Computing in SCIENCE & ENGINEERING
www.computer.org/cise

Lessons Learned from the SDSS Catalog Archive Server

Ani R. Thakar

Vol. 10, No. 6
November/December 2008

This material is presented to ensure timely dissemination of scholarly and technical work. Copyright and all rights therein are retained by authors or by other copyright holders. All persons copying this information are expected to adhere to the terms and constraints invoked by each author's copyright. In most cases, these works may not be reposted without the explicit permission of the copyright holder.

IEEE computer society

© 2008 IEEE. Personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

For more information, please see www.ieee.org/web/publications/rights/index.html.

Lessons Learned from the SDSS Catalog Archive Server

The Sloan Digital Sky Survey is one of the first very large archives in astronomy and other sciences. Experiences gained over the past decade of SDSS Catalog Archive Server development in software, hardware, and operational issues could provide useful advice for future archive projects.

The Sloan Digital Sky Survey, or SDSS, is a multi-institution, internationally funded project to map about half of the northern sky in unprecedented detail with a dedicated 2.5-m telescope and special-purpose instruments.[1] The SDSS Science Archive resulting from the survey contains the calibrated science data and is published in two formats: a raw image archive consisting of approximately 15 Tbytes of binary image files and accessible via an rsync/wget interface, and an approximately 6-Tbyte catalog archive that is loaded into a relational database management system (DBMS)[2] and is accessible via specialized Web-based tools and SQL query interfaces.

Large multiyear, multiterabyte archive projects can provide invaluable lessons. Given that SDSS is just the beginning of a data avalanche in the sciences, this is a good time to list the insights gained from the SDSS experience, along with any advice that might benefit future archive projects. On behalf of the SDSS Catalog Archive Server (CAS) team, I have collected the most generally applicable recommendations regarding software, hardware, and operational issues here for designers of other large scientific archives.

Software

We strongly recommend that the archive data be hosted in a DBMS unless

- the archive project has a large software development budget (unusual for scientific archives),
- there are specific and unusual requirements (such as high-security or mission-critical data), or
- the data access patterns are unusual and restricted (for example, in some particle physics data sets, the main type of query is searching for a few rare events in a huge data set).

A DBMS is the best and most versatile choice for guaranteeing data integrity, superior data mining capabilities, fast query performance, and high data availability.

DBMS Vendor Choice

The data repository is by far the most important part of the archive and the one most difficult to change or migrate to a new platform. Therefore, the DBMS must be the most reliable and longest-lasting software component. The DBMS vendor's maturity and stability are critical for a large archive project's success. Some disadvantages are associated with major commercial vendors such as Microsoft, Oracle, or IBM—for example, their

1521-9615/08/$25.00 © 2008 IEEE
Copublished by the IEEE CS and the AIP

Ani R. Thakar
Johns Hopkins University

This article has been peer-reviewed.

products might be much more expensive, and they might not be responsive to change requests from a single customer. Nevertheless, there are considerable advantages beyond financial viability and stability:

- *Market competitiveness.* A large vendor must remain competitive, so it's more likely to make timely upgrades to keep up with market trends. The DBMS product is also more likely to have all the essential features.
- *Product maturity.* An established product from a large vendor is less likely to have basic and major flaws; otherwise, it wouldn't have survived in the market. This is not to say that large vendors don't produce buggy products, but such products should be relatively rare, and their problems (hopefully) are well documented in the literature, so there should be no major surprises.
- *Size of development team.* DBMS software is complex enough that a large, well-managed software development effort is absolutely essential to create, maintain, and support a good product. A large vendor is much more likely to meet this requirement.

Several major noncommercial, open source DBMS products—including MySQL (www.mysql.com) and PostGreSQL (www.postgresql.org)—have done well in the DBMS market, have a significant installed base, and have a feature set that's increasingly competitive with commercial DBMSs. An open source DBMS would be a good option in my view, as long as the product can meet the scientific requirements and can be demonstrably scaled up or scaled out to the size of the archive under consideration. Not all DBMS products offer the scientific programmability and floating-point support that are critical for scientific archives, and certainly not all are scalable to the same degree. In opting for an open source DBMS, the same overall considerations apply: make sure there's enough long-term stability, maturity, and development support behind the product to keep up with the demands of the growing archive.

