FERMILAB-PUB-07-462-CD

DZero Data-Intensive Computing on the Open Science Grid

B. Abbott [%], A. Baranovski[*], M. Diesburg[*], G. Garzoglio[*], T. Kurca[+], P. Mhashilkar[*]

* Fermi National Accelerator Laboratory, P.O. Box 500, Batavia, IL, 60510, USA

+ IPNL, Universite Lyon 1, CNRS/IN2P3, Villeurbanne, France and Universite de Lyon, Lyon France

% The University of Oklahoma, 660 Parrington Oval, Norman, OK

abbott@nhn.ou.edu, abaranov@fnal.gov, diesburg@fnal.gov, garzoglio@fnal.gov, kurca@in2p3.fr, parag@fnal.gov

Abstract. High energy physics experiments periodically reprocess data, in order to take advantage of improved understanding of the detector and the data processing code. Between February and May 2007, the DZero experiment has reprocessed a substantial fraction of its dataset. This consists of half a billion events, corresponding to about 100 TB of data, organized in 300,000 files.

The activity utilized resources from sites around the world, including a dozen sites participating to the Open Science Grid consortium (OSG). About 1,500 jobs were run every day across the OSG, consuming and producing hundreds of Gigabytes of data. Access to OSG computing and storage resources was coordinated by the SAM-Grid system. This system organized job access to a complex topology of data queues and job scheduling to clusters, using a SAM-Grid to OSG job forwarding infrastructure.

For the first time in the lifetime of the experiment, a data intensive production activity was managed on a general purpose grid, such as OSG. This paper describes the implications of using OSG, where all resources are granted following an opportunistic model, the challenges of operating a data intensive activity over such large computing infrastructure, and the lessons learned throughout the project.

1. Introduction

Throughout the past several decades, high energy physics experiments have pushed their computational requirements to the limit of data intensive computing technology. In order to make the problem manageable and limit duplication of work and waste of resources, collaborations have typically divided computing activities in two broad categories: those for the “common good”, also called “production”, and those of interest to single groups or individuals. Production activities have been typically managed centrally and often deserved special infrastructures and human resources.

Data Processing is an example of a production activity. Detector data in a "raw" format is processed in order to change the description of physics events from a representation close to the detector hardware to a representation close to physical quantities. All analyses use this output to mine the data for specific physics measurements. This limits the duplication of computing cycles to transform data to manageable physics-related representations. While this computing model reduces the amount of computing resource needed on a day to day basis, it also implies that when the understanding of the detector improves (new algorithms are developed, calibration constants are refined, etc.) all data must be reprocessed.

In the case of the 2007 Data Reprocessing for the DZero experiment [1], the activity was ever more challenging. For the first time in the history of the experiment, in fact, DZero decided to conduct the reprocessing activity using the Grid opportunistically. In other words, during the activity, only a small fraction of all available computing resources were dedicated to the experiment.

The majority of the Grid resources were made available from the Open Science Grid (OSG) [2], with contributions from CCIN2P3 [3], LCG [4], WestGrid; Fermilab provided resources accessible via local interfaces. The OSG is a consortium of about 80 National Laboratories, Universities, and Institutions working together to provide a national computing infrastructure for e-science. The consortium is funded by the National Science Foundation and the US Department of Energy's Office of Science. Computing resources vary widely in specifications, ranging from supercomputers to small clusters of commodity computers. Particularly challenging for the DZero Data Reprocessing activity were

- the large diversity in the computing environment
- a network connectivity inadequate for running data intensive jobs at some sites
- the large variation in number of available computing resources from time to time (opportunistic model).

In abstract terms, the Data Reprocessing challenge was an optimization problem. The problem was selecting OSG resources according to the DZero requirements for availability, quality, and accessibility. This problem was addressed with an iterative approach, by adding resources a site at a time. The heuristic that guided our solution was maximizing the number of available resources, minimizing the number of sites in the system, and maximizing the bandwidth inter-connectivity. Details on this approach are discussed in this paper.

2. Scale of the Data Reprocessing Activity

The total amount of input data to be reprocessed was 90 TB. Since no cluster on the OSG was dedicated to DZero, the application could not be pre-installed at the worker nodes. Considering that the application size was about 1 GB, an extra total of 250 TB of information had to be transported to the Grid. Caching the application on storages local to the sites was a necessary optimization to limit the total bandwidth usage to a manageable level.

