NDIIPP Models for Mass Data Transmission and Storage

Mike Ashenfelder, Andy Boyko, Jane Mandelbaum, David Minor, Robert McDonald, and Richard Moore

Abstract

Preservation of digital content into the future will rely on the ability of institutions to provide robust system infrastructures that leverage the use of distributed and shared services and tools. The academic, nonprofit, and government entities that make up the National Digital Information Infrastructure and Preservation Program (NDIIPP) partner network have been working toward an architecture that can provide for reliable redundant geographically dispersed copies of their digital content. The NDIIPP program has conducted a set of initiatives that have enabled partners to better understand the requirements for effective collection interchange. The NDIIPP program partnered with the San Diego Supercomputer Center (SDSC) to determine the feasibility of data transmission and storage utilizing the best of breed technologies inherent to U.S. high-speed research networks and high-performance computing data storage infrastructures. The results of this partnership guided the development of the Library of Congress's cyberinfrastructure and its approach to network data transfer. Other NDIIPP partners, too, are researching a range of network architecture models for data exchange and storage. All of these explorations will build toward the development of best practices for sustainable interoperability and storage solutions.

Recently, as part of the National Digital Information Infrastructure and Preservation Program (NDIIPP) data transfer activities, an NDIIPP partner shipped two hard drives overnight to the Library of Congress containing archived websites. Instead of the packages arriving the next day to

LIBRARY TRENDS, Vol. 57, No. 3, Winter 2009 ("The Library of Congress National Digital Information Infrastructure and Preservation Program," edited by Patricia Cruse and Beth Sandore), pp. 541–555

the waiting data wrangler (who manages the movement and processing of bits), the packages roamed around Washington for several days. As the wrangler described it,

> The person who prepared the packages for shipping got my first name wrong, which is not so bad. Though I specified Federal Express as the shipping method, they shipped the packages via UPS and addressed the packages to the local Federal Express office. When the UPS delivery man went to the Federal Express office with a package for someone with my name, the FedEx woman told him that no one by that name worked there. The package went back to the UPS warehouse.
>
> Later that day, I tracked down the package and spoke with the UPS customer representative, who promised to deliver it the next day. A day later I checked the package tracking number on the UPS website and found that someone at the Library had signed for the package but UPS just wrote down a single vague name, which could have been a first or last name. Marshall or Arthur or something like that. I tried looking up that name in the Library's phone directory but several people had that name. After a few dead-end calls, it became obvious that I would have to wait until someone called me. Two days later the call finally came. The packages were sitting in one of the other Library buildings, a block away. I was promised that someone from my division would drop the packages off at my desk. The next day I got a phone call saying that the packages were too big for the administrative assistant to carry, and I'd have to come get them myself.
>
> I went on to:
>
> 1. Pick up the boxes and walk them to the Library's transfer-server area
> 2. Go online back at my office and submit a request for our system administrator to connect the drives to the transfer server
> 3. Run the UNIX command to transfer the data from the transfer server to another server, a way station on the way to permanent storage
> 4. Run checksums on the newly transferred data
> 5. Email the NDIIPP partner that we successfully received and verified the data
> 6. Submit a request for the system administrator to disconnect the drives from the transfer server
> 7. Get a FedEx shipping label from the Library's administrative offices in order to ship the drives back to the NDIIPP partner
> 8. Get the drives and the boxes they were shipped in from the transfer-server area
> 9. Take those boxes to our shipping division

> 10. Repack the boxes, tape them, label them and give them to our shippers
> 11. Email the NDIIPP partner the FedEx tracking number.
>
> When you add up all the labor and shipping time, it took about three weeks from the beginning to the end of the transfer.
>
> The odyssey of those drives represents all that is wrong with the current methods of transferring data. And more could have gone wrong. The drives could have been damaged in transit, the checksums might not have verified, the drives might have gotten shipped to Canada by mistake, and more. These have all happened to us.

Such stories—though all too common—are decreasing as the Library of Congress improves its network data transfer capability. And as they refine their tools and procedures their progress will benefit the NDIIPP partners, leading to an efficient interoperability among everyone. Half of the digital-preservation challenge is getting the data to its destination.

