Creating the CIPRES Science Gateway for Inference of Large Phylogenetic Trees

Mark A. Miller, Wayne Pfeiffer, Terri Schwartz
San Diego Supercomputer Center
9500 Gilman Drive La Jolla CA, 92093-0505
{mmiller, pfeiffer, terri}@sdsc.edu

ABSTRACT
Understanding the evolutionary history of living organisms is a central problem in biology. Until recently the ability to infer evolutionary relationships was limited by the amount of DNA sequence data available, but new DNA sequencing technologies have largely removed this limitation. As a result, DNA sequence data are readily available or obtainable for a wide spectrum of organisms, thus creating an unprecedented opportunity to explore evolutionary relationships broadly and deeply across the Tree of Life. Unfortunately, the algorithms used to infer evolutionary relationships are NP-hard, so the dramatic increase in available DNA sequence data has created a commensurate increase in the need for access to powerful computational resources. Local laptop or desktop machines are no longer viable for analysis of the larger data sets available today, and progress in the field relies upon access to large, scalable high-performance computing resources. This paper describes development of the CIPRES Science Gateway, a web portal designed to provide researchers with transparent access to the fastest available community codes for inference of phylogenetic relationships, and implementation of these codes on scalable computational resources. Meeting the needs of the community has included developing infrastructure to provide access, working with the community to improve existing community codes, developing infrastructure to insure the portal is scalable to the entire systematics community, and adopting strategies that make the project sustainable by the community. The CIPRES Science Gateway has allowed more than 1800 unique users to run jobs that required 2.5 million Service Units since its release in December 2009. (A Service Unit is a CPU-hour at unit priority).

Categories and Subject Descriptors
H.3.4 [Systems and Software] Distributed systems**.**

General Terms
Management, Design.

Keywords
Science Gateway, TeraGrid, Cyberinfrastructure, Systematics, Phylogenetics, CIPRES.

1. INTRODUCTION
Assembling the Tree of Life is one of the most complex and difficult problems in modern science. The process of inferring phylogenetic relationships among all living organisms requires the collection and analysis of large amounts of data from as many species as possible. Recent advances in DNA sequencing technologies have dramatically increased both the number of species for which data are available and the amount of DNA sequence data available for many species. Moreover, the rate at which new sequence data can be obtained has increased dramatically, so it is now possible to rapidly acquire additional data to resolve relationships that are uncertain. The increased availability of sequence data provides an unprecedented opportunity to explore evolutionary relationships more broadly across all taxa, and more deeply to resolve relationships between closely related taxa. This information is critical to understanding the history of the relationships between organisms as well as the origins of biological diversity.

Analyzing large amounts of sequence data requires access to substantial computational resources. The algorithms used to infer phylogenetic relationships from sequence data are NP-hard (c.f. [1, 2]). Thus, while the increase in the amount of data allows investigators to ask deeper questions and obtain more definitive results, the requisite analyses can only be accomplished using computational resources that scale with these large data sets. The issue of resource access represents a significant problem for many research groups, as they must now move beyond local desktop resources to perform sequence alignment and tree inference analyses.

The Web Portal is an obvious choice for providing location-independent access to computational tools and resources for inference of computational trees. Popular web applications such as the Biology Workbench [3], Anabench [4], NC Bioportal [5], and MIGenAS [6] provide access to molecular biology tools and data in an integrated environment, while projects such as Morphbank [7] and Morphobank [8] provide collaborative data sharing workspaces for systematics researchers. Because tree inference codes are computationally demanding, they are typically not offered in public web portals, and when they are available, the time allocated for analyses is inadequate for most research data sets.

In an attempt to meet community needs for access to tools and resources for inferring evolutionary relationships, the CIPRES (CyberInfrastructure for Phylogenetic RESearch) project created a prototype web portal. The CIPRES Portal V 1.0 permitted users to run the community tree inference tools GARLI [9], RAxML [10], PAUP [11], and MrBayes [12], both as standalone tools, and with

the tool Rec-I-DCM3, a disc covering method to speed and improve inference of large trees [13]. The demand for Tree Inference analyses on the CIPRES Portal quickly exceeded the computational resources available to the project, pointing to a need to redesign the CIPRES Portal for greater scalability while minimizing costs for the sake of sustainability.

