computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

INTERACTIVE ENVIRONMENT FOR MASSIVE NEUROSCIENCE SIMULATIONS IN GRID

Andrii Salnikov [1,2], Oleksandr Sudakov [1,2], Roman Levchenko [2], Ievgen Sliusar [1], Anton Savchenko [1]

[1] Information and Computer Centre, National Taras Shevchenko University of Kyiv, 4D Glushkova avenue, Kyiv, Ukraine, e-mail: cluster@cluster.kiev.ua, http://grid.org.ua
[2] National Scientific Centre for Medical and Biotechnical Research, National Academy of Sciences of Ukraine, 54 Volodymyrska str., room 232, Kyiv, Ukraine, e-mail: biomed@nas.gov.ua, http://biomed.kiev.ua/

Abstract: *End-user oriented system for massive computations in grid is proposed and implemented. The system provides support of user interfaces for input and output data staging, asynchronous jobs submission and control, tasks status and results monitoring. The main system components include web-portal, authentication components and jobs submitter that interact with grid infrastructure. The main advantages of described grid-portal are flexibility in computations back-ends support and possibility to interactively handle thousands of jobs. The proposed integrated environment was implemented in Ukrainian National Grid (UNG) infrastructure for massive simulations of non-linear dynamics in neuroscience.*

Keywords: *Grid, portal, neuroscience, massive simulation.*

1. INTRODUCTION

Grid [1] is one of the most popular tools for large scale high-throughput computing. Ukrainian grid-infrastructure was established in 2006 as Ukrainian Academic Grid (UAG) [2]. Since 2006 number of UAG application for scientific researches increases and leads to evolving to Ukrainian National Grid (UNG). New problems that could not be efficiently solved before now are able to utilize computing power and storage elements of the UNG. One of such new application is non-linear dynamics problems concerned with biological neurons. In 2010 the virtual organization "networkdynamics" [3] has been created specially for these tasks under the initiative of Laboratory for mathematical modelling of nonlinear processes of National Scientific Centre for Medical and Biotechnical Research, National Academy of Sciences of Ukraine (NSCMBR) in cooperation with Laboratory for parallel computing, Information and Computer Centre and Medical Radiophysics Department, Radiophysics Faculty National Taras Shevchenko University of Kyiv (KNU).

Analysis of neuronal networks is one of the leading trends in modern science aimed to understand mechanisms of human brain functions like memory, cognition, etc or nervous pathologies like Epilepsy, Parkinsonism etc. The main difficulty in such simulations is a large number of neurons (>10^3-10^5) and large number of parameters to be taken into account. Simulation requires consideration of 10^3-10^5 connected neurons and solution of 10^3-10^5 non-linear differential equations. Investigation of neuronal network behaviour requires computing of 10^2-10^3 trajectories, each of 10^6-10^7 steps. Computing time of this task is 3-6 months on the one modern CPU. Size of the compressed data file for each trajectory is 0.1−1 Gigabytes. Different dynamics trajectories are not coupled so may by computed in parallel on different resources in Grid.

The main problem of Grid application is the large number of jobs that should be handled interactively. When number of jobs grows up to hundreds or thousands automation is required to accomplish the research task. Automation is also required for jobs monitoring and aggregation of jobs results. The manual operation requires a lot of knowledge beyond the subject of researcher's science area, like internals of grid operation, job description languages, command line client tools usage for job submission, monitoring, management and data staging. Human factor is another disadvantage of the manual operation. One can easily mistype the input data or lose the job identifier. Thus we can conclude that without automation it is almost impossible to perform research that requires massive distributed computing.

Many software tools available for biological neurons simulations, but authors found no such tool designed for batch procession in grid. Thus special parallel software for neurons dynamics modelling was developed by authors of this paper [4]. This software was deployed for grid-computing with integrated environment for handling massive job submission. Despite several implementations of grid portals frameworks exist [5], [6] we implemented our own approach [7] that overcome shortcomings of thousands jobs handling present in the mentioned frameworks. In present works this approach is used for building of grid-portal for nonlinear dynamics simulations in neuroscience.

