LoDN: Logistical Distribution Network

Logistical Computing and Internetworking Laboratory
Computer Science Department, University of Tennessee

Micah Beck, Jean-Patrick Gelas, Dustin Parr, James S. Plank, Stephen Soltesz
gelas@cs.utk.edu

This paper introduces the Logistical Distributed Network (LoDN) tool based upon Logistical Networking technology. LoDN aims to solve a common distributed collaboration dilemma faced by end-users in the context of distributing large data sets.

1 Introduction

Today, thanks to the World Wide Web, large communities of people with a common interest (artists, scientists, ...) can produce and share large data sets with colleagues around the world. However, when these data sets are exceptionally large or in peak demand, the available content distribution technologies cause a bottleneck. FTP and HTTP servers, the most common mechanisms for data distribution, must be either manually mirrored to divide the server load, connected through a very high speed link, or hosted by a large and expensive high performance storage system in order to accelerate downloads. In the past, the relative scarcity of storage necessitated implementing policy to determine which content would be mirrored at each site (and when a mirror would be removed.) Web caching automated this process and made policy a simple function of content size and popularity [1].

Logistical Networking makes it possible to deploy a globally scalable storage service that is sharable by the world community. The Logistical Distribution Network or LoDN (pronounced Low-Down) is an experimental content distribution tool which allows users to store content in the *Logistical Network* and manage, retrieve, or share that stored content. Users of a given community can collaborate through a web portal dedicated to them. Content distributors can push and manage stored data in the Logistical Network through an environment similar to a familiar IMAP e-mail interface or a file browser. Like most anonymous FTP servers, an account may be required to push data into the Logistical Network; however, nothing but a browser and a JVM is required to pull the published data. Finally, content distributors can make LoDN data files available by including an active LoDN link on a web page, in an e-mail, or in the LoDN content directory itself [2, 3].

In this paper, we first depict three common distributed collaboration dilemmas faced by end-users. Next, we briefly introduce the concept of Logistical Networking. We then explain how we run a content distribution service on top of Logistical Networking technology. Next, we revisit the three previous scenarios and show how Logistical Networking can resolve each one. Finally, we conclude and give future directions.

2 Scenarios

In this section, we describe three typical scenarios commonly experienced by end-users, research teams, or large communities who want to collaborate with their peers by sharing large digital objects through the Internet.

- Scenario 1: "Private usage"

A user with Internet access works on some high resolution video files at home and wants them to be available for his colleagues, living and working in the same city, tomorrow morning. These files are too big to be held by a USB key. The transience of the files makes burning a CD inconvenient. Moreover, both of these individual data support are not well adapted for sharing data. The files are too big to be sent in a e-mail attachment, and the user has neither access to sufficient storage on a public FTP or Web server.

- Scenario 2: "Uncertain future"

  A research team, located in the Eastern US, publishes on their website an abstract of their results with links to some big files (e.g huge matrices, high-resolution photographs, etc.) available on their server. Surprisingly, these files are tremendously popular and are downloaded very often from the west coast of the US, as well as Western Europe. Because of the success, their server becomes overloaded and unresponsive. The researchers do not have access to the resources of distributed servers, for instance in California, France, Japan and Brazil, to serve as mirrors for their data.

- Scenario 3: "Massive daily data production"

  A globally distributed community of Earth Science researchers work collaboratively making constant reference to observations of mutual interest within massive datasets downloaded daily from earth-orbiting satellites. The satellite data is stored in a centralized storage facility with time-consuming retrieval required to download specific subsets to collaborating sites. This requires preplanning on the part of collaborating teams to arrange for downloading of datasets of common interest in advance of interactive collaborative sessions. If interest arises unexpectedly in some data that has not been distributed to all sites, interactive work must be suspended to allow time for data distribution.

3 Logistical Networking

The "working" storage offered by Logistical Networking differs from other network storage schemes because it follows the same scalable design paradigm as the Internet: A highly generic, best effort service provides the common foundation on which all higher-level services are built. The foundational service underpinning Logistical Networking is the *Internet Backplane Protocol (IBP)*[4]. IBP creates storage resources in the network and provides access to them in the form of time-limited allocations. The combination of a storage resource and a network service that speaks IBP is referred to as a *storage depot*. Depots are placed directly into the network, thereby augmenting the network with user-addressable storage.

