Jefferson Lab Mass Storage and File Replication Services[§]

Authors: Ian Bird (Ian.Bird@jlab.org), Ying Chen, Bryan Hess, Andy Kowalski, Chip Watson

Thomas Jefferson National Accelerator Facility, 12000 Jefferson Avenue,
Newport News, VA 23606, USA

Abstract

Jefferson Lab has implemented a scalable, distributed, high performance mass storage system - JASMine. The system is entirely implemented in Java, provides access to robotic tape storage and includes disk cache and stage manager components. The disk manager subsystem may be used independently to manage stand-alone disk pools. The system includes a scheduler to provide policy-based access to the storage systems. Security is provided by pluggable authentication modules and is implemented at the network socket level. The tape and disk cache systems have well defined interfaces in order to provide integration with grid-based services. The system is in production and being used to archive 1 TB per day from the experiments, and currently moves over 2 TB per day total. This paper will describe the architecture of JASMine; discuss the rationale for building the system, and present a transparent 3rd party file replication service to move data to collaborating institutes using JASMine, XML, and servlet technology interfacing to grid-based file transfer mechanisms.

Introduction

Thomas Jefferson National Accelerator Facility (Jefferson Lab) operates a high intensity continuous-wave electron accelerator facility (CEBAF) with experimental facilities investigating the quark and gluon structure of nuclei in the energy regime that overlaps traditional nuclear and high energy physics. The intense nature of the delivered electron beam and the desire for extremely high precision in the experiments leads to very high event rates and consequently data rates. Current experiments acquire data at between 10 - 22 MB/s, with several hundred Terabytes of data per year stored. Future expected upgrades of the accelerator and the experimental facilities, including the addition of a fourth experimental area, will mean an order of magnitude increase in data rates. At that time, data acquisition rates of 100 MB/s and data storage needs of 3 PB/year of raw, processed and simulated data are expected. These expectations are similar to those of LHC experiments in schedule and magnitude. The physics collaborations involved in these experiments are international with typically 150-200 scientists from institutions world-wide involved. The scientific, technical, and sociological requirements to employ computing- and data-grid technology are just as vital for success in this field as they are in LHC and other large experiments. Jefferson Lab is a member of the Particle Physics Data Grid (PPDG) project[1] in order to provide remote data services to current experiments as well as to investigate solutions for the future.

In order to support the storage and computational needs of the experiments the laboratory operates a facility with 2 StorageTek data silos equipped with 28 tape drives (9940, 9840, and RedWood), supplemented by some 25 TB of managed and unmanaged disk pools. These disk pools are constructed from systems of IDE or SCSI disks using RAID levels 0 or 5 according to the use of the system. Computation for reconstruction and analysis is provided on a farm of 350 Linux processors. In addition the laboratory is also building a Lattice QCD cluster, expected to reach 0.5 Teraflops early

[§] This research was sponsored by the U.S. Department of Energy; contract DE-AC05-84ER40150. Views and conclusions contained in this paper are the authors' and should not be interpreted as representing the official opinion or policies.

in 2002. This project is in collaboration with MIT and will also use data-grid services to provide access to remote collaborators. Figure 1 illustrates the data and computational facilities.

The JASMine Storage Manager

The mass storage system described here was built to provide robust, distributed, scaleable and fault-tolerant data access facilities to the experiments and their collaborators both local and remote. Until recently, Jefferson Lab used OSM as the basis of its storage management software, but overlaid by Java software that not only provided the user interface, but also provided disk cache management, request scheduling, and tape staging. As a replacement for OSM was sought it became clear that other existing storage systems lacked most of this functionality, or were not affordable or manageable by a small staff. Thus the JASMine system was built upon the existing Java layer to provide the functionality lacking in OSM. The resulting software is a lightweight, flexible, but scaleable distributed system capable of meeting the needs of existing and future high data rate experiments.