The amount of output produced by the application was 60 TB. In order to make storage of the output efficient, the files produced by each job had to be merged in larger files, of around 1 GB in size. To avoid excessive network traffic on the WAN, it was decided to gather all output files at a single site, Fermilab, where the merging application was then run locally.

The total amount of computation required during the activity was 500 GHz CPU years. Considering that at least 80% of the data was to be made available to the collaboration before the "summer conferences", DZero requested OSG to have available an average of 1,500 CPU for about 4 months. The plan was for DZero to run about 2 jobs per day on each CPU. Once reached a steady operational regime, this schedule could be achieved.

3. Resource Coordination

Production activities for DZero are managed via the SAM-Grid system [5]. The system is an integrated job, data, and information management infrastructure. Its main goal is coordinating the usage of Grid resources according to DZero requirements.

As one of its main responsibilities, SAM-Grid implements the selection of computing and storage resources during workload scheduling. For selection of storages and for distribution of data, SAM-Grid relies on the Sequential Access via Metadata (SAM) [6], a data handling system responsible for the management of storage elements and for job-related metadata cataloguing. For selection of computing resources, SAM-Grid relies on specialized deployments of standard middleware components, such as Condor-G [7] and the Globus Gatekeeper [8], to distribute the workload between OSG, LCG, and the other available resources. The SAM-Grid job selection infrastructure interfaces to grid-specific brokering systems, like the LCG resource broker [9] or the OSG Resource Selection System (ReSS) [10], for fine-grained intra-grid cluster selection. In addition, the SAM-Grid system provides interfaces for job management (job submission, job status tracking, job deletion, job failure recovery, etc.) as well as job workflow monitoring.

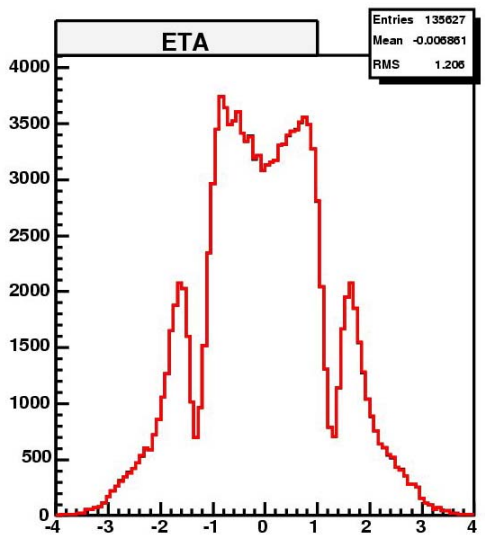
4. System Commissioning

The data reprocessing activity started with a ramp up phase of resource commissioning. Out of all the available computing facilities, on the order of 80 in the OSG only, resources were included in the system according to a long multi-step process, only partially automatable. First, network connectivity to storage resources had to be tested for sufficiently high bandwidth and low latency. Second, data reprocessing jobs were run on reference input data, so that output could be compared with expected results (site certification). Third, appropriate data transfer queues were created in the SAM system. This achieved two goals: (1) separating requests for data between sites with low and high network latencies; (2) shaping network traffic for applications with different data input patterns e.g. data reprocessing applications vs. data merging applications. Finally, sites with local storages where preferred, in order to enable caching of the input application.

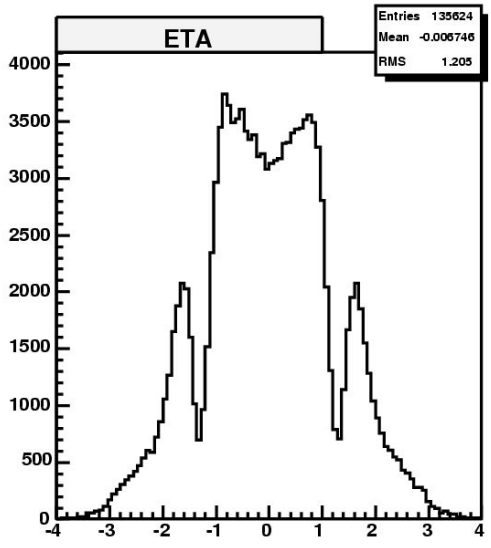
The following subsections explore in more detail some of these commissioning steps.

