My WorkSphere: Integrative Work Environment for Grid-unaware Biomedical Researchers and Applications

Zhaohui Ding[1], Yuan Luo[1], Xiaohui Wei[1]

zhaohui.ding@email.jlu.edu.cn; pp.jordan@email.jlu.edu.cn; weixh@jlu.edu.cn

College of Computer Science & Technology, Jilin University, Changchun, Jilin, China 130012[1]

Chris Misleh[2], Wilfred W. Li[2], Peter W. Arzberger[2]

cmisleh@sdsc.edu, wilfred@sdsc.edu, parzberg@ucsd.edu

National Biomedical Computation Resource, University of California, San Diego, La Jolla, CA, USA, 92093[2]

Osamu Tatabe[3]

tatebe@cs.tsukuba.ac.jp

Department of Computer Science, University of Tsukuba, Tsukuba, Japan[3]

Abstract: In order to deliver cyberinfrastructure to the general scientific and biomedical research community, transparent access and ease of use are of critical importance. Applications in systematic modeling of biological processes across scales of time and length demand more and more sophisticated algorithms and larger and longer simulations. The increased level of sophistication requires that cyberinfrastructure developers either work closely with the applications scientists, or develop middleware that flattens the learning curve for these scientists to use the grid willingly and transparently. Many life sciences researchers prefer to run applications in the grid environment without modifications, and without knowledge of specific computational resources being utilized. Here we report the latest advances in the use of Gfarm-FUSE (Grid Data Farm-Filesystem in UserSpaceE) as a computational data grid, with CSF4 (Community Scheduler Framework 4) as the metascheduler, through a GridSphere portal based environment, termed My WorkSphere. We describe the design and performance of this transparent grid computing environment using bioinformatics and computational biology applications as examples. All the components developed or utilized are open source and available freely.

Keywords: GridSphere，Metascheduler, Community Scheduler Framework 4, Gfarm, Bioinformatics

Introduction

The rapidly improving genome sequencing technology and genome assembly software have made possible the deciphering of the building blocks of diverse species, from microbial to humans. However, scientists are still learning ways not possible before to take advantage of this deluge of genomic information. Multiscale modeling, across the length scale from nanometers for molecules to meters for human bodies, as well as across the time scale from nano-seconds for molecular interactions to the length of human life, is crucial to the development of simulation systems that enable the understanding of the interactions of human physiology and the environment [1]. Despite the “tyranny of scale” (the inability of current computing technology to meet the requirement of full simulation of complete systems across scale [2]), the necessity of multiscale simulation activities is evident. With this type of knowledge, it's possible to develop predictive models and preventive strategies for human diseases and environmental disasters. One challenging task to the development

of multiscale biological systems models is to develop the necessary standards, tools and databases using state of the art computer science and technology [3,4].

The increased sophistication in developing multiscale models inevitably demands more attention from the researchers to the specific systems, leaving them little time to learn new programming models or computing technology. The advances in computing technology are turning commodity computing into reality. Nowadays, clusters with peak speed in the teraflop range are springing up across the globe, at $1/10^{th}$ the cost of a supercomputer with the same capacity 5 years ago. Whereas the newest challenge is to build petaflop scale supercomputers, it's not far fetched to imagine this newest breed would become common place within the next decade. Even as more and more "privately operated" computing facilities become available, grid computing will become even more important as researchers develop more complex models and increase in their desire to share spare computing cycles, and as grid data security satisfies the requirement of biomedical researchers in terms of confidentiality and fidelity.

However, there remain several major stumbling blocks before the highly focused biomedical researchers are willing to use grid computing on a daily basis. First of all, the ability to establish a unique identity that can be as convenient to use as a passport or identity card with password protection is essential. With such a single sign on (SSO) system, a user can be easily authenticated to all the resources with the proper authorizations. Such an identity must be recognized across different virtual organizations (VO's), or administrative domains. Currently the commonly used authentication system is the X.509 certificate system based the Public Key Infrastructure (PKI) specification (for more information, see [5]). This is available as Grid Security Infrastructure (GSI) in the Globus Toolkit, which includes a Simple CA (certificate authority) for creating and signing X.509 certificates. However, many applications do not support X.509 authentication, and that's limiting its widespread use. Other systems, such as Shibboleth, use SAML (Security Assertion Markup Language) to federate identities across virtual organizations. A NSF middleware initiative (NMI) funded project named GridShib [6] aims to bridge these two technologies. However, the additional administrative overhead has slowed the adoption of federated authorization systems so far.