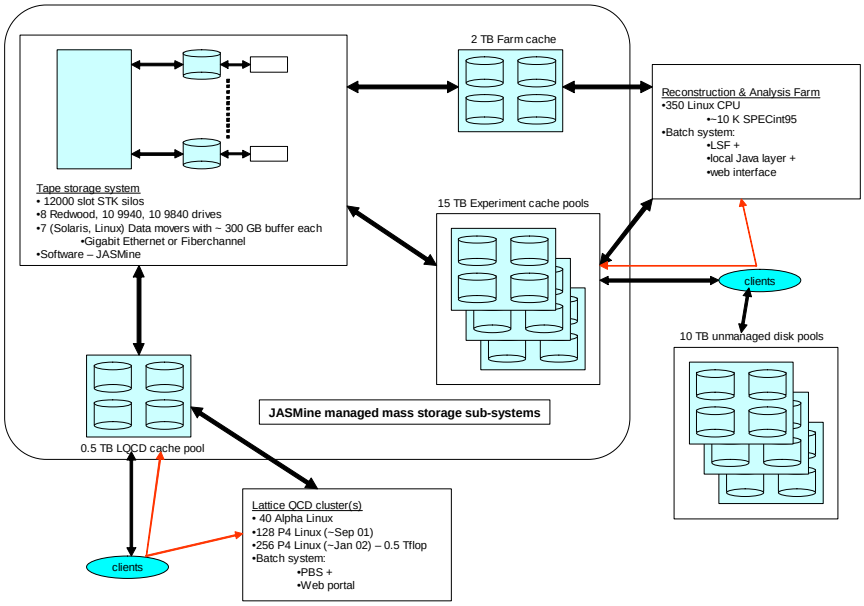

Figure 1: Jefferson Lab Mass Storage & Farms

Architecture and Features

The complete architecture of the JASMine (Jefferson Lab Asynchronous Storage Manager) system has been described elsewhere[2] and is illustrated in Figure 2. A brief overview of the main functions and features are given here. The system is entirely implemented in Java. The major components are a MySQL database that controls the system, distributed data movers, distributed cache managers and administrative and control daemons. Most services are replicated. The data movers, which run on Linux or Solaris systems, have local disk buffers for staging data to and from tape. Each data mover contacts the central database for work.

File movement uses an extensible protocol which uses serialized objects as messages passed over streams. For bulk data transfer the implementation reverts to raw data transfer over tcp for efficiency. The basic protocol is synchronous, multiple requests and asynchronous behavior is achieved with multiple threads. Security is provided within the protocol with an Authentication interface that allows any suitable authentication mechanism to be used. This is applied to all network connections and communications. It is planned to provide an implementation of the Globus Security Infrastructure (GSI)[3]. A further level of security is provided via a pluggable file transfer policy – for example each

client request is only authorized to perform the file transfers it requested. The same protocol is used for all data transfers within the system, and CRC32 checksums are used at each stage to verify integrity of the data.

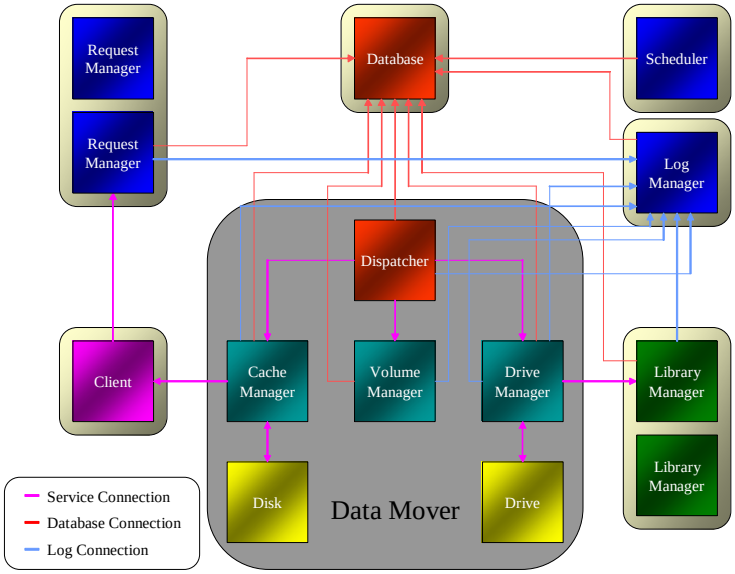

Figure 2: JASMine architecture

The system provides a hierarchical scheduling mechanism to prioritize access to the system based on individual or groups of hosts and users. This permits the system to be optimized so that production farms and associated users and groups get high priority access to tape - thus keeping the farms busy. It also prevents certain users or remote mini-farms from hogging system resources. Internally, however, the system will always optimize file access on tape so that any outstanding requests that use mounted tapes will be serviced first no matter what priority that user has.

