The Stanford Archival Vault: A reliable, long-term data archive

Brian Cooper, Hector Garcia-Molina
Department of Computer Science
Stanford University
{cooperb,hector}@DB.Stanford.EDU

December 17, 1999

1 Introduction

Information stored and managed in an archive today can be lost within years or decades if special care is not taken. The causes include media and system failures and format obsolescence. At Stanford we have implemented a prototype archival repository, the Stanford Archival Vault (SAV, pronounced "save"), for the long term preservation of digital objects. These objects may include documents, their metadata, and the programs for interpreting formats. Our repository does not entirely solve the preservation problem, but we believe it provides an extremely reliable storage infrastructure for preserving digital objects, even as hardware, software, and organizations evolve.

We propose a poster discussing issues related to the implementation of SAV. As we implemented and tested our SAV prototype, we identified some unexpected, important challenges that led us to modify our initial design, and to develop some new storage and replication techniques. We believe that the encountered challenges were not unique to our system, but represent some fundamental problems that will be faced in the design of any type of digital archival preservation system.

Specifically, this poster would address the following issues:

- *Replication*: Our SAV solution replicates digital objects to distributed sites, and continually compares replicates to detect errors. We have looked at effective ways to replicate data efficiently and sensibly so that a backup is available when corruption is detected in digital objects.
- *Write-once archives*: In order to prevent accidental erasure of data and inconsistencies within the archive, SAV does not provide any operations to delete or modify digital objects once they are written. However, many traditional data management algorithms rely on the ability to update objects; thus, new approaches must be used.
- *Automation*: Archives are most effective when they can operate autonomously for long periods of time. SAV is designed to perform archiving functions automatically.
- *Non-intrusiveness*: Data is migrated into the SAV without disturbing the data store from which the data is obtained (whether it be a traditional filesystem or the Internet). This non-intrusive approach does not require the data store to be modified in any way to facilitate archiving.
- *Scaleability*: A real archive will be responsible for storing gigabytes or terabytes of information. Despite this large data set, the archive should still operate effectively and efficiently.

This statement outlines some of the ideas that will be presented in the poster. Section 2 discusses the research we have done so far, and Section 3 examines the challenges that remain for future work. For a more complete description of the SAV system and details of the implementaion, please see the full papers we have written [1, 2].

2 Archiving digital information

The SAV is an implementation of an Archival Repository, a general design for a long term digital archive [3]. An Archival Repository is primarily concerned with reliable storage, and thus has very different features than a traditional storage system. These traditional systems focus on efficiency or low cost, often to the detriment of reliability. However, redesigning storage solutions from the perspective of archiving (as we have done with the Archival Repository) has presented several interesting research issues which we have examined. Our prototype incorporates many of our proposed solutions, and has been used to archive the Stanford Database Group's public web site, which contains almost 2 GB of data.

Reliable storage using automated replication

The philosophy of an Archival Repository is that data is replicated before it can become corrupted. Because it is unlikely that copies at geographically dispersed locations are simultaneously corrupted, this replication solution provides a way to recover from corruption by overwriting the corrupted object with a pristine object obtained from another site.

In the SAV system, administrators of different sites set up *replication agreements*. Sites participating in an agreement maintain copies of the data in the agreement. This happens automatically once the agreements are constructed, as each SAV instance periodically connects to other sites in the agreement to detect new versions of the replicated data and to correct corruption. Our architecture allows different *replication networks* to be constructed, in which sites participate in binary or multiway agreements.

A replication agreement defines both the sites that will copy data and the data to be copied. Our system uses the concept of a *replication set*, which specifies how to identify documents that should be replicated. For example, a set of web pages from a particular host may be replicated under a given agreement. When a new web page from the host is added to the archive, it should be automatically replicated in the same way as the other pages. We have proposed viewing the stored objects as nodes in a graph, with edges formed from relationships between objects (such as hypertext links). By annotating these edges, we can clearly specify the boundaries of collections of objects for purposes of replication while still allowing these collections to grow.

A *write-once* repository

A *write-once* repository specifies that data, once written, is never erased. The archive does not provide any operations for unintentional or malicious destruction of data. A write-once archive also simplifies reliability by allowing the archive to treat any modification or deletion of a digital object as corruption, and automatically replacing the "corrupted object" with a pristine object.

However, it may be desirable to record modifications to documents, so we propose using *version chains*, in which each version of an object points to the previous version. In this way, modifications to documents and to the structure of the graph of objects can still be recorded without violating the write-once policy. Objects can even be taken "out of circulation" by marking a new version as "do not circulate."