Besides the authentication and authorization of users, the ability to deploy existing applications seamlessly into a grid environment is rather important. As the demand for more complex simulations of biological systems increases, applications scientists have very little time to think about or even willing to learn how to develop their applications with the grid in mind. Quite often, researchers choose TeraGrid sites based on the type of commodity cluster operating system used in their own software development, to reduce or eliminate modifications required to scale up onto the TeraGrid. While many problems are best solved on petaflop scale supercomputers with low latency networks, the development and maintenance of such a system is extremely expensive, and the platform is often obsolete within a few years, given the exponential growth in processor speed and network bandwidth. The desire of biomedical researchers not to learn the "newest" and "fastest" computing technology, but to focus on solving their problems at hand means that the ability to deploy traditional applications continues to be very important. New programming models takes time to filter down to the educational curriculum to enable new generations of programmer to develop more grid-aware applications. On the other hand, many proven applications are prohibitively expensive to rewrite for the grid, and must be deployed and optimized as is.

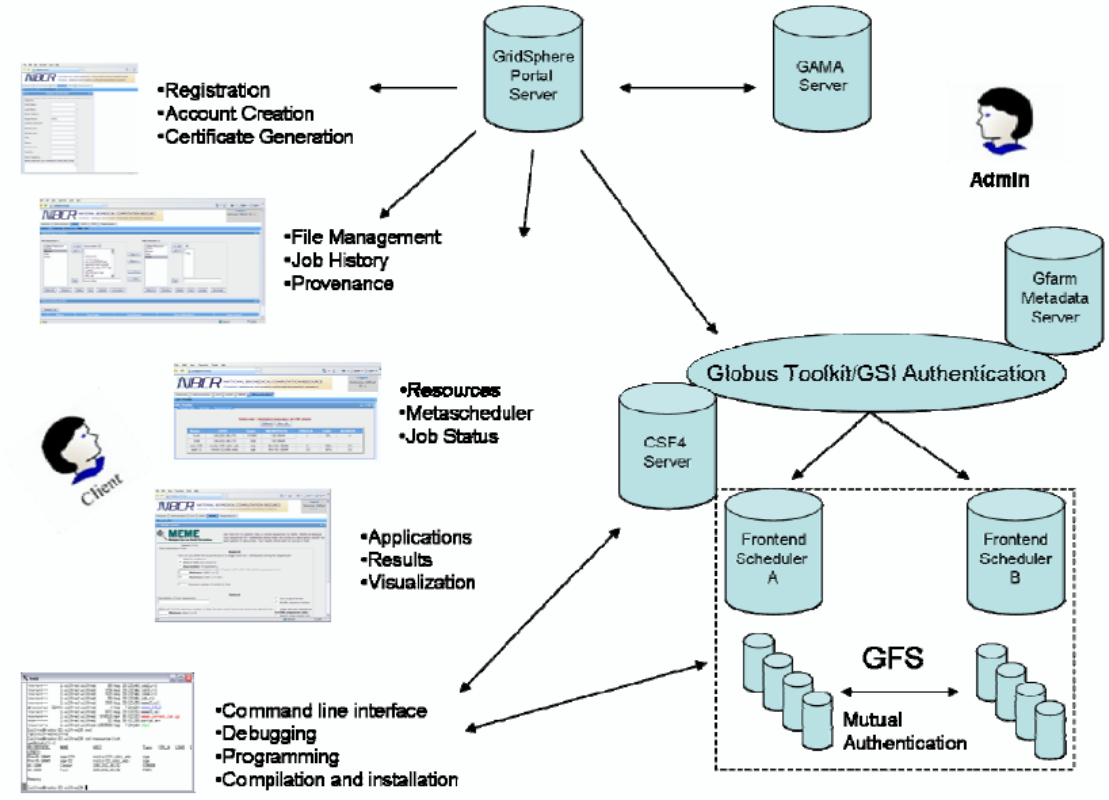

Figure 1. My WorkSphere utilizes open source components and provides an integrative environment for easy access to the grid computing environment.

