Hadoop Distributed File System for the Grid

Garhan Attebury *, Andrew Baranovski †, Ken Bloom *, Brian Bockelman *, Dorian Kcira ‡, James Letts §, Tanya Levshina †, Carl Lundestedt *, Terrence Martin §, Will Maier ¶, Haifeng Pi §, Abhishek Rana §, Igor Sfiligoi §, Alexander Sim ‖, Michael Thomas ‡, Frank Wuerthwein §

* University of Nebraska Lincoln

† Fermi National Accelerator Laboratory

‡ California Institute of Technology

§ University of California, San Diego

¶ University of Wisconsin Madson

‖ Lawrence Berkeley National Laboratory

***Abstract*—Data distribution, storage and access are essential to CPU-intensive and data-intensive high performance Grid computing. A newly emerged file system, Hadoop distributed file system (HDFS), is deployed and tested within the Open Science Grid (OSG) middleware stack. Efforts have been taken to integrate HDFS with other Grid tools to build a complete service framework for the Storage Element (SE). Scalability tests show that sustained high inter-DataNode data transfer can be achieved for the cluster fully loaded with data-processing jobs. The WAN transfer to HDFS supported by BeStMan and tuned GridFTP servers shows large scalability and robustness of the system. The hadoop client can be deployed at interactive machines to support remote data access. The ability to automatically replicate precious data is especially important for computing sites, which is demonstrated at the Large Hadron Collider (LHC) computing centers. The simplicity of operations of HDFS-based SE significantly reduces the cost of ownership of Petabyte scale data storage over alternative solutions.**

I. Introduction

HADOOP, a newly emerged Java-based software framework, supports applications that process vast amount of data in a timely manner [1]. It is extensively used in search engine and advertising businesses by Yahoo and other companies of scale up to thousands of computing nodes and Petabytes of data. The Hadoop Distributed File System (HDFS) is a key component of Hadoop that is designed to store data on commodity hardware with high access bandwidth across the cluster. The reliable data replication and detection of failure enable fast and automatic system recovery. The emphasis of high throughput instead of low latency makes HDFS appealing for batch processing. HDFS implements a simple data coherency model, write-once-read-many, that allows high throughput of data access. The portability of Hadoop makes integration easier with heterogeneous hardware and software platforms.

Building a Storage Element (SE) [2] based on HDFS is a natural choice for CPU-intensive and data-intensive Grid computing since Hadoop has many features well fit into the distributed computing architecture: high scalability, reliability, throughput, portability, capability of running on heterogeneous platforms, low maintenance cost, etc. But the integration between Hadoop and existing Grid software and computing models is non-trivial. In addition to the general requirements on the SE, a Virtual Organization (VO) may implement special requirements depending on its data operation mechanism and data access pattern.

For example, in the Compact Muon Solenoid (CMS) [3] experiment at the Large Hadron Collider (LHC) [4], each recognized Grid site, ranging from Tier-0 to Tier-3, needs to run a standalone SE to fulfill some common tasks according to CMS VO's computing needs: moving data between sites and allowing Grid jobs to access data at the local storage. To accomplish these tasks, some technical specification and preference [5] for the SE technology include but not limited to

- SE technology must have a credible support model that meets the reliability, availability, and security expectations consistent with the computing infrastructure
- SE technology must demonstrate the ability to interface with the existing global data transfer system and the transfer technology of SRM tools [6] and FTS [7] as well as demonstrate the ability to interface to the CMS software locally through ROOT [8]
- The SE must have well-defined and reliable behavior for recovery from the failure of any hardware components. This behavior should be tested and documented.
- The SE must have a well-defined and reliable method of replicating files to protect against the loss of any individual hardware system.
- The SE must have a well-defined and reliable procedure for decommissioning hardware which is being removed from the cluster; this procedure should ensure that no files are lost when the decommissioned hardware is removed.
- The SE must have well-defined and reliable procedure for site operators to regularly check the integrity of all files in the SE.
- The SE must have well-defined interfaces to monitoring systems so that site operators can be notified if there are any hardware or software failures.
- The SE must be capable of delivering at least 1 MB per second per batch slot for CMS applications. The SE must be capable of writing files from the wide area network at a performance of at least 125MB/s while simultaneously writing data from the local farm at an average rate of

9781-4244-3962-1/09/$25.00 ©2009 IEEE

20MB/s.