All stronger services, such as extended storage duration, reliability, and fault tolerance, are provided "end-to-end" by layers of middleware implemented on top of IBP. Using this scalable approach, Logistical Networking has been able to create a testbed that offers unbrokered access via the Internet to 41+ Terabytes of storage space, on over 415 locally maintained IBP storage depots spread across the USA and 30 other countries around the world (current figures at time of publication).

4 LoDN: Logistical Distribution Network

LoDN uses duplication and a process called "warming" to provide reliability for its storage service. The *Warmer* (defined in section 4.3) performs necessary periodic maintenance on stored data, such as renewing time-limited IBP storage allocations.

When a file is uploaded into the Logistical Network, the file content is broken into smaller data blocks. Each data block is then replicated and the replicas are distributed across multiple IBP depots. Distributing fragmented replicas of file content over multiple storage locations helps to ensure that stored content will remain accessible in case of a machine or network outage. Currently the distribution of data blocks is determined according to geographical parameters. Each registered IBP depot provides its geographical position (latitude and longitude) to the *L-Bone*[5], a directory and resource discovery service cataloguing

publicly accessible IBP storage depots. When storing data, clients may query the L-Bone for depots with specific characteristics, including minimum storage capacity, duration policy, proximity, etc.

When a user uploads a file into the Logistical Network, an *exNode* is generated. The exNode is an XML encoded metadata file that accommodates the aggregation of individual storage allocations into a "network file" by holding the metadata necessary to manage distributed content (where each data block is stored, when its storage "lease" expires, etc.) The exNode acts like a pointer to file content stored over the network, in the same way that the UNIX inode is a pointer to storage allocations on a hard drive. The exNode aggregates individual IBP allocations, thereby allowing the use of multiple allocations to achieve strong characteristics not offered by any single allocation (large file size, extended storage duration, reliability and fault tolerance, etc.) When a file is uploaded with LoDN, the resulting exNode is stored on the LoDN server. The Warmer operates on all exNodes in the LoDN server to maintain the integrity of the corresponding stored data.

LoDN gives fast access to stored data. At download, individual data blocks from a fragmented, distributed file can be retrieved from different depots, with each block being downloaded in parallel. LoDN also provides fast throughput by utilizing the geographical proximity of depots to upload and download sites. At upload, you can choose the location where your file is stored; you may store the data in a depot within your neighborhood or place the data on depots close to remote download sites. The LoDN download client automatically retrieves data from the closest depots storing replicas of the desired content.

4.1 Implementation

LoDN is comprised of four elements: Java upload and download applications for storing and retrieving data, a web interface for managing stored data and browsing public content, and a process called the *Warmer* (currently written in Perl).

On the system's end-user side, the software requirements are just a standard web browser (with frame support and *JavaScript* enabled), a modern Java Runtime Environment ($\geq$ 1.4.x), and a network access (modem, xDSL/cable, T3,...) to the Internet. The Java client applications (*LoDNPublisher* to upload files and *LoDNClient* to download files) rely on the Java *Web Start* technology [6]. This technology is based on the *Java Network Launching Protocol* (JNLP) which provides a way for an application to run from a codebase accessible over the network. It provides an environment similar to what would happen if URLs were supported in the *classpath*. Because these applications access the user's local file system, the class files (compressed and archived in `.jar` files) are signed with a certificate provided by a Certificate Authority (CA).

On the LoDN server side, we use *Apache* as web server to serve the `html` pages and to call the CGI scripts (written in Perl).

All the communications, between the client applications (upload and download), the LoDN server, the IBP depots and the L-Bone directory are made through the TCP protocol.

4.2 Upload and Download step-by-step

This subsection briefly describes the different steps for uploading and downloading a file in the Logistical Network through LoDN.

Upload step-by-step:

1. A LoDN user employs the Java client application (dynamically downloaded or updated from the web) to choose the file he wants to upload, his connection type, his location, and the number of copies to store (two by default).

2. The Java application contacts the L-Bone and waits for a list of depots with the requested location and available disk space.

3. The input file is partitioned into several blocks, which are uploaded to allocations on IBP depots. Copies of the individual data blocks are then uploaded to other allocations as required to achieve the

requested number of copies, n. Notice that n replicas of the same data block are uploaded to n different depots. During this operation, the corresponding exNode file is built.

