DATA AND COMPUTATIONAL GRID DECOUPLING IN RHIC/STAR – AN ANALYSIS SCENARIO USING SRM TECHNOLOGY

E. Hjort*, L. Hajdu, J. Lauret†,
D. Olson*, A. Sim*, A. Shoshani*

* LBNL, Berkeley, CA, 94720, USA
† BNL, Upton, NY 11973 USA

Abstract

In this paper, we describe the integration of Storage Resource Management (SRM) [1] technology into the grid-based analysis computing framework of the STAR experiment at RHIC. Users in STAR submit jobs on the grid using the STAR Unified Meta-Scheduler (SUMS) [2] which uses Condor-g [3] to send the jobs to remote sites. The input and output files are transferred between sites by 2-step transfers utilizing a DRM running at each site. In each case one transfer is a local transfer between the remote site worker node where the job was executed to and remote site DRM cache, and the other transfers are between the remote site DRM to the submission site DRM. The advantages of this method include SRM management of transfers to prevent gatekeeper overload, release of the remote worker node after initiating the second transfer so that the computation and data transfer are independent tasks, and seamless mass storage access if HRM's are used. Additionally, this light weight storage solution requiring only a few client executables on the worker node and one instance of a server process at each site could be deployed "*on the fly*" providing information is available as per the manageable storage at a target site. This makes our approach the sole Storage Element solution deployable a-posterior and requiring little human intervention while providing all the benefits of SRM managed space, optimizing the storage accessible by a given virtual organization.

STAR GRID COMPUTING OVERVIEW

Grid Computing Objectives

STAR is a TPC-based experiment at the Relativistic Heavy Ion Collider (RHIC) [4]. Over the past five years STAR has generated hundreds of TB of DST-level files for user analysis, and distributed those files between STAR's computing sites using SRM technologies for bulk file transfer. STAR is also a member VO (Virtual Organization) in the Open Science Grid (OSG) consortium and this paper will describe our efforts to integrate SRM technologies with the STAR Unified Meta-Scheduler (SUMS) to run grid jobs on OSG sites providing SRM/DRM storage elements.

While simulation based production has been tackled by many virtual organization, the problem of accessing grid resources for user analysis has been sparse. One of the main reasons is the absence of a convenient mechanism to bring files in and out of a site, considering the perhaps too rich (but needed) site policies, firewall rules or acceptance and availability of tools managing storage. Such tools and middleware have seen their existence in concrete implementation and deployment such as Xrootd [5] or dCache [6], managing large pools of space usually with a back-end handshake with mass storage. However, they are often either difficult to deploy or serve only reserved or dedicated space allocated to specific VO's. This is inadequate for opportunistic running on the Grid, the grail of the distributed computing program and at the very heart of user analysis. While the problem may be complex, simple and lightweight approaches could be employed to resolve this important issue providing a careful and staged approach. For user analysis jobs our first objective is to develop a seamless, grid-based, OSG-compliant method for users to run their jobs at any and all STAR institutions. The next objective would be to extend the method to non-STAR sites which would include OSG sites and possibly sites on other grids as well. Grid computing at STAR institutions offers advantages such as load balancing across sites and convenient access to remote resources, and eventually running on non-STAR sites would give users access to even more resources.

STAR Analysis Jobs

Local, non-grid STAR analysis jobs are based on SUMS for job submission. STAR users have been using SUMS almost exclusively for local job submission for about three years. The user describes the set of input files to use or the dataset he task need to work on, a description of the job execution (program and argument) and a destination for the output files in an xml input file. SUMS then performs the appropriate queries of the STAR file catalogue, constructs a job execution script, and submits the jobs to the local batch system. Details such as batch system syntax and matching execution node with locally stored files are handled internally by SUMS so that users can utilize different sites through a common interface.

SUMS is also in use for grid-based job submission to remote sites. In this mode SUMS submits the jobs to Condor-g which then submits them across the grid to the remote site. Again, the details of the remote submission and execution are hidden from the user and from the user's point of view the process is very similar. An important difference between local and remote job execution is that all files related to a job must be transferred across the WAN in a non-local scenario.

SUMS uses a number of methods to do this. For small files such as scripts SUMS uses Condor-g to transfer them as input files, and stderr and stdout are also left handled by Condor-g using the gridmonitor instead of file streaming. For larger input and output files, however, a managed, scalable approach is needed so that the computational quanta is not held hostage by a workflow which would be part of a unique script, executing on a worker node. In other words, decoupling of computational and storage resources must be achieved to allow efficient use of computational resources and best stability. To reach this goal, we have adopted DRM-managed gridftp transfers.

