однако, надо отметить, что использование этого механизма для резервирования ресурсов - не самый удобный вариант, поскольку требует либо унифицированной поддержки в рамках VO, либо знания пользователем деталей конфигурации конкретных сайтов.

Атрибуты типа заданий отсутствуют - пользователь использует свой скрипт запуска и подходящий RTE. Соответственно, и параметры запуска задаются явно в скрипте пользователя или определяются окружением RTE. Компиляция также выполняется из пользовательского скрипта перед запуском.

Созданием необходимых RTE несложно обеспечить среду для любых задач и обеспечить возможность прохождения всех тестов. Но, как отмечено выше, такой механизм нельзя считать универсальным, поэтому почти все результаты в Табл.2 помечены как условные (*). Исключением является тест O1, который не требует специальной настройки RTE.

Сравнение реализаций. В качестве резюме этого раздела приведем итоговую таблицу с характеристиками, определяющими возможности каждой грид-инфраструктуры по запуску параллельных заданий.

Таблица 3. Реализация систем приема и обработки заданий.

	Язык описания заданий	Резервирование cpu	Способ и параметры запуска	Окружение	Pre/Post сценарии
GTK5.2.1	RSL	count, host_count	job_type	enviroment	-
ARC	xRSL	count, RTE	user_script, RTE	RTE Environment	user_script
UNICORE	JSDL1.0, SPMD ext, UNICORE -specific ext	CPUs, Nodes, CPUsPerNode	ExecutionEnv, NumberOfProcesses*, ProcessesPerHost*, ThreadsPerProcess*	ExecutionEnv SPMDVariation	User precommand, User postcommand
ГридННС	JSON	count extension: nodes*	job_type (auto)	SoftEnv	user_script (для single- задач)
gLite	JDL	CPUNumber, HostNumber*, SMPGranularity*, WholeNodes*, CERequirements*	user_script, MPI_Start	Requirements Environment	user_script, Prologue, Epilogue

*/ Поддерживается не всеми версиями ПО или требуется дополнительная настройка.

Заключение

В результате проведенного анализа и тестирования реализаций современных грид-систем с точки зрения возможностей их использования для широкого круга параллельных задач можно сделать следующие выводы:

· Все системы поддерживают запуск MPI-задач в режиме один процесс на ядро без строгих требований к распределением ядер по узлам (задачи типа M1).

· Общей проблемой является передача параметров для запуска MPIEXEC. С одной стороны, набор должен быть согласован с запрошенными ресурсами, с другой – достаточно гибким для правильного запуска. Реализация с фиксированным набором типов заданий, где параметры запуска автоматически рассчитываются из требований к ресурсам, сильно ограничивает круг задач(Globus, ГридННС).

· Для корректной работы произвольных OpenMP и гибридных задач, желательно резервирование узлов целиком под один многопоточный процесс.

· На крупных smp-узлах и при особых требованиях к геометрии запуска (например, задачи типа O2,M2 и сложнее) целесообразно использовать механизм типа cpuset.

Наиболее эффективным и законченным решением на сегодняшний день показала себя реализация UNICORE. Другие проекты без дополнительной конфигурации пока обеспечивают лишь базовую функциональность для простейших MPI- и OpenMP-задач.

Литература

[1] Linux cpuset. http://www.clusterresources.com/torquedocs21/3.5linuxcpusets.shtml
[2] RSL. http://www.globus.org/toolkit/docs/latest-stable/gram5/developer/gram5DeveloperGuide.pdf
[3] Формат описания заданий и задач для ГридННС. http://www.ngrid.ru/ngrid/support/user/job_description
[4] JSDL Specification, v.1.0. http://www.ogf.org/documents/GFD.56.pdf
[5] JSDL SPMD Application Extension. http://www.ogf.org/documents/GFD.115.pdf
[6] Unicorex. http://www.unicore.eu/documentation/manuals/unicore6/files/unicorex/unicorex-manual.pdf
[7] JDL. https://edms.cern.ch/file/590869/1/WMS-JDL.pdf
[8] MPI-Start. http://grid.ifca.es/wiki/Middleware/MpiStart/
[9] xSRL. http://www.nordugrid.org/documents/xrsl.pdf

ИСПОЛЬЗОВАНИЕ ТЕХНОЛОГИЙ РАСПРЕДЕЛЕННОЙ ВИЗУАЛИЗАЦИИ НА ПРИМЕРЕ МВК НИЦ «КУРЧАТОВСКИЙ ИНСТИТУТ»

И.А. Ткаченко

Национальный Исследовательский Центр «Курчатовский институт»
Россия, 123182, Москва, пл. Академика Курчатова, д. 1.
tia@grid.kiae.ru

Статья посвящена системам распределенной визуализации на примере настройки визуализационной части МВК НИЦ "Курчатоский институт". Рассматриваются вопросы выбора подходящей распределенной файловой системы, настройка и режимы работы систем распределенного рендеринга, настройка и использование систем распределенной визуализации для вывода изображения на видеостену.

Файловые системы для распределенной визуализации и рендеринга

Любая распределенная система, работающая с центральным файловым хранилищем, будет загружать его тем больше, чем больше узлов в ней задействовано. В случае систем рендеринга и визуализации это становится очень заметно из-за того, что работа ведется с файлами большого объема.

Распределенная система рендеринга ParaView [1] решает эту задачу следующим образом: один из процессов назначается главным и распределяет между собой и остальными обрабатываемые данные. В результате на каждом узле образуется та часть данных, с которой он работает и обращения к центральному файловому хранилищу минимальны: требуется скачать данные на главный узел, который распределит их между остальными.

В случае распределенной визуализации все намного сложнее. Всегда есть какой-то узел, который подготавливает визуальный ряд для отображения и есть набор узлов на которые он его транслирует. Тут уже нет смысла как-то заранее подготавливать изображение: подготовкой видео ряда (воспроизведением видео файла или отображением изображений) занимается только один узел. В этом случае выбор файловой системы для хранилища должен определятся отображаемыми данными. Например, pvfs2 [2], являясь распределенной файловой системой, хорошо подходит для работы с большими файлами, что делает возможным быстро создать локальную копию отображаемых данных и в дальнейшем сетевую файловую систему не задействовать, а при работе с маленькими файлами производительность сильно проседает, доставляя большие неудобства при работе с большим числом обычных изображений.

В МВК НИЦ «Курчатовский институт» мы использовали две сетевых файловых системы: pvfs2 для работы с файлами большого объема и обычный nfs для маленьких файлов.

Рендеринг с использованием GPU

ParaView – свободное, мультиплатформенное приложение, разработанное для визуализации наборов данных больших и малых размеров. Основу ParaView составляет система VTK [3], которая содержит библиотеку C++ классов и несколько интерфейсов для языков высокого уровня, поддерживает широкий набор алгоритмов визуализации, включая скалярные, векторные, тензорные, текстурные и др. методы.

ParaView работает как на персональных рабочих станциях, так и на многопроцессорных суперкомпьютерах с распределенной памятью или кластерах. Система ParaView включает удобный интерфейс, написанный с использованием библиотеки Qt, реализацию распределенных вычислений и параллельный сервер для визуализации. Средства распределенной обработки, реализованные в ParaView , позволяют выполнять преобразования

больших объемов числовых данных в параллельном режиме и собирать затем результаты на одном компьютере.

Работа ParaView была успешно протестирована в Windows, Linux, MacOS и на различных рабочих станциях Unix, а также кластерах и суперкомпьютерах. Вследствие того, что большинство функций ParaView реализованы в виде библиотек, возможна замена существующего пользовательского интерфейса новым клиентским приложением. Также, ParaView поддерживает pvpython приложение, позволяющее осуществлять визуализацию и пост-обработку данных посредством написания скриптов на языке Python.

Библиотека компонентов пользовательского интерфейса доступна каждому приложению ParaView. Библиотека сервера ParaView реализует слой абстракции, необходимый для осуществления параллельной, интерактивной визуализации. Она делает параллельную работу прозрачной для клиентского приложения.

Архитектура системы визуализации ParaView

Система ParaView имеет трехзвенную клиент-серверную архитектуру и имеет три логических уровня:

Сервер данных – блок, отвечающий за чтение, фильтрацию и запись данных. Все информационные объекты, видимые в браузере, содержатся в сервере данных. Сервер данных может быть параллельным.

Сервер обработки – отвечает за рендеринг. Сервер обработки может быть также параллельным, в таком случае осуществляется параллельный рендеринг.

Клиент – отвечает за осуществление визуализации. Клиент контролирует создание объектов, исполнение и уничтожение их на серверах, но не содержит данных. Клиент также содержит графический пользовательский интерфейс. Клиент - всегда последовательное приложение.

Эти три логических уровня не всегда разделены физически. Логические блоки часто встроены в одно приложение, таким образом, отпадает необходимость в создании средств взаимодействия между ними. Существует три модели, в которых может функционировать ParaView .

Первая модель – автономная. Клиент, сервер данных и сервер обработки встроены в одно приложение. При работе с приложением ParaView идет автоматическое подключение к встроенному серверу, таким образом, возможно использование всех функций ParaView.

Следующая модель – клиент-сервер. В данной модели выполняется программа pvserver на параллельной машине, к которой пользователь подсоединяется посредством клиентского ParaView приложения. Программа pvserver содержит вместе серверы данных и рендеринга, таким образом, обработка данных и рендеринг осуществляются на ней. Соединение клиента и сервера осуществляется с использованием сокетов, относительно медленного механизма взаимодействия, в следствии чего передача данных через сокеты должна быть минимизирована.

Третий вариант функционирования - client - render server - data модель. В данном случае все три уровня работают как отдельные программы. Сервер рендеринга и сервер данных соединяются посредством множества сокетов, один для каждого процесса на сервере рендеринга.

Работа параллельной версии ParaView обычно включает в себя следующие этапы:

- подключение к удаленным компьютерам,
- выделение параллельных узлов,
- запуск параллельной программы,
- создание соединений и проход через брандмауэры.

Настройка вычислительных узлов для работы с системами рендеринга

Для рендеринга изображения ParaView требуется наличие дисплея, с поддержкой OpenGL и высокой производительностью. Стандартная Linux конфигурация инициализирует

дисплей, только если к видеокарте подключен монитор. Такая модель совершенно неприемлема для вычислительных GPU кластеров: никто не будет подключать по несколько мониторов к каждому серверу только ради того, что бы Linux проинициализировал дисплей. Так же графические карты nVidia, ориентированные на высокопроизводительные вычисления, не имеют видеовыходов.