The TeraGrid Science Gateway program[14] offers a solution that is well suited for the present use case. It is designed to link domain science communities to a scalable and sustainable set of resources on the TeraGrid. The Science Gateway program allows the community to create a domain work environment where researchers can concentrate on domain problems without facing the complexities of deploying jobs on distributed resources and without the costs of acquiring and managing high-end computational resources.

This paper describes the redesign and reimplementation of the CIPRES Portal as the CIPRES Science Gateway (CSG), a scalable, sustainable resource for systematics and evolutionary biology.

2. MOTIVATION

As implied by its name, the CIPRES project's goal was to create tools and infrastructure to meet the computational needs of a well-established phylogenetic research community. For this reason, development of the CIPRES Portal was user-focused and community-centric. In the three years following the release of the CIPRES Portal V 1.0 (in May, 2007), we worked closely with the target community to understand how best to meet user needs for access to computational resources. User requirements were gathered from conversations at professional meetings and through issue reports/feature requests submitted to the CIPRES Portal. The CSG design as well as implementation of its capabilities has been dictated exclusively by user feedback.

The typical use case for phylogenetic research involves collecting specimens in various locations around the world for 6-12 months, obtaining DNA sequence data from these specimens, and then analyzing the DNA sequence data to infer evolutionary relationships among these specimens. The data analyzed may represent a few or many species, one or more genes, and long or short DNA (or protein) sequences. The focus may be on resolving relationships within a small group (or clade), or placing an entire clade accurately with respect to other clades in the larger Tree of Life.

The use case described above requires easy access to high-end computational resources, but has no need for public database access. Each user creates their own data sets to analyze. Users manipulate their data sets as plain text files in one of a handful of community formats. The data sets are typically small (less than 5 MB), and can be easily exchanged even over fairly slow network connections. Each data set has persistent value to the individual user or user group and is proprietary until published, so data sharing across the community is not desirable prior to publication.

While users require computational resources to run tree inference analyses, they have not requested additional tools to display and edit the resulting trees (i.e. post-tree analysis). These tools have low computational requirements and typically feature sophisticated graphical user interfaces, so they are well suited to the desktop environment available to most users. Moreover, while there are only a handful of tools for inferring phylogenetic trees, many software packages are available for displaying, analyzing, and editing phylogenetic trees. Each user's environment for post-tree analysis reflects the personal choices, as well as preferences of their home laboratory and the current practices of their sub-specialty of biology or biomedicine.

Based on the use case just described, we designed and implemented the CIPRES Science Gateway to meet the following design objectives:

1. Provide simple browser-based access to community tree inference codes.
2. Store uploaded files and records of job runs in a persistent, login-protected area.
3. Provide interfaces that are tailored to the user's expertise, from full command line options to simpler interfaces with fewer choices
4. Provide easy access to the fastest available tree inference codes run on scalable, sustainable computational resources.
5. Make it possible to add new tools quickly and update existing tools and interfaces as new releases appear.
6. Minimize job loss due to system/hardware errors, and elegant recovery from failures.
7. Minimize job loss due to file format/file translation errors.
8. Distribute jobs transparently across all available resources.
9. Provide programmatic methods to deploy jobs from and deliver results to third party environments for post-tree analysis.

The approaches used to meet objectives 1-6 are described in Section 3. Community usage and management of the resource in its current state is described in Section 4. Plans to address design objectives 7-9 are described briefly in Section 5.

3. ARCHITECTURE/IMPLEMENTATION

3.1 Basic Architecture

The CIPRES Portal V 1.0 was created using a set of CORBA-based software libraries from the CIPRES Project. These libraries were adequate to create a web application that could check uploaded file formats and deliver files to community tree inference codes run on a computational cluster. However, adding new command line options for each program was difficult, and the libraries did not provide a mechanism for adding login-protected user areas, so the architecture of CIPRES Portal V 1.0 was not adequate to meet the requirements of our project.