2. HANDLING OF COMPUTATIONAL JOBS IN GRID ENVIRONMENT

Developed system treats a grid job as a set of executables and data specified by job description. Job description contains information about executable itself, location of input and output data and requirements for execution: processor count, system memory, local storage capacity, execution wall-time, etc. Job description is required to allow the grid-scheduler to find computing element that fulfils job resource needs. There are two common description semantics used: JSDL – Job Submitting Description Language and xRSL – eXtended Resource Specification Language. [8]

Job passes various states in grid environment:

1. Preparing the job input files, executable parameters specification and writing the job description. Requirements to computing and storage elements need to be chosen. Wall-time can be computed taking into account algorithm complexity and job parameters (number of elements, etc).
2. Job submission and scheduling. Based on the job description grid scheduler choose appropriate computing element. Information about available processors, disk space and resource load published in common grid information system and available for scheduler. There are two approaches of scheduler implementation in middleware: separate service (for example, Workload Management System (WMS) in gLite) and integration of brokering into client tools (like Advanced Resource Connector [9] clients). Passing to separate service minimizes user interface (UI) tools execution time, but job additionally has to wait in service queue and single point of failure exists. Choosing computing element directly from UI tools requires retrieving resource specification from information system and thus it takes more time to proceed but it passes job to computing element immediately. Both approaches, depending on the number of resources and their response times, waste from dozens of seconds to couple of minutes before submitting to a target computing element (CE).
3. Preparing input files on CE. Depending on middleware implementation, input files can be uploaded to a computing element bundled with a job, or downloaded by CE from specified URI. In turn, file downloading can be performed on gateway node or, after the submission to Local Resource Management System (LRMS), on worker node.
4. Pending state. When all information about job including input files has been retrieved, job is waiting for submission to LRMS. Due to CE queue limits and resources available a significant time can be elapsed before pending job become submitted.
5. Submission to LRMS. Computing element reads the job description, locates job input files and parameters. Based on requirements from the job description, job is passed to the LRMS. Each cluster have own LRMS. For end users it doesn't matter how grid job is scheduled locally.
6. In LRMS queue. Job is processed by LRMS scheduler, gets passed to selected queue and waiting for resources to be available for actual execution.
7. Running. At this stage the desired execution of program on selected worker node finally takes place. During this process some additional commands can be executed before and after job computation. These actions are produced on demand of job script itself and to ensure correct middleware operation.
8. Finishing. Execution of specified program finished and output data needs to be handled. There are several possibilities to handle data: leave it on the CE to be called for retrieval on client or upload it to specified URI on grid Storage Element (SE). The first method is less demanding – client retrieval is always available. Second method requires SE protocols support, frequently outbound internet access on Worker Node (WN) and valid user delegation available (see below).
9. Finished. CE finished output files handling and job operation is almost complete.
10. Job information deleted. After configured amount of time, information about the job removed from grid information system.

After submission to the scheduler (or to a CE directly) unique identifier is assigned to the job for further operations. Information about job state and parameters gets published and continuously updated in the grid information system.

Delegation is the process of transferring some limited rights and privileges to another party, primarily grid service. Use of delegation techniques is a common requirement for a wide range of grid applications [10]. Integrated environment, like any other grid-service, needs to obtain user delegation for further job submission and handling.

The mechanism of delegation commonly used in grid – transferring of the user proxy certificate. Proxy certificate is a short lived certificate signed by user's personal long-term certificate. Usually proxy certificate is valid for 12 hours that is less than job life cycle time. Often, the user proxy expires while the job is still running (or even waiting in a queue).

When the job with expired proxy tries to upload output files to SE – data stage-out will fail [11]. To avoid this situation delegation renewal is required. Delegation renewal can be accomplished via client tools (such as arcrenew for ARC middleware) or via MyProxy service. Like output files uploading, using client tools is the most robust way, which is independent on remote CE deployment. Interaction with MyProxy needs to be configured both on remote CE and MyProxy server.