Для решения это проблемы было применен следующий способ: X серверу сообщалось, что на каждой карте нужно инициализировать дисплей, не смотря на то, что к видео выходу ничего не подключено. Делается это указанием параметра **`Option "UseDisplayDevice" "none"`** в конфигурационном файле **`/etc/X11/xorg.conf`** в разделе **`Section "Screen"`**.

Распределенные системы визуализации

Современные компьютерные технологии позволяют делать распределенными и удаленными не только вычисления, но и вывод визуальных материалов: источником изображения может быть как результат вычислений, проводимых на кластере, так и заранее подготовленный визуальный ряд. Причем, компьютер, на котором визуальный ряд создается, совершенно не обязан находиться в непосредственной близости с устройством, которое этот видеоряд показывает: достаточно иметь хороший и надежный канал для передачи.

Такой подход можно более подробно рассмотреть на примере системы визуализации, развернутой в НИЦ «Курчатовский институт».

Распределенная система визуализации SAGE

Устройствами вывода изображения являются 12 тонкорамочных мониторов, составленных в стену 4x3 и подключенные к 3м компьютерам, составляющих визуализационный кластер. В единое визуализационное пространство они объедены с помощью SAGE [4], программного продукта ориентированного на построение распределенных систем визуализации. Она имеет управляющий модуль, который сообщает клиентам, желающим вывести изображение на видеостену, какую часть видео потока нужно направлять на конкретные узлы визуализации, в зависимости от расположения изображения на стене. На каждом визуализационном узле запускается вторая часть SAGE, которая принимает видеопоток и выводит его в часть экрана, указанную управляющим модулем. Такая организация рабочего процесса позволяет одновременно исключить наличие узких мест при передаче данных: данные передаются только туда, где они отображаются, и позволяет получать данные из любого удаленного источника, имеющего соответствующий канал связи.

Однако, есть и свои недостатки: в передаче участвует сжатый пиксельный поток. Это означает, что клиентское приложение должно быть модифицировано таким образом, что бы, при передаче данных X серверу, передавать их на SAGE. Этого можно добиться тремя способами: передачей рабочего стола целиком, модификацией приложения, подменой библиотек.

Передача рабочего стола целиком

Этот способ является самым простым, наиболее универсальным и наименее затратным, с точки зрения передачи данных. На клиентской машине запускается VNC сервер или x11vnc [5] сервер, на котором осуществляются необходимые операции по отображению произвольных данных. На кластере визуализации запускается модифицированный VNC клиент, который передает данные на SAGE, по высокоскоростному каналу, обслуживающему кластер.

Этот метод является кроссплатформенным: VNC сервер может быть запущен как на Windows, так и на Linux или iOs.

Он так же является наиболее простым, с точки зрения реализации, но работа с приложениями может оказаться не тривиальной из-за ограничений VNC сервера. В частности, при использовании обычного VNC сервера под Linux возникают трудности с использованием

OpenGL. Эти трудности позволяет преодолеть программа VirtualGL, переносящая обработку 3D графики на другой Linux дисплей.

Модификация приложения

Если доступен исходный код приложения, то можно передавать изображение непосредственно на SAGE из буфера вывода, воспользовавшись SAGE библиотекой sail и ее функциями. В оригинальную программу должны быть добавлены следующие действия, для работы с SAGE:

- **установка соединения с головным узлом SAGE-кластера**.
  На этапе инициализации приложения оно должно установить соединение с головным узлом SAGE-кластера и с каждым узлом, отвечающим за вывод изображения, передать некоторую информацию о себе на головной узел.
  Инициализация выполняется следующим образом:

```
sailConfig scfg;
sail sageInf;
…
scfg.init("some_sage-config_for_app");
scfg.setAppName("some_app_name");
scfg.resX = sage_w;
scfg.resY = sage_h;
scfg.winWidth = sage_w;
scfg.winHeight = sage_h;
scfg.pixFmt = sage_pixel_fmt;
scfg.rowOrd = sage_row_order;
sageInf.init(scfg);
```

  Здесь задается используемый конфигурационный файл, название приложения, размеры окна приложения, используемый формат пикселя и порядок строк в передаваемом изображении.
  Основная информация о конфигурации кластера содержится в конфигурационном файле. Существенно важными являются следующие переменные, определяющие будущее поведение приложения, при взаимодействии с SAGE-кластером:
  - `fsIP` – определяет адрес головного SAGE узла,
  - `fsPort` – определяет порт головного SAGE узла,
  - `pixelBlockSize` – размер передаваемых блоков,
  - `nwProtocol` – используемый сетевой протокол,
  - `asyncUpdate` – разрешено или нет асинхронное обновление дисплеев для данного приложения.
- **передача изображения**
  Каждое обновление изображения, перед выводом его на экран, должно инициировать обновления изображения на видеостене, т.е. должен пересылаться буфер изображения на узлы визуализации:

```
unsigned char *sage_buf =
      (unsigned char *)sageInf.getBuffer();
struct sage_image *im_sage = &sshow->cur->im_sage;
memcpy((void *)sage_buf,
      (void *)im_sage->data,im_sage->byte_size
);
sageInf.swapBuffer();
```

- **закрытие соединения с SAGE-кластером**

При завершении работы, приложение должно сообщить SAGE, что больше от него данных не поступит и его окно на видеостене можно уничтожить. Делается это вызовом функции `sageInf.shutdown();`

Подмена библиотек

Бывают ситуации, когда исходный код приложения недоступен или слишком сложен, но при этом оно использует некоторые стандартные библиотеки для вывода изображения на экран. В этом случае, подменив их на модифицированные соответствующим образом аналоги, можно так же добиться вывода изображения на видеостену.

```
#!/bin/sh
export SAGE_GL_WIDTH=800
export SAGE_GL_HEIGHT=600
export LD_PRELOAD=$SAGE_DIRECTORY/bin/libmyGL.so
export SAGE_GL=/usr/lib64/libGL.so.1
$*
export -n LD_PRELOAD
```

Данный скрипт использует libgl, уже модифицированный для работы с SAGE, что позволяет отображать приложение на видеостене.

Минусом данного метода является необходимость поддержания актуальной версии модифицированной библиотеки, что может стать затруднительным, если в оригинальную будут внесены существенные изменения.

XDMX

Кроме вывода изображения на видеостену с использованием SAGE существует более простой, но при этом менее гибкий способ: объединение всех доступных дисплеев в один виртуальный, при помощи Linux приложения `xdmx` [6]. К преимуществам данного способа относятся простота установки и настройки (необходимые пакеты включены практически во все распространенные дистрибутивы, а настройка – задание положения каждого дисплея на общей видеостене). Недостатком же является существенное ограничение возможностей. `Xdmx` создает виртуальный рабочий стол на одном из компьютеров, как правило, для этого нужна выделенная машина, управляющая видеостеной. За счет того, что рабочий стол виртуальный он не поддерживает некоторые расширения реальных, например, `RANDR`.

Настройка головного узла XDMX

Настройка головного узла заключается в создании конфигурационного файла, с описанием какой дисплей узла в каком месте видеостены находится:

```
virtual kiae {
  display "172.17.24.1:0.3"  @0x0;
  display "172.17.24.1:0.4"  @1920x0;
  display "172.17.24.1:0.1"  @3840x0;
  display "172.17.24.1:0.2"  @5760x0;
  display "172.17.24.2:0.3"  @0x1080;
  display "172.17.24.2:0.4"  @1920x1080;
  display "172.17.24.2:0.1"  @3840x1080;
  display "172.17.24.2:0.2"  @5760x1080;
  display "172.17.24.3:0.3"  @0x2160;
  display "172.17.24.3:0.4"  @1920x2160;
  display "172.17.24.3:0.1"  @3840x2160;
  display "172.17.24.3:0.2"  @5760x2160;
}
```

и запуска xdmx сервера

```
#!/bin/sh
startx -- \
  /usr/bin/Xdmx :5              \
  +extension GLX                \
  +extension RANDR              \
  +xinerama                     \
  +extension XINERAMA           \
  -input 127.0.0.1:2.0,console  \
  -fontpath tcp/172.17.24.1:7100 \
  -fontpath tcp/172.17.24.2:7100 \
  -fontpath tcp/172.17.24.3:7100 \
  -configfile  xdmx.conf        \
  -config kiae
```

Настройка узлов видеостены

Настройка узлов видеостены для использования `xdmx` состоит в разрешении подключения к соответствующим дисплеям с управляющего узла:

1. Добавление адреса управляющего узла в `/etc/X0.hosts`
2. Разрешение подключений к X серверу извне, добавлением
   ```
   [security]
   DisallowTCP=false
   ```
   в `/etc/gdm/custom.conf`

Заключение

Использование кластерного подхода к обработке графических данных, в сочетании с правильным выбором файловой системы, существенно повышает удобство использования системы в целом, а кластерный подход к визуализации данных позволяет получить изображение в сверхвысоких разрешениях.

Описанные выше методы позволили создать в МВК НИЦ «Курчатовский институт» высокопроизводительную систему для обработки и визуализации данных, которая может использоваться для научной и образовательной работы.

Литература

[1] ParaView - Open Source Scientific Visualization. Web site http://www.paraview.org/
[2] Parallel Virtual File System, Version 2. Web site http://www.pvfs.org/
[3] The Visualization Toolkit Web site http://www.vtk.org/
[4] Scalable Adaptive Graphics Environment. Web site http://www.sagecommons.org/
[5] x11vnc - VNC server for real X displays. Web site http://www.karlrunge.com/x11vnc/
[6] xdmx - Distributed Multihead X. Web site http://dmx.sourceforge.net/

ПРИМЕНЕНИЕ ТЕХНОЛОГИИ ВИРТУАЛИЗАЦИИ ДЛЯ ИЗУЧЕНИЯ ПРИНЦИПОВ ФУНКЦИОНИРОВАНИЯ КОМБИНИРОВАННЫХ ВЫЧИСЛИТЕЛЬНЫХ ИНФРАСТРУКТУР

Н.П. Храпов
Центр грид-технологий и распределённых вычислений ИСА РАН, Москва, Россия
nkhrapov@gmail.com

В настоящее время для изучения и практического освоения новых программных продуктов широко используются технологии виртуализации. На основе технологий виртуализации в рамках одной вычислительной машины полностью моделируется (виртуализуется) работа нескольких других вычислительных машин. Данная статья посвящена применению технологий виртуализации для обучения технологиям интеграции разнородных вычислительных грид-ресурсов. Рассматриваются проблемы, являющиеся специфическими для реализации различных типов грид-инфраструктур в виртуальной среде. Также в рамках статьи описываются современные методы обучения грид-системам и технологиям их интеграции посредством виртуальной среды.