4.1. Site Certification

In the early days of the DZero experiments, data reprocessing activities were conducted on a few closely-controlled computing clusters. As more and more computing resources were made available through grid interfaces, systems like SAM-Grid made it possible to manage the complexity of running data intensive activities over large distributed system [11]. The increased diversity of the available resources, however, made it necessary to verify that the physics results were invariant with respect to where they were produced. To accomplish this goal, the DZero experiment developed a site certification process.



LTU



D0farm

Figure 1: Two physics histograms produced from the same input data on two different computing clusters. Data produced on the Louisiana Optical Network Initiative (LONI) cluster at Louisiana Tech University (LTU) (left) look identical to data produced at the Fermilab DZero Farm (right).

When a new site was considered for inclusion in the pool of resources, a data reprocessing job was run on a specific input dataset. The reprocessed events were then organized in several physics histograms, which were used as a benchmark for the site. The histograms were then compared by a physicist with references produced on known hardware. Because of the need for manual intervention, site certification was always a relatively long process, lasting, in average, a week.

Figure 1 shows the comparison for one such histogram.

4.2. Data Accessibility

Data reprocessing jobs require two pieces of input: the application, about 1GB in size, and the data, in average 300 MB. In many cases, the application could be cached at storages local to the computing sites; however, for sites where local storages were not available, the application had to be transported over the WAN from the "closest" grid storage. In OSG, 8 sites out of 12 had local storages; 4 of them were available for grid access and 4 for local access only (3 via SRM [12] interfaces, 1 via NFS). On the other hand, input data was always transported from Fermilab, where a Mass Storage System, Enstore [13], holds the entire data set for the experiment. It is not desirable caching such data because it is processed once, unless application failures occur. The output from all the jobs was transported to temporary storages at Fermilab, where it was merged (sec. 2).

Overall data access was optimized by monitoring data throughput and categorizing sites in three broad groups: those with "fast", "slow", and "unacceptable" network connectivity. Sites considered "unacceptable" had network connectivity much worse than their peers and were not included in the pool of resources. Figure 2 shows measurements of data throughput from a site included in the pool and one rejected.

The connectivity categories were used to direct data access requests initiated from "fast" and "slow" sites to different data transfer queues, tagged as "fast" and "slow". These special data queues were created at each storage elements managed by SAM. Having multiple data queues is preferable to having a single one. With a single data queue, transfers from sites with "slow" connectivity slow down data access from sites with "fast" connectivity. Thus, data access for the whole grid is negatively affected.

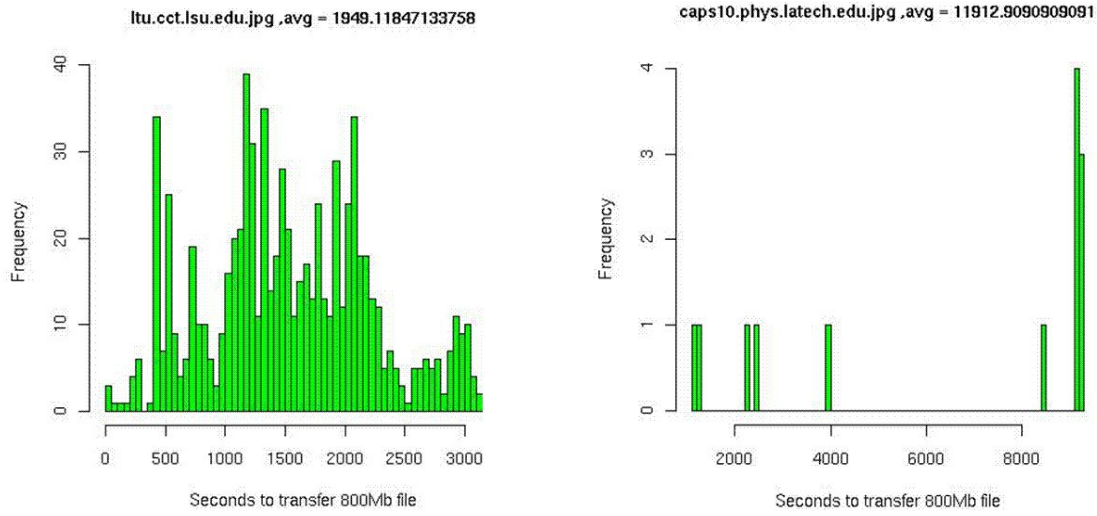


Figure 2: An example of the histograms used to accept or reject OSG sites based on their network connectivity. For each OSG site, a histogram shows measurements of the time necessary to transfer 800 MB from 2 OSG storages (at Oklahoma and Indiana University). The site on the left took an average of 2000 seconds to transfer the file; the cluster on the right an average of 12,000 seconds.