As a result, the web application for the CIPRES Science Gateway (CSG) was based on an entirely different architecture called the Workbench Framework (WF) [15]. The WF is a software development kit (SDK) designed to generically deploy analytical jobs and database searches to a generic set of computational resources and databases. The WF contains modules to manage submission of jobs to analytical tools on various computational resources and modules to manage queries to data resources. A schematic of the WF architecture used for the CIPRES Science

Gateway is shown in Figure 1. The modules in the WF are as follows:

Broker Module. The Broker Module provides access to all application-specific information in a Central Registry This Registry contains information about all data types required as input and output for each application along with the formats accepted by each tool. Concepts and concept relationships are formulated in XML files and read by Central Registry API-implementing classes. By defining tools and data types in a single location within the application, new tools and data types can be added with no impact on the functioning of the application outside the Registry.

User Module. The User Module manages all user-initiated activities. This module passes user-initiated queries and tasks from the interface to the executive portions of the infrastructure via data and task management modules. It also stores all user data and task information in a MySQL database (although any conventional RDBMS can be used). A user management module supports individual user roles, permitting the assignment of individual user accounts, the sharing of data between accounts, and selective access to tools and data sources that may be proprietary.

Tool Module. The SDK design includes a Tool Executive Module that manages the translation of tasks submitted by users into command lines and submission of the command line strings along with user data for execution by appropriate compute engines. As part of this process, the Tool module handles data formatting for jobs, and job stagingaging. It also keeps track of which tools can be run on which computational resources, and the status of those resources. The design allows great flexibility in determining what assets the CSG can access for job execution. Computational resources can be added through editing the tool resource configuration file, and the application can send command lines and receive output via essentially any well defined protocol (e.g. Unix command line, web services, ssh, drmaa, gram, gsissh, etc.).

Presentation Layer. The CSG Presentation Layer accesses SDK capabilities through the J2EE front controller pattern [16] which involves only two Java Classes. As a result, the WF is neutral with respect to interface access. We sought a presentation layer that would provide lightweight access through a web browser and that would preserve flexibility for alternative access routes. We investigated a range of architectures, from web-services based architectures to the Enterprise Java Beans (EJB) framework. We adopted an architecture based on Linux, Apache Tomcat, MySQL, and Java Struts2. This open source software stack provides the capabilities needed to manage the web application in a sustainable manner and provides sufficient structure and stability. The Presentation Layer supports access by browser (thin) clients and will support programmatic access via SOAP and ReST services.

The browser interface is based on the look and feel of popular e-mail clients and supports data and task management in user-created folders. The interface allows users to create a login-protected personal account. Registered users can store their data and records of their activities indefinitely. Alternatively, the tools can be used under a guest account, but data in guest accounts is lost once the user migrates away from the web site. The

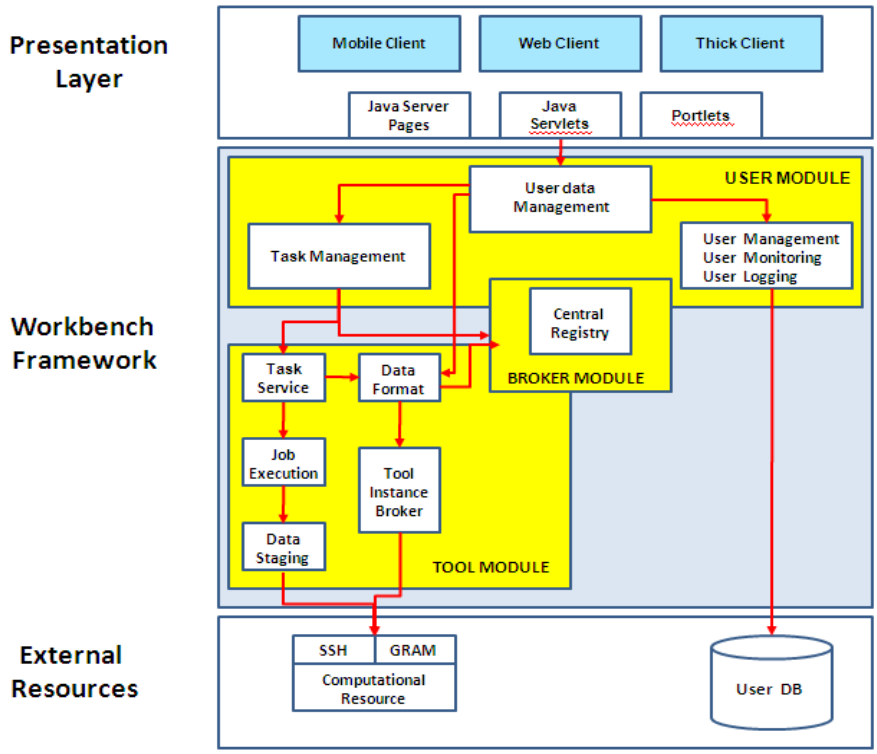

Figure 1. Schematic Diagram of the CIPRES Science Gateway architecture

application currently provides access to approximately 20 command line tools. Uploaded user data is checked for format. Users can also manually specify data types and formats.