Additionally, as the requirement for computing power increases, the data processing and memory requirement also becomes bigger. Network and disk input/output bandwidths pose additional challenges. Quite frequently, the challenge for commodity cluster configuration is to design the systems with sufficient network attached storage (NAS) that can meet the demand of the applications with the best price/performance ratio. The price of a low latency network equipped cluster is now comparable to one without just a few years ago. While this is good news, it's still a far cry from meeting the demand of applications, whose combined I/O can easily overwhelm 10 Gbps networks. As the cost of large local disks become cheaper, the development of a grid filesystem that takes advantage of local disk I/O yet supports existing applications becomes critical. Lastly, as more and more users demand more intuitive and interoperable computing environment, the ability to schedule jobs transparently and to access data transparently with high throughput becomes important.

My WorkSphere

My WorkSphere is an integrative environment which leverages open source components to provide easy access to grid computing resources. The major components are shown in (Figure 1). In this particular environment, we utilize the GridSphere portal framework and Gridportlets [7,8], the GAMA server and portlet [9], the Opal web service toolkit [10], the MEME Opal portlets [11], the community scheduler framework 4 (CSF4) [12], Gfarm [13], Globus Toolkit and Commodity Grid Kits [14,15,16], and Rocks [17]. While most components have been previously described, we present here the latest advances in the development of this integrative environment, including the newest development in

software packages and performance evaluation of the system. The purpose of My WorkSphere is to prototype an environment where users can easily gain access to grid computation resources; run jobs without worrying about what resources they are using, and deploy applications once and use it everywhere on the grid. My WorkSphere will serve as the NBCR portal to the TeraGrid Science Gateways for the biomedical community researchers.

GridSphere

GridSphere portal framework provides a development environment for JSR 168 compliant portlets, which are java based web components called servlets that run inside the Apache tomcat container [18]. It is currently used in a number of projects internationally, in particular, academic institutions. The GridSphere portal is easy to setup, and there are a set of portlets bundled as Grid Portlets that provides basic functions of file browsing, credential management, and job execution, as long as the proper resources are configured for use with the Globus toolkit. The Grid Portlets is shipped with the Java COG kit, and doesn't require the Globus Toolkit installation on the portal server, except for basic grid security requirements such as valid host certificates and trusted certificate authorities (CA), as required by COG. GridSphere and GridPortlets provide a set of APIs for development of additional portlets that can take advantage of the credential management and account management services.

GAMA (Grid Account Management Architecture)

GAMA is co-developed by the GEON, NBCR, and SDSC at UCSD. GAMA 1.3 has two major components, the GAMA server and the GAMA portlet. The GAMA portlet fully automates the user account creation on the portal, and the X509 certificate creation, and signing. In this case, the GAMA CA is operated by NBCR, and is trusted as needed by PRAGMA [19] partners and the TeraGrid Science Gateways [20]. GAMA server may also be used to create host certificates, and serve as a MyProxy server [21]. The advantage of having a project/organization based GAMA server is reduced administrative overhead and maximum flexibility.

The GAMA portlet automatically retrieves user proxy from the GAMA server upon user login into the portal. This is one more step towards SSO, where login once allows one to access resource readily. One additional feature of GAMA is the management of user certificates, and import of user certificates into a new GridSphere portal, which simplifies the upgrade process of GridSphere. The account approval process may also be linked to automatic cluster account creation and grid-mapfile entry addition. Currently gxmap is being used to automate this process, and GAMA version 2 plans to further automate this process.