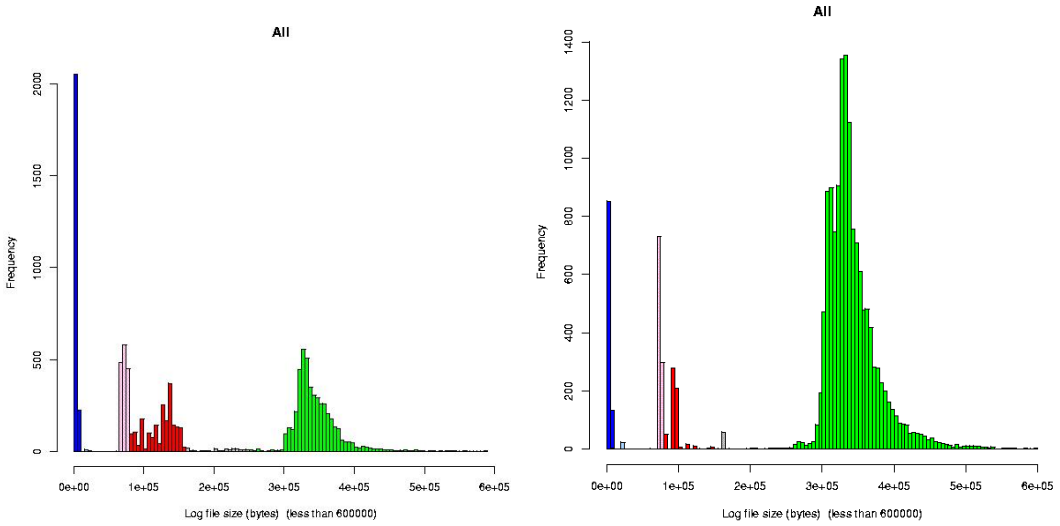
While the former measurement is a fairly typical result among OSG sites, the latter was sensibly higher. The site from the right was not included in the DZero pool of resources.

5. Monitoring and Troubleshooting Operations

Monitoring and troubleshooting the system was one of the challenges of the operations. Job failures were caused by three major factors:

- Site / Grid (OSG / LCG) problems: these were typically site gateway and worker nodes configuration problems. The site administrators and the OSG troubleshooting team were responsible for addressing these.
- Data delivery problems: these were either data handling services problems or storage element problems. The SAM-Grid system group was responsible for the former, the site administrators and the OSG troubleshooting team for the latter. In many occasions, only careful inspections of log files could distinguish between the two cases.
- Application failures: these were caused by software bugs or input data corruptions. The DZero data reprocessing operation team were responsible for addressing these in collaboration with the DZero offline software group.

In order to investigate and properly triaging the problems, a monitoring system was developed to attempt an automatic categorization of the failures. The system plotted the histogram of the output size of all jobs submitted in a 5 days interval. Because the data reprocessing application is the same for every job, the length of its output strongly correlates with the type of failure for the job. Figure 3 shows an example of such histograms.



blue	0-8 kB	OSG CE or Worker Node configuration problem; lost standard output
aqua	8-25 kB	SAM-Grid service / hardware failure; could not start bootstrap executable.
pink	25-80 kB	SAM problem: could not get application, possibly raw files
red	80-160 kB	SAM problem: could not get raw files, possibly application
gray	160-250 kB	Possible D0 application crash
green	>250 kB	Success

Figure 3: Two histograms of the size of the data reprocessing application output for jobs run throughout 5 days. The size of the output is a strong indication of the type of job failure. The histogram on the left was created on Mar 27, 2007, before the involvement of the OSG troubleshooting team; the one on the right on Apr 17, 2007. The two left-most bins on each histogram (in blue) indicate failures of the OSG infrastructure. The proportion of OSG-related failures with

respect to other types of failures and successes decreased significantly after the intervention of the OSG troubleshooting team.