In the past few years, the Library of Congress had two significant events that influenced—and continue to influence—transfer operations: membership in the Internet2 consortium and a collaborative project with the San Diego Supercomputer Center. This report is an account of the SDSC project: the requirements, the challenges, the methods, the tools, and the results. The outcome of this project altered the Library's practices and the lessons continue to reverberate.

Collaboration between the Library and SDSC

Up to a few years ago transferring data to and from the Library of Congress primarily consisted of small sets of metadata files transferred over the network, or larger sets of content delivered on tangible media such as CDs. Network transfers were frustratingly slow, and the transfer rate—the speed at which the data moves from one place to another—was a small percentage of what it is today. The procedure for shipping tangible media had not changed much over time, with the additional complication of security-related mail delays implemented on Capitol Hill. The Library had begun to receive requests for more regular and sizable data transfers, particularly for digital content designated to be ingested and managed by the Library.

The SDSC project was a test of large-scale data transfer and storage. At the heart of the project was the issue of *trust*, specifically how the Library of Congress could trust SDSC to reliably transfer and store terabytes of the Library's precious digital assets. The Library needed assurances that SDSC could move the data from our servers to their servers intact, and that once the data was at SDSC it was not at risk for loss or corruption.

The project also provided an opportunity to explore the impact of high-end networking between digital libraries.

The content consisted of two different types of digital data from two divisions within the Library:

- Approximately 5 TB of harvested Web Sites, tens of thousands of individual files that comprised the Web Sites bundled and compressed into 500 MB ARC-formatted files.
- Approximately 580 GB of digital image files—high-density master TIFF files and their derivatives—and associated files that support their display.

It did not matter what the file types were, since the Library was only concerned with moving bits from one place to another.

The Library devised several test scenarios, including: uploading, downloading, and deleting data; deleting data without telling anyone at SDSC to see how the system would react; and conducting "doomsday" tests simulating a crisis in which the Library's copy might become unavailable.

Storage Preparation at SDSC

SDSC created on-site user accounts for the Library and the Library, in turn, created accounts on a Library-hosted development server for the SDSC technicians. SDSC leveraged much of their existing infrastructure and processes, and their storage was divided into three parts:

- An ingestion machine, which initially hosted the data arriving from the Library
- The back-end storage, which included live disks and tape-based archival storage systems
- Access machines, which housed the various mechanisms used by the Library staff

This model was based on the standard storage model used by SDSC's Data Central group (see fig. 1).

This diagram shows several SDSC storage management systems. The two most important are the High Performance Storage System (HPSS) and SAM-QFS. HPSS was developed by IBM in conjunction with several Department of Energy laboratories; SAM-QFS was a product from Sun Microsystems. While they have slightly different feature sets, both systems provided the same functions for this project: the ability to store large amounts of data on both disks (for fast access) and archival tapes (for long-term reliability). SDSC stored identical copies of the Library data on both systems for reliability. It also provided tests for sharing and logging data among systems.

The pieces of storage infrastructure were functionally separate from each other. The ingest machines were separate from the storage machines, which were separate from the front-end machines. This provided an opportunity for more specific storage system management.