Opal web service toolkit and MEME portlets

Opal web service toolkit [22] is developed to make the deployment of existing applications as web services easy. It leverages the Globus GRAM for job submission, and requires a simple configuration file for the location of the application, and command line options may be supplied as required. A number of web services for popular applications such as MEME [23], APBS [24], and AutoDock [25] are available and used by any clients that support web services. For applications such as AutoDock

that require license agreements, Opal based web services support GSI authentication so that only users with appropriate licenses are issued certificates, and allowed to access AutoDock based web services.

Opal based web services also makes the development of generic web service client possible in application frameworks such as Gemstone [26]. It also means that portlets may be developed that uses the same application based web services. The MEME portlet is one such example [11], and the same MEME web service is also used by Gemstone, or any other web service clients if desired.

Community Scheduler Framework 4(CSF4) and CSFportlets

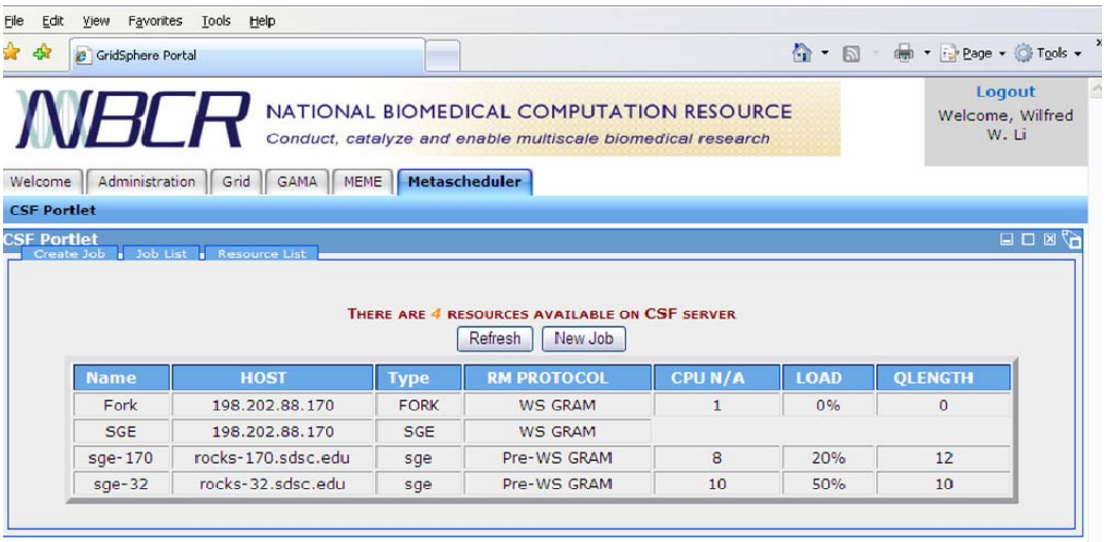

Figure 2. CSFportlets enables a visual interface to accessible resources, and eliminates the specification of specific resource for computation.

CSF4 is the first WSRF compliant metascheduler released as a contribution to the Globus Toolkit 4 (GT4). The latest release supports resource listing, job listing and job history, including support for the Sun Grid Engine (SGE) [27], TORQUE [28], and commercial schedulers such as LSF. It also leverages the Globus WS-GRAM for support of FORK, and Condor [29]. CSF4 supports user developed scheduling policy plugins [30], and supports proxy delegation for access of Gfarm filesystem [31].

A new prototype CSF4 portlet for GridSphere has been developed, providing a visual interface to CSF4 functionalities (Figure 2). Through either CSF4 command line interface, or the portal interface, a user no longer has to worry about where his application will be run, except for specifying the required resource characteristics, such as memory, number of processors, or application type (serial or MPI).

As CSF4 schedules jobs over Globus GRAM and local jobmanager-adapters, there are some additional overhead over the use of ‘globusrun’ or ‘globusrun-ws’. The average time it takes for a globusrun job to execute is about 12 sec, which is largely dependent on the local batch scheduler’s scheduling interval. The use of CSF4 adds about 30 sec on top of GRAM jobmanager, when the default scheduling interval of CSF4 is 20 sec. Therefore, besides the scheduling interval, which is a

user configurable option, the overhead is very small with the use of CSF4 through the portal interface.