A process for triaging problems to the right group had to be developed. In particular for problems on the OSG, we could take advantage of the ticketing service offered by the OSG Grid Operation Center (GOC) [14]. Problems submitted to the system are tracked and followed up for resolutions. Despite the help from GOC, the sheer number of resources made it impossible for the single person that we could dedicate from our group to keep up with site problems. The OSG Troubleshooting team was therefore involved to interact with system administrators, investigate failures, propose solutions, follow up with the resolution, etc. The team was instrumental to the success of the data reprocessing activity. Figure 3 shows the reduction of OSG system problems after the OSG troubleshooting team was involved.

6. Lessons Learned

The DZero data reprocessing effort faced different types of challenges in operating the infrastructure in the phases before and after the completion of resource commissioning. Both phases of operations lasted several months.

During the phase of commissioning, about 50% of job failures were due to configuration problems at OSG sites and about 50% to data delivery problems. For the data reprocessing application, a job is considered successful if the produced data is successfully stored to storage.

Typical site configuration problems included computing element access authentication and authorization failures, scratch areas permission problems, system library incompatibilities, and wrong reports of the job status (middleware failures). In addition, despite the OSG process to report cluster downtimes to the Grid Operation Centre, unscheduled downtimes affected operations for the duration of the activity. Most configuration problems were addressed with the help of the OSG troubleshooting team. The lesson learned is that one should not undergo computing activities of such a magnitude without the support of a troubleshooting team acting as a liaison between the user and the system administrators.

Data delivery problems were mainly due to lack of storage systems local to the computing sites and insufficient network connectivity to storages. In our model, local storage systems were used to cache the application. Given its large size and the hundreds of concurrent jobs potentially running on each cluster, uncontrolled access from worker nodes to a shared file system (a typical configuration on many clusters) tended to make the system unstable. It is preferable storing the application in a local storage system like dCache [15] and accessing it via an SRM interface.

Addressing data delivery problems, we learned that sites must be categorized according to their connectivity to global storages. Requests for data access from “slow” sites must be queued together, separately from requests from “fast” sites (sec. 4). In addition, as expected, sites with poor connectivity to storages are useless to run data intensive applications.

As the efforts toward commissioning resources diminished, the resource pool became more stable. Configuration and data delivery problems started to become more seldom. In this more reliable environment, problems at the global level became more apparent. In particular, the lack of a stable Grid-level resource selection service manifested in over- and under-subscription of cluster usage. Resource selection, in fact, was left to the operators of the infrastructure, responsible for job submission. This resulted in a less than optimal utilization of the resources as some clusters received fewer jobs than they could process, while others queued up jobs that eventually failed since grid services, such as data handling, had timed out. We learned that for a computational challenge of this magnitude, an automatic resource selection system is necessary to reduce the need for job recovery and for simplifying operations.

Another lesson learned in the final weeks of the activity is that DZero data reprocessing operations would have been more efficient if most job recoveries had been spread in time. This consideration is probably valid for all workflows that include an operationally intensive phase of failure recovery.

Despite the automation of the job recovery procedure, in fact, identifying jobs to be resubmitted was considered a human intensive operation. Recovering jobs shortly after job failures, would have avoided a final "tail" of intensive operations right at the end the activity, a time where personnel focus tends to dwindle.

7. Conclusions

The DZero collaboration has reprocessed 90 TB of data using fully distributed, opportunistic resources, contributed by OSG, LCG, CCIN2P3, Westgrid, and Fermilab from February to May 2007. We described the challenges of system commissioning, troubleshooting, and operations. Commissioning was approached as an iterative problem. Resources were added a site at a time, categorizing their network connectivity to storages and comparing the output of physics results with standard references. Troubleshooting required the development of a monitoring tool to categorize failures. Site configuration problems have been addressed with help from the OSG troubleshooting team. Operations were coordinated via the SAM-Grid system. The system understands the DZero workflow and its requirements and coordinates the selection of computing as well as storage resources.

450 million events were reprocessed and made available to physicists for the "summer" conferences. Figure 4 shows the integrated number of events produced vs. time.