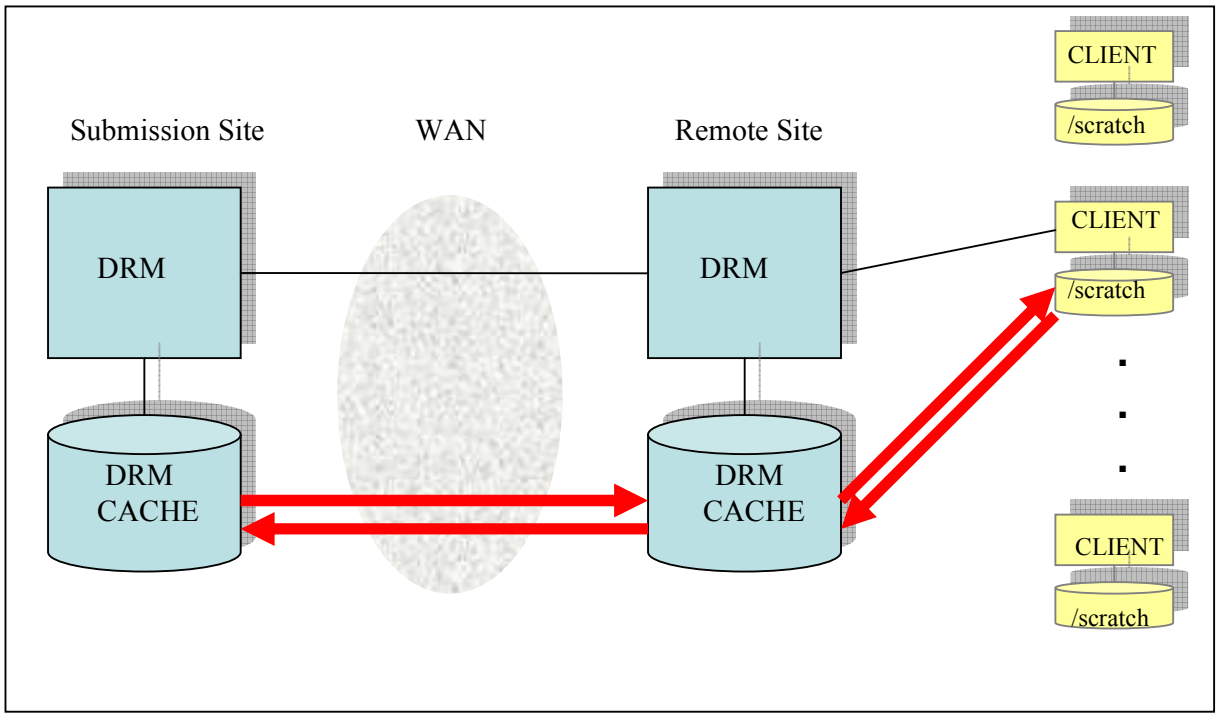

Figure 1. Schematic illustration of the STAR Analysis Method showing DRM installations at both the submission site and the remote execution site along with remote site worker nodes.

DRM Description

DRM stands for Disk Resource Manager [1] and consists of a server managing a disk cache area. Some of the functionality of a DRM includes space reservation, pinning/unpinning of files, automatic disk space management and configurable file transfers. By using appropriate settings the load on the gatekeeper running the DRM may be controlled, for example, by limiting the number of concurrent gridftp transfers. If there are not enough gridftp sessions available to transfer a file then that file will be queued until a gridftp session becomes available.

Clients to the DRM server include `srm-put` to put files into the DRM cache, `srm-get` to get files out of the DRM cache, `srm-copy` to get files from a remote DRM, and various other clients including `srm-ping` to check the status of a server, `srm-ls` to list a server's files, and `srm-release` to unpin a pinned file.

STAR ANALYSIS METHOD

Method Details

Figure 1 illustrates our general DRM transfer method. It shows a DRM running both at the job submission site and at the remote job execution site. Also shown are worker nodes at the remote site. The worker nodes have a local disk for input and output files and have access to the `srm-put`, `srm-get` and `srm-copy` clients. The input and output files are each transferred in 2 steps. In this way DRMs can buffer the transfers and network connections are channelled through the gatekeepers (or other nodes with appropriate network connectivity). As will be shown, incoming network connections on the worker nodes are not needed but an outgoing connection is used for one of the transfers. This corresponds to a typical network configuration for most clusters where outgoing connections from worker nodes are allowed but incoming connections are blocked. Solutions for clusters that block outgoing connections from worker nodes are outside the scope of this paper but could be constructed with any method that initiates the final transfer from

someplace other than the remote worker node, perhaps as an "border" or "edge" node running the service and transfer and handled within a DAG.

The first transfer of input files starts when job execution begins on the remote site worker node. Here `srm-copy` is used to transfer input files from the submission site to DRM cache space on the remote gatekeeper. A logical file name is assigned by the client for future reference in the second transfer. Once this transfer is complete a second transfer uses `srm-get` to put the input files onto the local disk of the worker node by referring to the logical file name assigned in the first transfer. After this transfer is complete the input files are in place on the worker node and file processing can begin, releasing the space in the site DRM cache. Note that for the input files no external network connections to the worker nodes are required as files are imported to the local node using a reference (logical name) to a file previously "*pushed*" into DRM space by proxy-ing the file transfer to the SRM layer in a totally asynchronous manner.