Технологии виртуализации

Технологии виртуализации позволяют в рамках одного компьютера смоделировать работу одного или нескольких других компьютеров. На каждый из виртуальных компьютеров можно будет установить отдельную ОС. Посредством технологий виртуализации решается широкий спектр учебных и практических задач. Технологии виртуализации позволяют в рамках одного современного компьютера развернуть учебную инфраструктуру из нескольких виртуальных машин.

В настоящее время существует множество как коммерческих, так и свободных реализаций программного обеспечения виртуализации. Программа, осуществляющая виртуализацию на компьютере, будем называть *гипервизором*. Операционную систему, под управлением которой функционируют виртуальные машины, в рамках данной работы будем называть *базовой*. Операционную систему, работающую в виртуальной машине – *гостевой*. По принципу функционирования технологии виртуализации можно разделить на три типа:

- *Полная виртуализация* – такой тип виртуализации, при котором центральный процессор базовой машины моделирует полный набор аппаратного обеспечения виртуальной машины. Недостатком данного типа виртуализации является низкая эффективность работы. Преимуществом данного подхода к виртуализации являются минимальные требования к аппаратно-программной платформе как базовой, так и виртуальной машины.
- *Паравиртуализация* – тип виртуализации предполагающий модификацию виртуальной операционной системы. При данном подходе в рамках виртуальной машины может быть установлена операционная система, ядро которой специальным образом модифицировано для запуска в виртуальной среде. Необходимость адаптации ядра операционной системы накладывает серьёзные ограничения на область применения данной технологии. Преимуществами использования данной технологии является высокая производительность, гибкое управление работающими виртуальными машинами, возможность запуска виртуальных машин при отсутствии аппаратной поддержки технологии виртуализации.
- *Аппаратная виртуализация* – виртуализация на аппаратном уровне. Реализация аппаратной виртуализации возможна только если процессор и материнская плата базовой машины поддерживают технологию аппаратной виртуализации. Большинство

современных процессоров (таких как intel и AMD) имеют встроенную поддержку виртуализации. Недостатком применения данной технологии являются ограничения на аппаратное обеспечение базовой машины. Преимуществом использования данной технологии является максимально возможная эффективность и минимальные ограничения на применение гостевой ОС.

Разновидности вычислительных грид-систем

В настоящее время вычислительные грид-системы разделяют на два основных типа: грид-системы из персональных компьютеров (ГСПК) и сервисные грид-системы.

Грид-системы из персональных компьютеров в первую очередь предполагают, что в качестве вычислительного ресурса будут использоваться домашние или офисные компьютеры. Если взять современный домашний, офисный компьютер или ноутбук, и проследить среднюю по времени загруженность его процессора, то можно увидеть, что процессор в целом использует всего несколько процентов своего вычислительного ресурса, а более чем 90% (а это колоссальные вычислительные мощности) простаивают. Современные персональные компьютеры оснащаются достаточно мощными процессорами, это является необходимым для обеспечения возможности комфортной работы для пользователя в моменты пиковой нагрузки на процессор. Моменты пиковой нагрузки на процессор при стандартном использовании персонального компьютера являются кратковременными, а большую часть времени процессор не является загруженным. Технологии ГСПК позволяют использовать незадействованное при стандартном использовании процессорное время для научных вычислений.

Принцип работы ГСПК состоит в том, что на вычислительных узлах устанавливается и настраивается клиентское программное обеспечение, которое выполняет периодические запросы удаленному серверу на наличие заданий для своей платформы. Если на центральном сервере таковые задания имеются, то клиентская машина скачивает задание в виде исполняемого файла с необходимыми данными, и запускает его, результат работы приложения возвращается обратно на сервер.

Существует несколько современных реализаций ГСПК (BOINC, XwHep, OurGrid, Condor, X-com, SARD). Наибольшее распространение к настоящему моменту получила платформа BOINC (Berkeley Open Infrastructure Network Computing).

Программное обеспечение BOINC можно разделить на две компоненты: клиентскую и серверную части. Клиентская часть устанавливается на вычислительном узле. В её задачи входит:

1. Подключиться к одному из проектов, к какому именно указывает владелец машины.
2. Запрашивать задания у центрального сервера.
3. Скачивать задания с сервера, если они там есть.
4. Запускать у себя скачанные задания.
5. Результаты работы заданий отсылать обратно на сервер.

Серверная часть программного обеспечения BOINC выполняет следующие действия:

1. Создает задания для пересылки на вычислительные узлы.
2. Отвечает на клиентские запросы, отправляет задания на вычислительные узлы.
3. Получает результаты работы задания и передает их для дальнейшей обработки.
4. Содержит в себе web-сервер для получения информации о проекте через web-интерфейс.

Аналогично распределённое приложение для инфраструктуры BOINC можно разделить на две основные компоненты: клиентскую и серверную части распределённого приложения.

Клиентская часть распределённого приложения и есть исполняемый файл, запускаемый на вычислительном узле. Она выполняет основную вычислительную нагрузку. Серверная часть распределённого приложения создает задания (в терминологии BOINC расчетные блоки — workunits) для клиентских узлов. Как правило, расчётный блок состоит из исполняемого файла клиенткой части, объединённый со специфическим для конкретного задания входного файла с данными. После отправки задания в вычислительную инфраструктуру серверная часть распределённого приложения ждет результатов задания. Получив из инфраструктуры все

результаты заданий, серверная часть производит их обработку, и создает единый результат работы распределённого приложения.

В *грид-системах сервисного типа* взаимодействие между поставщиком и потребителем ресурсов осуществляется посредством набора сервисов. Функционирование сервисом обеспечивается специальным программным обеспечением промежуточного уровня (ППО). Основными реализациями ППО сервисного грида являются gLite [1], Globus, Unicore. Общий для них набор сервисов: безопасность, авторизация, поиск ресурсов, доступ к удалённым данным.
Для обеспечения информационной безопасности в инфраструктуре сервисных грид-систем используется механизм цифровых сертификатов. Таким образом необходимым условием функционирования узлов инфраструктуры сервисного грида является взаимодействие с сертификационными центрами.

Комбинированные грид-инфраструктуры

Для интеграции разнородных вычислительных грид-ресурсов в Институте Автоматизации Исследований Академии Наук Венгрии (MTA SZTAKI) было разработано программное обеспечение Generic Grid to Grid Bridge (3G-Bridge) [2]. Программное обеспечение 3G-Bgidge позволяет запускать задания из инфраструктуры сервисного грида на выполнение в инфраструктуре грид-систем из персональных компьютеров и наоборот. Механизм мостов позволяет осуществлять интеграцию сервисных гридов и ГСПК на системном уровне, т.е. прозрачным для пользователя образом. На данный момент этот подход реализован для связи грид-инфраструктуры EGEE/EGI с несколькими ГСПК (Рис. 1). Суть подхода состоит в специальном программном компоненте, который, опираясь на абстрактное понятие задания, может быть использован для интеграции двух грид-систем. По выполняемым функциям интегрирующее программное обеспечение можно подразделить на два типа:

- *Мост EGEE ⇒ DG*, обеспечивающий запуск заданий сервисного грида в инфраструктуре ГСПК. Данное соединение функционирует как Computing Element (CE) сервисного грида, где задания вместо вычислительных узлов направляются в инфраструктуру грида из персональных компьютеров (BOINC, XWHEP, OurGrid). Взаимодействие различных типов грид-систем обеспечивается тремя основными программными компонентами:
  1. Функционирующий на стороне gLite модифицированный Computing Element, который отправляет принятые из инфраструктуры сервисного грида задания на удалённый мост. Данный CE поставляется в качестве модуля YAIM, и может быть установлен и настроен вместе с другими компонентами gLite.
  2. На стороне сервера ГСПК функционирует специальный адаптер, отвечающий за получение заданий, их преобразование для новой инфраструктуры, и выполнение.
  3. Репозиторий приложений (Application Repository — AP), содержащий информацию о всех приложениях, проходящих через данный мост.
- *Мост DG ⇒ EGEE*, позволяющий, наоборот, запускать задания ГСПК в инфраструктуре EGEE. Поскольку принцип работы и основное программное обеспечение зависит от типа подключаемой инфраструктуры ГСПК, для каждой из них создана отдельная реализация моста:
  1. **Мост BOINC ⇒ EGEE,** который функционирует как клиент BOINC, отправляя скачанные задания в виртуальную организацию EGEE. В инфраструктуре сервисного грида задание запускается специальной программой (jobwrapper), которая запускает приложение BOINC, и эмулирует для него окружение клиента BOINC.
  2. **Мост XWHEP ⇒ EGEE,** подключающий рабочие узлы EGEE к гриду XWHEP путём запуска рабочих компонентов инфраструктуры в виде заданий EGEE.
  3. **Мост OurGrid ⇒ EGEE,** который непосредственно запускает задания на вычислительных узлах виртуальной организации EGEE.

