An Architecture for Scaling NVO Services to TeraGrid

Roy Williams, Caltech
Bob Hanisch, STScI
Joe Jacob, JPL
Ray Plante, NCSA
Alex Szalay, JHU

The term "cyberinfrastructure" has been adopted by the US National Science Foundation to mean "advanced computing engines, data archives and digital libraries, observation and sensor systems, and other research and education instrumentation [linked] into a common framework". One of the largest awards in this program is the TeraGrid, a linkage of large supercomputer centers based on the Globus software. Another cyberinfrastructure program is the National Virtual Observatory, a linkage of astronomical data publishers into a service-oriented framework.

There are different philosophies behind the TeraGrid and the NVO architecture. This note explains a proposed service-oriented architecture for TeraGrid nodes that is an attempt to bridge these ways of working, and a prototype instantiation at Caltech.

TeraGrid began with the big supercomputer centers, with an emphasis on a named individual doing "big-iron" computing that uses massively parallel machines and stores the results on massive data systems. TeraGrid usage traditionally begins with an account request that may require a 15-page proposal. However, there is now an attempt to broaden access with the *science gateway* thrust. A community may set up a remote web server – not a TeraGrid resource – that speaks the language of the user community, then starts and monitor jobs on TeraGrid. Alternatively, community developers may deploy science services directly on TeraGrid resources, through the recent installation of Application Servers such as GT4 and Clarens.

The NVO began with the idea of publishing and exposing astronomical data through simple, anonymous protocols. Anyone who has data can set up a web server, put simple, standardized CGI scripts in place, and register those with the VO so that others can find them. As this international framework has matured, SOAP over HTTP has complemented (but not replaced) CGI scripts. The NVO is introducing *compute services*, which combine a security model and the idea of batch jobs. The former is needed if big resources are to be used, although the NVO is sensitive to the open-access expectations of an astronomy research community that does not want to learn about X.509 certificates. A batch job is thus interpreted as a sequence of service calls. Instead of immediate results, the user gets a service to monitor the job and collect results. The job runs in the background and writes results into a temporary storage area called *myspace*, which acts as the repository of state for the job.

Thus the TeraGrid and NVO are approaching a middle ground from opposite poles. TeraGrid, working with a dozen portal-style science gateway projects, is exploring a variety of ways to support interaction between portals and TeraGrid resources, with focus

on authorization and how to provide ways for gateways to execute their services and applications on TeraGrid resources.

In this note, we try to fill in some details about how gateway developers can run compute services on national cyberinfrastructure like TeraGrid. This means that non-sysadmins deploy and control services on the TeraGrid infrastructure, with all the power and flexibility, but also the vulnerability to attack. For organizations that have chosen SOAP and HTTP as their service fabric, it means that services are deployed in the usual way, through a WAR file or other deployment protocol.

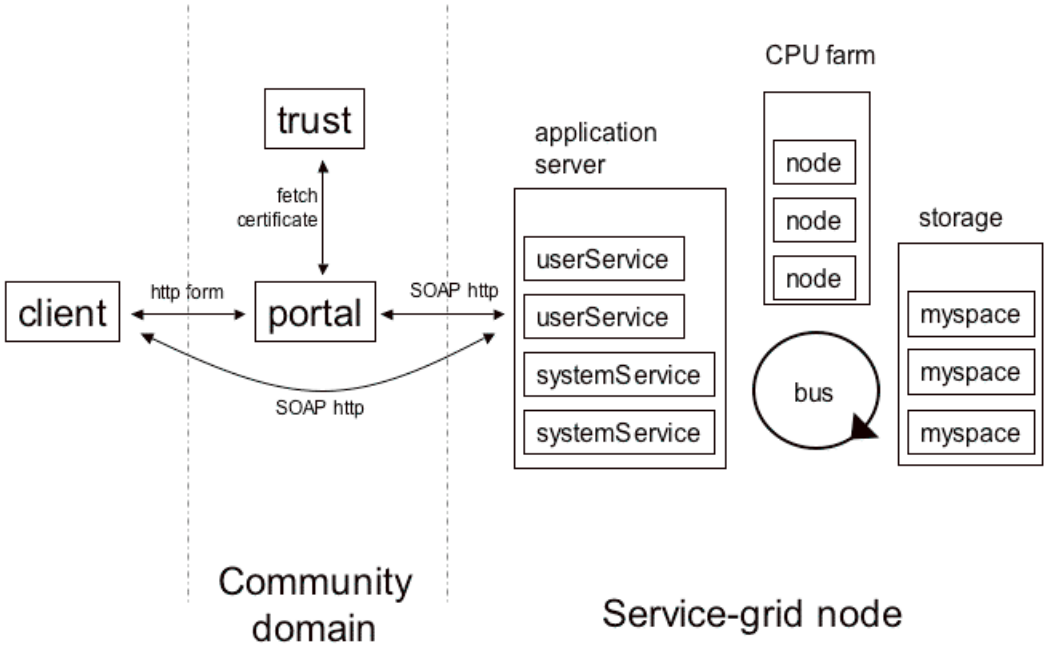
When a request comes to a deployed service, it must be accompanied by a certificate. NVO is building a "graduated security" model to allow easy access for beginners, but also powerful access for sophisticated users. The service container is able to understand the certificate in the context of the known user accounts ("gridmap file"), so that power users have powerful resources, but anonymous users are not rejected.