Gfarm Grid File System

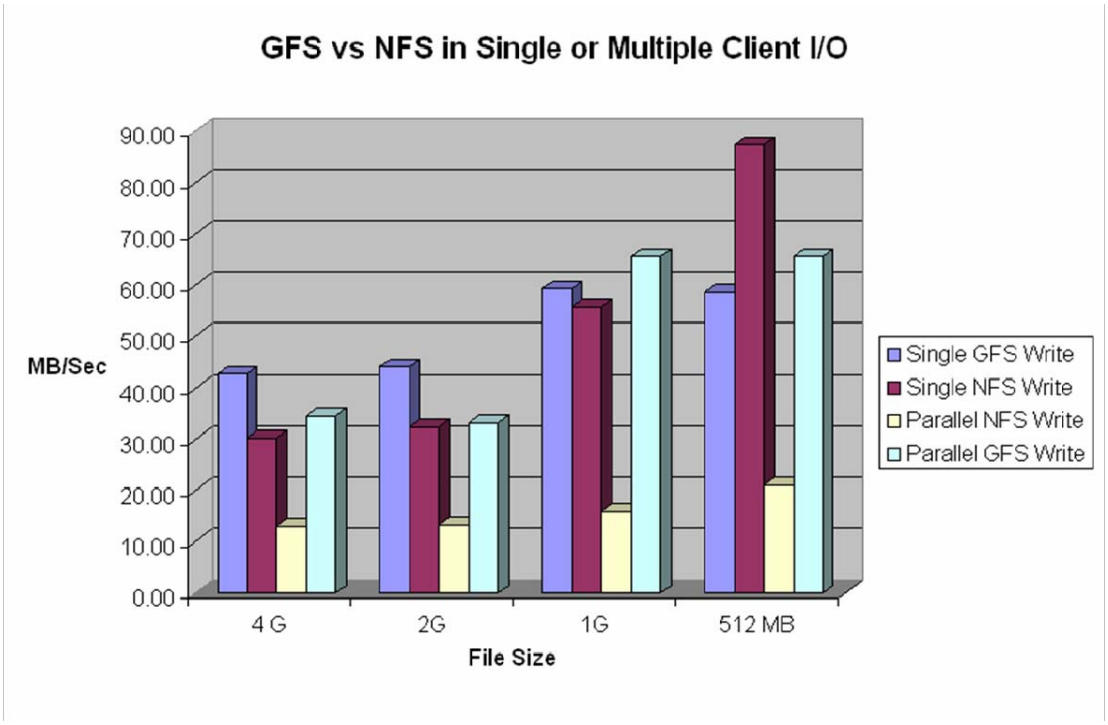

Figure 3. Gfarm-FUSE outperforms NFS as the number of clients increases. The machines have 2 GB of RAM, 3.0 GHz Dual Xeon processors.

Gfarm is developed as a petascale high performance file transfer and storage system [13], and it won the StorCloud Challenge Award during SC '05. As part of the collaborative activities under PRAGMA between the Biosciences, Resources and Data Grid working groups, we have used Gfarm to as a testbed for the deployment of existing bioinformatics applications [32,33].. While initial activities used the LD_PRELOAD environment variable and Syscall-hooking library to run pre-existing applications, the recent incorporation of FUSE [34] into the Linux kernel 2.6.14 has prompted us to use Gfarm-FUSE. This has eliminated the problems with glibc version incompatibilities, and resulted in a much more stable environment. The major advantage of using Gfarm-FUSE as a grid filesystem is two folds:

1. All applications may run without modifications through a unified filesystem view. This includes both serial and parallel applications. The corollary of this is that users do not need to learn about the grid, and may develop and compile applications within the Gfarm-FUSE file system using familiar tools.
2. Data intensive applications may take advantage of local filesystem I/O, reducing network bandwidth requirements.