In the case of SDSS, the choice of a commercial DBMS vendor was a result of two coincidental developments: our inability to meet the performance requirements with our original choice of an object-oriented DBMS[3] (which should also serve as a cautionary tale for those intending to adopt an object-oriented DBMS platform), and our serendipitous collaboration with Jim Gray (Microsoft Research) and the consequential port of the SDSS early data release (EDR) data to SQL Server as an alternate (test) platform.

Although we didn't have the resources to do a comparative evaluation of the available DBMS products at the time, we did learn the lesson that a major vendor of a mature DBMS technology was a much better choice. There's no question that we benefited tremendously from the "divine help" that Gray was able to provide us in porting, maintaining, and tuning the SDSS catalog archive in SQL Server. But without the advanced query optimization capabilities, extensive programmability (functions and stored procedures), and native floating-point (single- and double-precision) support that were built into SQL Server, we wouldn't have stuck with this platform for very long.

SQL Server also happens to have considerably cheaper licensing terms (especially for academic institutions) than most of its major competitors in the DBMS market, a fact that is of no small importance for the tight budgets of most scientific archives. On the other hand, it isn't available for non-Windows platforms, which is also important in the scientific market.

Although we have received several requests for the SDSS catalog data from others wishing to port it to different DBMS platforms, to date we are unaware of a single successful port of the full SDSS catalog data and functionality to a platform other than SQL Server. We believe this is due to the data size and the schema complexity, with its large number of functions and stored procedures.[2]

Making the Most of the DBMS

Because it's likely that the DBMS will be the most expensive component in the project, it makes sense to make it work for you as much as possible. Modern relational DBMS products have come a long way and now offer many features not traditionally associated with databases.

Move the Analysis Inside the DBMS

In the large-data paradigm, avoid moving data unnecessarily as much as possible; modern relational databases provide the capabilities necessary to do a lot of the computation inside the database, right next to the data. Besides, databases are exceptionally good at returning only the data necessary for a given task and returning it fast. We've relentlessly employed the mantra "bring the analysis to the data" instead of the other (traditional) way around, and we believe this has served us well.

Minimize Custom Software Development

Our first version of the CAS was unsuccessful for two main reasons: the commercial object-oriented DBMS had severe performance flaws, and it required us to develop too much software outside the DBMS to meet the science archive requirements (including a query manager and a query optimizer). We were so overwhelmed with the size of the data that it was beyond our capabilities to develop additional custom software components to handle the query processing.

In fairness, the relational DBMS products available at the time also lacked some of the features that we needed (such as advanced query optimization), and our lack of resources to devote to software development were a result of underestimating the impact of the data size and underbudgeting for the software effort (2.5 full-time development resources). However, because this is likely to be a chronic condition in large science archive projects, we strongly recommend minimizing any custom software development as a rule of thumb.

True Cost of Software Development

A DBMS is admittedly a blunt instrument for certain computational tasks, and a better, faster way to do specific complex computational tasks often exists outside the database, via custom software. However, try this only with a full appreciation of custom software's true cost and always keep in mind that the data's sheer size will tax every available resource. No matter how innovative and efficient the custom software might be, there are good reasons to avoid this beyond the strain it places on the operations and development effort.

First, it's notoriously hard to benchmark performance in such a way that the benchmarks are applicable to changing hardware conditions (such as load), configurations, and platforms. A piece of code might perform extremely well in isolated tests and not so well under real conditions.

You also need to budget maintenance and support of software developed in-house for the project's duration. In conjunction with this, complex software can become a liability once the original developers have moved on. This is common in academia, where students and postdoctoral associates often develop the software.

The bottom line is that science archive projects that lack the proper infrastructure and manpower shouldn't be in the business of developing major software products unless absolutely no other option is available. Software development should be left to the professionals as much as possible.

Don't Reinvent the Wheel

This point is related to my two previous recommendations, but it's worth reiterating. To minimize the development effort, tools and interfaces must utilize off-the-shelf software products as much as possible. Traditionally, the scientific community is reluctant to spend money on commercial software and tends to build ad hoc software. In the long run, however, buying off-the-shelf products ends up being significantly cheaper than in-house development. Only very large labs and academic institutes can afford to maintain professional software development shops that can cost-effectively produce high-quality, reliable software.