We believe that security can be maintained even though non-sysadmins would be able to deploy services; and we believe that scientific return of the national cyberinfrastructure can be greatly increased by this architecture. The architecture obviates the need for an independent server between user and TeraGrid, providing a closer connection that is more flexible, higher bandwidth, and lower latency than can be done with the remote-handling approach. What is needed are some reasonable rules and agreements – e.g., only power users can deploy services, critical code should be audited (as a CGI script should be), and TeraGrid sysadmins should enclose services in bulkheads (e.g., *chroot-jail*) to restrict the effect of any malicious or naïve usage. Policy issues concern, for example, what code can be deployed, what review it must undertake, and what privileges it is given. Identifying and resolving these issues is a priority. Some of these issues may be easier with standardized software for service deployment, monitoring, and management.

In this model (see figure), the Application Server (AS) is a web-service container that accepts HTTP connections that may be SOAP messages, and may have a certificate attached. Some of the available services are system services, controlled by sysadmins, for example, allowing exposure of data as web pages, or service deployment. Other services ("userServices") that could perhaps make a mosaic of astronomical images or mine a database of galaxies are deployed by users or by projects on behalf of the community. The AS would run on a single node, and is not meant to actually run computation, but rather to farm out jobs to a large farm of processing nodes. One candidate for AS is the GT4, offering excellent security, integrated registry, dynamic service deployment and powerful authorization framework. In the NVO and High Energy Physics community, there is a lot of interest in Clarens, a standalone web server with security extensions based on Apache SSL. Clarens is open-source from Sourceforge, and has Java and Python APIs.

Clients can connect to the AS directly through an API that is automatically published when a service is deployed (WSDL file), and send a certificate with a request. In this scenario, clients may be human users or other services. Alternatively, a human user can use a community web portal, which can fetch a certificate from a certificate store, and form the

request from user selections on a web page. A user can also make an anonymous request (no certificate), which can run in a very restricted way. Note that the web server that is behind this portal is concerned only with building SOAP requests rather than Globus requests.



The AS will launch jobs on the CPU farm, utilizing big data and big bandwidth to achieve significant results. A *myspace* area is first allocated for the job (in a regularly purged scratch space), and results and logs written there. The user has been immediately given a monitoring URL that can be used to check the progress, which will reveal the contents of the queues and *myspace*, and when the job is finished, give access to the completed data product.

The CPU farm can be either space-shared or time-shared. TeraGrid currently supports only space-shared nodes – meaning the whole CPU is controlled by a single user and jobs are queued and perhaps delayed for hours or days. In time-shared mode, requests made to a service are spawned (not queued) to a CPU and begin to run immediately, even if the CPU is under heavy load. CPUs could also run a database such as SQLServer or Postgres, and the spawned jobs would consist of execution of database queries.

The AS, the CPUs and other resources, and the *myspace* instantiations are connected by a data bus, which could be effected by cross-mounted file systems. The nodes and storage resources could also be geographically distributed, and connected through a global file system. The *myspace* area could contain files, but also database tables, and these data objects could be exposed through URLs at various levels of security; thus the result of a compute service could be returned to the user as "your data can be found at http://service.sdsc.TeraGrid.org/3ba7d8eff884a/mosaic.fits"

The following points elaborate the advantages of this architecture.

Security: A graduated model from anonymous client, through community- (NVO) authenticated, to a strongly authenticated power user. In this way, users can discover the scientific potential of a published service before going to the trouble of registering and getting a certificate. They can get more done by filling in a form at the community trust server, which grants a weak certificate. By making the effort to become a TeraGrid power user, they can fully exploit the national cyberinfrastructure.

Asynchrony: Services can run for a long time. The paradigm is "start-monitor-monitor-get result" in place of "request-response". We can build services that either return the result immediately, or return a jobID for a batch job. This provides a smooth scaling from simple and immediate, to sophisticated and asynchronous, so that users have the gentle learning curve that will build confidence and acceptance.

Trusted users can deploy services: The original roles were computing customer and computer center. The new role of service developer acts as a broker between these. TeraGrid power-users would be allowed to deploy services directly on TeraGrid machines, within a well-defined security framework.

Service-oriented architecture: The old supercomputer centers and the paradigm of the batch job are being replaced by a service-oriented architecture on a networked infrastructure. Services can be deployed quickly on workstations or supercomputers, and services can be deployed, managed, and upgraded by their makers.

From clicking to scripting: Users can utilize a service by clicking on a web page, and alternatively there is an API available for scripting large numbers of jobs. Authentication for web clicking comes from a certificate store, but scripted access is assumed to provide a certificate directly from the client along with the request.

Workflow: TeraGrid resources are encouraged to work together, and there is an effort to work with other big-iron grid resources through certificate trust agreements, all orchestrated by Globus. However, the architecture outlined here allows TeraGrid to play a part in SOAP-mediated workflows (CEA/Astrogrid, Kepler, Triana, Wf-XML, etc).