We have done some performance measurements, and shown in Figure 3. The results show the average of 3 experiments using 'dd' to write files of 500 MB to 4 GB in size in 16k block size, either using a

single client or 3 clients in parallel. GFS outperforms NFS in single client mode except in the case of writing the 500 MB file, which is likely due to memory caching. Similar results are seen using 'iozone' (data not shown). However, when multiple clients are used, GFS-FUSE consistently outperforms NFS from 3 to 5 times. The advantage of Gfarm will become even more obvious as Gfarm-FUSE is using local file system I/O, the scalability isn't hindered by network bottlenecks as NFS is. Using Gfarm's built-in I/O measurement tools, one can easily achieve 220 MB/s read speed and 50 to 110 MB/s write speed with 6 clients. On the other hand, the network bottleneck applies if remote file transfer is required.

Running Grid-unaware Applications in Gfarm-FUSE

We have used MEME as an example to illustrate the use of Gfarm-FUSE. The Gfarm-FUSE system described here uses PostgreSQL 8.1.4 as the metadata server, a newly added feature in addition to LDAP. However, in our preliminary experiment, the file creation and registration using PostgreSQL is significantly slower than the use of LDAP. This suggests that the use of a relational database requires additional tuning on the persistence layer of Gfarm. For example, the untaring of MEME, creating more than 500 files, requires more than 1 min, vs 74 ms in NFS. Despite the aspect of slow file registration, we have successfully run the entire MEME web server, which includes MPI programs written in C, PERL scripts, and shell scripts, and Apache web server, using the Gfarm-FUSE filesystem. Based on the preliminary job execution studies, the execution and response time of different applications are comparable to NFS. For example, there is no noticeable difference in the Apache web server response time, the execution time for the sample dataset provided by MEME is only about 3% slower for MEME or MAST.

What are the advantages of using Gfarm-FUSE if there is a slight performance penalty for a single run?

1. The applications installed in one cluster on the Gfarm filesystem are also accessible on another cluster, with automatic support for architecture matching of binaries.
2. There is no need for stage-in or stage-out of application input or outputs. Gfarm keeps track of the data's locations, with metadata possible in future versions of Gfarm now that relational database support is available.
3. The ability to take advantage of local I/O for data intensive applications significantly improves the support for multiple clients over NFS.
4. The uniform location of applications in the Gfarm filesystem across sites simplifies job scheduling.

With the testbed developed for this prototype environment, a quarter Terabytes of total storage is achieved, and the automatic replication feature of Gfarm may be used to ensure duplicate copies are available in different clusters at distinct physical locations.

Summary and Discussion

The GridSphere portal framework [7] enables the reuse of functionalities exposed as portlets, but portable for different JSR168 compliant portal containers. In this work environment termed My

WorkSphere, a user can apply for access to compute resources through GAMA (Grid Account Management Architecture) portlet, which automates the user X509 certificate creation, signing, and proxy management [9]. A user who has never used the grid can start using the grid through the portal environment. The same X509 certificate may also be used directly, if desired, since it's a properly signed certificate by the organization running the GAMA server.

We have developed a prototype CSF4 portlet that allows users to submit jobs to CSF4 managed resources in a GridSphere portal through a web browser. A CSF4 server can forward the job request, along with the user credentials, to different heterogeneous clusters using the GRAM (Grid Resource Allocation & Management) Protocol. CSF4 uses the Java Commodity Grid Kit and GT4 (Globus Toolkit 4) delegation service to support full delegation of user proxies to both Pre-WS GRAM and WS-GRAM. The jobs are submitted to different clusters with local batch schedulers, such as SGE, dynamically based on metadata provided by MDS (Monitoring and Discovery System) or through FCFS (first come first serve)/round robin scheduling algorithm. CSF4 is capable of working with different local schedulers, like LSF, PBS (Portable Batch System), SGE and Condor, and via both GT4 WS GRAM and GT2 GRAM (Pre-WS GRAM).