- The new SE should be filled to 90% with CMS data. Failures of jobs due to failure to open the file or deliver the data products from the storage systems should be at the level of less than 1 in 10^5 level.

The rest of the paper describes a paradigm that takes HDFS as a key technology for the Grid SE solution. Initial integration of HDFS with other modular grid components to form a Grid SE was proposed by Bockelman in [9]. Later it received strong support by Open Science Grid (OSG) [10] and CMS. The development is under the scientific high performance computing domain. Comparing to the implementation of HDFS in business that is characterized as large scale but actually more dedicated for specific usage and well-controlled access to data, the scientific Grid computing needs to handle various types of applications and data access patterns, which may differ by several orders of magnitude in the required throughput and other benchmark measures with respect to data handling and I/O of the system. A more distributed, heterogeneous, chaotic and open environment is assumed and targeted for the applications of HDFS for the Grid.

II. HDFS-based Storage Element Architecture

The HDFS architecture is described in [11]. A brief review of major components and features of HDFS is provided in the following

- HDFS has a master/slave architecture. An HDFS system consists of a single NameNode and a number of DataNodes. HDFS exposes a file system namespace and allows user data to be stored in files. Internally, a file is split into one or more blocks and these blocks are stored in a set of DataNodes.
- NameNode is a master server that manages the file system namespace and regulates access to files by clients. The NameNode executes file system namespace operations like opening, closing, and renaming files and directories. It also determines the mapping of blocks to DataNodes.
- DataNodes, usually one per node in the cluster, manage storage attached to the nodes that they run on. The DataNodes are responsible for serving read and write requests from the file systems clients. The DataNodes also perform block creation, deletion, and replication upon instruction from the NameNode.

In order to build a SE which provides I/O and other Grid data services to the VO(s), we consider following software are needed to be seamlessly integrated with HDFS to fulfill all the basic required functionalities of the SE.

- **FUSE** [12]
  It is a linux kernel module, allows file systems to be written in userspace and provides POSIX-like interface to HDFS. With FUSE mount, the whole file system of the SE will be ”local” to the WN, which is crucial for user applications running at the WN to access data at the local storage.
- **Globus GridFTP server** [13]
  It provides WAN transfer between SE(s) or SE and worknernode (WN). Due to the fact that HDFS only supports synchronous write, we developed a special plugin to GridFTP. It is able to assemble asynchronous transfered packets to be sequentially written to HDFS, which is necessary for GridFTP running at multiple-stream-transfer mode to achieve optimal performance in file transfer.
- **BeStMan server** [14]
  It provide SRMv2 interface to HDFS. several plugins are developed to enable selection of GridFTP servers.

There are optional components that may be layered on top of HDFS depending on specific needs of a site or VO's computing strategy, for example XrootD [15], Apache HTTP server and Fast Data Transfer (FDT) [16].

We adopt a generic integrated architecture of HDFS-based SE, as shown in Fig. 1, that includes multiple level of data access and several possible ways of sharing those services on limited pieces of hardware, which is shown efficient, scalable and extensible :

- **Data access in the local domain**
  Direct data access via FUSE mount is the primary method to access the data at local domain, since a POSIX-compliant file system is necessary for many applications to access data at the SE. The support of this method largely depends on the configuration of the site, since it is not within the GSI authentication and authorization. The export of HDFS to the FUSE mount of a machine is subject to the security consideration of the site.
  Direct data access via Hadoop client. This method is similar to direct access via FUSE mount, except that it can only be used for file transfer. In the case of data access via FUSE, a site must explicitly authorize the export of the HDFS to the machine to be mounted, while in this case a more open and general access can be offered to a pool of machines for file transfer purpose.
  The BeStMan server provides a standard data provision and access method via SRMv2 protocol. It works as a frontend of transfer servers and on top of POSIX-compliant file system.
  The GridFTP server also provides a standard data provision and access method. A site can run multiple GridFTP servers to maximize the throughput of the file transfer with load-balance.
- **Data access between two SEs**
  SRM and GridFTP are the standard data transfer mechanism between SEs.
  Direct data access via FUSE mount between two SEs or a secure remote machine and SE is technically feasible, although it provides the easiest and simplest access to data including file transfer and running applications against data, the data access is limited by the throughput of single FUSE mount, which is more appropriate for user applications interactively accessing data at the SE locally.
  Direct data access via Hadoop client can also be supported between SEs, but it is also subject to security consideration.
- **Data and system security**
  Generally strong X509 authentication and authorization

are needed for the data access to the SE if creation and deletion of files are involved and initiated from a public host. These operations can be allowed via local FUSE mounted machines, since these machines are under the administration of local domain.