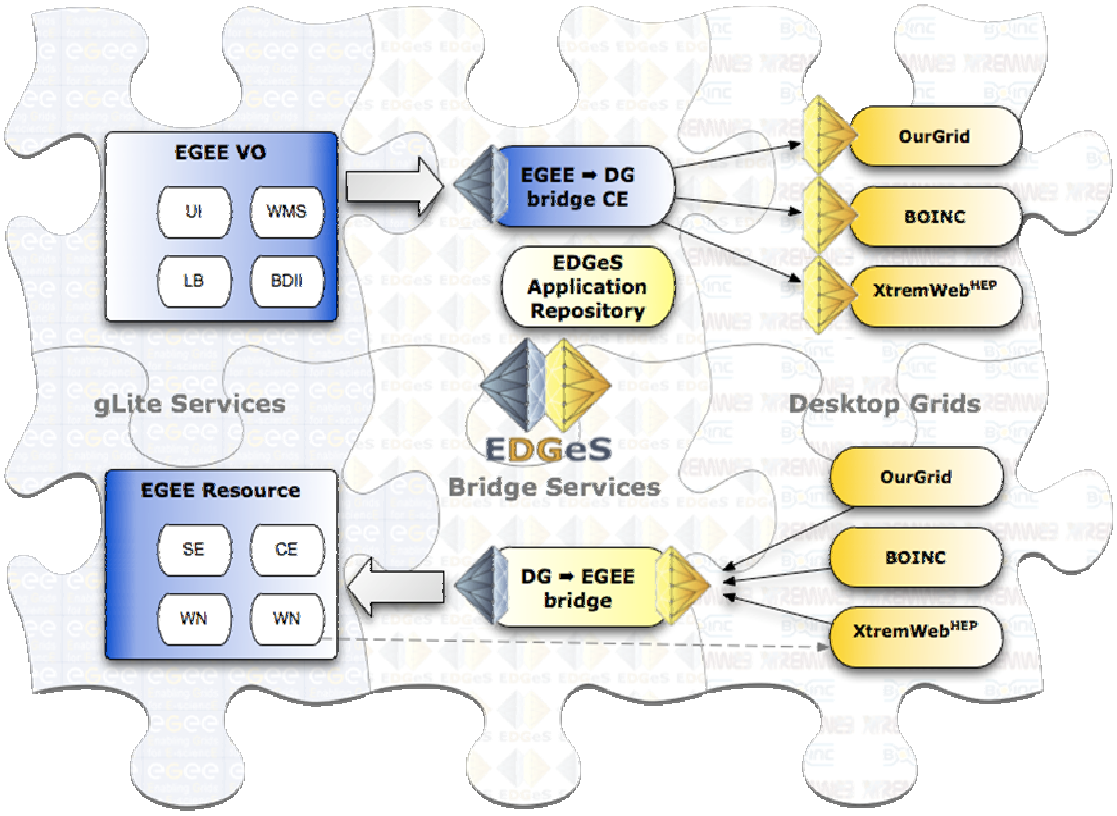


Рис. 1: Основные элементы инфраструктуры, обеспечивающие взаимодействие разнородных грид-систем

Реализация учебного грид-полигона на базе виртуальных машин

Основным препятствием для запуска сервисных грид-систем на локальных виртуальных инфраструктурах является необходимость взаимодействия элементов грид-сети с внешними сертификационными центрами. Проблемы, связанные с локальным использованием сервисных грид-систем также возникнут для сегмента сервисных грид-систем комбинированной вычислительной инфраструктуры. Для использования ГСПК в рамках локальной грид-инфраструктуры технических сложностей нет.

В рамках MTA SZTAKI для демонстрации своих разработок был предложен макет комбинированной вычислительной грид-инфраструктуры [3]. Макет представляет собой набор из пяти виртуальных машин.

Для обеспечения совместимости с различными платформами виртуализации виртуальные машины имеют формат qemu qcow2. Данный формат полностью совместим с технологией виртуализации XEN. Кроме того программной утилитой qemu-img образы виртуальных машин могут быть сконвертированы в образы виртуальных машин Vmware и VirtualBox.

Сегмент сервисного грида комбинированной инфраструктуры основан на технологии gLite. Сегмент ГСПК – на технологии BOINC. Список компонентов сервисного грида и ГСПК, содержащихся в виртуальных машинах приведён в таблице 1.

Таблица 1. Список виртуальных машин и основных компонентов программного обеспечения.

№	Название ВМ	Операционная система	Список компонентов
1	edgi-testui	SL* 5, 64 бит	Пользовательский Интерфейс (User Interface - UI).
2	edgi-testvoms	SL 5, 64 бит	Сервис Управления Виртуальными Организациями (Virtual Organization Management Service).
3	edgi-testwms	SL 4, 32 бит	Система Управления Нагрузкой (Workload Management System – WMS).
4	edgi-testce	SL 4, 32 бит	Вычислительный Элемент (Computing Element – CE) инфраструктуры glite, кэш репозитория приложений элемента 3g-bridge.
5	edgi-testboinc	Debian 5.0	BOINC-сервер, BOINC-клиент, 3g-bridge.

*SL - Scientific Linux.

Для решения проблемы взаимодействия элементов gLite с сертификационными центрами в рамках макета используется локальный сертификационный центр, и все элементы инфраструктуры сервисного грида настроены на использование данного локального сертификационного центра. Кроме того в рамках учебного макета используется упрощенная система сертификации SimpleCA.

Заключение

В настоящее время приобретают всё большую популярность как ГСПК, так сервисные гриды. Вместе с грид-системами также приобретают популярность технологии их интеграции. Задача подготовки специалистов в области грид-систем и технологий их интеграции приобретает всё большую актуальность.

Рассмотренный в данной статье макет виртуальной вычислительной инфраструктуры даёт учащемуся возможность локально работать только с отдельной реализацией системы 3g-Bridge (gLite to BOINC). Построение аналогичных макетов для интеграции других типов грид-систем заметно улучшит качество подготовки специалистов по грид-системам.

На основе виртуальных макетов инфраструктур могут быть организованы практические занятия для студентов [4]. Кроме того мощности современных вычислительных машин позволяют в рамках персонального компьютера студента развернуть виртуальный макет для испытания технологий распределённых вычислений в домашних условиях.

Литература

[1] gLite. http://glite.cern.ch/

[2] E. Urbach, P. Kacsuk, Z. Farkas, G. Fedak, G. Kecskeme'ti, O.Lodygensky, A. Cs. Marosi, Z. Balaton, Zolta'n; G. Caillat, G. Gomba's, A.Kornafeld, J. Kova'cs, H. He, R. Lovas: EDGeS: Bridging EGEE to BOINC and Xtrem Web, Journal of Grid Computing, 2009, Vol 7, No. 3, pages 335 -354

[3] Страница загрузки макета: http://www.edgi-grid.eu/downloads/vmimages/v2.0/

[4] V.A. Sukhomlin, A.P. Afanasiev, A.L. Kalinichenko, M.A. Posypkin, S.A. Stupnikov, O.V. Sukhoroslov, On Professional Training and Education in the Field of Grid Technologies and Distributed Computing // тезисы конференции GRID'2010, http://grid2010.jinr.ru/files/pdf/sukhomlin.pdf

ЭЛЕКТРОННАЯ БИБЛИОТЕКА НАУЧНОГО ЦЕНТРА[1]

С.К. Шикота[2], С.А. Крашаков[1,2], Л.Н. Щур[1,2]
[1] Институт теоретической физики им Л.Д. Ландау РАН
Россия, 142432, Московская область, г. Черноголовка, пл. ак. Семенава, 1а
[2] Научный центр РАН в Черноголовке
Россия, 142432, Московская область, г. Черноголовка, Институтский просп., 8
sveta@chg.ru

В статье излагается проект создания электронной библиотеки публикаций (ЭБП) сотрудников научного центра. Цель системы двунаправленная: во-первых – это привлечение внимания широкой аудитории к научным достижениям сотрудников НЦЧ РАН, во-вторых – предоставление сотрудникам НЦЧ РАН возможности для поиска научных контактов по смежным областям исследований внутри НЦЧ РАН. Для достижения намеченной цели предлагается система, обеспечивающая доступ к полнотекстовым публикациям научных сотрудников НЦЧ, а также содержащая актуальную информацию об индексах цитирования. Обсуждаются некоторые технические детали распределенной библиотечной системы.

1 Постановка задачи

Для успешного проведения научных исследований необходим доступ к научной информации как в области проводимых исследований, так и в смежных областях науки. Особенно это актуально при выполнении работ по приоритетным направлениям развития науки, таким, например, как индустрия наносистем или информационно-телекоммуникационные технологии. В настоящей работе излагается постановка задачи по разработке ЭБП, предоставляющей, в частности, открытый доступ к публикациям сотрудников учреждений Научного центра РАН в Черноголовке.

Назначение ЭБП двунаправленное. Первое - привлечение внимания широкой аудитории к научным достижениям сотрудников НЦЧ РАН. Второе – это предоставление сотрудникам НЦЧ РАН возможности для поиска научных контактов по смежным областям внутри информационного поля НЦЧ РАН.

В качестве показателя низкого уровня информированности научных сотрудников о проводимых в соседних учреждениях и лабораториях исследованиях и разработках даже внутри Черноголовки можно привести пример предыстории присуждения Нобелевской премии по физике А. Гейму и К. Новоселову за проведенные ими в Университете Манчестера пионерские исследования физических свойств графена [1]. Исследования проводились на образцах, изготовленных в Институте проблем технологии микроэлектроники и особо чистых материалов РАН (ИПТМ РАН), одном из учреждений НЦЧ РАН (часть соавторов в цитируемом источнике – штатные научные сотрудники ИПТМ РАН). Этот факт практически неизвестен широкой научной общественности в стране и мире. Удивительно то, что он также неизвестен и в самой Черноголовке. О чем это свидетельствует? Первое, о высоком уровне имеющихся в ИПТМ РАН технологий, позволяющих создавать экспериментальные образцы наноразмера. Второе, о наличии распределенного международного научного коллектива. Третье, о том, что в ИПТМ проводятся работы на мировом уровне по одной из критических технологий. Четвертое, о том, что в настоящее время отсутствует эффективно работающий механизм обмена научной информацией.

[1] Работа выполнена при поддержке РФФИ, грант 11-07-00385, и в рамках программы РАН “Высокопроизводительные вычислительные системы и научные телекоммуникации”.

Предлагаемая информационная система призвана обеспечить доступ широкой общественности к результатам работ, проводимых в Научном центре РАН в Черноголовке.

2 Научный центр РАН в Черноголовке

Научный центр РАН в Черноголовке [2] объединяет 13 учреждений, которые, с одной стороны, являются самостоятельными юридическими лица, с другой – находятся под управлением Научного центра. Девять учреждений – это научные учреждения четырех отделений Российской академии наук (отделения физических наук; химии и наук о материалах; наук о Земле и нанотехнологий и информационных технологий), четыре – учреждения технической и социальной сферы.