One of the goals for high-reliability storage was the elimination of sin-

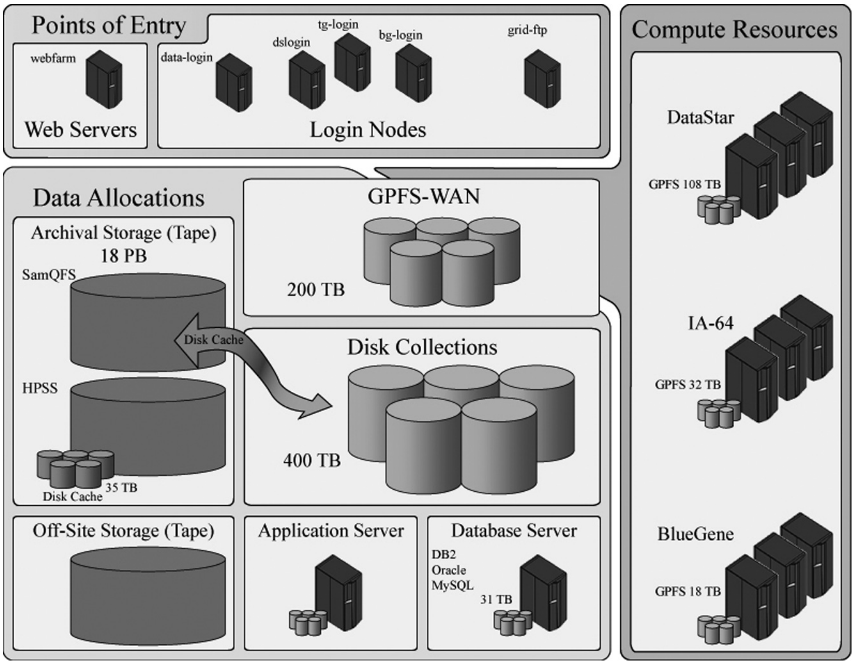

Figure 1. Storage Infrastructure at SDSC. Courtesy of Bryan Banister, Storage Systems Group, SDSC 2006.

gle points of failure. By isolating the sections of storage infrastructure, there were fewer opportunities for one event to take everything down.

This complex infrastructure required more management, and different SDSC groups managed different pieces. While this normally works well, there was an instance—which we will describe later—where this complexity led to an accidental cutoff of access for the Library.

During the course of the project, SDSC acquired a piece of test equipment from COPAN Systems (http://www.copansystems.com): a Massive Array of Idle Disks (MAID), an alternative to traditional tape libraries. It consists of many hard disks of which only maybe 25 percent is spinning at one time. It does not offer the immediate retrieval of an always- spinning hard disk, like our standard sharable enterprise storage, but it does offer a quicker retrieval time then fetching data from a tape library.

SDSC's MAID array was one of the first production models in the world. With the agreement of staff at the Library, SDSC put a third copy of all the Library data on the MAID system to see how it performed.

SDSC storage solution provided four points of storage for the Library data:

- Spinning disk
- Massive array of idle disks (MAID)
- Tape (copy 1)
- Tape (copy 2)

SDSC had increased its storage presence at two remote sites: the National Center for Atmospheric Research (NCAR) in Boulder, and the Pittsburgh Supercomputer Center (PSC). As a demonstration of enhanced data storage, SDSC replicated a portion of the Library data at these sites. The replicated data was considered to be a static archive, not directly accessible by anyone at the Library and only available if needed in an emergency.

SDSC used its collection-management system, the Storage Resource Broker (SRB, http://www.sdsc.edu/srb/index.php/Main_Page). SRB provided all of the data management, replication, logging, and auditing services required by the project, and maintained metadata about each stored object. Because the data sets within the project had different structures, they were stored in SRB in different ways, but to the end user these differences were irrelevant.

Drive Shipment

Five months before the beginning of this project, the Library—as a member of the Internet2 consortium (http://www.internet2.edu/)—installed a high-bandwidth connection to the Mid-Atlantic Crossroads (MAX) gigapop (http://www.maxgigapop.net), a network access point at the University of Maryland. From there the University of Maryland connected to the Internet2 network. The Library's development team for this project was not aware of any Library tools or processes for transferring large quantities of data over Internet2. Up until then, large data transfers to the Library were primarily done by shipping tangible media such as hard drives.

Shipping drives to SDSC, while familiar, was time consuming and labor intensive, depending on the time it took for:

- The Library to copy the data to disk, run checksums, pack the drive in a box and ship the drive to SDSC
- SDSC to receive the disk, install it, verify the checksums and transfer the data to servers.

In time the Library's data-transfer discussions with SDSC centered almost exclusively on the processes for transferring data over Internet2. Still, the Library decided to also transfer data to SDSC on disks in order to measure how long the end-to-end process would take. It was a comparative metric: how long it took to ship a drive versus how long it would take to transfer data online.