Using and adapting existing standards falls into much the same category. SQL might not be a good fit for all applications and certainly isn't the scientific community's preferred choice. However, its wide acceptance as the lingua franca of the database world makes it the best choice for querying the database rather than developing a nonstandard, expensive, and potentially performance-degrading layer between the user and the data. There were other emerging and existing World Wide Web Consortium (W3C) standards that we decided to adopt along the way to provide some of our services, such as SOAP/XML Web services. This approach has benefited us tremendously, and we certainly recommend it to those designing other large archives.

Hardware

Although we don't have any reason to recommend specific hardware vendors or configurations for large data sets, some technologies worked well for us, and we found some general guidelines useful.

CPU and Disk

For the Web servers, we're using Intel Xeon 2.8-GHz processors (Dell PowerEdge 1750 or other manufacturer, such as SuperMicro) with 2 Gbytes of RAM. For the database servers (Data Release 6), we have dual-core AMD Opteron 248 2-GHz processors with approximately 4 Gbytes of RAM. We have a cluster of three Web servers for load balancing and high availability, and several database servers for each data release. SQL Server (especially 2005) makes good use of multiple CPUs, so quad-core and higher machines provide scalable performance for CPU-intensive queries.

However, the vast majority of our queries are I/O-bound, so the sequential I/O speed is the most important metric for us. We've been using serial advanced technology attachment (SATA) disks for all the recent data releases. They're set up as

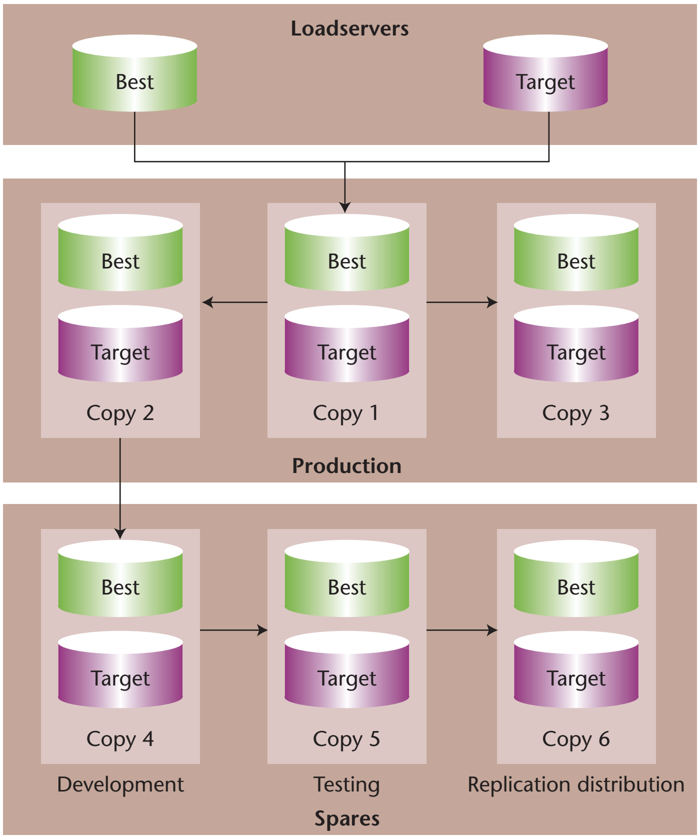

Figure 1. Copies of the SDSS Archive that are necessary for maintaining high availability. We needed three copies for production and three spare copies for development, testing, and replication and distribution to mirrors.

redundant arrays of independent disks (RAID 0) for the load servers, for data ingest where we need maximum read and write throughput. For the production servers, where we need decent read performance and fault tolerance, we use RAID 10. Earlier in the project, we used small computer system interface (SCSI) and parallel ATA drives, but now only the Web servers have SCSI, whereas all the database servers have 3Ware SATA drives.

We also looked into the storage-area-network (SAN) option a few years ago, but at that time, the price/performance ratio was too high for us. We are also considering serial attached SCSI now for the upcoming releases of SDSS-II and looking ahead to SDSS-III.

We tested RAID 5 for our production configuration but decided it wasn't a good option because the write throughput was considerably lower and wasn't reliable enough to justify the performance hit. This might have been a specific problem with the controllers we were using (3Ware), but they performed well in benchmarks so we had no a priori indication that there would be a performance problem in practice. In particular, we found RAID 5 to be unacceptably slow during index rebuilds (these operations were much faster with RAID 10), and the disks were generally more likely to fail (possibly owing to a complex RAID algorithm). Things were also slow during recovery and rebuild of a degraded disk.