For security concern, the remote direct file operation via FUSE or Hadoop client should be prohibited.

A GSI-compliant mechanism like using GUMS server can be implemented to map the user X509 SSL certificates to local user or group accounts which are used by HDFS.

According to the roadmap of HDFS, access token, users authentication and data encryption will be supported that provide strong security for HDFS at the local domain in the future.

- **Sharing services in the cluster**

  The computing cluster consists of a number of WorkerNodes (WN). HDFS as a distributed storage solution can make best use of those WNs and form a pool of DataNodes, which lead to the sharing between WN and DataNode. The sharing between GridFTP and computing nodes (WN and/or DataNode of HDFS) is also feasible. The capability of HDFS supporting high throughout in inter-DataNode file replication and distribution allows a wide choices of configuration for the cluster and won't put physical limit or bottleneck on the service if the service is shared by a sufficient number of distributed hardware.

  From computing architecture point of view, the model of sharing data services with computing nodes is an economical choice for sites running commodity hardware to achieve high-end performance if the overall throughput is the major concern.

  An extra advantage of sharing service lies on the fact that HDFS distributes data to the whole cluster. Applications accessing data will naturally interact with a pool of DateNodes. The performance is mainly determined by the scalability of the SE and load balance among many pieces of hardware mediated by the computing model. For a Grid SE solution, the distributed and shared model is the most flexible and resilient one for serving a large group of users and various applications with their data access patterns.

III. Performance of HDFS-based SE at CMS VO

Three CMS Tier-2 centers, University of Nebraska Lincoln (UNL), California Institute of Technology (Caltech) and University of California San Diego (UCSD), deployed HDFS-based SE. During past 12 months, the scale of the HDFS at these sites increased continuously. Currently HDFS is the major file system for the SE of these site running around 100 DataNodes and providing total disk space ranging from 300 to 500 Terabytes each. These sites serve the CMS collaboration with up to thousands of grid users and hundreds of local users to access the datasets in HDFS at daily basis. The data operation (transferring and replicating datasets) and Monte Carlo production run by the CMS VO also intensively use HDFS.

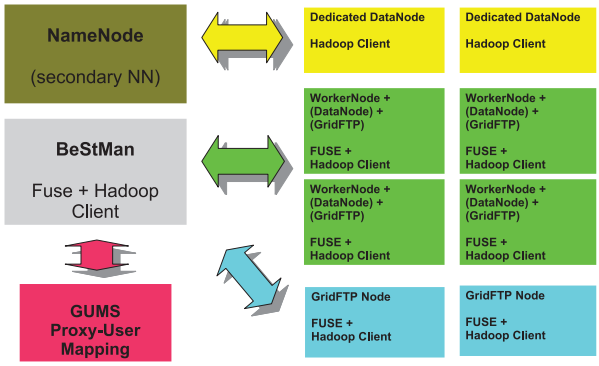

Fig. 1. Architecture of HDFS-based Storage Element

The experience of these sites shows that HDFS brings many benefits over alternative file system

- Reliability: it stops loss of files because of an excellent file replica scheme run by HDFS.
- Simple deployment: most of the deployment procedure is streamlined with fewer commands done by the administrators
- Easy operation: it is a stable system that requires little effort for the system and file recovery due to hardware or software failure, and user support.
- Proved scalability for supporting a large number of simultaneous Read and Write operation
- High throughput for serving the data for grid jobs running at the site

The performance of the presently deployed HDFS-based SEs is summarized in the following

- It stably delivers 10MB/s to applications in the cluster while the cluster is fully loaded with data processing jobs. This exceeds CMS application's requirement on I/O by an order of magnitude.
- HDFS NameNode serves 2500 request per second as shown in Fig. 2, which is sufficient for a cluster of thousands of cores to run I/O intensive jobs.
- It achieves sustained WAN transfer rate at 400MB/s, which is sufficient for the data operation of CMS Tier-2 centers including transferring datasets and staging out of user jobs. Fig. 3 shows the network throughput of data transfer as a typical data operation with the new SE.
- It can simultaneously process several thousand client's requests at the BeStMan server and achieve sustained endpoint processing rate above 50Hz as shown in Fig. 4, which is sufficient for high-rate transfer of gigabytes-sized files and uncontrolled I/O requested by random user jobs from across the CMS VO running on more than 30 thousand CPUs mainly provided by over 60 Tier-1 and Tier-2 sites.
- Extremely low file corruption rate is observed, which

mainly benefits from robust and fast file replication of HDFS.