3. INTERACTIVE PREPARATION OF MASSIVE JOBS

Composition of thousands of input files without automation is complicated. However, researches that require massive computations in most cases focused on enumeration of some parameters. Structure of input files with predominant number of parameters left the same, while changing initial conditions.

Set of jobs with enumeration of several accessible parameters handled as one project we called a jobset.

Jobset require methods, to specify all enumeration once a time. Proposed approach designed to bring special symbols that indicate parameter enumeration:

• To specify several parameter values the syntax: *{p1,p2,…,pN}* is used;

• Parameter numerical values range can be specified as: $\{P_{start}:P_{step}:P_{end}\}$. For example, specification *{0.1:0.05:0.3}* is equivalent to *{0.1,0.15,0.2,0.25,0.3}*.

Integrated environment needs to parse provided jobset files looking for defined semantics and generate set of input files for each job. The good practice is to include exact parameter value on every job in jobset into job name published to information system.

As mentioned above submission of job requires significantly long time. For example, UNG consolidate 24 clusters, and average job submission time in this segment is about 100 seconds. This submission time is provided by client tools available in ARC middleware installed in UNG [2].

Using serial submission of 1000 jobs requires more that twenty-four hours: 100*1000/3600=27.7. Submission of 12000 jobs consume a two weeks. Even if submission time can be reduced with blacklisting of some "bad" clusters, and by aggregating jobs submit (several jobs per submit) submission of thousands jobs may take hours or days.

Existing implementations of grid portals and frameworks are not suitable for massive submission, because of using interactive submission from user interface [5], [6]. Integrated environment needs to separate the process of input data preparation from actual grid job submission to accomplish the goal.

We suggested the following methodology for massive submission with integrated environment:

• User creates a jobset by means of UI;

• UI passes jobset to the portal server along with user identity;

• Server stores information about jobset in a database and notify user about submission via integrated environment;

• Batch server backend checks database on regular basis and looks for new jobs;

• Server retrieves user delegation and submits jobs from jobset non-interactively in background.

4. NETWORKDYNAMICS WEB PORTAL

To investigate neuron networks behaviour and determine which set of input parameters lead to synchronization it is necessary to recompute enumerations of data. Massive distributed grid computing is the efficient way for that researches, so integrated environment requires for end-user scientists.

On demand of networkdynamics VO, integrated environment has been implemented. User interface to solution has been built using Adobe Flex technology to be implemented as a cross-platform web-based solution – portal for non-linear networks investigations.

Backend implements described methods to support massive computing and analysis. Backend is written in the PHP [12] and provides the JSON RPC for interaction with the UI. Backend separation provides an ability to completely change UI implementation or to use more than one different interfaces (including non web-based).

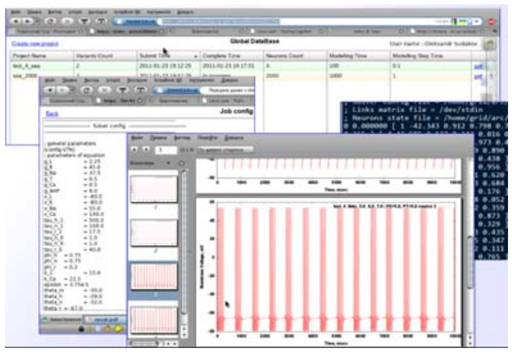

Fig. 1 – Web portal screenshot

Portal is available at the web site https://chimera.biomed.kiev.ua for VO members only due to authentication restrictions.

User authentication on the backend uses HTTPS client certificate authentication method and require PKCS12 user certificate to be imported into the web browser. User certificate Distinguished Name (DN) serves as unique identity to all user-dependent operations on the backend.

If HTTPS authentication has successfully passed, VO authentication is performed next. Integrated environment connects to VOMS server via SOAP interface [13] (PHP-SOAP is used for this purpose) and checks membership of authenticated user.