Industry experts in the database field also warned us against using RAID 5. The wisest strategy is to assume that you're going to lose disks, or at least blocks on a disk, and use mirroring (RAID 10) for maximum fault tolerance. The additional disks are well worth the extra price.

Storage Cost of High Availability

Maintaining a reasonably fault-tolerant production environment while allowing for smooth and efficient data loading requires a certain minimum amount of disk storage. As Figure 1 shows, in addition to the loadserver, three copies of each data set are kept spinning on disk: the two production copies (one live and one warm spare) and the legacy copy that contains all the data served up to date. We maintain a ping-pong configuration for the loading—that is, we alternate between two servers so that while one of them is being loaded, the other is pressed into service as the live production server. To store backups of the task databases that are created during the loading process, each server must have twice the amount of storage required to store one copy of the archive. Generally, it's also a good idea to have this amount of spare storage in case recovering the database from a backup (temporarily requiring twice the amount of space) is necessary. This means that in addition to the space required for the legacy copies, disk storage equal to at least five times the size of the archive must be available.

Our experience has shown that even this sometimes isn't enough if we allow for disk fragmentation, disk errors, and so on. We recommend having six times the space required for one instance of the archive to be available. This might sound like a lot, but when dealing with terabyte data sets, you want to avoid having to reload the databases or having to copy them from one server to another as much as possible. Shuffling data between servers to make room isn't a good option. As an example, copying the BestDR6 3-Tbyte database from one server to the other takes the better part of two days, even with Gigabit Ethernet! The bottom line is that for high availability and less operational headaches, we recommend six copies in all of any given data set, with at least

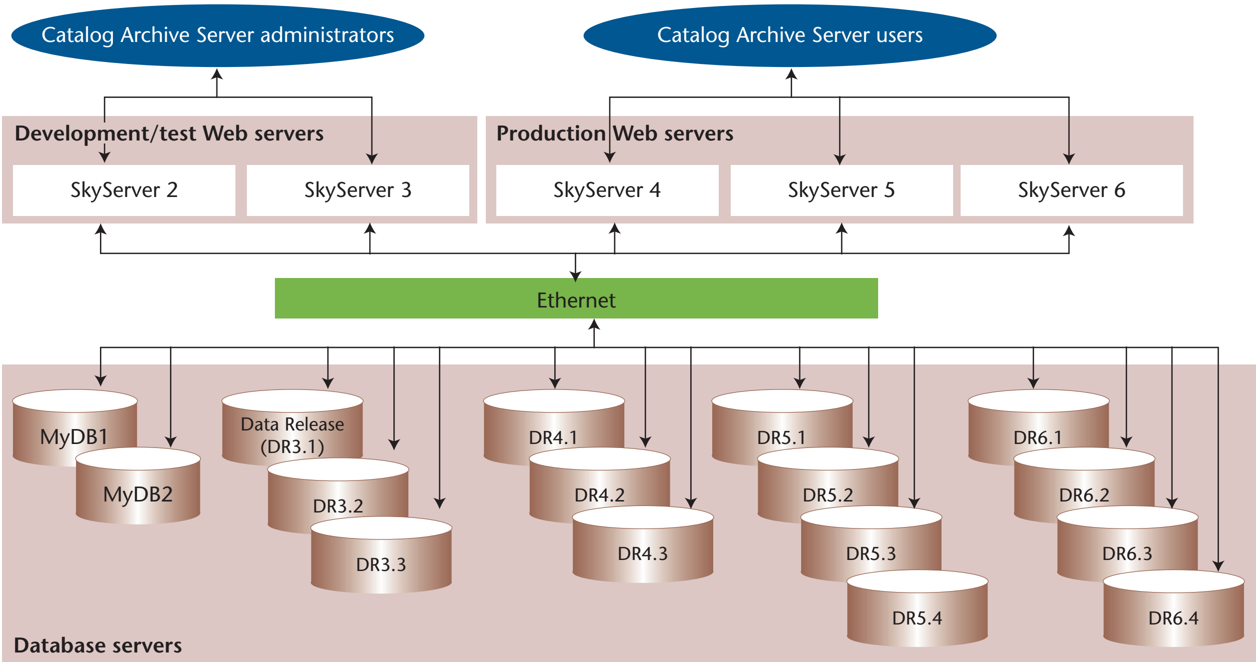

Figure 2. SDSS production server configuration at Fermilab, showing Web and database servers. Offline servers hosting spare copies of the data sets are not shown.

three of them on fast disks (the remaining copies can be on slower media).