- Decommissioning of a DataNode uses less than 1 hour, restarting NameNode in 1 minute, checking the image of file system from memory of NameNode in 10 sec, which are fast and efficient for the operation and system administration.
- In general, we expect that many of these performance characteristics scale with the amount of hardware provided.

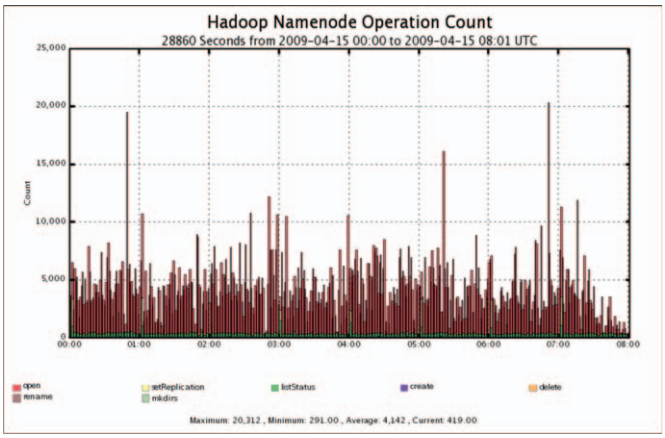

Fig. 2. The operation count at the NameNode of HDFS for a period of 8 hours. The operation of NameNode mainly involves file opening, replication, creation, deletion, renaming and etc. File opening (an indication of Grid jobs accessing data at the SE) dominates the operation.

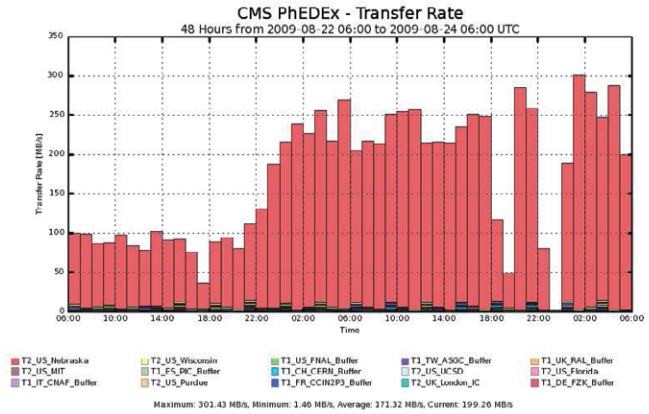

Fig. 3. Inter-site WAN Transfer between HDFS-based SEs of UNL and Caltech using CMS data transfer tool, PhEDEx. The transfer rate was measured in a period of 48 hours. Both site also received data from other CMS Tier-1 and Tier-2 sites. The average transfer rate for one site is 300 MB/s and peaked at 400 MB/s

IV. Operation

The monitoring and routine check are important daily tasks for the administration of the SE. Following is the summary of what have been accomplished and standardized to support the SE operation based on the experience of the sites running HDFS-based SE:

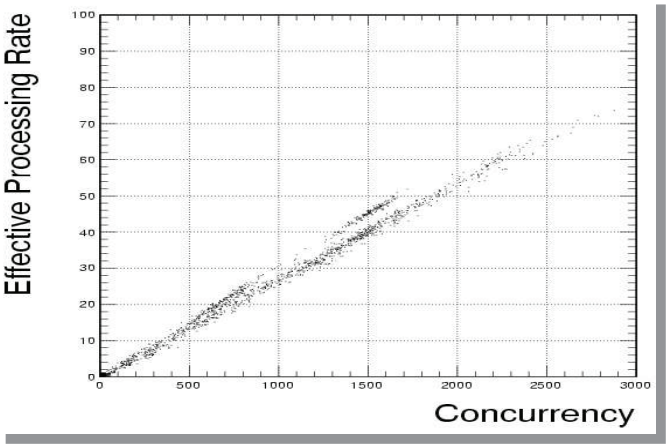

Fig. 4. The effective processing rate vs. number of concurrent client requests. The client's request is configured to be mainly processed by BeStMan server and not to involve much GridFTP and HDFS operations, to illustrate the performance of BeStMan server exclusively. Here the client requests for the test are "srmls" and "lcg-ls", to list the files in a directory at the SE. The BeStMan can process this command with the namespace of the file system via FUSE, which is very fast. So the processing rate is largely determined by the scalability of BeStMan itself. As we can see, the scalability of BeStMan server scales with the increasing number of simultaneous client requests.