For simple access to data and application deployment, Gfarm-Fuse (File System in User Space) is used to enable a familiar Unix environment in which compilation, installation of software is only required once per platform, and the filesystem is transparently mounted and unmounted without user intervention. We have compiled and deployed into Gfarm-Fuse data grid commonly used bioinformatics software such as MEME and BLAST, and other computational mathematics, biology and chemistry applications, such as FFTW, APBS, and AutoDock. We are beginning to utilize My WorkSphere as a production environment to further evaluate the performance and scalability of the system. In addition, we have made available the documentation on the configuration of the major components through the NBCR public wiki and started training users through the NBCR Summer Institute [35].

As more and more clusters, and grid of clusters, as virtual organizations, spring up, the ability to simplify the scheduling of tasks across virtual organizations becomes critical. The Open Science Grid, utilizes a condor-based system, with dedicated compute and storage resources, but not a global filesystem. Such systems use VOMS (Virtual Organization Membership Service) to automate the user creation processes on remote systems through the dynamic generation of temporary users during job execution. Once a job is finished, the user data is moved to specified data central, and may be accessed independently of the computing resource. Such systems provide a central resource broker which yields information about the resources available, and a command line interface for seasoned users in the high energy physics community. Biological applications have also been deployed to such systems.

Conclusion and Future Work

We are currently working towards several goals: 1) Use of CSF4 as a metascheduler for Opal based web services, with Gfarm as an data and application repository; 2) Support data uploads from client side, either from CSF portlet or an Opal based web service client; 3) Increase the performance of Gfarm metadata server, and metadata cache server; 4) Deploy the system for production use; 5) Prepare the release of the entire suite as Rocks roll(s) for easy setup and maintenance. 6) As a

TeraGrid Science Gateways portal, we are developing ways to interoperate with the TeraGrid, as well as other VO's within the OSG.

Acknowledgements

PWA and WWL wish to acknowledge PRAGMA as supported by NSF Grant No. INT-0216895 and INT-0314015; NBCR as supported by NIH NCRR P41 RR08605; CAMERA project as supported by the Moore Foundation. XW wish to acknowledge Jilin University under Grant No.419070200053 and Grant No.420010302338, National Natural Science Foundation of China under Grant No.60473099 and Outstanding Youth Science Foundation of Jilin Province under Grant No.20040119.

References

1. Hunter PJ, Borg TK (2003) Integration from proteins to organs: the Physiome Project. Nat Rev Mol Cell Biol 4: 237-243.
2. Oden JT, Belytschko T, Fish J, Hughes TJ, Johnson C, et al. (2006) Revolutionizing Engineering Science through Simulation.
3. Hunter PJ, Li WW, McCulloch AD, Noble D (2006) Multi-scale Modeling Standards, Tools, Databases for the Physiome Project. Computer: In Press.
4. Li WW, Arzberger PW, McCulloch A (2006) Developing End-to-End Cyberinfrastructure for Multiscale Modeling in Biomedical Research. CTWatch Quarterly. pp. 6-17.
5. Siebenlist F, Magaratnam N, Welch V, Neuman C (2004) Security for Virtual Organizations: Federating Trust and Policy Domains. In: Foster I, Kesselman C, editors. The Grid: Blueprint for a New Computing Infrastructure. 2nd ed. Amsterdam: Morgan Kaufmann.
6. GridShib (2006) GridShib: Integrating federated authorization infrastructure (Shibboleth) with Grid technology (the Globus Toolkit). http://gridshib.globus.org/.
7. Gridsphere (2004) GridSphere. http://www.gridsphere.org.
8. Novotny J, Russell M, Wehrens O (2004) GridSphere: a protal framework for building collaborations. Concurrency and Computation: Practice and Experience 16: 503-513.
9. Bhatia K, Chandra S, Mueller K. GAMA: Grid Account Management Architecture. 1st IEEE International Conference on e- Science and Grid Computing; Melbourne, Australia. pp. In Press.
10. Krishnan S, Stearn B, Bhatia K, Baldridge K, Li WW, et al. Opal: Simple Web Service Wrappers for Scientific Applications. International Conference for Web Services.
11. Li WW, Krishnan S, Mueller K, Ichikawa K, Date S, et al. Building cyberinfrastructure for bioinformatics using service oriented architecture. Sixth IEEE International Symposium on Cluster Computing and the Grid Workshops Singapore. pp. 39-46.
12. CSF (2005) Community Scheduler Framework. http://sourceforge.net/projects/gcsf/ 2005.
13. Gfarm (2004) Grid Data Farm. http://datafarm.apgrid.org/software/#download.
14. von Laszewski G, Foster I, Gawor J, Lane P (2001) A Java Commodity Grid Kit. Concurrency and Computation: Practice and Experience 13: 643-662.
15. COG (2004) Commodity Grid Kits. http://www-unix.globus.org/cog/.
16. Globus (2004) The Globus Alliance. http://www.globus.org.
17. ROCKS (2005) Rocks Cluster Distribution. http://www.rocksclusters.org.
18. Tomcat (2006) Apache Tomcat. http://tomcat.apache.org/ 2006.