4. Once the upload operation is completed, the exNode file is uploaded to the LoDN server, which will maintain the corresponding data (i.e. *Warm* it).

Download step-by-step:

1. A LoDN user clicks on a LoDN link (posted in a LoDN directory, on a web page, in an email, etc.) This event starts the Java download application, which requests that the user specify the connection type in use.

2. Before downloading the stored data, the Java client retrieves the exNode for the corresponding data and downloads it to the user system.

3. Thanks to this exNode, the Java Download application is now able to contact the required IBP depots and retrieve the individual blocks of stored data and reassemble the file on the user's local disk.

4.3 Maintaining Data integrity in a Logistical Network: The Warmer

To offer maximum scalability, IBP storage allocations are lightweight and best effort; and for maximum sharability, they are also time-limited. However, using best effort, time limited storage means that there are no guarantees about the reliability and sustainability of stored data. Data held in an IBP allocation can be lost or become inaccessible at any time due to network or machine outage, if space is suddenly reclaimed by the local system, or deletion upon expiration of the time limited allocation (storage "lease" runs out).

Blocks of data are stored on depots in allocated pre-defined fixed space size, called "capabilities" for a dynamically negotiated fixed period of time, also called a "lease". The existing IBP interface allows a client or an application to renew this lease with the *refresh* operation. The interface also allows the user to copy data to another depot with the *augment* operation (thereby duplicating a block of data), or delete a data block with the *trim* operation. Using these mechanisms, we can provide robustness and persistence even though the underlying storage is best-effort and time limited by replicating and migrating data between allocations.

In order to ensure data integrity, the LoDN server runs a process called the *Warmer* over all the exNode files for which the server is in charge. The term integrity is used here in the context of block availability and correctness. If a file's block is missing or corrupted we can say that the file integrity is compromised. However it is not definitive (final), as one or more copies may still be available on depots which do not respond at the time that the file integrity check process is running. The Warmer must: (1) Walk through the exNode files, (2) process the status of each block in each exNode, and (3) take action if needed (*refresh*, *trim* or *augment* data) according to local policy (max number of copies, supplemental lease time,...).

The Warming operations must be applied periodically. On the current LoDN server we choose to warm the complete content directories twice a day by utilizing the *crontab* utility.

The Warmer is currently a Perl script using the LoRS tools [7]. The commands used are: `lors_ls` to establish the list of the blocks and their status (1=available, -1=unavailable), `lors_trim` to remove the unavailable blocks, `lors_augment` to add one or more copies of the blocks, and `lors_refresh` to renew the storage lease.

5 Back to the scenarios, now with LoDN!

Let's see how LoDN can help to solve the scenarios introduced in section 2.

- Scenario 1: "Private usage"

  In this first scenario, the user just needs an account on a LoDN server, to upload his files to his LoDN account, and then to publish the files by clicking a simple switch. Next, he can send a LoDN link to

his colleagues by e-mail. By following this link, his colleagues will be able to download the documents easily and quickly (they do not even need a LoDN account if the data is published in a public LoDN directory).

In this context, these files do not need to be replicated broadly all over the world. Two or three copies on IBP depots in their city (or in the vicinity) should be enough.

Here, the objective of the Warmer is to keep a few copies of each block available in the original area (close to the upload site). We can also imagine that these files are of interest for one other colleague somewhere else on Earth (e.g attending a meeting). The user can specify that any additional copies made by the warmer should be stored near the colleague's remote location; in this case, the warmer will migrate the data according to the intent expressed by the user.

- Scenario 2: "Uncertain future"

  In this second scenario, our research team members use LoDN to upload their data to IBP depots located in their area by default (Eastern US), and put a LoDN link on their project web page, pointing to the data. There is no need to store the actual data files on their server. Data consumers download the data directly from IBP storage, not the researchers' server. Their server will not be clogged with downloads, even under peak demand.

  The research team had no idea about the future demand for their data and even less idea about where the data would be consumed. This is a perfect example of dynamic data logistics. Here, the role of the Warmer is, first, to move data where it is in highest demand (most often downloaded). In our scenario, some copies must be created on depots available on the West coast and in Western Europe. Note that the idea is not very far from web caching. Second, the Warmer should increase or decrease the number of copies according to the actual demand.