В учреждениях НЦЧ РАН работает примерно 2700 научных сотрудников и аспирантов. Общая численность штата составляет около 4500 сотрудников.

Все учреждения Научного центра объединены высокоскоростной компьютерной сетью ChANT [3], которая обеспечивает сотрудников высококачественным доступом на скоростях от 1 до 10 Гбит/с к локальной инфраструктуре сети: вычислительным кластерам, базам данных, информационным системам, грид-ресурсам, «облачным» сервисам, а также предоставляет доступ к другим научным сетям и сети Интернет на скорости 355 Мбит/с (рис.1).

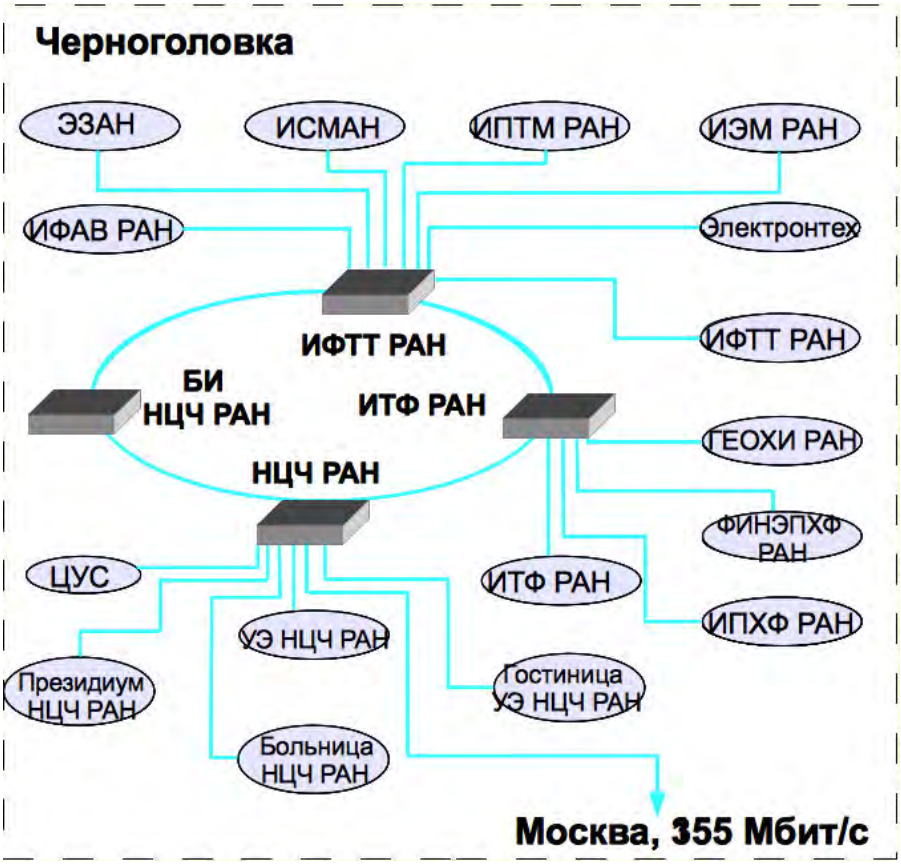


Рис. 1: Компьютерная сеть ChANT

Каждое учреждение Научного центра имеет свою собственную библиотеку. Сложность заключается в том, что библиотеки имеют различный юридический статус: 1) филиал Библиотеки естественных наук РАН (БЕН РАН); 2) филиал Библиотеки Научного центра (БНЦ РАН), которая в свою очередь входит в структуру БЕН; 3) собственная библиотека организации. Информация о публикациях сотрудников также разрозненна. Отсутствуют механизмы поиска.

Таким образом, существующая инфраструктура дает уникальную возможность по построению ЭБП, обеспечивающей доступ к публикациям сотрудников Научного центра непосредственно с рабочих мест.

3 Архитектура системы

В настоящее время в России и мире имеется большое число информационных систем для работы с научными и библиографическими данными, например, научная электронная библиотека «Киберленинка» [4], библиографическая база данных INSPIRE (SPIRES) по физике высоких энергий [5]. Многие научные учреждения имеют собственные информационные системы, представляют в электронной форме данные о публикациях сотрудников, о проводившихся или ведущихся научных исследованиях и проектах, о результатах исследований. Однако, по нашим данным, региональных информационных систем на уровне научного центра, подобных разрабатываемой в Научном центре РАН в Черноголовке, не имеется. Для решения поставленных задач представляется целесообразным разработка системы, состоящей из следующих компонент:

- Пользовательский интерфейс, обеспечивающий доступ по http протоколу к публикациям сотрудников учреждений Научного центра с различным уровнем доступа: администратор (он же сотрудник библиотеки), зарегистрированный пользователь (он же читатель библиотеки) и гость. Интерфейс пользователя также должен обеспечивать возможность поиска и отображения информации в соответствии с уровнем доступа.
- Собственно база данных, содержащая библиографическую информацию с привязкой каждой публикации к одному или нескольким авторам, работающих или работавших (на момент публикации) в одном из институтов НЦЧ и к одному или нескольким институтам НЦЧ, в которых выполнялась работа. Корректность данных по каждому из институтов должна контролироваться уполномоченными представителями соответствующих учреждений.
- База данных по индексу цитирования, предназначенная для обеспечения выдачи актуальной информации об impact факторе журнала и индексе цитирования статьи, столь необходимой в последнее время для подсчета ПРНД сотрудника и предоставления отчетов для АСУ РИД РАН. База данных будет поддерживаться библиотечными работниками.

Электронная библиотека публикаций разрабатывается как часть единой электронной системы библиотечных сервисов (ЕЭСБС) Научного центра РАН в Черноголовке [6].

4 Подсчет индекса цитирования

На сегодняшний день имеется несколько систем для подсчета индекса цитирования. Наиболее полным и широко признанным источником информации о цитировании, охватывающем все области знаний, по которым ведутся исследования в институтах НЦЧ РАН, является ISI Web of Knowledge (бывший Science Scitation Index) компании Thomson Reuters [7]. Система, осуществляет поиск по 12000 журналов и 148000 материалов конференций, глубина – 1980 г. Недостаток заключается в том, что система реферирует всего лишь порядка 100 российских изданий. Система SciVerse Scopus [8], разработанная издательством Elsevier, индексирует более 18000 наименований научно-технических и медицинских журналов примерно 5000 международных издательств. Глубина цитирования – до 1996 г. Система подсчета индекса цитирования есть и в электронной библиотеке препринтов arXiv.org [9], существующей с августа 1991 г. В России также имеется своя система - российский индекс научного цитирования (РИНЦ) [10], которая еще находится в стадии разработки и имеет ряд недостатков [11].

Результаты научных исследований публикуются не только в журналах «из списка ВАК». Нередко препринты имеют большее значение, хотя и не принимаются чиновниками от науки. Яркий пример – это серия из трех работ Григория Перельмана [12, 13, 14], которые доказывают гипотезу Пуанкаре (связное замкнутое трехмерное множество гомеоморфно сфере)

и за которую Перельману была присуждена медаль Филдса (высшая международная награда для ученых, занимающихся чистой математикой). Эта серия работ формально не опубликована, а размещена в архиве препринтов arXiv.org. Статьи широко цитируются внутри архива: первая работа из этой серии имеет индекс цитирования (на дату подготовки статьи) – 502, вторая – 273, третья – 150. Включение препринтов в нашу систему важно! Наша задача – обеспечение наиболее полного охвата цитируемых источников и цитирующих их публикаций.

В качестве источника информации о цитировании нами выбрана система Web of Knowledge, как имеющая наибольшую глубину и систему поиска двух типов: стандартную (Search) и расширенную (Cited Reference Search). Последняя позволяет осуществлять поиск в полной базе, включающей дополнительные базы (типа Chinese Citation Index, к которой мы не имеем подписки) и также ссылки на другие публикации, найденные в статьях из реферируемых источников.

Нами получено предварительное согласие о возможности использования доступа к WoK из ЭБП по программному интерфейсу.

Подсчет индекса цитирования предполагается выполнять по следующему алгоритму:

1) Для каждого из авторов, сотрудников НЦЧ, оператором формируются (по определенному расписанию) запросы типа Cited Reference Search со всеми вариантами написания фамилии и инициалов. Результаты поиска, включающие информацию об авторах (ФИО, место работы), библиографическую информацию, индекс цитирования сохраняются в базе цитирований.

2) Оператор путем анализа ссылок, выданных в результате поиска, руководствуясь списком публикаций в БД публикаций и здравым смыслом, исключает работы, принадлежащие однофамильцам. В случае затруднений, оператор может обратиться к автору(ам) работы за разъяснениями. Исключенные ссылки не удаляются из базы, а только отмечаются, как удалённые (и при последующем анализе, в принципе, могут быть включены). Одновременно осуществляется привязка каждой из ссылок к тому или иному институту. При обнаружении работ автора, не включенных в БД публикаций, следует рассмотреть вопрос об их добавлении в БД публикаций. Все результаты должны сохраняться в системе, и при последующих поисковых запросах работ этого же автора, должны быть особо выделены только новые работы, чтобы оператор без нужды не анализировал уже проанализированные ранее (если конечно, речь не идет об устранении ошибок предыдущей обработки).

3) После завершения анализа подсчитываются суммарный индекс цитирования, индекс Хирша, другие индексы (если это будет необходимо), и результаты сохраняются в карточке автора.

4) По завершении анализа всех публикаций авторов того или иного института НЦЧ аналогичные индексы сохраняется в карточке института.

Информацию об индексе цитирования предполагается обновлять примерно один раз в месяц.

5 Обеспечение доступа к полнотекстовым версиям

В соответствии с договором об авторских правах, который заключается между автором и издательством, публикующим научный труд, автор имеет право выкладывать авторский вариант на собственном сайте. Таким образом, предполагается в качестве ссылки на полный текст статьи указывать ссылку на персональную страничку автора, содержащую искомую статью. Для пользователя с уровнем доступа «Гость» будет доступна только аннотация и ссылка на электронную версию издательства. Для пользователя, имеющего статус читателя библиотеки, будет доступна ссылка на персональную страницу автора.