The Library ran checksums on its data, using the SHA-1 algorithm and bundled the resulting checksum files with its source data. The Library then shipped the data, which spanned nineteen hard drives, to SDSC.

Once the data arrived at its destination several days later, SDSC technicians connected the drives to their servers and transferred the data. It took about four hours to transfer data from disk to drive, and one to four hours to run a checksum on the contents of each drive. Given the number of drives and that SDSC transferred the data from the drives during their workday, it took over a week to complete the transfer. The entire process to transfer approximately 5 TB shipped via drives, end to end from the Library to SDSC, took a few weeks.

There was one odd checksum incident that stumped everyone for a few days. It is unusual but worth noting.

When SDSC transferred the contents of one of the shipped disks and verified the checksums they got a surprise: not one of the checksums matched. There was an error for every file that they transferred.

The Library reran a check on its source files and found that the data had not changed since they ran the original checksums. But SDSC confirmed that not one of the checksums on the transferred data matched.

After a few days of head scratching and intense investigation, the Library discovered that SDSC was using an older version of SHA1deep (Federal Information Processing Standards, 1995) for the checksums; the Library had v. 1.12 and SDSC had the older v.1.11. The Library also discovered that there was a known bug on 64-bit machines (which both the Library and SDSC have), the SHA-1 portion of the toolset doesn't compile correctly on AIX (which SDSC was using), and that version 1.12 fixed. However, if both institutions used the same older version (1.11) there would have been no discrepancy in the checksums.

SDSC reran the checksums using MD5deep (http://md5deep.sourceforge.net/), not the more-advanced SHA1deep, and the checksums verified. So the Library agreed to only run MD5 for the rest of the project.

Tuning Up

Given the tediousness of the shipping process, network data transfer became more and more appealing to the Library. It seemed like a quicker and cleaner automated option, with fewer steps and less human involvement. But there were many lessons for the Library to learn.

It was not enough for the Library to connect to the Internet2 network; the Library also had to increase the speed and efficiency of its own internal network environment. A super fast transfer rate over the Internet2 pipe was not useful if the Library moved data onto it at a relatively sluggish rate. It would be like pedaling a bicycle on a superhighway. The potential for speed is great but the vehicle was relatively slow.

As of 2007, the data transfer rate over the MAX gigapop to the Internet2 node at the University of Maryland was approximately 1 Gb/s. To estimate how much data the Library might potentially transfer in a day, they used the following formula as a point of reference: *100 Mb/s = 1 TB* *per day*. So at 1 Gb/s (ten times 100 Mb/s) the Library could potentially transfer 10 TB per day; that rate is not likely but it is possible. In time the Library came to appreciate the more modest transfer rate of 100 Mb/s as a good target. A faster transfer rate would be desirable—but 1 TB a day was fine.

Most operating systems have a default setting that limits the amount of system memory that can be used by any one TCP connection. This default setting is lower than what modern networks can handle, so the Library had to tune its servers to increase the transfer rates.

SDSC and the Library optimized the network components between the Library's development server and the Internet2 network. The Library installed a new Linux-based router at the edge of its test environment, replaced some cables and did some reconfiguration. Eventually, using the network bandwidth testing tool Iperf (SourceForge, n.d.) within the Library, SDSC and the Library recorded *internal* transfer rates of 605 Mb/s (6 TB per day).

Parallelization and Transfer Tools

Once the network was optimized—the pipes were cleared, so to speak—the transfer mechanism became the Library's next challenge. SDSC technicians pointed out to the Library that transferring files in a *single* stream would be a waste of the enormous Internet2 bandwidth potential and any data-transfer tool the Library used should maximize the bandwidth by splitting data into multiple concurrent streams. "Parallelization" became the Library's first important and significant lesson.

SDSC technicians conducted an initial single-stream Iperf test from SDSC to the Library development server and measured the network performance at a low 4.26 Mb/s. As they opened new simultaneous streams the transfer rate climbed. Eventually, they achieved transfer rates of approximately 240 Mb/s via fifty-five streams. Beyond that they discovered a thread threshold: fifty-five threads were about optimum and sixty-five offered no significant transfer-rate advantage.