19. PRAGMA (2004) Pacific Rim Applications and Grid Middleware Assembly. http://www.pragma-grid.net/.
20. TGSG (2006) Tera Grid Science Gateways. http://www.teragrid.org/programs/sci_gateways/.
21. Novotny J, Tuecke S, Welch V. An Online Credential Repository for the Grid: MyProxy. High Performance Distributed Computing (HPDC).
22. Krishnan S (2006) Opal Web Service Toolkit. http://nbcr.net/services.
23. Bailey TL, Williams N, Misleh C, Li WW (2006) MEME: discovering and analyzing DNA and protein sequence motifs. Nucleic Acids Res 34: W369-373.
24. Baker NA, Sept D, Joseph S, Holst MJ, McCammon JA (2001) Electrostatics of nanosystems: application to microtubules and the ribosome. Proc Natl Acad Sci U S A 98: 10037-10041.
25. Morris GM, Goodsell DS, Huey R, Olson AJ (1996) Distributed automated docking of flexible ligands to proteins: parallel applications of AutoDock 2.4. J Comput Aided Mol Des 10: 293-304.
26. Baldridge K, Bhatia K, Greenberg JP, Stearn B, Mock S, et al. GEMSTONE: Grid Enabled Molecular Science Through Online Networked Environments. Life Sciences Grid Workshop; Singapore. World Scientific Press.
27. SGE (2006) Sun Grid Engine. http://gridengine.sunsource.net/.
28. TORQUE (2006) TORQUE Resource Manager. http://www.clusterresources.com/pages/products/torque-resource-manager.php.
29. Litzkow M, Livny M, Mutka M. Condor - a hunter of idle workstations. Proceedings of the 8th International Conference of Distributed Computing Systems June 1988. pp. 104-111.
30. Wei X, Li WW, Tatebe O, Xu G, Hu L, et al. Implementing data aware scheduling in Gfarm using LSFTM scheduler plugin mechanism. International Conference on Grid Computing and Applications; Las Vegas. pp. In Press.
31. Wei X, Ding Z, Li WW, Tatebe O, Jiang J, et al. GDIA: A Scalable Grid Infrastructure for Data Intensive Applications. IEEE International Conference on Hybrid Information Technology, ICHIT 2006; Cheju Island, Korea. IEEE Computer Society. pp. Accepted.
32. Wei X, Li WW, Tatebe O, Xu G, Hu L, et al. Implementing data aware scheduling on Gfarm by using LSF™ scheduler Plugin. International Symposium on Grid Computing and Applications; Las Vegas, NV.
33. Li WW, Arzberger PW, Yeo CL, Ang L, Tatebe O, et al. Proteome Analysis using iGAP in Gfarm. The Second International Life Science Grid Workshop 2005; Grid Asia 2005, Singapore. World Scientific Press.
34. FUSE (2006) Filesystem in USEr space. http://fuse.sourceforge.net/ 2006.
35. NBCR.WIKI (2006) NBCR Public Wiki. https://nbcr.net/pub/wiki/index.php?title=Main_Page.