6 Проблемы, требующие решения

При разработке ЭБП следует учитывать некоторые особенности научных публикаций:

1) Некоторые российские журналы имеют две версии: русскую и английскую. При этом каждая из версий может иметь различную нумерацию страниц и даже томов. Таким образом, публикация в русской и английской версии должна быть учтена как одна публикация с разными выходными данными.

2) Как уже говорилось выше, многие статьи публикуются в виде препринтов (особенно это свойственно для публикаций в области физико-математических и компьютерных наук). При наполнении базы данных необходимо учитывать препринт и последующую публикацию за одну публикацию и суммировать их индекс цитирования.
3) Многие публикации могут цитироваться различными способами: как статья (глава) в книге и как статья в периодическом издании, если данная книга является сериальным изданием; как публикация в трудах конференции и как статья в журнале или сериальном издании и т.д.
4) Важным моментом при разработке ЭБП является привязка публикации к конкретным авторам и организациям. Сделать это непросто по нескольким причинам. Во-первых, имеются однофамильцы и даже полные тезки (особенно по ФИО сотрудников), нередко в одной организации. Во-вторых, имеется множество вариантов написания фамилий (и имен) в переводных публикациях. В-третьих, имеются различные варианты написания названия организации в публикуемых статьях. ЭБП должна различать такие публикации при вводе и выводе информации. Многие существующие библиографические базы пытаются решать эту проблему, однако ни одна из них не решает ее в полной степени правильно и в каждой из них встречаются ошибки. По нашему мнению, единственным способом точной привязки статей к конкретным авторам является децентрализованный экспертный подход. В конечном счете, необходима коррекция листа публикаций самими авторами.

Исходя из этого, основу предлагаемой нами подсистемы составляет общая база публикаций (без разделения на тип публикации) с привязкой каждой публикации к одному или нескольким авторам, работающих или работавших (на момент публикации) в одном из институтов НЦЧ и к одному или нескольким институтам НЦЧ, в которых выполнялась работа. Корректность данных по каждому из институтов должна контролироваться уполномоченными представителями соответствующих учреждений.

Литература

[1] K.S. Novoselov, A.K. Geim, S.V. Morozov, D. Jiang, Y. Zhang, S.V. Dubonos, I.V. Grigorieva, and A.A. Firsov, Science 306, 666 (2004).
[2] Научный центр РАН в Черноголовке http://scc.chant.ru
[3] С.К. Шикота, Л.Н. Щур, С.А. Крашаков, А.Ю. Меньшутин, М.В. Григорьева. Региональная сеть для науки и образования ChANT как инфраструктура для Грид-приложений. Труды 4-ой международной конференции ГРИД'2010 "Распределенные вычисления и Грид-технологии в науке и образовании", Дубна, 28 июня - 03 июля 2010 г., с 345-351.
[4] Научная электронная библиотека «КиберЛенинка», http://cyberleninka.ru/
[5] High-Energy Physics Literature Database INSPIRE, http://inspirehep.net/
[6] С.К. Шикота, С.А. Крашаков, Л.Н. Щур Единая электронная система библиотечных сервисов, Труды Всероссийской научной конференции «Электронные библиотеки: перспективные методы и технологии, электронные коллекции» RCDL-2012, Переславль-Залесский, 15-18 октября 2012 года.
[7] Web of Knowledge, http://wokinfo.com/
[8] Scopus, http://www.scopus.com/
[9] Электронный архив http://arxiv.org/
[10] Российский индекс научного цитирования (РИНЦ), http://elibrary.ru/project_risc.asp
[11] Н.Е. Каленов, О.В. Селюцкая. Некоторые оценки качества Российского индекса научного цитирования на примере журнала «Информационные ресурсы России» // Информационные ресурсы России, 2010, № 6. — С. 2-13
[12] G. Perelman, The entropy formula for the Ricci flow and its geometric applications, arXiv:math/0211159v1
[13] G. Perelman, Ricci flow with surgery on three-manifolds, arXiv:math/0303109v1
[14] G. Perelman, Finite extinction time for the solutions to the Ricci flow on certain three-manifolds, arXiv:math/0307245v1

ПРОБЛЕМЫ РАЗВИТИЯ ВЫСОКОПРОИЗВОДИТЕЛЬНОЙ ИНФРАСТРУКТУРЫ НАУЧНОГО ЦЕНТРА[1]

Л.Н. Щур[1,2], С.К. Шикота[2]
[1] *Институт теоретической физики им. Л.Д. Ландау РАН,*
[2] *Отдел прикладных сетевых исследований РАН,*
Научный центр РАН в Черноголовке,
142432, Черноголовка, Российская федерация
shchur@chg.ru, sveta@chg.ru

Анализируется задача планирования, построения и развития информационной и телекоммуникационной инфраструктуры научного центра. Основная цель состоит в том, чтобы доставить современные инфо-коммуникационные технологии непосредственно на рабочий стол научных сотрудников. Проблема разделяется на набор технических, организационных и методологических задач. Обсуждается доступное для решения задач свободно распространяемое программное обеспечение и связанные соответствующие потребности в развитии аппаратной части. Мы расширили наш предварительный анализ [1] и демонстрируем реализацию представленных идей в Научном центре Российской академии наук в Черноголовке. Среди примеров, проекты ВидеоГрид, МиниОблако, 10 гигабитная опорная сеть и платформа КомпФиз.

Введение

Современная информационная и телекоммуникационная инфраструктура научного центра необходима для обеспечения научной работы, выполняемой сотрудниками, студентами и аспирантами.

Существует две части решения этой задачи. Первая часть, это задача интеграции уже существующих в исследовательских центрах информационных и телекоммуникационных (ИКТ) ресурсов в единую систему, обеспечивающая работу виртуальной организации. Это дает возможность совместной работы распределенных коллективов. Вторая часть, это задача разработки инфраструктуры для осуществления исследований в определенной области науки, что дает исследователям возможность удаленного доступа к ИКТ ресурсам, разработанным третьей стороной. Это так называемая концепция совместного (разделяемого) использования ресурсов, объединенных порталом, хабом, и т.п.

Откуда происходит такая необходимость? Можно задать вопрос – а зачем это надо? Для этого есть несколько причин. На самом деле, эти причины не являются несвязанными.

Во-первых, усиливается конкуренция исследований, особенно в хорошо финансируемых областях науки, таких, как исследования в области получения энергии, в области поиска и исследования новых материалов, в области нанотехнологий. Для успешного проведения исследований и выдерживания конкурентного уровня, необходимо обеспечивать быстрое распространение достигнутых результатов, иметь надежный и своевременный доступ к результатам, полученным другими исследователями, необходимо обеспечить возможно короткий исследовательский цикл.

Во–вторых, наблюдается глобализация исследовательского процесса, что, в свою очередь имеет много причин (смотри подробное обсуждение этого процесса в недавней статье [2]. В контексте обсуждаемой задачи не важно то, что, глобализация, в частности, обусловлена пограничностью исследований по приоритетным направлениям науки.

В третьих, это проблема воспроизводства кадров в условиях быстрого накопления информации и обновления знания.

[1] Работа выполнена в рамках гранта РФФИ 11-07-00471.

В четвертых, это проблема достижения критической массы исследовательской группы [3]. Анализ данных научных фондов Великобритании и Франции указывает на существование величины числа членов научного коллектива, ниже которой успешность коллектива резко падает. Критическая масса коллектива сильно зависит от области знаний, и для успешности важно необходимо набрать определенное число научных сотрудников, что возможно, например, за счет виртуализации коллектива.

В пятых, осуществление научной деятельности по критическим технологиям требует часто вычислительных мощностей большого объема, а также передачи, хранения и обработки данных большого размера.

Именно ИКТ способны обеспечить устойчивое проведение научных исследований и потенциально привести к их успешности.

1 Анализ проблемы

Для постановки задачи необходимо провести анализ проблемы. Проблема разделяется на набор технических, организационных и методологических задач.

Анализ необходимо начать от проблем конечного адресата развиваемой системы, то есть научного сотрудника (аспиранта, студента). Для этого уместно выделить основные моменты осуществления научной работы (здесь уместно вспомнить английский термин этого процесса - workflow). Подробный анализ можно найти в нашей недавней работе [1]. Здесь мы лишь кратко перечислим основные составляющие, на которые может быть разложена научная деятельность в области естественных наук:

- научная деятельность члена научного коллектива, в том числе:
  - ✓ проведение экспериментальных исследований;
  - ✓ проведение теоретических исследований;
  - ✓ проведение вычислений, численных экспериментов и моделирования;
  - ✓ обработка результатов вычислений, их визуализация;
- преподавательская деятельность члена научного коллектива:
  - ✓ подготовка учебных материалов — лекций, методических пособий;
  - ✓ чтение лекций и проведение семинаров;
  - ✓ научная работа со студентами и аспирантами;
- административная деятельность члена научного коллектива:
  - ✓ участие в научных советах;
  - ✓ реферирование статей, экспертиза грантов, заявок и т.п;
  - ✓ решение текущих административных вопросов — финансирование, материальное обеспечение исследований;
- обсуждение текущей научной работы на семинарах научного коллектива;
- подготовка научных статей, научных отчетов, устных и стендовых докладов, презентаций;
- общение с коллегами из других коллективов;
- участие в семинарах и конференциях в качестве докладчика и слушателя;
- написание заявок в научные фонды на проведение научных работ и конференций.

Необходимо провести глубокие исследования по составу и характеру научной деятельности коллектива с целью выявления общих и важных черт научной деятельности для дальнейшего использования результатов при проектировании востребованных компонент инфокоммуникационных систем.

Следует отметить, что стиль выполнения научной деятельности достаточно консервативен, и имеются определенные барьеры для использования новых систем и методов, которые в той или иной степени влияют на стиль работы. Как правило, переход на новые системы требует определенного времени и большого терпения разработчиков. Уменьшение этого времени возможно при тесном сотрудничестве разработчиков со специалистами в предметной области, для которых создается система.

Что касается технического решения, то в настоящее время имеется большой набор программного обеспечения, подходящего для наших целей. Большая часть его имеет открытый код (СПО – свободно распространяемое программное обеспечение), что позволяет создание систем для определенных целей за достаточно разумное время. Кроме того, большая часть СПО не сильно уступает по надежности промышленным системам [4]. Если учесть необходимость построения систем для определенной задачи, при небольшом финансировании и за относительно короткое время, то выбор СПО для построения системы представляет собой предпочтительное решение.