Adjusting the number of streams was not enough to optimize the rate though. SDSC technicians had to adjust other parameters, including tuning the Library's firewall and adjusting the TCP stack and buffer. When SDSC and the Library were finished, four months after the project began, the Library consistently and reliably transferred files online with an average rate of 200Mb/s, about 2 TB per day.

SDSC noted that when a transfer interruption occurs a good tool should continue the transfer from the point of interruption. Also the tool should aggregate the files, combining many files into a collective unit. Aggregation is not essential, but it reduces the number of files to transfer; the fewer files you have to transfer, the better. This was another important lesson. Transfer tools require extra communication for each file transferred over the network, a "handshake" between the endpoints to request, respond, transmit, and acknowledge the transfer of the data. This transaction takes time (see fig. 2).

Though it is only a matter of milliseconds, the mass of transactions can accumulate. Also, the greater the physical distance between the transfer points—such as the 2,500 miles between Washington, DC and San Diego, California—the greater the potential for latency and the more noticeable the lag time as requests, responses, data packets, and acknowledgments whiz past each other. A "waiting period" between the request and response might occur. Reducing the number of transferred files not only lowers the risk and cumulative effect of latency but also reduces the risk of a timeout or transfer interruption.

GridFTP

The Library and SDSC discussed a few different tools for high-speed data transfer. SDSC favored GridFTP, an application from the Globus toolkit (http://globus.org/toolkit/) that is optimized for high-bandwidth networks. SDSC's experience is that GridFTP is commonly used in the high-performance computing world. After some discussion the Library agreed to use GridFTP, and SDSC took responsibility for its installation and administration.

GridFTP installation required custom configuration and many iterative steps between the Library and SDSC. The setup was complicated by the nature and complexity of enterprise computer system security configurations. The Library's computing infrastructure is designed to meet the requirements of serving Congress and the public; it uses firewalls and role-based user privileges for security. GridFTP has greater security needs, using certificate-based authentication that binds cryptographic information.

The Library felt GridFTP was much more than they needed for simple data transfer. One Library technician commented that it was "like driving a Maserati to the corner store."

In time GridFTP transferred the bulk of the Library's data over the network. And at 2 TB per day it did not take long to pull 5 TB of data to SDSC. But the Library knew that when they were on their own without the benefit of SDSC's GridFTP expertise, they would need a simpler transfer tool that could be easily installed and used in multiple scenarios. The Library deferred its search for another data-transfer tool until after the SDSC project ended.

File System Structures

Another issue arose with file system structure compatibilities. When the Library looked for the files after a transfer, many appeared to be missing. This was eventually traced to a complex file structure at the Library side (with symbolic or virtual links) that did not carry over through the transfer.

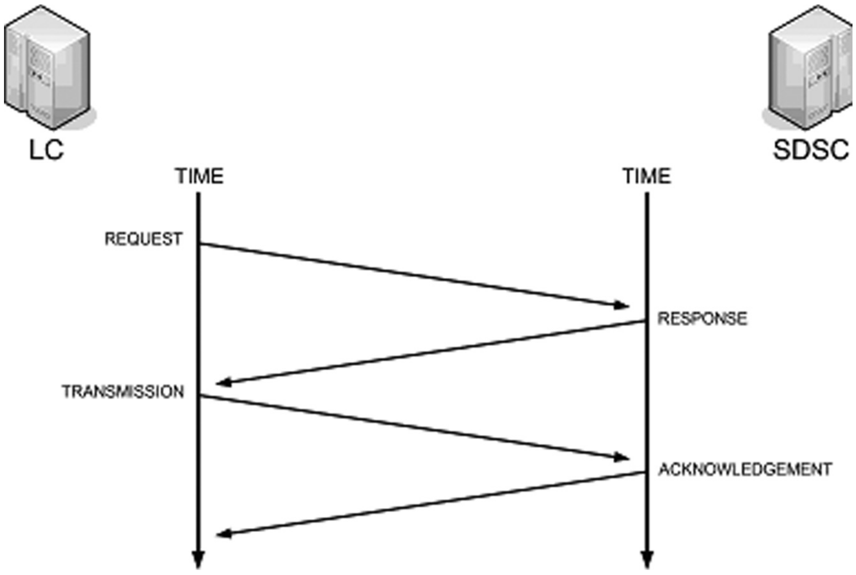

Figure 2. Data handshake. Courtesy of the Library of Congress 2008.