Networkdynamics VO uses PHP VOMS-Admin 0.6 software with membership database multi-master replication enabled: two instances at NSCMBR and KNU. Thus backend involves use of several VOMS servers for authentication to achieve fault tolerance.

Backend has the database of jobs under control of the integrated environment. MySQL backend is utilized for this purpose, and several database schemas have been tested to optimize performance. Every jobset contain thousands of jobs and when number of jobsets grows, backend must able to handle millions of jobsets.

Saving parameters and input file links in relational database is more processor and time intensive operation, compared to storing it as serialized arrays within the same jobs table. Database access operations in networkdynamics implementation do not require searching for every parameter, so indexes are available only for several fields: job and jobset ids, authors, searchable parameters (separate fields of job table) and job status. Future system enhancements will change this behaviour to more universal.

Monitor script selects jobs having transient status (already finished or failed jobs don't get queued, because nothing has to be done for final status). The querying of grid LDAP-based information system is performed. Querying the data also has the specific features. Efficient querying applies request-time filtering techniques by jobid to eliminate continuous interactions with LDAP servers. In the system with thousands of jobs filter size grows, and LDAP cannot handle megabytes of filter specifications. Filter data requires separation to manageable pieces (4KB in general).

LDAP response contains information actual job status for every jobid. This information is used for updating the database. Monitor script is executed on regular basis by means of CRON UNIX subsystem. To prevent concurrent script running, when CRON interval become smaller then LDAP query time locking mechanism has been implemented as well.

As mentioned above, every grid operation requires delegation from user. Networkdynamics portal implements transparent delegation retrieval from MyProxy server. MyProxy supports "authorized retrievers" policy that describes subjects able to retrieve user delegation without password providing service SSL credentials.

To allow transparent delegation, user need to upload proxy certificate to KNU MyProxy server (px.grid.org.ua) that supports NSCMBR portal certificate as authorized retriever.

Job submission also relies on CRON. Prepared jobs have the initial state in database, that can be queried and job will be submitted. Locking mechanism becomes more important for submission, because of long submission times. Input files are generated and passed to remote CE at submit time, while statically linked programs are available for download from specified URL. This approach use CE caching mechanisms [11], which prevent clusters from making thousands of code copies.

Unfortunately, due to different systematic-less configuration of clusters in UNG, automatic results uploading to SE on every CE become impossible. So we forced to stick on manual results retrieval. Retrieving process, like submission and monitoring, operates the same way by querying finished jobs from database and retrieving results using client tools. Third party transfer allows storing space-consuming results on SE.

Job script contains a code for visualizing results – plotter that stores images in PDF file. PDF files are stored on portal file backend locally, and are available for download from the UI for the first step analysis.

Aggregation of jobset PDF files is also supported, so users can simply review the results of the whole jobset.

5. USING ARC1 CLIENTS IN GRID-PORTAL

A new generation of ARC middleware client tools (starting from version 1.0.0) supports multiple endpoints to submit job as well as multiple sources of information about available VO resources.

Endpoints supported by ARC A-REX computing element are: GridFTP (legacy interface that rely on Globus Toolkit and compatible with ARC 0.x clients), xBES – extended OGSA BES interface implementation and a new-one EMI-ES interface that is now under development in the EMI project.

ARC clients are capable of submitting jobs not only to A-REX interfaces, but also to gLite CREAM CE or UNICORE/X execution service. The future development of portal will cover this way of submission to other infrastructures.

Information sources supported by ARC clients are LDAP resource information systems (including own Nordugrid LDAP schema and Glue2.0 schema), WSRF XML resource information endpoints, LDAP-based EGIIS indexing service and EMI Registry indexes.

Compared to the old clients (which were initially used in the first versions of grid-portal) where grid job ID was the ultimate source of information, a new clients require additional info to operate with jobs (like the exact URI of endpoint where job was submitted and the information source). This additional information is stored in a database local to ARC client (in current implementation file jobs.xml is used). Database contains every required (or even optional, like human-readable job name) parameters associated with every grid job ID.