A significant feature of the system is the cache manager component. This component is used within JASMine in a variety of ways – as a stage disk manager within the data movers, as a cache server managing a disk pool, or deployed as a remote client. A cache server is a pool of disk that is automatically managed by the system. The protocol above is extended with database hooks, the database then shared between several cache servers to create a high-throughput disk pool. Any cache server in the pool can look up a file location. In that sense it is serverless, but data transfer is always direct between the client and the node holding the file. Thus adding servers (and disk) to the pool increases the throughput with no overhead and provides tolerance against failure. The policy used to manage a pool can be one of those existing, or externally defined. Current policies include reference counting (for stage disks), least recently used (cache pools), or explicit deletion for some long-lived data. The cache server can be deployed to a remote site to act as a remote client to the JLAB mass store as the basis for a transparent file replication service. This is useful for collaborators who have a Linux system with a disk pool who wish to stage data remotely for analysis. Network efficiency is improved by support within the protocol for multiple simultaneous file transfers and with a Java stream for parallel file transfer currently under development.

Data Grid and Web Services

In order to provide a useful service to the user community, one of the first requirements is a reliable end-to-end remote file transfer capability. There are several possible scenarios – using the cache server as a remote client as described above, or providing a standard interface into the JASMine system so that it can function as a black box and respond to data-grid requests for files. Both of these strategies are being followed to achieve the short- and long-term goals of the experiments. The cache server software is being deployed to Florida State University in conjunction with the CLAS

experiment[4], and to MIT as part of the collaboration on Lattice QCD to manage data transfers between those facilities. As part of the long term strategy, work is proceeding within PPDG to define and implement standard interfaces to a storage system which will enable Jefferson Lab storage systems to be integrated as a node on the data grid providing file transfer and replication services in a standard way.

A natural extension to the basic data grid services is to provide access to those services to web-based applications. At Jefferson Lab a data management portal has been implemented using Java servlets in the Apache web server with the Tomcat servlet engine. The servlets make metadata available in the form of XML data structures. Other servlets are used to apply style sheets to the XML to permit web browsing. The replica catalog service has been implemented as a simple MySQL database accessed by the servlets via JDBC, and supports a recursive directory structure. A grid node system acts as the front-end to the disk caches and tape storage. A minimal set of operations includes staging files to and from tape, adding files to cache, and the ability to pin/unpin files in cache. For data transfer, this node translates the logical file name to the URL of a server able to provide or receive the file. Actual transfer mechanisms are independent of this service, but will include gridFTP as a minimum. Authentication based on X.509 certificates is available now, and will be made to interoperate with GSI.

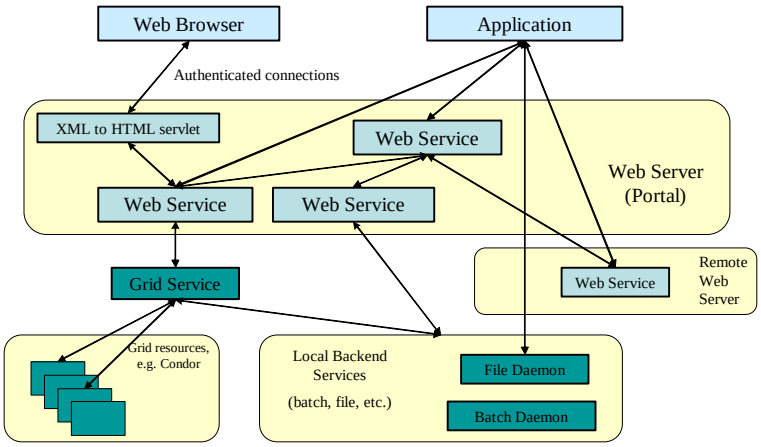

Figure 3: Three-tier web services architecture

The web service layer will provide web-based (browser or application) file selection and queuing of 3rd party file transfers. The data management web service is viewed as one component of an integrated portal (Figure 3) allowing experimenters web access to batch and data management services, including experiment-specific layers to permit file and compute operations based on experiment metadata. It is foreseen that the use of web technologies such as XML, servlets, and standards-based protocols such as SOAP will aid the rapid development and deployment of these services using available tools.

---

[1] Particle Physics Data Grid collaboration. http://www.ppdg.net
[2] Building the Mass Storage System at Jefferson Lab. Ian Bird, Bryan Hess, Andy Kowalski. *Proceedings of the 18th IEEE Symposium on Mass Storage Systems, April 2001*.
[3] Globus: http://www.globus.org
[4] Cebaf Large Acceptance Spectrometer (CLAS). http://www.jlab.org/Hall-B