Separate Web and Database Servers

For several reasons, not the least of which is security, we advise having separate Web and database servers so that user connections are made only to the Web servers and the database servers stay offline. This is also recommended for load balancing, of course. For popular data access sites handling several hundreds of thousands of Web hits per day or more, a cluster of Web servers connected to a layer of database servers will be necessary, as we have for the main SDSS site. Figure 2 shows our current configuration, which includes MyDB servers for the CasJobs/MyDB query workbench service[4] and load segregation (separating quick queries from long, intensive queries) for better utilization of resources.

Disk Failure Rates

We experience at least a 5 to 10 percent annual failure rate for disks. Of the more than 20 servers in service at Fermilab, at any given time one or two have disk problems, and the disks either must be replaced or the RAID must be rebuilt. Although this is anecdotal evidence, Carnegie Mellon scientists recently found failure rates up to 13 percent in a 2007 study,[5] much higher than manufacturers' claims. A study by Google found similar failure rates: an 8 to 10 percent annual failure rate for two- to three-year-old disks.[6] At supercomputing centers with large disk farms (such as the US National Center for Supercomputing Applications), even higher failure rates can occur, partly because disks in the same manufacturing batch tend to fail at the same time.

From a performance viewpoint, disks in the same batch might work better together, but they also tend to fail together. What all this means for archive administrators is that having extra copies of the data is crucial for high availability.

Moore's Law

Even if hardware performance increases at Moore's law rates, that doesn't mean that it's economically feasible for a large archive project to keep upgrading hardware at the rate necessary to keep up with the latest technology and performance enhancements. In SDSS, typically more than 80 percent of our servers and disks are more than two years old at any given time. As I mentioned earlier, disks tend to be replaced far more often than entire servers.

Operational Recommendations

We have a few other recommendations that large scientific archive projects might find useful.

IEEE computer society

PURPOSE: The IEEE Computer Society is the world's largest association of computing professionals and is the leading provider of technical information in the field.
MEMBERSHIP: Members receive the monthly magazine *Computer*, discounts, and opportunities to serve (all activities are led by volunteer members). Membership is open to all IEEE members, affiliate society members, and others interested in the computer field.
COMPUTER SOCIETY WEB SITE: www.computer.org
OMBUDSMAN: Email help@computer.org.

Next Board Meeting: 18 Nov. 2008, New Brunswick, NJ, USA

EXECUTIVE COMMITTEE

President: Rangachar Kasturi*
President-Elect: Susan K. (Kathy) Land, CSDP;* **Past President:** Michael R. Williams;* **VP, Electronic Products & Services:** George V. Cybenko (1ST VP);* **Secretary:** Michel Israel (2ND VP);* **VP, Chapters Activities:** Antonio Doria;† **VP, Educational Activities:** Stephen B. Seidman;† **VP, Publications:** Sorel Reisman;† **VP, Standards Activities:** John W. Walz;† **VP, Technical & Conference Activities:** Joseph R. Bumblis;† **Treasurer:** Donald F. Shafer;* **2008–2009 IEEE Division V Director:** Deborah M. Cooper;† **2007–2008 IEEE Division VIII Director:** Thomas W. Williams;† **2008 IEEE Division VIII Director-Elect:** Stephen L. Diamond;† ***Computer* Editor in Chief:** Carl K. Chang†
** voting member of the Board of Governors † nonvoting member of the Board of Governors*

BOARD OF GOVERNORS

Term Expiring 2008: Richard H. Eckhouse; James D. Isaak; James Moore, CSDP; Gary McGraw; Robert H. Sloan; Makoto Takizawa; Stephanie M. White
Term Expiring 2009: Van L. Eden; Robert Dupuis; Frank E. Ferrante; Roger U. Fujii; Ann Q. Gates; CSDP; Juan E. Gilbert; Don F. Shafer
Term Expiring 2010: André Ivanov; Phillip A. Laplante; Itaru Mimura; Jon G. Rokne; Christina M. Schober; Ann E.K. Sobel; Jeffrey M. Voas