So, unlike old implementation, where only grid job ID had to be stored in the portal database, with a new clients it is required to preserve jobs.xml file.

Common jobs.xml is not efficient for massive computations – there are performance issues with parsing a huge XML document and there are many different users that access the portal simultaneously.

General scalable approach is to save separate jobs.xml file for every job submitted in portal database. Networkdynamics web portal is project-oriented and all operations are performed in terms of jobsets instead of independent jobs including status updates. With a new ARC clients (starting from 2.0.0 version) it is possible to perform batch operations and retrieve information about every job in jobs.xml with a single command.

Taking into account all this considerations, it was decided and implemented in the current version of portal to save separate jobs.xml files for every jobset in the internal database and substitute it for every ARC client command invoked from shell wrappers.

UNG now contains many A-REX instances with different endpoints – legacy GridFTP available on every cluster, xBES and EMI-ES are supported on several installations. Networkdynamics web-portal is now ready to submit to EMI-ES transparently.

6. UTILIZING STORAGE INFRASTRUCTURE OF UNG

ARC 1.x comes with built-in support for accessing Storage Resource Manager (SRM) compatiple storage elements and LCG File Catalog (LFC) data catalogue services, which are de-facto standards in the global grid infrastructures like EGI and WLCG.

Clusters that support netowrkdynamics VO deploy ARC 1.x and higher and hence support data staging using SRM and LFC as well as GridFTP.

Portal was extended to automatically stage-out jobset results to the storage infrastructure of the VO, which consists of storage elements and central redundant data catalogue deployed at KNU.

EMI StoRM SRM implementation was deployed on CHIMERA cluster as a primary storage for networkdynamics computations. It provides dedicated disk storage resources for the VO accessible via SRM and raw GridFTP protocols.

LFC catalogue is used to track replicas of files as well as access permissions. Portal uses LFC references when producing job description which allows off-loading data staging from the portal to the accepting computing element. After job completion at the target CE, it automatically stages out data to the CHIMERA SE and registers file location in the central LFC catalogue.

7. GRID-PORTAL API

A test version of new grid-portal API is developed based on the experience in creation and applications of networkdynamics web-portal. A new API includes three main parts: job monitor, job submit system, results access system. Grid-portal API contains software components that simplify creation of end-user web interfaces to grid jobs. *JavaScript* components of the dynamic user-interface communicate with server (section 3) using *JSON-RPC* protocol. For data parsing and representation components *jQuery*, *dataTable* and *jsTree* are utilized. To optimize server access times for large number of jobs and large data files HTML5 *localStorage* objects are used if available in client browser. AJAX library is used for asynchronous communications between server and client if *localStorage* is unavailable.

Monitoring system periodically communicates with server using AJAX library to get, renew or change status of running tasks. Minimum information is transferred over the network and only if task status changes occur.

Job submitting interface receives existing task profiles from server, stores task profile on server and submits jobs based on task profiles specified by user. Task profiles include set of input and output jobs data that may be edited by user (section 3). Profile data is sent and received by the same methods as in monitoring interface. Development of job workflow support is now in progress.

Visualization interface provides visual representations and aggregation of job results as plots, pictures, diagrams, animations etc. Downloading of job results data files and building of data aggregation maps for multiple jobs are implemented now.

8. APPLICATION RESULTS

Application of proposed system was performed for investigation of condition when the network of 2000 neurons transfers from chaotic (healthy) to synchronized (Parkinsonism disease) state. For this realistic simulation Rubin-Terman model [14] was used and 640 trajectories of 10 seconds dynamics on 2 clusters of UNG were computed.