- Scenario 3: "Massive daily data production"

  In this scenario, LoDN could be used to store data that is likely to be used by collaborators, typically the most recent observations, at multiple sites distributed throughout the network. Data from the satellites could be stored in the Logistical Network temporarily, a month for example, and cycled out as newer data is added. Predicted interest in a particular location or time could result in the staging of large amounts of possibly-interesting data to the Logistical Network, making access possible by collaborators with relatively low overhead.

  Logistical Networking is well suited for massive data storage and delivery. In this scenario, the Logistical Network could be used as a huge storage buffer. A "sell by" tag could be applied to the data. When the date expired, the Warmer would stop renewing the storage lease for the expired data. However, if the exNodes that represented datasets of interest to particular users were copied by users, and warmed independently of the central distribution service (either using the same LoDN server as that service, or using a warmer of their own) that data would continue to live in the network until the user decided to stop warming it. Thus, the exNode allows storage management policy to be distributed as well as the data itself.

6 Conclusion and Future work

LoDN is a file directory based on the IBP protocol and the exNode file description, both designed for Logistical Networking by the LoCI research group. Members can access their LoDN directory to manage their files and sub directories. They can also access other users' published data. If a user has not explicitly published a file, other users will not be able to access it.

LoDN provides a reliable storage service, through the automated duplication of data and periodic renewal of time limited storage allocations by a process called the Warmer. The Warmer helps to maintain reliability and fault tolerance by actively managing files, for instance augmenting the number of available copies when it falls below specified limits. LoDN gives fast access to the data. At download, the individual blocks of a

fragmented, geographically distributed file can be downloaded in parallel from different depots. LoDN also increases performance by allowing the user to position data close to possible download sites. You can upload your file to a depot in your area and then create replicas on any of the 415 depots located around the world.

The main advantages of the LoDN approach over peer-to-peer networks in the context of file sharing or content distribution network are: users do not have to share their own local disk space because data are stored on publicly available (exposed) storage resources. Users do not have to stay connected to keep their published data available to other users. And finally, a user uses their own computer only to upload, download, and remotely manage his/her data through the LoDN web interface. The LoDN user does not need to store either the file content (actual data) or metadata about the stored content (location and expiration of each data block, etc.) on their local system.

This last point raises a number of questions which should be addressed in future work. First, the problems of ownership and property have not yet been taken into account. Second, no explicit mechanisms have been implemented to limit the size of uploaded data. This could allow malicious users to easily produce a Denial of Service attack.

In the near future, we will provide new metadata to help LoDN and the Warmer to better maintain data and deal more efficiently with dynamic change (e.g locality, time out, last access time, etc.). In addition, we plan to post more useful metadata on the LoDN directory, so that users can utilize statistical information about files in their directory (number of hits, file size, number of copies, etc.)

7 Acknowledgements

This work is supported by the National Science Foundation (NSF) Next Generation Software Program under grant # ACI-0204007 and NSF Middleware Program under ANI-0222945, the Department of Energy Scientific Discovery through Advanced Computing Program under grant # DE-FC02-01ER25465, and a grant from the Joint Institute for Computational Science (JICS). The infrastructure used in this work was supported by the NSF CISE Research Infrastructure program, EIA-9972889 and Research Resources program EIA-022444. Travel funds to WACE 2004 (Nice, France) were provided by Microsoft Research.

The authors would like to acknowledge the early contribution design work of Jeremy Millar former member of the LoCI team, Scott Atchley for his technical help, Jenée Mitchell for her great editing and documentation, Stéphanie Chêne the very first user of LoDN, Terry Moore and all the users around the world for their valuable comments to improve the LoDN interface.

References

[1] M. Rabinovich and O. Spatscheck. *Web caching and Replication*. Addison-Wesley, 2001.

[2] The LoDN server. `http://loci.cs.utk.edu/lodn`.

[3] Jean-Patrick Gelas. An introduction to LoDN in French, July 2004. `http://loci.cs.utk.edu/lodn/doc/Intro-fr.htm`.

[4] Micah Beck, Terry Moore, and James S. Plank. An End-to-End Approach to Globally Scalable Network Storage. In *SIGCOMM'02*. ACM, August 2002. Pittsburgh, Pennsylvania.

[5] The L-Bone server. `http://loci.cs.utk.edu/lbone`.

[6] Introduction to the Java Web Start architecture. `http://java.sun.com/products/javawebstart/architecture.html`.

[7] The LoRS tools web page. `http://loci.cs.utk.edu/lors`.