External Resources. The generic design of the WF architecture supports access to a wide variety of computational resources and databases, whether local or remote. Access can be accomplished through a combination of individual mechanisms, including ssh, GRAM/Globus, SOAP, ReST services, etc. At present, parallel tools run on TeraGrid resources, and serial tools are run on the fee-for-service UCSD Triton Resource.

3.2 Command-Line Tool Scalability

The WF achieves scalable implementation of new tools by taking advantage of the XML specification of the Pasteur Institute Software Environment (PISE) for command line tool interfaces [17]. PiseXML files specify point-and-click interfaces and use embedded Perl scripts to transform user input into Unix command lines. The WF interface generation tool uses the instructions in the XML files to create .jsp pages that collect user input and assemble the input into command line strings that can be delivered to individual tools. This strategy of reusing PiseXML files offers scalability from several perspectives:

- Developers can expose, test and edit interfaces quickly.
- All command-line options of any tool can be exposed easily.
- Hierarchical logic (preconditions) and control parameters can be exposed in the .jsp interface form using Javascript. Functionally, this means click boxes and entry boxes in the interface are activated or deactivated based on choices the user makes, and that the interface can prevent inappropriate entries and provide users with robust error messages.
- The presentation is flexible, since all interfaces in the application can be modified at once by changing the module that interprets the XML file. This means new presentation paradigms can be adopted simply by modifying the .jsp creation software, and new features can be introduced through modification of the XML standard. This technology allows us to sustain the evolution of the resource to meet the needs of a dynamic community.
- The XML specification we use is closely related to the PISE and Mobyle XML [18] specifications, so interfaces created for either of these projects can be incorporated into the CSG portal, and vice versa.

3.3 Access to the Fastest Available Codes

The CIPRES Science Gateway offers three parallel codes on TeraGrid: MrBayes, RAxML, and GARLI. We conducted extensive benchmarking for these codes to determine the optimal configuration (in terms of processes and threads) for runs of various types on each TeraGrid resource. The portal interface was then adapted to automatically configure the correct number of processors and threads based on user-entered problem specifications. The codes currently offered by CSG for use on TeraGrid are described further below:

MrBayes: The initial TeraGrid portal release offered the standard version of MrBayes (3.1.2), which is parallelized with MPI in a coarse-grained fashion across the different run-chain instances of a single analysis. In February 2010, the standard version was replaced with a new hybrid MPI/OpenMP version of MrBayes 3.1.2 [12] in the CSG interface. This version adds OpenMP code to exploit fine-grained parallelism within each run-chain instance. The OpenMP code allows jobs to be divided over a larger number of processors, therefore reducing the clock time required for individual job runs. To our knowledge, this code is the fastest version of MrBayes available anywhere. MrBayes hybrid code jobs execute on up to 32 cores.

RAxML: The initial portal release offered RAxML V. 7.2.3. This version confined runs to a single 8-core node. To improve the flexibility of the code, MPI code was added, creating a hybrid MPI/Pthreads version [19]. This innovation makes RAxML more scalable for three common types of analyses:

1. Multiple maximum likelihood searches on the same data set, starting from different initial trees.
2. Multiple bootstrap searches, which are maximum likelihood searches on data sets obtained by randomly resampling the columns of the multiple sequence alignment.
3. A comprehensive analysis that combines the two preceding analyses.

The new hybrid code parallelizes over the number of searches, so multiple nodes can be used. The hybrid code was provided via the CIPRES Portal in February of this year as RAxML 7.2.6. This code is the fastest version of RAxML available anywhere. The hybrid code executes on up to 40 cores.