Rubin-Terman model is one of the most realistic models for neurons responsible for certain diseases, like Parkinsonism. It involves six nonlinear differential equations per neuron that describes different electrochemical parameters. Neurons are coupled together by one dynamic variable. Simulation requires consideration of 10^3 connected neurons and in turn solution of system of non-linear differential equations. Investigation of neuronal network behavior requires computing of 10^3 trajectories, each of 10^6 steps. Computing time of this task is about 4 months on the one modern CPU. Total size of the compressed data files for simulation is about 1 Terabyte. About 100 CPUs were used and it took about 15 days that was about 50 times faster then on supercomputer in Juelich where 32 CPUs were available for such jobs [4]. The main scientific result was that network can be transferred from healthy to pathological state only when the links number and strengthens reduced.

Another application of proposed system is investigation neuronal networks described by Kuramoto-Sakaguchi model. Kuramoto models [15] treats each neuron as phase oscillator. Each oscillator is described by one nonlinear differential equation. Set of such oscillators are coupled. Physical meaning of phase is position of neurons membrane potential pulses in time. Investigation of hyper-chaos and synchronization regions in parameters space by Lyapunov exponents method requires obtaining of 10^{6-7} trajectories with 10^{5-6} steps. Computing time of this task corresponds to 1-3 months on single modern CPU. Simulation of 10^6 trajectories of 10^5 steps on 3 clusters (115 CPUs) took about 3 days. The main scientific results were determination of initial conditions and parameters ranges where chaotic and synchronized behavior can exist.

Some examples of massive jobs results representations are depicted in Fig. 2-Fig 5 [16]. Fig. 2 describes an example of aggregated parameters map. Each pixel of this map corresponds to a single trajectory for 1000 neurons simulated with certain parameters. Vertical and horizontal axes correspond to coupling radius and initial phase. Pixel colour describes number of frequency clusters in the network. This plot is one of the possible representations of bifurcation diagram.

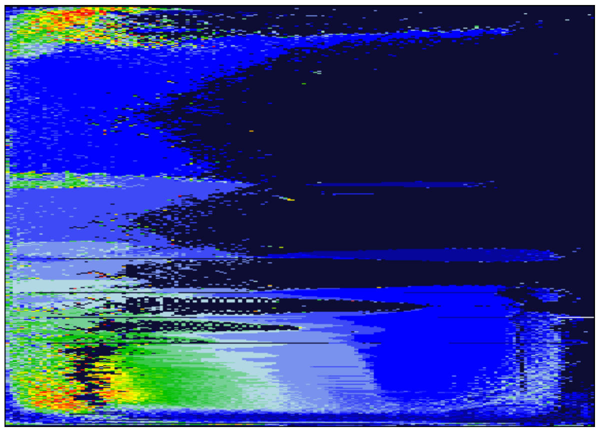

Fig. 2 – Aggregated parameters map (79000 trajectories)

Clicking at appropriate place in map one can see all data available for trajectory, like animations of trajectory dynamics in time (Fig. 3, Fig. 4), phase space diagrams (Fig. 5), parameters values etc. Animations may be easily compared by visual inspection. Parameters sets may by easily imported into other software for data analysis.

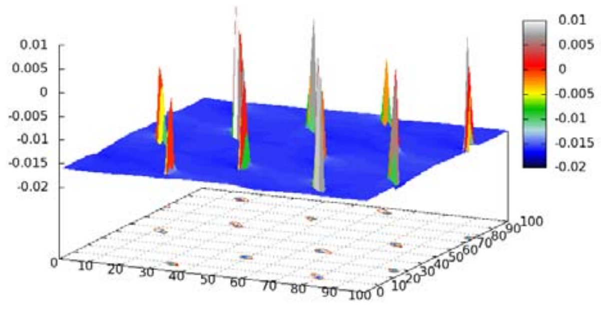

Fig. 3 – Dynamics of neurons frequencies (2d network 100x100 neurons)