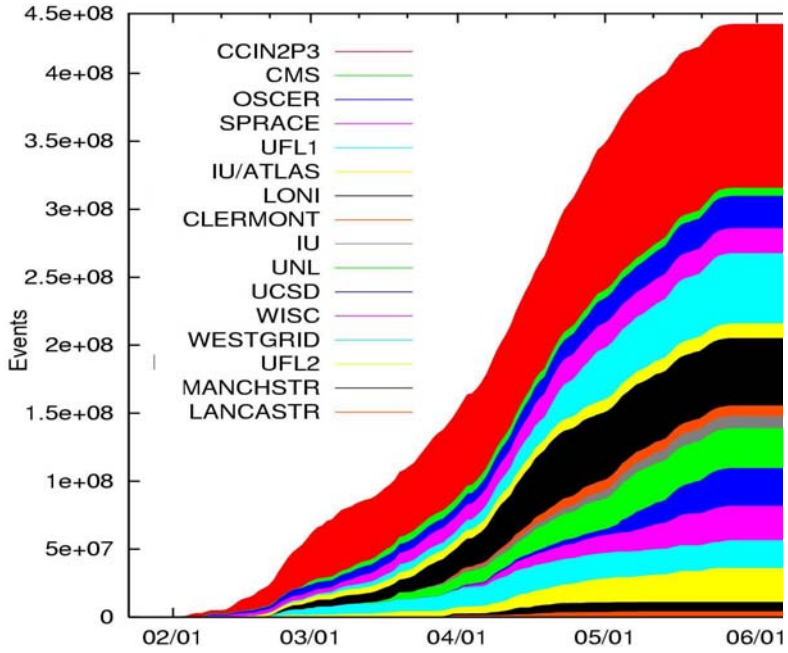


Figure 4: the integrated number of events produced on the OSG, LCG, CCIN2P3, and Westgrid vs. time.

References

[1] The D0 Collab., "The D0 Upgrade: The Detector and its Physics", Fermilab Pub-96/357-E.

[2] OSG: www.opensciencegrid.org

[3] CCIN2P3: http://cc.in2p3.fr/cc_accueil.php3?lang=en

[4] J. Apostolakis, G. Barrand, R. Brun, P. Buncic, V. Innocente, P. Mato, A. Pfeiffer, D. Quarrie, F. Rademakers, L. Taylor, C. Tull, T. Wenaus, "Architecture Blueprint Requirements Technical Assessment Group (RTAG)", Report of the LHC Computing Grid Project, CERN, Oct. 2002

[5] G. Garzoglio, A.Baranovski, H. Koutaniemi, L. Lueking, S. Patil, R. Pordes, A. Rana, I.

Terekhov , S. Veseli, J. Yu, R. Walker, V. White, ”The SAM-GRID project: architecture and plan.”, Nuclear Instruments and Methods in Physics Research, Section A, NIMA14225, vol. 502/2-3 pp 423 – 425

G. Garzoglio "A Globally Distributed System for Job, Data, and Information Handling for High-Energy Physics", Ph.D. Thesis, DePaul University, Chicago, March 2006, Fermilab-Thesis-2005-32

[6] J. Frey, T. Tannenbaum, M. Livny, I. Foster, and S. Tuecke, ”Condor-G: A Computation Management Agent for Multi-Institutional Grids”, in Proceedings of the 10th International Symposium on High Performance Distributed Computing (HPDC-10), IEEE CS Press, Aug. 2001Globus

[7] G. Avellino et al., ”The EU DataGrid Workload Management System: towards the second major release”, in Proceedings of Computing in High-Energy and Nuclear Physics (CHEP03), La Jolla, California, March 2003

[8] ReSS: https://twiki.grid.iu.edu/twiki/bin/view/ResourceSelection

[9] J. Snow, D. Wicke, M. Diesburg, G. Garzoglio, G. Davies, "DZero Data Reprocessing with SAM-Grid", in Proceedings of Computing in High Energy Physics (CHEP06), Mumbai, India, Feb 2006

[10] I. Bird, B. Hess, A. Kowalski, D. Petravick, R. Wellner, J. Gu, E. Otoo, A. Romosan, A. Sim, A. Shoshani, W. Hoschek, P. Kunszt, H. Stockinger, K. Stockinger, B. Tierney and J. Baud, ”SRM (Storage Resource Manager) Joint Functional Design”, Global Grid Forum Document, GGF4, Toronto, Feb. 2002

[11] Enstore: www-isd.fnal.gov/enstore/

[12] GOC: www.grid.iu.edu/

[13] M. Ernst, P. Fuhrmann, T. Mkrtchyan, J. Bakken, I. Fisk, T. Perelmutov, D. Petravick, "Managed Data Storage and Data Access Services for Data Grids", Chep 2004, Interlaken, Switzerland, Sep. 2004