GARLI: The Portal offers GARLI 1.0, which includes an MPI implementation. There are two conditions for MPI use: when the number of searches conducted (nsearch) is increased in parallel with the number of cores employed (this is common, but not a default situation) and when bootstrapping is employed (this is part of routine analysis). In both situations, the parallel efficiency is very high (77-94%), because these situations are embarrassingly parallel and can be scaled efficiently to a large number of cores. GARLI executes on up to 100 cores.

3.4 Access to scalable compute resources

The CIPRES Portal V 1.0 deployed serial jobs to a 16-node/128-processor computational cluster purchased for the CIPRES Project. While serial runs were efficient, they increased the wall time for each job. Moreover, to insure fair access to all users, jobs were restricted to 72 hours. This imposed a strict upper limit on the size of jobs that could be run through the portal, which in turn prevented the portal from meeting the needs of users with very large data sets.

The CIPRES Science Gateway offers parallelized MrBayes, RAxML, and GARLI codes described above on the Abe and Lonestar TeraGrid clusters. These clusters each have more than 1,000 nodes with 8 and 4 cores per node, respectively, which provides a huge improvement in scalability over the original CIPRES Cluster. Abe also allows run times of up to 7 days, while Lonestar allows runs of 2 days.

3.5 Minimal job loss from external system errors

In the current use case, many large jobs require 7 days to execute, and such jobs often remain in a resource queue for several days

before beginning to execute. As a result, the job submission and monitoring mechanism used by CSG must be robust to scheduled and unscheduled outages of external compute resources, pausing of resource schedulers, and outages of the web application. Our initial implementation employed a system where a unique process was used to monitor each job created. Not only did this present scalability problems as the number of submitted jobs grew, but loss of a monitoring process for any reason caused the application to lose track of the job's progress. When this happened, job results could not be returned automatically to the user. Jobs that had a combined queue and run time of over 14 days suffered a significant risk of loss. In the initial release of the CIPRES Science Gateway, up to 15% of all jobs could be lost due to resource and application outages under normal use conditions.

To mitigate this issue, a new submission mechanism was developed. Instead of having one process monitor each job, the web application stages the input, submits the job, and creates an entry in a Running Task table. The CSG web application handles job submission by creating a job submission script, staging it to the TeraGrid host along with the job's input files, and doing a Java runtime exec of Globus "gsissh" to remotely run commands on the TeraGrid host that submit the job script to the scheduler. File staging is handled via the Java CoG Kit GridFTP API.

A *curl* command in the job submission script notifies a servlet in the web application when the job finishes, and the servlet marks the entry in the Running Task table as DONE. However, when jobs time out or are terminated abnormally or the CSG web application is down, the "curl" notification will not occur. To compensate, a daemon process named "checkJobsD" contacts Teragrid hosts (via "gsissh qstat" or "gsissh bjobs", for example) to see which jobs have finished. A separate daemon named "loadResultsD" polls the Running Task table for jobs that are DONE and transfers the results from the execution host to CSG's database.

This approach is more scalable, because a separate process is not required to monitor each remote job and because the web application and the daemons that monitor jobs and transfer results can be run on separate hosts. The new mechanism uses fewer socket connections, has a lower CPU requirement, and requires less memory. The approach is more robust because it does not lose track of jobs and results when connectivity to TeraGrid machines is temporarily lost, when the web application goes down, or a TeraGrid host goes down.

4. MANAGING PORTAL USAGE

In December 2009, the tools for accessing MrBayes, RAxML, and GARLI on TeraGrid resources were released in the CSG. The number of unique users of these tools increased from 90 to 280 per month between December 2009 and August 2010. In each month, an average of 100 new users ran jobs for the first time. In July and August, approximately 180 users returned to run additional jobs. In a recent survey, 84% of 187 respondents reported that the Gateway allowed them to perform work that would be difficult or impossible to accomplish in any other way [24].

Figure 2 shows the rate of job submission and Service Unit (SU) consumption for the Gateway (an SU is a CPU-hour at unit priority). The number of jobs run per month has increased 4-fold since December 2009, while the number of SUs used per month has increased approximately 10-fold. The CIPRES Science Gateway is currently the most active TeraGrid gateway, as measured by both the number of users and the number of Service Units consumed per month. Importantly, the rate of SU consumption is still increasing, and it could be several months before we know if a plateau has been reached.

