Interconnecting DSpace and LOCKSS

Mushashu Lumpa, Ngoni Munyaradzi, and Hussein Suleman

Department of Computer Science, University of Cape Town,
Cape Town, South Africa
{mlumpa@cs.uct.ac.za,ngoni.munyaradzi@uct.ac.za,hussein@cs.uct.ac.za}

Abstract. Repository managers increasingly use toolkits such as DSpace to manage submission of and access to resources. However, DSpace does not support the highly desirable distributed replication functionality provided by LOCKSS. This paper describes an experiment to seamlessly interconnect DSpace and LOCKSS in a generalisable manner. An experimental prototype confirms that this is indeed possible, and that the interoperation can be efficient within the constraints of the systems.

Keywords: interoperability, harvesting, replication

1 Introduction

Lots of Copies Keeps Stuff Safe (LOCKSS) [7] is a popular software toolkit to replicate digital collections. LOCKSS is noted for its voting system whereby multiple replicas of an object on distributed servers are used to ensure the authenticity of the object [4]. If a single copy of an object is corrupted, it is outvoted as the authoritative copy and updated with an authoritative copy from one of the other servers. Thus, many copies may ultimately preserve the data, as suggested by the name LOCKSS.

This replication feature is not currently considered by most institutional repository (IR) toolkits, which focus on Web-based submission and access. Thus, the administrator of an IR would need one software tool for the management of the repository (e.g., DSpace) and another for the replication function (e.g., LOCKSS).

This paper reports on an experiment to interconnect DSpace and LOCKSS, where the former is used for submission/access and the latter for replication. The feasibility of interconnecting these systems seamlessly was investigated, using an approach that is generalisable to other systems. Metadata and digital objects are then transferred between the systems as necessary using METS [6] and MODS [5] for packaging of objects and specification of metadata, with OAI-PMH [3] as the underlying transfer layer.

2 Data Harvesting

The OAI-PMH only defines the harvesting of metadata so a mechanism for access to the complete digital objects was necessary. The solution adopted was the use of a special metadata format that returns a minimal metadata record with only a URL from which the full packaged digital object can be obtained. Thus a standard OAI-PMH harvester can be used, followed by the additional step of transferring each packaged digital object individually. This solution, a minimalist version of that suggested by Van de Sompel, et al [9], is described bellow. This approach is arguably superior to METS records transferred over OAI-PMH because the packages are self-contained and can be stored and manipulated without modification of the references to objects, which would in such a solution need to initially point to the originating repository [1] [8].

The **metsPackage** metadata format is defined as containing only a single element that specifies the URL from which a METS package can be downloaded. The *metsPackage* metadataPrefix is used to request a response in the **metsPackage** metadata format.

Once the harvester has obtained this list of URLs to digital objects, it can download each package individually. These packages are in the standard METS package format and contain METS metadata in an XML file as well as all digital objects that make up the item. All files are then compressed using the ZIP algorithm.

In the experimental system that was developed, a new verb - GetPackage - was added to the OAI-PMH set to download a METS package. This was simply a development convenience - this can also be implemented using a completely different Web application. To be consistent with the OAI-PMH verbs, GetPackage takes a single parameter that is the identifier of the item whose package is being requested.

When the server creates a **metsPackage** response, it inserts GetPackage URLs into the identifier fields.

When the GetPackage verb is encountered by a server, it will dynamically create the identified METS package and return it to the harvester client. The harvester then unpackages it and ingests the metadata and digital objects into its local system.

3 Evaluation and Analysis

3.1 Experiments

The plugins were evaluated for criteria of correctness and speed. Results from the test of the speed of the LOCKSS system interface show a roughly linear increase in time to scan through the filestore as the size of the filestore increases and negligible difference in performance for shallow and deeply-nested filestores. Very small filestores appear to perform much better than larger ones, probably as a result of disk caching.A linear increase in time is expected in an unindexed data store, so this result confirms what was expected. The results for the GetPackage verb experiment show that transfer time of individual packages are reasonably consistent, with an average of 1.6 seconds per package for transfer. Variations in time are due to the differences in file sizes. The system can be deemed to be scalable with the number and sizes of files.

3.2 Post-Analysis

Numerous stumbling blocks were encountered during the development that could inform the design of future repositories and systems for distributed preservation. These are discussed below.