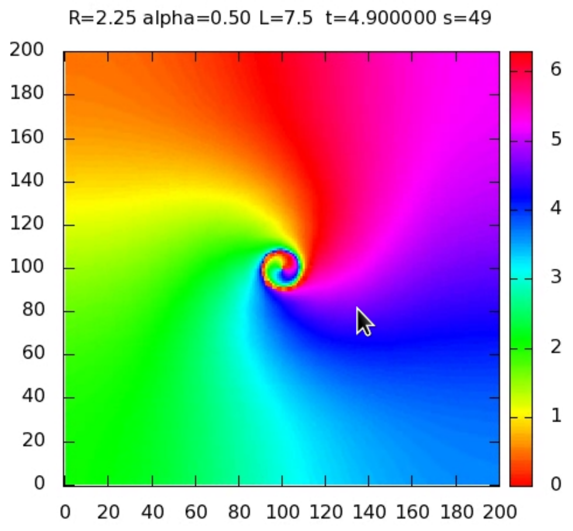

Fig. 4 – Dynamics of neurons phases (2d network 200x200 neurons)

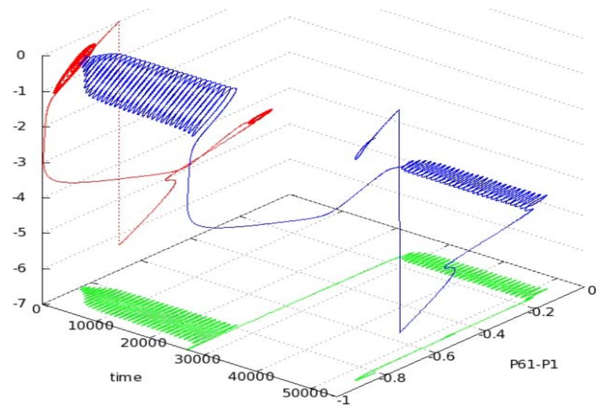

Fig. 5 – Phase space diagram for two neurons

9. CONCLUSIONS

Developed job management system allows handling thousands of grid jobs form a single web-based user interface and provide a framework for a complete automation of grid-computing cycle: from input parameters preparation, submitting and managing jobs in background to staging-out a job results to the storage infrastructure.

Backend implementation of the latest framework build on the top of Nordugrid ARC1 clients that allows a transparent usage of different information sources (including gLite Top-BDII, ARC EGIIS and EMI EMIR) as well as job management endpoints, making the whole system GLUE2 and EMI-ES ready.

Data storage infrastructure used involves SRM-storages and LFC file catalogue, to automatically stage-out data for further processing.

Proposed framework provides an open RPC API for easy extension with other interfaces that can be used for completely different projects.

First applications of the grid-portal has been devoted to brain structures simulations during the Parkinsonism disease and obtaining a new results in the field of neuroscience and non-linear dynamics using Kuramoto-Sakaguchi phenomenological model for phase oscillators and Hodgkin-Huxley-Katz realistic neuron models.

Proposed solution provides the way of research automation in the grid infrastructures to efficiently utilize computing resources.

10. ACKNOWLEDGEMENTS

All computations were performed in Ukrainian National Grid Infrastructure. Research and developments were held using computing clusters of the Information and Computer Centre of National Taras Shevchenko University of Kyiv and National Scientific Centre for Medical and Biotechnical Research, National Academy of Sciences of Ukraine. Funds were provided in the framework of The Ukrainian State Scientific and Technical Programme on Development and Application of Grid Technologies in 2009–2013.

11. REFERENCES

[1] I. Foster and C. Kesselman, *The Grid: Blueprint for a New Computing Infrastructure,* Morgan Kaufmann, 1999.

[2] M. Zynovyev, S. Svistunov, Y. Boyko, and O. Sudakov, Ukrainian grid infrastructure: Practical experience, *4-th IEEE Workshop IDAACS'2007*, (September 6-8, 2007, Dortmund, Germany, pp. 165-169.

[3] Virtual organization membership service of Ukrainian National Grid (UNG), National Taras Shevchenko University of Kyiv, 2011. [Online]. Available: http://grid.org.ua/voms/

[4] R. Levchenko, O. Sudakov, and Y. Maistrenko, Parallel software for modeling complex dynamics of large neuronal networks, *17th International Workshop on Nonlinear Dynamics of Electronic Systems*, June 2009, Rapperswille, Switzerland, pp. 34-37.