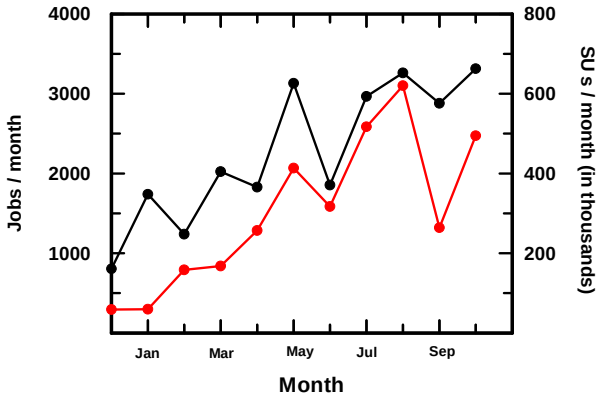

Figure 2. Jobs Run (black) and Service Units consumed (red) per month by the CIPRES Science Gateway Dec, 2009 – Aug, 2010.

The upward trend demonstrates the success of the CSG as a resource for phylogenetic tree inference, but it also presents a challenge. The TeraGrid annual allocation for the CSG is 2.7 million SUs, and at the present rate of consumption that allocation will be consumed in less than 6 months. Clearly a management plan is required to control the use of resources, and to ensure fair access to as many users as possible.

We examined the rate of consumption for individual users between April and August, 2010. Table 1 shows user accounts that were active in this time period, binned according to monthly SU consumption. The results show that 75% of users consumed less than 100 SUs per month. In the aggregate, these users consumed less than 20% of all SUs. The jobs run by these users do not represent heroic computations, but users relate that access to TeraGrid for jobs of this magnitude makes it possible to complete analyses overnight that might require 10 days on their local resources.

Table 1. Monthly Per User consumption of SUs between April and August, 2010.

SUs /month	Number of Users	% total SU
< 100	536 (75.3%)	19
100 - 500	86 (12.1%)	9
500 - 2000	53 (7.5%)	10
2000 - 5000	24 (3.4%)	17
5,000 - 10,000	9 (1.3%)	14
> 10,000	3 (0.4%)	31

At the other end of the spectrum, three power users consumed 31% of all SUs used between April and August; and the top 12 users consumed 45% of all SUs. This level of consumption by individual users is not sustainable in the current TeraGrid Science Gateway community allocation model. It is therefore necessary to develop and implement a policy that provides access to the largest possible user group, while at the same time ensuring responsible use of TeraGrid resources.

Based on the usage demographic shown in Table 1, the following policy was devised for managing use of TeraGrid resources through the CIPRES Science Gateway. First, the Gateway will continue to allow any user to create an account and deploy phylogenetic analyses on the CIPRES Science Gateway. This is in keeping with the "open access" spirit of the TeraGrid Science Gateway program. To insure usage consistent with the "equal access" spirit of TeraGrid, any user who consumes more than 2% of the total community allocation for any given month will be contacted and asked to establish a personal account (see below). Users who consume more than 3% of the total community allocation in a month will lose the ability to submit new jobs to the TeraGrid, pending establishment of a personal account.

The mechanism for establishing a personal account is by requesting a personal or project allocation from the TeraGrid Resource Allocation Committee (TRAC). Typically a user would request up to 50,000 SUs as a development allocation. The user will provide this information to the web application through a web form, and jobs run from their account will be charged to their personal allocation. In the event that more than 50,000 SUs are needed, the user can request a research allocation from the quarterly TRAC competition. The goal of this policy is to allow users to continue to access computational resources through a simple and familiar interface, while at the same time requiring that high-end users are subject to peer review and remain accountable for use of their computational time. The TRAC peer review process insures that when very large amounts of resources are used to address a scientific question, the approach used to address that scientific question has been validated, and that the use of these resources follows best known practices of TeraGrid users for efficiency.

Implementing the usage policy described above requires a set of administrative tools. The functionalities required include: tools to disable submissions from a user account, tools to monitor SU consumption by individual users, and tools that allow users to charge their usage directly to a personal allocation obtained from TRAC. The ability to disable submissions from specific user accounts was accomplished simply by adding a column for the property canSubmit to the User data table, where the property canSubmit is required for job submission. Users who exceed their allowed monthly usage will have their canSubmit privileges suspended until the end of the current calendar month.