SDSC suggested that it could ignore the Library link structure and generate a new structure that would allow correct functionality. But that would violate one of the Library's project requests: to retain the same structure established at the Library.

Eventually SDSC created a workaround, making a link at the top of the file system that simulated the path the files were looking for, which in turn pointed to where the files were on the SDSC file system. SRB tracked all the files, followed the links and registered the real files and not the link files. But this problem highlighted another lesson: identify potential issues in file system practices.

Data Reliability versus Availability

In the middle of the project there was an incident that resulted in the Library and other SDSC clients losing access to their SDSC-hosted data for almost two weeks. One of SDSC's main storage systems (SAM-QFS) developed problems with its metadata services, and there was no way for the Library to see its data, though it was still there. During the repair period, SDSC took the SAM-QFS file system offline rather than risk providing a resource that was unpredictable and unreliable.

There was no problem with the Library's data stored at SDSC. SAM-QFS was only one of three separate file systems on which the data was stored. And even on the SAM-QFS file system there was never a threat of data loss or corruption, since only the metadata servers were malfunctioning.

However, this incident exposed how SDSC had designed access to the data for Library staff: the primary front-end for the data was itself located on SAM-QFS. This meant that whenever SAM-QFS was down, access was restricted. Once SDSC had repaired the problem, they took corrective measures to eliminate this potential point of failure in data access.

This incident highlighted another important issue for the Library: there is a difference between *reliability* and *availability* of data.

The Library's data was stored *reliably* at SDSC. Given the four layers of storage and replication, and the offsite geographic replication at NCAR in Colorado, the Library's data was intact, 100 percent of the time.

However, during this incident, it caused frustration when the data was not available. It was clear that, in the risk-assessment phase of such projects, we need to consider potential points of failure. It is difficult to anticipate, or mitigate, all potential points of failure. There may always be a single point of failure, including the Library's own Internet2 network line or the commercial Internet network line. In two large, complex organizational infrastructures such as the Library and SDSC there are many things that could go wrong.

System Status Reports

One feature that helped the Library develop trust in SDSC was the continuous reassurance that SDSC was constantly monitoring the Library's data. SDSC set up a system that e-mailed regular system-status notifications.

It took several iterations between the Library and SDSC to prune the dense technical language in the e-mail. The Library had to consider who might receive such notifications and who the audience might be. Eventually the Library decided that:

- their potential audience was broad; and
- most recipients probably would not read the e-mail notifications, especially if the notifications were automatically generated and arrived with any predictable regularity. Such e-mails would not get more than a two-second glance at the header at best.

The Library and SDSC created the notices so that the data status displayed in a simple statement in the Subject field. The Subject of an "OK" e-mail read, "SDSC storage report: no changes in Library files since (date)."

There was no need for the recipient to open up the e-mail; this was enough. If there was a problem the system would state it and the e-mail would contain more information.

While the notifications were comforting and efficient, there was one incident toward the end of the project that reminded the Library of the need to occasionally monitor the monitoring system.

After the Library and SDSC finished testing and were in the final stage of documenting the project, SDSC started dismantling their system. The part of the system that monitored the system's status was shut down, but the automatic notification part was still active. During this time the Library conducted a spot test where they deleted a few files without notifying SDSC, just to see what alarm notice the system might generate. The system continued to send e-mails that read, "SDSC storage report: no changes in Library files since (date)."

In all fairness, the Library knew that SDSC was shutting down the system and it probably would not function normally. The deceptive e-mail notifications would not have happened if the system was actively monitoring itself. But it did show that under certain circumstances it's possible to receive false status assurances from the system.