[5] Lunarc Application Portal – User's guide, University of Lund, 2006. [Online]. Available: http://laportal.sourceforge.net/docs/users_guide .pdf

[6] P-GRADE Grid Portal, Sztaki, 2005. [Online]. Available: http://portal.p-grade.hu/

[7] A. Salnikov, I. Sliusar, O. Sudakov, O. Savytskyi, and A. Kornelyuk, Moldyngrid virtual laboratory as a part of Ukrainian academic grid infrastructure, *Proc. IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications,* 21-23 September 2009, Rende (Cosenza), Italy, pp. 237-240. [Online]. Available: http://moldyngrid.org

[8] The NorduGrid Collaboration, Web site. [Online]. Available: http://www.nordugrid.org

[9] M. Ellert et al., Advanced resource connector middleware for lightweight computational grids, *Future Gener. Comput. Syst.*, (23) 1 (2007), pp. 219-240.

[10] GridSiteWiki, Grid Security for the Web Web platforms for Grids, 2007. [Online]. Available: http://www.gridsite.org/wiki/Delegation_protoc ol

[11] ARC Clients: User's Manual, Nordugrid, 2011. [Online]. Available: http://www.nordugrid.org/ documents/arc-ui.pdf

[12] T. Converse and J. Park, *PHP 4 Bible*, New York, NY, USA: John Wiley & Sons, Inc., 2000.

[13] R. Alfieri et al., From gridmap-file to VOMS: managing authorization in a grid environment,

Future Gener. Comput. Syst., (21) 4 (2005), pp. 549-558.

[14] D. Terman, J. Rubin, A. Yew, and C. Wilson, Activity patterns in a model for the subthalamopallidal network of the basal ganglia, *J. Neurosci.*, vol. 2, 2002, pp. 2963-2976.

[15] O.E. Omel'chenko, M. Wolfrum, S. Yanchuk, Y.L. Maistrenko, O. Sudakov, Stationary patterns of coherence and incoherence in two-dimensional arrays of non-locally-coupled phase oscillators, *Phys. Rev. E 85*, 036210 (2012).

[16] A. Salnikov, R. Levchenko, O. Sudakov, Integrated grid environment for massive distributed computing in neuroscience, Proc. *6-th IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications,* 15-17 October 2011, Prague, Chech Republic, – pp. 198-202.

Andrii O. Salnikov, *Master of physical and mathematical sciencies, Engineer of Parallel Computing Laboratory at Information and Computer Center of National Taras Shevchenko University of Kyiv.*

Scientific interests: grid and high performance computing

Oleksandr O. Sudakov, *Candidate of physical and mathematical sciencies (Ph.D.). Associate Professor of Medical Radiophysics Department, Head of Parallel Computing Laboratory at Information and computer Center National Taras Shevchenko University of Kyiv. Graduated from Radiophysics Faculty Taras Shevchenko Kyiv University in 1996. Ph.D. thesis in Procession of magnetic resonance tomography signals in 2002.*

Scientific interests: high performance computing, physical processes in biological systems.

Roman I. Levchenko, *Candidate of technical sciences (Ph.D.), Engineer of National Scientific Center for Medical and Biotechnical Research, NAS of Ukraine. Ph.D. thesis in automatic dynamic parallelizing of calculations for heterogeneous multiprocessing computers in 2011.*

Research Interests: Numerical methods and nonlinear dynamic, high performance computing systems and information technologies in physic, dynamic parallelizing for heterogeneous computer systems.

Ievgen A. Sliusar, *Master of physical and mathematical sciences, Engineer of Parallel Computing Laboratory at Information and Computer Center of National Taras Shevchenko University of Kyiv.*

Scientific interests: grid and high performance computing.

Anton I. Savchenko, *National Taras Shevchenko University of Kyiv, Faculty of radiophysics, Quantum radiophysics department, 4-th year bachelor student.*

Scientific interests: computer science, distributed computing, front-ends development.