2 Подходы других групп и сообществ

Какие есть примеры построения систем с похожими функциями?

Портал GViz (TeraGrid Visualization Portal) [5] был разработан для удаленной визуализации расчетов, проводимых на распределенной системе TeraGrid [6]. В систему была включена возможность запуска сервера ParView на вычислительных ресурсах TeraGrid и использование соответствующего ParaView клиента на рабочей станции пользователя для отображения визуальной информации. Портал TGViz построен на системах uPortal [7] и GridSphere [8]. Для запуска задач на TeraGrid используется система GRAM [9]. Такой подход предусматривает возможность реализации удаленного просмотра каталогов, а также удаленного запуска приложений.

В рамках проекта SIDGrid (Social Informatics Data Grid) [10] были разработаны портал по информатике и дата-центр для научных сотрудников в области социологии и исследования социального поведения. Система SIDGrid позволяет загружать в нее большие объемы данных, а также предоставляет возможность масштабирования анализа данных до размеров вычислительных ресурсов TeraGrid.

В портале OLSGW (Open Life Science Gateway) [11] реализована методика поддержки описания выполнения приложений и их интеграции в портал путем генерации вэб-интерфейса для их выполнения.

Технологической платформой программы «Университетский кластер» является портал Unihub [12], построенный на основе технологии HubZero [13], служит технологической платформой программы «Университетский кластер».

3 Примеры внедрения

В настоящем разделе мы демонстрируем реализацию представленных идей в Научном центре Российской академии наук в Черноголовке (НЦЧ). Научный центр объединяет 15 учреждений Российской академии наук. В нем проводятся научные исследования по физике, математике, химии, биологии, минералогии, физиологии, материаловедению, микроэлектронике, и по многим другим направлениям фундаментальной науки. Институт РАН входят в состав четырех отделений РАН.

Учреждения НЦЧ объединены в компьютерную сеть ChANT (Chernogolovka Academic Network) [14]. Сеть ChANT имеет оптоволоконную опорную сеть в 1 гигабит в секунду (Gbps) для обеспечения доступа Институтов к сети Интернет и кольцевую оптоволоконную опорную сеть в 1 Gbps для доступа к информационно-вычислительным ресурсам НЦЧ РАН.

Для целей обеспечения совместной работы научных коллективов и для скоростного обмена с большими хранилищами данных и для доступа к вычислительным ресурсам создается 10 гигабитная оптоволоконная опорная сеть с топологией восьмерки. Она построена на коммутаторах Allied Telesys.

Такой подход позволяет настраивать каналы для выполнения конкретных работ, а также проводить эксперименты с целью совершенствования программного обеспечения.

В НЦЧ РАН развернута программно-аппаратная система для обеспечения работы распределенных научных коллективов ВидеоГрид. Она построена с использованием свободно распространяемого программного обеспечения (СПО) AccessGrid [15]. Ядро системы состоит из сервера venue и четырех рабочих серверов. Сервер venue обеспечивает регистрацию пользователей и поддержку виртуальных залов, а также хранение и доступ к мульти-медийным данным. Рабочие обеспечивают работу прием и передачу видео и аудио потоков, их передачу

на звуковоспроизводящую аппаратуру и отображение на экраны, захват аудио потока из зала и захват видео изображения от четырех видеокамер. Система используется для проведения распределенных видео-семинаров [16].

Для проведения исследовательских работ в области вычислительной физики создана программно-аппаратная платформа КомпФиз [17]. Платформа базируется аппаратно на Мини-Облаке, программная часть базируется на СПО HubZero [13].

Для целей отработки технологий приема, передачи и отображения информации большого объема создан программно-аппаратный комплекс Wall. Он состоит из матричного дисплея, составленного из двадцати 27-дюймовых мониторов, способного отображать примерно 50 миллионов пикселей. Разрешение монитора позволяет, например, отображать карту всей Московской области в таком разрешении, что возможно найти объект с точностью до дома. Видеопотоки передаются параллельно, для чего используется кластер из 12 серверов. Каждый сервер обслуживает два монитора, координацию ведет управляющий сервер. Один сервер находится в режиме горячего резерва. Общая производительность кластера составляет 640 терафлоп.

Заключение

Мы изложили подход к развитию программных и аппаратных ресурсов научно-исследовательского центра, при котором создаваемые информационные, вычислительные и телекоммуникационные ресурсы основаны на подходе «*researcher oriented approach*». Основная задача подхода состоит в том, чтобы доставить весь спектр новейших информационных и телекоммуникационных ресурсов непосредственно исследователю-предметнику, причем прямо на его рабочий стол.

Литература

[1] Л.Н. Щур, А.Ю. Меньшутин, С.К. Шикота, *Инфокоммуникационное обеспечение исследовательского центра: задачи и инфраструктура //* Информационное общество. 2011. Т. 6. С. 58-68.

[2] Л.Н. Щур, *Глобализация научных исследований и инфокоммуникационные технологии* // Информационное общество. 2012. Т. 5. С. 99-99.

[3] R. Kenna, B. Berche, *Critical mass and the dependency of research quality on group size* // Scientometrics. – 2011. - V. 86 – P. 527-540.

[4] В.П. Иванников, *Что такое СПО* // сб. трудов Перспективные компьютерные системы: устройства, методы и концепции, по ред. Р.Р. Назирова и Л.Н. Щура – 2011- Москва, С. 106-114.

[5] J.A. Insley, M.E. Papka. Prototyping Simple Access to Visualization Resources// http://www.globus.org/alliance/publications/clusterworld/0605Grid.pdf

[6] https://www.xsede.org/

[7] Проект uPortal, http://www.jasig.org/uportal/

[8] Проект GridSphere, http://www.vlab.msi.umn.edu/gridsphere/gridsphere

[9] Проект GRAM, http://dev.globus.org/wiki/GRAM

[10] Levow G.-A., Waxmonsky, S., Bertenthal, B., et al. SIDGrid: A Framework for Distributed, Integrated Multimodal Annotation, Archiving, and Analysis// http://anl.academia.edu/MarkHereld/Papers/828780/SIDGrid_A_Framework_for_Distributed_Integrated_Multimodal_Annotation_Archiving_and_Analysis

[11] Wu, W., Edwards, R., Judson, I.R., et al. TeraGrid Open Life Science Gateway // TeraGrid 2008 conference, June 9-13, 2008, Las Vegas. P. 97-105.

[12] Технологическая платформа программы «Университетский кластер», http://unihub.ru

[13] McLennan, M., Kline, G. HUBzero Paving the Way for the Third Pillar of Science // HPC in the Cloud. 2011. № 9, P. 11-18.

[14] Сеть ChANT, http://www.chant.ru

[15] Проект AccessGrid, http://accessgrid.org

[16] Проект ВидеоГрид, http://comphys.ru/projectscomphys/video-grid

[17] Проект КомпФиз, http://comphys.ru/projectsanr/comphys-mon

INDEX

Ciubăncan M.	Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), Măgurele, Romania	95
Constantinescu Ş.	Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), Măgurele, Romania	95
Davronov R.R.	Institute of Mathematics of National University of Uzbekistan	13
Degtyarev A.B.	St. Petersburg State University, Russia	60, 81
dell'Agnello L.	INFN-CNAF, Bologna, Italy	204
Deng Z.Y.	Institute of High Energy Physics, Beijing, China	85
Di Girolamo A.	CERN, Geneva, Switzerland	212
Dmitrienko P.V.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23
Dubenskaya Yu.Yu.	Scobeltsyn Institute of Nuclear Physics Lomonosov Moscow State University, Russia	89
Dulea M.	Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), Măgurele, Romania	95
Dulov O.	Steinbuch Computing Center (SCC), Karlsruhe Institute of Technology (KIT), Germany	104
Dumitru B.A.	Institute of Space Science, ISS, Magurele, Romania	165
Elizbarashvili A.	Ivane Javakhishvili Tbilisi State University, Georgia	111
Elmsheuser J.	Ludwig-Maximilians-Universitat Muenchen, Germany	212
Eroshkin A.V.	St. Petersburg Electrotechnical University "LETI", Russia	81
Feofilov G.A.	Laboratory of Ultra-High Energy Physics, St. Petersburg State University, St. Petersburg, Russia	18
Field L.	CERN, Genève, Switzerland	220
Filozova I.A.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	117
Fischer M.	Karlsruhe Institute of Technology, KIT, Germany	34
Fuhrmann P.	DESY, Hamburg, Germany	220

Hejbal J.	Institute of Physics, Academy of Sciences of the Czech Republic, Czech Technical University in Prague, Fac. of Nuclear Sciences and Physical Engineering, Prague, Czech Republic	212
Ilyin V.	Skobeltsyn Institute of Nuclear Physics, Moscow State University, National Research Centre "Kurchatov Institute", Moscow, Russia	133
Irimia F.L.	Institute of Space Science, ISS, Magurele, Romania	165
Ivănoaica T.	Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), Măgurele, Romania	95
Ivanov A.N.	St. Petersburg State University, Russia	176
Kadochnikov I.S.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23, 195
Kiryanov A.K.	St. Petersburg Nuclear Physics Institute, Gatchina, Russia	144, 147
Klopov N.V.	St. Petersburg Nuclear Physics Institute, Gatchina, Russia	144, 147
Kodolova O.	Skobeltsyn Institute of Nuclear Physics Moscow State University, Moscow, Russia	133
Kompaniets M.V.	St. Petersburg State University, Russia	18
Kónya B.	Dept. of Physics, Lund University, Lund, Sweden	220
Korenkov V.V.	Laboratory of Information Technologies , Joint Institute for Nuclear Research, Dubna	23, 30, 130, 133
Korkhov V.V.	St. Petersburg State University, Russia	176
Kotlyar V.	State Research Center of Russian Federation Institute for High Energy Physics, Protvino, Moscow region, Russia	150, 171
Kouba T.	Institute of Physics, Academy of Sciences of the Czech Republic, Prague, Czech Republic	212
Kovalenko V.N.	St. Petersburg State University, Russia	18
Kryukov A.P.	Scobeltsyn Institute of Nuclear Physics, Lomonosov Moscow State University, Russia	89
Kukhtenkov V.	State Research Center of Russian Federation Institute for High Energy Physics, Protvino, Moscow region, Russia	150
Kulabukhova N.V.	St. Petersburg State University, Russia	176