It is difficult to use traditional object management techniques if objects cannot be modified. For example, the traditional way to keep a set of objects (such as a directory on a filesystem) is to maintain a set object, and then add or remove objects from the set by modifying the set object. Because such changes are recorded in a write-once archive by creating new objects instead of changing or deleting old objects, it is necessary to create indexes in order to preserve efficiency. These indexes can ensure that objects can be found and manipulated efficiently, even as the archive evolves.

Migrating data into the repository

An archive is only useful if information is stored in it. We have implemented a software package called the InfoMonitor that monitors a traditional (non-archival) data store and copies objects into the SAV. This system is designed to be non-intrusive so that data objects can be migrated into the archive without modifying the traditional store or requiring the constant supervision of users. This makes the migration process less painful, which means that information is more likely to be archived.

This migration tool must translate between the storage systems of the traditional data store and the archive. For example, the InfoMonitor deals with differences in naming systems, and automatically creates version chains to represent modifications to objects. In this way, the InfoMonitor acts as a mediator, and can be extended to migrate data from a wide variety of differents stores into the common paradigm of the archive.

In order to be non-intrusive, the InfoMonitor cannot depend on the traditional store to provide signals when objects are created or updated. Instead, the InfoMonitor must detect such events on its own. We have implemented techniques to efficiently determine when changes have occurred and new objects must be copied into the SAV.

Scaleability and user interface

An archive of the Internet must deal with terabytes of information, and even archiving smaller collections can require gigabytes. As a result, we have investigated methods for ensuring the scaleability of the system. For example, we have examined ways to efficiently migrate large amounts of data from site to site. This involves quickly determining when collections at two sites differ, either because of new objects or due to corruption. We have also begun to investigate how operations can be performed incrementally, so that bandwidth usage can be reduced if necessary.

One aspect of scaleability is how to build a user interface that is useful despite dealing with large numbers of objects. We have constructed an interface that allows for effective navigation of the objects in the repository, even when the objects number in the hundreds of thousands (or more). This is done by providing several different views of objects, including hierarchical and temporal views, and by providing filtering capability to quickly find objects that have certain properties. We have also investigated how to reduce the overhead of the interface to avoid impacting the rest of the system.

3 Future work

Although we have built a working SAV system, and have successfully used the InfoMonitor to migrate data into the archive, there are still several issues that we would like to explore further. These include:

- **Archiving the web.** We have already archived the Stanford Database group's web site, but would like to extend our system to archive larger portions of the Internet. This involves archiving content from systems where we do not have direct access to the filesystem of the server, and thus using HTTP or another Internet protocol to retrieve information in a sensible way. We would also like to explore how to define meaningful subunits of the web and archive just those subunits.
- **Other replication models.** Our replication agreement model for replicating data is just one possibility. We would like to identify and examine other (potentially less structured) models for copying data to remote sites.
- **Preserving meaning.** We have so far concentrated on preserving bits reliably. Preserving meaning is a much harder problem. Nonetheless, we would like to examine this issue, perhaps starting by looking at how to preserve the meaning of "file formats" so that the order of bits can be understood far into the future.
- **Terabyte scaleability.** Our system currently works for replicating sets consisting of a few gigabytes. However, multiplying the size of the data set more than three orders of magnitude is likely to introduce new challenges. We would like to examine whether our current methods are still sufficient for terabyte sized sets, and if not, how our system can be extended to deal with such sets.

4 Conclusion

A vital component of an Internet archiving system is the archive itself, which is responsible for storing the data reliably for a long period of time. Simply copying large volumes of data onto magnetic tape or optical disks is insufficient if the data becomes corrupted. We propose a poster examining our experiences designing and implementing the SAV system. This system ensures long term preservation by using automated replication to create and verify backups. Moreover, SAV is a write-once repository, which protects data from unintentional or malicious erasure. We have created the InfoMonitor to migrate data into the repository automatically and non-intrusively. We have also examined ways to deal with large numbers of archived objects, including the challenges inherent in building an interface that scales well. These solutions provide a useful substrate for storing information as well as a good starting point for future research in archiving.

Acknowledgements

The proposed poster discusses research that is joint work with Arturo Crespo, and is part of the Stanford Digital Library Project.

References

[1] Brian Cooper, Arturo Crespo, and Hector Garcia-Molina. Implementing a reliable digital object archive. http://www-db.stanford.edu/pub/papers/arpaper.ps, 1999. Submitted for publication to ACM DL 2000.

[2] Brian Cooper and Hector Garcia-Molina. Designing and implementing layered archival storage systems. http://www-db.stanford.edu/pub/papers/fmpaper.ps, 1999. Submitted for publication to ACM SIGMOD 2000.

[3] Arturo Crespo and Hector Garcia-Molina. Archival storage for digital libraries. In *Proceedings of the Third ACM DL Conference*, 1998.