Once the input files have been processed and the output files written to the worker node disk they are transferred back to the submission site in a similar two-step transfer. The first transfer uses `srm-put` to deposit the output file into the DRM cache and assign logical file name. The second transfer uses `srm-copy` to transfer the output file from the job execution site back to the job submission site. Of the four transfers this is the only one in which the client call goes to a remote DRM over the WAN. This is necessary because `srm-copy` works in pull mode and therefore requires an outgoing connection from the worker node. Since the call-back from the DRM server requires an incoming connection to the worker node this transfer is done without waiting for call-backs. This has the advantage that the worker node is released back to the local batch system immediately.

Advantages of the Method

One advantage of this method is that it takes advantage of standard grid middleware and requires no STAR specific installations at remote sites. Another advantage for STAR users is that the look and feel of grid submission is very similar to local jobs submission that they are already familiar with – the grid-based parts of the job such as WAN file transfers are hidden and the same local files and catalogues are used. Output files are returned to the submission site just as though the job was run locally with perhaps the sole caveat to have them back in a delayed manner. To our experience, this is not a problem as SRM could be later queried for the status of an incoming transfer, allowing for accurate reporting of a job workflow, including the completion status depending on its output completion status. Making the use of the grid in STAR as transparent as possible is an important part of this method as typical users tend to be inclined to use only as much technology as they need to get their job done. A final but important benefit of this method is that by using DRM-managed transfers the load on the gatekeepers can be controlled. This is accomplished by limiting the allowed number of concurrent gridftp sessions and if necessary using a separate node or set of nodes for SRM file transfers since this method is independent from job submission. For example, if a user submits 100 jobs to a remote site and they all start at the same time and try to transfer files simultaneously the gatekeeper might be come overloaded. In this method a limited number of transfers would be allowed and some jobs would wait for their input files.

TESTING AND PERFORMANCE

The main testbed involves the two main sites for STAR computing: the tier 0 site at RCF is used for job submission and the tier 1 site at PDSF/NERSC is used for job execution. A third, smaller site at Wayne State University is also used for smaller scale testing.

For large-scale testing, we used production-level STAR simulation jobs which perform reconstruction of simulated events. Slightly more complex than a pure simulation job, these jobs require one input file of about 300 MB which is sourced from HPSS storage by extending the DRM to be an HRM (Hierarchical Resource Manager). With a job execution time of 5-10 hours and up to 100 jobs running concurrently we expect up to 20 jobs to finish per hour. Each job produces a total of 700 MB in five files so this requires 14 GB/hour (~4MB/s) on average. For simplicity we've been using a single gridftp session for our WAN transfers which results in a transfer rate of about 5 MB/s. This rate is comparable to the net rate we require to minimally sustain the data transfer so we expect some amount of DRM management to take place in terms of queuing files during busy times for later transfer. For the case we studied, the data transfer being slightly higher than the job IO throughput allows file transfer to keep up with the data produced at the remote site. In other words, the introduction and use of SRM technology in such poor network transfer conditions is still a good case for efficiently harvesting remote resources otherwise unusable or "locked" while waiting for a direct file transfer. Our later goal of running larger-scale tests and migrating most production to a Grid based enterprise would however be severely impacted by the WAN transfer performance. It is noteworthy to mention that while this poor performance is not completely understood, it does not seem to be limited by DRM (gridftp) as other transfer protocols result in similar performance.

In our tests we have observed that DRMs working as designed – i.e., buffering the data transfers and managing the load on the gatekeepers during times of intense activity. We have observed latencies of 2 hours without problems during our testing of this method on a large-scale basis for about four months. Many thousands of jobs have been run and many TB's of data transferred. DRM stability has been excellent, and the problems we do observe are typically not DRM-specific and as such

the method also provides a powerful testing tool for STAR's grid computing in general.

SUMMARY

The combination of SUMS-based submission of STAR user analysis jobs combined with file transfers utilizing SRM technologies enables a seamless, scalable solution for STAR's grid computing needs. By working within the OSG framework we expect that extending the method to other non-STAR sites will be straightforward. The SRM/DRM implementation is relatively lightweight which make it attractive to site administrators particularly at smaller sites. The method has seen extensive use with excellent results on the STAR testbed and efforts are underway to extend the method to other sites.

REFERENCES

[1] The SRM project, http://sdm.lbl.gov/srm-wg/

[2] The STAR Unified Meta-Scheduler CHEP04, Interlaken, Switzerland http://indico.cern.ch/contributionDisplay.py?contribId=318&sessionId=7&confId=0 ; SUMS web site resources: http://www.star.bnl.gov/STAR/comp/Grid/scheduler/

[3] The Condor project: http://www.cs.wisc.edu/condor/

[4] RHIC: http://www.bnl.gov/RHIC/

[5] The xrootd project, http://xrootd.slac.stanford.edu/

[6] The dCache Project, http://www.dcache.org/