EXECUTIVE STAFF

Executive Director: Angela R. Burgess; **Director, Business & Product Development:** Ann Vu; **Director, Finance & Accounting:** John Miller; **Director, Governance, & Associate Executive Director:** Anne Marie Kelly; **Director, Membership Development:** Violet S. Doan; **Director, Products & Services:** Evan Butterfield; **Director, Sales & Markting:** Dick Price

COMPUTER SOCIETY OFFICES
Washington Office. 1828 L St. N.W., Suite 1202, Washington, D.C. 20036-5104
Phone: +1 202 371 0101 • Fax: +1 202 728 9614
Email: hq.ofc@computer.org
Los Alamitos Office. 10662 Los Vaqueros Circle, Los Alamitos, CA 90720-1314
Phone: +1 714 821 8380 • **Email: help@computer.org**
Membership & Publication Orders:
Phone: +1 800 272 6657 • Fax: +1 714 821 4641
Email: help@computer.org
Asia/Pacific Office. Watanabe Building, 1-4-2 Minami-Aoyama, Minato-ku, Tokyo 107-0062, Japan • Phone: +81 3 3408 3118
Fax: +81 3 3408 3553 • Email: tokyo.ofc@computer.org

IEEE OFFICERS

President: Lewis M. Terman; **President-Elect:** John R. Vig; **Past President:** Leah H. Jamieson; **Executive Director & COO:** Jeffry W. Raynes; **Secretary:** Barry L. Shoop; **Treasurer:** David G. Green; **VP, Educational Activities:** Evangelia Micheli-Tzanakou; **VP, Publication Services & Products:** John Baillieul; **VP, Membership & Geographic Activities:** Joseph V. Lillie; **VP, Standards Association Board of Governors:** George W. Arnold; **VP, Technical Activities:** J. Roberto B. deMarca; **IEEE Division V Director:** Deborah M. Cooper; **IEEE Division VIII Director:** Thomas W. Williams; **President, IEEE-USA:** Russell J. Lefevre

revised 17 June 2008

IEEE

Usage and Traffic Logging

Our analysis of the first five years of the SDSS CAS usage illustrates the kind of information and analysis that's possible when archive usage is tracked and logged to a weblog database.[7] However, there are several other reasons why fastidiously logging all user activity is important. We've found that the SDSS traffic database has helped us in the following important ways:

- Usage/traffic profiles are a great management resource. Statistics and trends derived from the usage data are invaluable for funding proposals and reviews. They help management decide if the archive is meeting user requirements. They are also useful to monitor the impact of press releases or articles about SDSS.
- The traffic data can be used for resource management and load balancing. We have used it occasionally to find crawlers and inconsiderate users that hog resources and make the system unusable for others. We have also used it to track server performance.
- Analyzing the user SQL queries can guide schema and interface design. We have looked at how often the SDSS photo flags were being used and if users were able to get clean photometry data. Another important check was to see if users were filtering out invalid data values properly.
- Studying user queries revealed users' level of comfort with SQL. We were able to assess how quickly users were learning SQL and what fraction of users could formulate complex queries.
- Tracking failed queries and errors uncovers bugs in the system. Even looking at the incidence of HTTP error codes is a quick way to find broken links or services.
- Analyses of the usage and performance data guide mirror site design. It was useful for prospective mirror sites to budget hardware for their user communities.

Operational Databases

In addition to the science data, it's a good idea to stuff the operational and commissioning-test data into the DBMS to facilitate operations, quality assurance, and testing. This can be in a separate database if necessary. Pipeline inputs and outputs can also be tested efficiently if the data is in a database. SDSS has an operational database called OpDB, and even though it isn't a relational database, it helps SDSS scientists run tests, plan spectroscopic targeting, and perform other operations. In general, the faster you get your data into a DBMS, the better.

Data Releases

Most archives will need to release data at least periodically during the data collection phase. If there will be official data releases, they should be planned and budgeted with the knowledge that they will have to be supported indefinitely. Data, once published, cannot be retracted and should be considered immutable, especially because it will be used in research publications. As such, there will usually be a need to maintain online access to multiple data releases simultaneously (as Figure 2 shows). It's often cleaner to do it this way than to subsume older releases in new ones. It's also important to optimize the data loading process and pipeline for such periodic incremental data releases.[8] These considerations argue for having the smallest number of data releases necessary to meet the scientific requirements.