- Integration with general grid monitoring infrastructure, including Nagios [17], Ganglia [18], and MonALISA [19]. The system information of CPU, memory, network for the NameNode, DataNode and the whole system are monitored by these tools.
- Incorporation with HDFS built-in monitoring tools, including Hadoop web service, Hadoop Chronicle, and Jconsole. These tools provide the status and in-depth information of the file system and user spaces.
- Analysis of logs in NameNode, DataNode, GridFTP and BeStMan servers are also part of the daily tasks and especially useful for system debugging.
- Regular low-stress test, i.e. running test analysis jobs accessing local dataset at the SE, load test of file transfer between SEs as shown in Fig. 5. These tests are part of the daily validation of the site mainly involving local and remote I/O of the SE.
- Intentional failure in various parts of the SE to test the system recovery mechanism.

The installation, upgrading, bug fix and deployment of new patches are important part of the administration of HDFS, which is also considered as an important factor in the cost of ownership of large scale of data. A streamlined release procedure of HDFS-based SE was tested among several sites as mentioned earlier. A more structured scheme on top of that is under development by the OSG.

- All software components needed for deploying HDFS-based SE are packaged as RPMs with add-on configuration and scripts necessarily to a site to deploy with minimal changes to adapt to the specific needs and requirements of a site. The release is essentially an integration based on various original open sources that provide all the necessary packages including HDFS, FUSE, BeStMan, GridFTP plugins, BeStMan plugins,

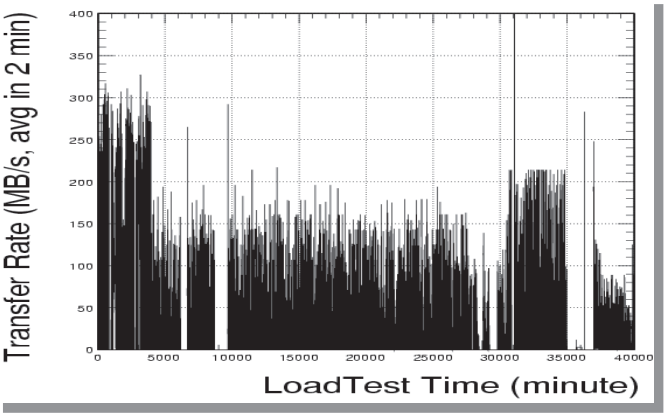

Fig. 5. A low stress test of inter-site WAN transfer between SEs of Caltech and UCSD. The transfer rate was measured in a period of 30 days. The average transfer rate is 150 MB/s.

etc.

- Consistency check and validation are done in selected sites before the formal release via OSG. A testbed is used for common platforms and scalability test.
- Future release procedure will be fully integrated into standard OSG distribution: Virtual Data Toolkit (VDT) [20]. There is possibility of some intersection with external commercial packagers, i.e., using selected RPMs from Cloudera [22]. Various flavors for different linux platforms are under the scope of the release.

We believe this release procedure significantly reduce the amount of efforts for a site to deploy HDFS-based SE from scratch and maintain it in a long run. The average downtime for a major upgrade is expected to take less than a few hours.

V. Site-Specific Optimization

Some optimization can be done for a site based on expected usage patterns and specific hardware condition. Several factors need to be evaluated and taken into the configuration when deploying HDFS-based SE:

- Size of file block. This mainly determines how well the data is distributed across the cluster and how many blocks of data the applications need to access.
  In addition to site-wide configuration, users can define their own size of blocks via HDFS client. The combination of site- and user-level configuration provides a good solution for data handling of various needs.
- Number of file replicas. This is mainly determined by a tradeoff among disk space, file security, throughput and latency in the data access.
  In addition to site-wide configuration, users can also define the number of file replicas to prioritize the data security over its "cost", use of disk space.
- Architecture of sharing data services (DataNode and GridFTP) and computing nodes. As discussed in previous Section, HDFS-based SE can fit into various configurations, especially for a site running commodity hardware. Some application's special data access patterns and requirements are possible factors that a site may need to consider running a standalone high-end storage system on top of HDFS, which doesn't share with computing nodes.
- Memory allocation at WorkerNode for GridFTP, applications, DataNode. This mainly applies to the case when sharing of resources in the cluster is implemented.
- Selection of data transfer servers by the SRM endpoint. Various models can be choosed: using the load and memory usage of GridFTP based on real-time-monitoring or a simple random selection from GridFTP pool if the data transfer service is highly distributed.
- Data access with MapReduce technology [21]. This is a special data processing tool that allows application directly accessing data blocks at the local DataNode and ignores the boundary of a file.
  Normally the data access model is developed and defined at the VO level. The use of MapReduce for the data processing will be an interesting and promising solution for those VOs that need to process very large amount of data.
- Rack awareness. This technology lets the application only access data closer to the computing nodes (e.g. in the same rack) that takes advantage of better network bandwidth. It mainly applies to a large cluster, in which the network latency and bandwidth have important impact on the throughput of the I/O and CPU utilization.
  To make the data local to the application requires integration between applications and data distribution at HDFS-based SE. As data volume and cluster size will increase dramatically in the future, this may be an important approach for a large site to optimize its performance in data handling.

VI. Conclusion

Hadoop-based storage solution is established and proved functioning at CMS Tier-2 sites and within the OSG community. CMS has adopted HDFS-based SE in its computing strategy.

- It is flexible in the architecture choice for a site, which incorporates various grid components. Its high scalability and stability well fit into the map of Grid computing. It is shown seamlessly integrated and interfaced with various grid middleware.
- VO and grid sites benefit from reliable HDFS file replica and distribution scheme, and high data security and integrity. It is shown excellent I/O performance for CPU- and data-intensive grid applications.
- HDFS carries low cost of deployment and maintenance, particularly useful to operate a large scale file system. It has very low limit for required hardware. It significantly reduces the manpower to maintain the system and increases quality of service. It is shown easy to adapt to new hardware and changing requirements.
- Standard release and support are becoming available for the scientific computing community.

References

[1] Apache Hadoop Project. Available: http://hadoop.apache.org.

[2] Storage Element on Open Science Grid. Available: https://twiki.grid.iu.edu/bin/view/Documentation/AboutStorageElements.
[3] The Compact Muon Solenoid Experiment. Available: http://cms.web.cern.ch/cms/index.html.
[4] The Large Hadron Collider. Available: http://lhc.web.cern.ch/lhc.
[5] Requirements for US CMS Tier-2 Facilities. Available: https://twiki.cern.ch/twiki/bin/view/CMS/USCMSTier2SERequirements
[6] Storage Resource Manager. Available: https://sdm.lbl.gov/srm-wg/doc/SRM.v2.2.html.
[7] The gLite File Transfer Service. Available http://egee-jra1-dm.web.cern.ch/egee-jra1-dm/FTS.
[8] A Data Analysis Framework. Available: http://root.cern.ch.
[9] Brian Bockelman, *Using Hadoop as a Grid Storage Element*, Journal of Physics, Conference Series 180 (2009) 012047
[10] Open Science Grid. Available: http://www.opensciencegrid.org.
[11] Hadoop Distributed File System. Available: http://hadoop.apache.org/common/docs/current/hdfs_design.html.
[12] File System in Userspace. Available: http://fuse.sourceforge.net.
[13] GridFTP Protocol. Available: http://www.globus.org/grid_software/data/gridftp.php.
[14] Berkeley Storage Manager. Available: https://sdm.lbl.gov/bestman.
[15] Scalable Cluster Architecture for Low Latency Access Using xrootd and olbd Servers. Available: http://xrootd.slac.stanford.edu.
[16] Fast Data Transfer. Available: http://monalisa.cern.ch/FDT.
[17] Nagios Monitoring System. Available: http://www.nagios.org.
[18] Ganglia Monitoring System. Available: http://ganglia.sourceforge.net.
[19] Monitoring Agent using a Large Integrated Services Architecture. Available: http://monalisa.caltech.edu/monalisa.htm.
[20] Virtual Data Toolkit. Available: http://vdt.cs.wisc.edu.
[21] Map/Reduce Technology. Available: http://hadoop.apache.org/mapreduce.
[22] Cloudera Project. Available: http://www.cloudera.com