Usage by individual accounts is monitored by merging job submission records from the CSG Web application (which associate the TeraGrid resource scheduler jobid with the user account) with records from the TeraGrid job database (which associates the TeraGrid resource scheduler jobid with the SU charge). This is currently done manually, however, we are creating tools to perform this task automatically, making it possible to monitor usage in real time, and to create automatic triggers to impose the usage policies described above.

The ability to charge to a personal user allocation rather than to the CSG community allocation can be accomplished in a straightforward way. A web form will be provided that allows the user to associate their personal TeraGrid allocation identifier with their CSG account. Jobs submitted from their CSG login will then be charged to their personal allocation instead of the CSG community allocation. Implementation of this capability is in progress.

5. FUTURE WORK

In view of the heavy use of the CSG to run large computational jobs, our future work will focus on improving the efficiency of the resource and on simplifying the user experience. The key priorities in the regard are design goals 7-9 in Section 2.

5.1 Minimize job loss due to file format/file translation errors.

Approximately 5% of all job submissions fail each month because users upload an input file in a format that is not correct for their selected code, or because the input file contains a formatting error. To improve the efficiency of user submissions we plan to create infrastructure to insure that each uploaded dataset is formatted correctly and that the dataset selected for a given job is appropriate for the code selected. The WF contains a module that detects the format of uploaded files, and the WF Central Registry is designed to associate each code with its accepted input and output formats. Future work will focus on further development and implementation of these capabilities for the CIPRES Science Gateway so that users will be warned 1) if they upload a data set with a formatting error, and 2) if they attempt to analyze a data set in a format that is not understood by their chosen code.

5.2 Distribute jobs transparently across all appropriate resources

At present the CSG runs each code on a single TeraGrid machine. This can be problematic when the resource used is under heavy load or is offline for maintenance. Adding a meta-scheduling capability to the CSG would allow user submissions to be deployed automatically on the most appropriate resource, based on availability and current traffic levels. We plan to add this capability using a tool such as SWARM [20], a job scheduling Web service framework developed specifically for use in TeraGrid Science Gateway applications.

5.3 Simplify Job Submission/Results Retrieval

Our design goal for the CSG is to integrate its functionalities as seamlessly as possible into the workflow of all users. The CSG user community is diverse, and each user has a specific set of tools for creating data matrices and a specific set of tools for post-tree analysis. Various user groups conduct analyses in web applications such as Morphobank [8] or Morphbank [7], via locally installed interfaces (e.g. Mesquite [21] or Topali [22] a new simple graphical user interface for RAxML [23]). We plan to provide programmatic access to CSG capabilities so individuals using one of these community tools can deploy jobs to the CSG and receive their results within their normal working environment. This will be less cumbersome than submitting jobs

to the CSG via a web browser. The current presentation layer for the CSG is designed to support access via both thick and thin clients, and programmatic access by ReST services is supported.

6. CONCLUSIONS

The CSG has accomplished the primary goal of the Science Gateway program. It provides a significant and growing population of domain scientists with access to fast tree inference codes and scalable TeraGrid resources. The overhead of accessing these resources would have been prohibitive for the vast majority of these users. In a recent user survey 96% of respondents state that the CSG benefits their research in a tangible way, while 84% indicate that it permits them to do research that would be difficult or impossible to conduct in any other way. The success of the CSG also raises some issues as well. The magnitude of user demand for computational resources is already stretching the limits of what can be managed in the context of the TeraGrid Science Gateways program. The challenge will be to develop policies and techniques to manage this demand, so successful Science Gateway projects can be both scalable and sustainable, while using the national cyberinfrastructure in a responsible way.

7. ACKNOWLEDGEMENTS

This work was supported in part by National Science Foundation awards NSF EF 01-31648 (MM and TS) and OCI-932251 (WP) as well as National Institutes of Health award 5R01GM073931 (MM). The work was also supported in part by National Science Foundation funding of TeraGrid computational resources at the National Center for Supercomputing Applications and the Texas Advanced Computational Center. The authors thank Nancy Wilkins-Diehr, Paul Hoover and Lucie Chan for helpful discussions.

8. REFERENCES

[1] Morrison, D. A. 2006. Multiple Sequence Alignment for Phylogenetic Purposes. Australian Systematic Biology 19, 479-539.

[2] Morrison, D. A. 2006. Phylogenetic analyses of parasites in the new millennium. Advances in Parasitology 83, 1-124.