Data Distribution

Distributing the data to mirror sites and even individual users can take a considerable slice out of the archive's operational budget. Allocating resources for this ahead of time is important. We've been lucky that the National Center for Data Mining at the University of Illinois, Chicago, has agreed to serve as our official data distribution site (http://sdss.ncdm.uic.edu). NCDM has developed UDT, a fast UDP-based peer-to-peer (P2P) data transfer protocol, along with an application called Sector that can transport data at several times the maximum speed achievable with TCP/IP. Using Sector, researchers transported the 1.3-Tbyte BestDR5 (compressed) data set from the University of Illinois, Chicago, to the SC '06 floor in Tampa, Florida, with a sustained data transfer rate of 8 Gbytes/s over a 10 Gbytes/s link, and a peak rate of 9.18 Gbytes/s.[9]

Without some such fast protocol that makes it easier to squeeze the maximum performance out of TCP/IP and a data distribution site that has high-bandwidth connections to the rest of the world, distributing terabyte or larger data sets to mirror sites can be a real challenge, if not impossible. In fact, such a fast data transport protocol can be invaluable even for making local copies for backup or load balancing, which can take days even with Gigabit Ethernet. Our experience is that with TCP/IP alone, the data transfer starts off at wire speed, but the speed drops off at some point, and most of the data is copied at a fraction of the maximum speed available. At the very least, the resources (personnel and hardware) required for massive data distribution must be part of the archive planning.

The SDSS project has just completed its first two phases—SDSS-I and SDSS-II—and is now poised to begin a significantly different science program with SDSS-III (www.sdss3.org). The final data release for SDSS-II is DR7. In spite of some missteps and hard lessons learned, this ground-breaking multi-institution, international project, and the resulting science archive have been an amazing success and should serve as a model in many ways for future and much larger archive projects.

Acknowledgments

Funding for the SDSS and SDSS-II was provided by the Alfred P. Sloan Foundation, the US National Science Foundation, the US Department of Energy, NASA, the Japanese Monbukagakusho, the Max Planck Society, and the Higher Education Funding Council for England.

References

1. A. Szalay, "The Sloan Digital Sky Survey," *Computing in Science & Eng.*, vol. 1, no. 2, 1999, pp. 54–62.
2. A.R. Thakar et al., "The Catalog Archive Server Database Management System," *Computing in Science & Eng.*, vol. 10, no. 1, 2008, pp. 30–37.
3. A.R. Thakar et al., "Migrating a Multiterabyte Astronomical Archive from Object to Relational Databases," *Computing in Science & Eng.*, vol. 5, no. 5, 2003, pp. 16–29.
4. N. Li and A.R. Thakar, "CasJobs and MyDB: A Batch Query Workbench," *Computing in Science & Eng.*, vol. 10, no, 1, 2008, pp. 18–29.
5. B. Gengler, "Hard Disk Failure Rates 'Up to 13pc,'" *The Australian*, 17 Apr. 2007; www.australianit.news.com.au/story/0,24897,21553519-15321,00.html.
6. E. Pinheiro, W.-D. Weber, and L.A. Barroso, "Failure Trends in a Large Disk Drive Population," 2007; http://research.google.com/archive/disk_failures.pdf.
7. V. Singh et al., *The SkyServer Traffic Report—The First Five Years*, tech. report, Microsoft, 2007.
8. A. Szalay et al., "The sqlLoader Data Loading Pipeline," *Computing in Science & Eng.*, vol. 10, no, 1, 2008, pp. 38–49.
9. Y. Gu et al., "Distributing the Sloan Digital Sky Survey Using UDT and Sector," *Proc. 2nd IEEE Int'l Conf. E-Science and Grid Computing* (E-Science 06), IEEE CS Press, 2006, p. 56.

***Ani R. Thakar** is a research scientist in the Center for Astrophysical Sciences at the Johns Hopkins University. His research interests include science with large databases, data mining, and simulations of interacting galaxies. Thakar has a PhD in astronomy from the Ohio State University. Contact him at thakar@jhu.edu.*

For more articles on the SDSS archive, see our January/February 2008 issue at http://csdl2.computer.org/persagen/DLAbsToc.sp?resourcePath=/dl/mags/cs/&toc=comp/mags/cs/2008/01/mcs01toc.xml. *—ed.*