Other Transfer Tools and Protocols

After the SDSC project ended, the Library expanded data transfer collaborations with other NDIIPP partners, building on the experience, lessons, and successes of the SDSC project. As the Library explored a range of other data transfer tools and protocols with other NDIIPP partners, further research and experience led the Library to refine and simplify its approach. It became obvious to the Library that the simpler the data transfer tools and protocols are, the greater the chance of consistent success among diverse institutions.

In the spring of 2008, the Library and the California Digital Library jointly developed a simple format for transferring digital content. The BagIt (California Digital Library, n.d.) format specification is based on the concept of "bag it and tag it," where digital content is packaged (the bag) along with a small amount of machine-readable text (the tag) to help automate the content's receipt, storage, and retrieval. There is no software to install.

BagIt builds in part on the successes of the SDSC data-transfer project and the Library of Congress's Archive Ingest and Handling Test (2005). BagIt streamlines the transfer process by reducing the number of "moving parts." Conforming to the BagIt format provides users a smooth, orderly transfer with little or no human intervention.

A bag consists of a base directory, or folder, containing the tag and a subdirectory that holds the content files. The tag is a text-file manifest, like a packing slip, that consists of just two data elements: (1) an inventory of the content files in the bag and (2) a checksum for each file.

This example shows a typical line from a manifest:

```
408ad21d50cef31da4df6d9ed81b01a7 data/docs/example.doc
```

Each line contains a string of characters—the checksum—followed by the name of the file ("example.doc") and its directory path. There are a few variations of a bag format, though the preferred version, an "empty" bag, lists URLs instead of simple directory paths. A script then consults the tag, detects the URLs, and retrieves the files over the Internet, in parallel, ten or more at a time, reducing the data-transfer time. If a user wants to add more detailed metadata, he or she may do so in another optional file.

There is no limit to the number of files or directories a bag may contain, but since a bag is a unit of transfer, it should be sized to make transfers easier, based on media sizes or expected network transfer rates. The Library recommends that bag sizes should be approximately 650 GB or so, in general, to avoid tying up network traffic.

BagIt has become such a standard at the Library that it routinely refer to any digital package in a data transfer as a bag, and it rely on the structure of each bag to be consistent with the BagIt specification.

Tools

The Library has been eager to explore data transfer tools and to work with NDIIPP partners in building and maintaining a diverse and robust set of tools and methods. The most commonly used tool and the one with the most consistently reliable success, has been the simple rsync (http://samba.org/rsync/) utility. This has been used in data transfer projects with multiple NDIIPP partners, including:

- The California Digital Library
- Channel 13/WNET (public television)
- University of California, Santa Barbara (National Geospatial Data Archive)
- Stanford University (National Geospatial Data Archive)
- Internet Archive

One key to the Library's success with rsync is parallelization, running several simultaneous instances to distribute and download the contents of a bag. Even though optimal network transfer requires both parallel streams *and* TCP buffer sizing, we can achieve sufficient throughput through parallelization of standard protocols like HTTP and rsync, using tools like wget or rsync managed by simple wrappers. The Library developed a parallelization tool (https://sourceforge.net/projects/loc-xferutils/) that implements a Python-based (http://www.python.org/) wrapper around wget (GNU Operating System, n.d.) and rsync.

Other data-transfer tools the Library has used in its NDIIPP-partner projects include SRB client (SDSC, n.d.), Fast Data Transfer (FDT; http://monalisa.cern.ch/FDT/) the Logistical Networking Toolset (http://www.lstore.org/pwiki/pmwiki.php), and LOCKSS (http://www.lockss.org).

One of the challenges the Library now faces is the automation of the ingest process. They are developing a submission application to receive, tag, index, and store the data.

As the Library defines processes for transferring content over a network rather than on physical media, it expects to use *both* the commodity

Internet and the Internet2 network. BagIt and rsync are wise choices for either network.

Lessons Learned and Best Practices

The Library has benefited from its collaboration with all of its NDIIPP partners. There is much for everyone in the digital library community to learn, not only from SDSC's years of expertise but also from each other's experiences, and from plain old trial and error. For an additional resource, the Library recommends the Department of Energy's "Guide to Bulk Data Transfer over a WAN" (2007).