Kutovskiy N.A.	Laboratory of Information Technologies, Joint Institute for Nuclear Research, National Scientific and Educational Centre of Particle and High Energy Physics of the Belarusian State University, Minsk, Belarus	23, 30, 180, 186
Kyaw Zaya	St. Petersburg State University, Russia	71
Legger F.	Ludwig-Maximilians-Universitat Muenchen, Fakultaet fuer Physik, Garching, Germany	212
Lensky I.I.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	186
Levchuk L.	National Science Center"Kharkov Institute of Physics and Technology", Kharkov, Ukraine	133
Li W.D.	Institute of High Energy Physics, Beijing, China	85
Lin L.	Soochow University, Suzhou, China	85
Magradze E.	II. Physikalisches Institut, Georg-August Universitaet Goettingen, Goettingen, Germany	212
Medrano Llamas R.	CERN, Geneva, Switzerland	212
Mitsyn S.V.	Laboratory of Information Technologies, Joint Institute for Nuclear Research, Dubna	23
Mitsyn V.V.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23
Negri G.	CERN, Geneva, Switzerland	212
Nicholson C.	Graduate University of Chinese Academy of Sciences, Beijing, China	85
Nilsen J. K.	Dept. of Physics, University of Oslo, Oslo, Norway	220
Noferini F.	Centro E. Fermi, Rome, INFN Sezione di Bologna, Bologna, Italy	204
Novodvorsky E.G.	St. Petersburg Nuclear Physics Institute, Gatchina, Russia	144
Oberst O.	Karlsruhe Institute of Technology, KIT, Germany	34
Oleshko S.B.	St. Petersburg Nuclear Physics Institute, Gatchina, Russia	144, 147
Oleynik D.A.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23, 30, 130, 189, 195

Petrosyan A.S.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23, 30, 130, 189, 195
Petzold A.	Karlsruhe Institute of Technology, KIT, Germany	158
Plăcintă C.	Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), Măgurele, Romania	95
Popova E.	State Research Center of Russian Federation Institute for High Energy Physics, Protvino, Moscow region, Russia	150
Posypkin M.	Moscow State University, Moscow, Russia	43
Prosperini A.	INFN-CNAF, Bologna, Italy	204
Pyae Sone Ko Ko	State Marine Technical University of St. Petersburg, Russia	54, 66
Quast G.	Karlsruhe Institute of Technology, Karlsruhe, Germany	34, 200
Ratnikova N.	Karlsruhe Institute of Technology, KIT, Germany, ITEP - Institute of Theoretical and Experimental Physics, Russia	34
Ricci P.P.	INFN-CNAF, Bologna, Italy	204
Rinaldi L.	Instituto Nazionale Fisica Nucleare, Bologna, Italy	212
Röcker S.	Karlsruhe Institute of Technology, KIT, Germany	34, 200
Ronchieri E.	INFN-CNAF, Bologna, Italy	204
Ryabinkin E.	National Research Center “Kurchatov Institute”, Moscow, Russia	171
Ryabov Y.F.	St. Petersburg Nuclear Physics Institute, Gatchina, Russia	144, 147
Sapunenko V.	INFN-CNAF, Bologna, Italy	204
Savin N.	State Research Center of Russian Federation Institute for High Energy Physics, Protvino, Moscow region, Russia	150
Schovancová J.	Institute of Physics, Academy of Sciences of the Czech Republic, Prague, Czech Republic	212
Sciacca G.	Albert Einstein Center for Fundamental Physics and Laboratory for High Energy Physics, University of Bern, Bern	212
Semenov R.N.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	30, 186

Serfon C.	Ludwig-Maximilians-Universitat Muenchen, Fakultaet fuer Physik, Garching, Germany	212
Sevcenco A.	Institute of Space Science, ISS, Magurele, Romania	165
Shabratova G.S.	Laboratory of High Energy Physics Joint Institute for Nuclear Research, Dubna	23, 171
Shamardin L.V.	Scobeltsyn Institute of Nuclear Physics Lomonosov Moscow State University, Russia	89
Shevel A.Y.	High Energy Physics Division St. Petersburg Nuclear Physics Institute, Russia	216
Shmatov S.	Laboratory of High Energy Physics Joint Institute for Nuclear Research, Dubna	130, 133
Smirnova O.	Dept. of Physics, Lund University, Lund, Sweden	220
Soldatov A.	State Research Center of Russian Federation Institute for High Energy Physics, Protvino, Moscow region, Russia	150
Stan I.	Institute of Space Science, ISS, Magurele, Romania	165
Strizh T.A.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23
Sukhoroslov O.V.	Centre for Grid Technologies and Distributed Computing, Institute for Systems Analysis, Russian Academy of Sciences, Moscow, Russia	228
Tatarenko V.	G.V. Kurdyumov Institute for Metal Physics, National Academy of Sciences of Ukraine, Kiev, Ukraine	123
Thurein Kyaw Lwin	St. Petersburg State Marine Technical University, St. Petersburg, Russia	76
Tikhonenko E.A.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23, 133
Tkachenko I.	National Research Center “Kurchatov Institute”, Moscow, Russia	171
Toporkov V.V.	National Research University “MPEI”, Moscow, Russia	234
Trofimov V.V.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23
Tselishchev A.S.	National Research University “MPEI”, Moscow, Russia	234
Uzhinskiy A.V.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23, 30
Vagnoni V.	INFN Sezione di Bologna, Bologna, Italy	204

Van Der Ster D.C.	CERN, Geneva, Switzerland	212
Vaniachine A.V.	Argonne National Laboratory, Argonne, USA	243
Vasile I.T.	Department of Computational Physics and Information Technologies, Horia Hulubei National Institute for R&D in Physics and Nuclear Engineering (IFIN-HH), Măgurele, Romania	95
Vasyunin D.A.	St. Petersburg State University, St. Petersburg, Russia, University of Amsterdam, Amsterdam, the Netherlands	176
Vechernin V.V.	Laboratory of Ultra-High Energy Physics, St. Petersburg State University, St. Petersburg, Russia	18
Vollmer G.	Karlsruhe Institute of Technology, KIT, Germany	200
Vorobyev I.S.	Laboratory of Ultra-High Energy Physics, St. Petersburg State University, St. Petersburg, Russia	18
Wayand S.	Karlsruhe Institute of Technology, KIT, Germany	34
White J.	Helsinki Institute of Physics, Helsinki, Finland	220
Xiao Yun	Computer Network Information Center, Chinese Academy of Sciences, Beijing, China	249
Ye Myint Naing	St. Petersburg State Marine Technical University, St. Petersburg, Russia	76
Yemelyanov D.M.	National Research University “MPEI”, Moscow, Russia	234
Zarochentsev A.	Laboratory of Ultra-High Energy Physics, St. Petersburg State University, St. Petersburg, Russia	18, 171
Zeise M.	Karlsruhe Institute of Technology, KIT, Germany	34
Zgura S.	Institute of Space Science, ISS, Magurele, Romania	165
Zhang X.M.	Institute of High Energy Physics, Beijing, China	85
Zhang Zuli	Computer Network Information Center, Chinese Academy of Sciences, Beijing, China	249
Zhemchugov A.	Joint Institute for Nuclear Research, Dubna	85
Zhiltsov V.E.	Laboratory of Information Technologies Joint Institute for Nuclear Research, Dubna	23, 133,
Zolotarev V.I.	St. Petersburg State University, Russia	60
Zvada M.	Karlsruhe Institute of Technology, KIT, Germany	34, 158, 200
Астахов Н.С.	Лаборатория информационных технологий ОИЯИ, Дубна	254

Пярн А.В.	Московский государственный университет им. М.В. Ломоносова, Факультет вычислительной математики и кибернетики, Москва, Россия	371
Раппопорт А.М.	Центр Грид технологий и распределенных вычислений, Институт системного анализа РАН, Москва, Россия	377
Русакович Н.А.	Объединенный институт ядерных исследований, Дубна	349
Степанова М.М.	С.-Петербургский Государственный Университет, физический факультет, кафедра вычислительной физики, С.-Петербург, Россия	383
Стесик О.Л.	С.-Петербургский Государственный Университет, физический факультет, кафедра вычислительной физики, С.-Петербург, Россия	383
Стриж Т.А.	Лаборатория информационных технологий ОИЯИ, Дубна	254
Тихоненко Е.А.	Лаборатория информационных технологий ОИЯИ, Дубна	254
Ткаченко И.А.	Национальный Исследовательский Центр «Курчатовский институт», Москва, Россия	389
Трофимов В.В.	Лаборатория информационных технологий ОИЯИ, Дубна	254, 343
Храпов Н.П.	Центр грид-технологий и распределённых вычислений ИСА РАН, Москва, Россия	395
Шалабай А.И.	ФГАОУ ВПО «Сибирский федеральный университет», Красноярск, Россия	365
Шикота С.К.	Отдел прикладных сетевых исследований РАН, Научный центр РАН в Черноголовке, Черноголовка, Россия	400, 405
Шматов С.В.	Объединенный институт ядерных исследований, Дубна	254
Щур Л.Н.	Институт теоретической физики им. Л.Д. Ландау РАН, Отдел прикладных сетевых исследований РАН, Научный центр РАН в Черноголовке, Черноголовка, Россия	400, 405
Яковлев А.В.	Лаборатория информационных технологий ОИЯИ, Дубна	349

Научное издание

Distributed Computing and Grid-Technologies in Science and Education

Proceedings of the Fifth International Conference

Распределенные вычисления и грид-технологии в науке и образовании

Труды пятой международной конференции

Д11-2012-127

Ответственная за подготовку сборника к печати *Т. А. Стриж.*

Сборник отпечатан методом прямого репродуцирования
с оригиналов, предоставленных оргкомитетом.

Подписано в печать 05.12.2012.
Формат 70 × 100/16. Бумага офсетная. Печать офсетная.
Усл. печ. л. 33,78. Уч.-изд. л. 51,29. Тираж 200 экз. Заказ № 57849.

Издательский отдел Объединенного института ядерных исследований
141980, г. Дубна, Московская обл., ул. Жолио-Кюри, 6.
E-mail: publish@jinr.ru
www.jinr.ru/publish/