[3] Subramaniam, S. 1998. The biology workbench - A seamless database and analysis environment for the biologist. Proteins-Structure Function and Genetics 32, 1, 1-2.

[4] Badidi, E., De Sousa, C., Lang, B. F., Burger, G. 2003. AnaBench: a Web/CORBA-based workbench for biomolecular sequence analysis. BMC Bioinformatics 4, 63 - 72.

[5] The NC BioPortal Project [http://www.ncbioportal.org/]

[6] Rampp, M., Soddemann, T., Lederer, H. 2006. The MIGenAS integrated bioinformatics toolkit for web-based sequence analysis. Nucl Acids Res 34, (Web Server issue), W15–W19. .

[7] Erickson, G., Jörgensen, P., Jörgensen, C., Riccardi, G., Ronquist, F., van Engelen, R. 2007. Morphbank: Web Image Database Technology for Comparative Morphology and Biodiversity Research http://www.morphbank.net/

[8] O'Leary, M. A., Kaufman, S. G. 2008. MorphoBank 2.7: Web application for morphological phylogenetics and taxonomy. http://www.morphobank.org.

[9] Zwickl, D. J. 2006. Genetic algorithm approaches for the phylogenetic analysis of large biological sequence datasets under the maximum likelihood criterion. The University of Texas at Austin.

[10] Stamatakis, A. 2006. RAxML-VI-HPC: Maximum likelihood-based phylogenetic analyses with thousands of taxa and mixed models. Bioinformatics 22, 21, 2688-2690.

[11] Swofford, D. L. 1999. PAUP*. Phylogenetic Analysis Using Parsimony (*and Other Methods), version 4.0 http://paup.csit.fsu.edu/

[12] Ronquist, F., Huelsenbeck, J. P. 2003. MrBayes 3: Bayesian phylogenetic inference under mixed models. Bioinformatics 19, 12, 1572-1574.

[13] Roshan, U., Moret, B. M. E., Williams, T. L., Warnow, T. 2004. Rec-I-DCM3: A Fast Algorithmic Technique for Reconstructing Large Phylogenetic Trees. In: IEEE Computational Systems Bioinformatics Conference (CSB) 98-104.

[14] Wilkins-Diehr, N., Gannon, D., Klimeck, G., Oster, S., Pamidighantam, S. 2008. TeraGrid Science Gateways and Their Impact on Science. Computer 41, 11, 32-41.

[15] Rifaieh, R., Unwin, R., Cleary, J. M., Brown, C. J., Miller, M. A. 2007. SWAMI: The Next Generation Biology Workbench http://www.ngbw.org

[16] Core J2EE Patterns - Front Controller [http://java.sun.com/blueprints/corej2eepatterns/Patterns/FrontController.html]

[17] Letondal, C. 2001. A Web interface generator for molecular biology programs in Unix. Bioinformatics 17, 1, 73-82.

[18] Neron, B., Tuffery, P., Letondal, C. 2005. Mobyle: a Web portal framework for bioinformatics analyses. In: NETTAB 2005.

[19] Pfeiffer, W., Stamatakis, A. 2010. Hybrid MPI/Pthreads parallelization of the RAxML phylogenetics code. In: Ninth IEEE International Workshop on High Performance Computational Biology (HiCOMB 2010) Atlanta.

[20] Pierce, M., Pallickara, S. 2008. Swarm: Scheduling large-scale jobs over the loosely-coupled hpc clusters. . In: IEEE Fourth International Conference on eScience: 285-292.

[21] Maddison, D. R., Maddison, W. P. 2007. Mesquite: A Modular System for Evolutionary Analysis http://mesquiteproject.org/mesquite/mesquite.html

[22] Milne, I., Lindner, D., Bayer, M., Husmeier, D., McGuire, G., Marshall, D. F., Wright, F. 2008. TOPALi v2: a rich graphical interface for evolutionary analyses of multiple alignments on HPC clusters and multi-core desktops Bioinformatics 25, 1, 126-127.

[23] Silvestro, D., Michalak, I. 2010. RAxML Graphical User Interface http://sourceforge.net/projects/raxmlgui

[24] Miller, M.A. 2010. CIPRES Science Gateway survey results. http://www.phylo.org/tools/survey2.html