The Library has built its cyberinfrastructure to the point that large-scale data transfer and ingest—building on work with NDIIPP partners—is now part of its daily workflow. Each NDIIPP partner has presented a different set of challenges. These are some of the lessons that have been learned so far:

- Simpler is better, as demonstrated by rsync and BagIt
- Parallelization—multiple simultaneous data threads—is the most efficient use of network transfers, either over the commodity Internet network or the Internet2 network
- *Reliability* and *availability* are two separate issues of digital storage. Data can be stored with 100 percent reliability in its integrity and yet not be available 100 percent of the time.
- Multiple levels of storage and replication (including geographic replication) are essential to guarantee reliability of storage. However, at the scale of terabytes, petabytes, and more, each instance of data replication has an associated cost to consider.
- For file system synchronization, structural elements and metadata (e.g., file names) must be coordinated.

The Library and its NDIIPP partners continue to learn from these data-transfer projects. These lessons—and others we have yet to learn—will benefit all types of institutions, including smaller, less well-funded institutions and libraries. Our cyberinfrastructure must enable the expanding data preservation community to reliably and easily transfer and manage data. The greatest test of our community's intelligence and expertise is how much complexity we can eliminate from the usability of our tools. The timeless wisdom of the Tao Te Ching still holds true, "Embrace simplicity."

References

California Digital Library. (n.d.). BagIt. Retrieved July 8, 2008, from http://www.cdlib.org/inside/diglib/bagit/bagitspec.html

Federal Information Processing Standards. (1995, April 17). Secure hash standard. Retrieved July 8, 2008, from http://www.itl.nist.gov/fipspubs/fip180-1.htm

GNU Operating System. (n.d.) Wget. Retrieved July 8, 2008, from http://www.gnu.org/software/wget/

Library of Congress. (2005, June). Archive Ingest and Handling Test. Retrieved January 15, 2009, from http://www.digitalpreservation.gov/partners/aiht/high/ndiipp_aiht_final_report.pdf

SDSC. (n.d.). SRB client interfaces. Retrieved July 8, 2008, from http://www.sdsc.edu/us/resources/srb/clients.html

SourceForge (n.d.). Iperf. Retrieved July 8, 2008, from http://sourceforge.net/projects/iperf

U.S. Department of Energy. (2007). *Guide to bulk data transfer over a WAN*. Retrieved July 8, 2008, from http://fasterdata.es.net/

Mike Ashenfelder is a digital media projects coordinator in the Office of Strategic Initiatives at the Library of Congress, working with NDIIPP partners on large-scale data transfer and digital preservation. He also writes for the digitalpreservation.gov communications team.

Andy Boyko is a senior software engineer at Apple, Inc. Previously he was a digital media projects coordinator in the Office of Strategic Initiatives at the Library of Congress, working on large-scale ingestion, access, and long-term preservation in digital repositories.

Jane Mandelbaum is a project manager, Information Technology Services, Library of Congress, focusing on enterprise-level systems and digital asset management. She is the author of *Small Project Automation for Libraries and Information Centers.*

David Minor is the project manager for Library and Archival projects with the San Diego Supercomputer Center and lead of the Data Preservation Initiatives Group. He received his BA degree from Carleton College and his MLIS from the University of Wisconsin-Madison. He has worked previously for libraries at the Universities of New Mexico and Wisconsin as well as Penn State University.

Robert H. McDonald is the lead for Strategic Data Alliances at the San Diego Supercomputer Center. He received his BM and MM degrees from the University of Georgia and his MLIS from the University of South Carolina. Prior to his position at SDSC he led technology programs in the research libraries of Auburn University and Florida State University.

Richard Moore is the San Diego Supercomputer Center's director of the Production Systems Division, which operates the center's supercomputers, storage, and networking systems. He received his BS degree from the University of Michigan in 1975 and his PhD in Astronomy from the University of Arizona in 1980. After a postdoctoral position at Caltech, he led aerospace research teams at the Aerospace Corporation and then Photon Research Associates.