- Read-only LOCKSS
  LOCKSS uses a read-only system partition for its operating system and all its software. This ensures that the system is stable and not prone to viruses or other attacks on the integrity of the system. Unfortunately, this also makes it impossible to add extensions to the system or update any part of the system. During the development of the LOCKSS plugin, it was necessary to copy over the plugin's files manually each time the server was restarted, as a temporary solution.
- Metadata Restrictions
  LOCKSS uses HTTP headers for its metadata while DSpace uses qualified Dublin Core. Neither system appears to allow deviation from their baselines. This poses a challenge when interconnecting the systems as metadata cannot be translated completely between HTTP headers and Dublin Core.
- Encoding structure (communities)
  LOCKSS stores structural information in the directory structure of its file-store while DSpace stores the community and collection information in its database. In both cases, there is structure in the repository but this structure is not reflected in the metadata. A solution to this is to encode the structure into the OAI-PMH set entry in a record's header. However, the structure is still not part of an offline METS package, which typically includes internal structure only. If it was possible to include structural membership information, this would still be piecemeal and empty collections would not be replicated when a repository is replicated.
- Configuration transfer
  One solution to the problem of missing structural information is to explicitly store and transfer the entire structure of a repository. This can be part of the configuration information that is necessary to restore a repository completely.

4 Related Work

Early efforts at interoperability were focused on showing the feasibility of repository-to-repository migration, for example a transfer from Greenstone to DSpace and vice versa in the Stoned project [10]. Stoned supported import and export functions for digital objects and metadata using a suite of different mechanisms.

Van de Sompel, et al [9] proposed an automated mechanism for inter-repository harvesting that is based on the OAI-PMH and the use of a complex and descriptive packaging metadata format (SCORM, MPEG-21 DIDL, METS, etc.) to represent digital objects. These metadata formats could contain direct links to the digital objects in a live repository. Tansley [8] used a similar approach.

Finally, Duracloud promises to bridge the gap between repositories and replication by providing its own network based on cloud data storage [2] . Unlike LOCKSS, which relies on spare capacity, Duracloud will expect repository managers to explicitly allocate resources to replication. The software is open source so there is promise for experimentation with different models in the future.

5 Conclusions

This paper has described an experiment to interconnect these 2 systems to bring the benefits of both worlds closer together for repository managers.

In the process of developing an interoperability solution, the more general problem of data harvesting has been discussed. While metadata harvesting is commonplace, OAI-PMH does not address how to access the digital objects using a machine interface. This makes common services such as search engines, data mining and replication difficult. The solution adopted in this paper is not only compatible with the existing semantics of the OAI-PMH, but minimalist.

The system evaluation has demonstrated that the plugins and approach to data harvesting are both feasible and efficient. While reasonably successful, it is clear that even with the most advanced and popular tools, replication is not a simple task. Many aspects of system design can be improved to enable more effective/efficient replication, such that future attempts to interconnect similar systems will hopefully be less intensive!

References

1. Bekaert, J., Van de Sompel, H.: A Standards-based Solution for the Accurate Transfer of Digital Assets. D-Lib Magazine 11, (2005)
2. Duracloud, `http://www.duraspace.org/duracloud.php`
3. Lagoze, C., Van de Sompel, H.: The open archives initiative: Building a low-barrier interoperability framework. In: 1st ACM/IEEE-CS Joint Conference on Digital Libraries, pp. 54–62. ACM, New York (2001)
4. Maniatia, P., Rosenthal, D.S.H., Roussopoulos, M., Baker, M., Guili, T., Muliadi, Y.: Preserving peer replicas by rate-limited sampled voting. ACM SIGOPS Operating Systems Review, 37–59 (2003)
5. McCallum, S.H.: An introduction to the Metadata Object Description Schema (MODS). Library Hi Tech 22, 82–88 (2004)
6. Metadata Transmission Encoding Standard, `http://www.loc.gov/standards/mets/`
7. Reich, V., Rosenthal, D.S.H.: LOCKSS (Lots of copies keep stuff safe). New Review of Academic Librarianship 6, 155–161 (2000)
8. Tansley, R.: Building a Distributed, Standards-based Repository Federation: The China Digital Museum Project. D-Lib Magazine 12, (2006)
9. Van de Sompel, H., Nelson, M.L., Lagoze, C., Warner, S.: Resource Harvesting within the OAI-PMH Framework. D-Lib Magazine 10, (2004)
10. Witten, I.H., Bainbridge, D., Tansley, R., Huang, C.Y., Don, K.: A bridge between greenstone and DSpace. D-Lib Magazine 11, (2005)