D-Lib Magazine

May/June 2016
Volume 22, Number 5/6

Customization of Open Source Applications to Support a Multi-Institution Digital Repository Using DSpace

Youssef Benchouaf
Colorado State University
youssef.benchouaf@colostate.edu
orcid.org/0000-0001-6204-8868

Daniel Hamp
Colorado State University
daniel.hamp@colostate.edu
orcid.org/0000-0002-8945-1363

Mark Shelstad
Colorado State University
mark.shelstad@colostate.edu
orcid.org/0000-0002-8064-4679

DOI: 10.1045/may2016-benchouaf

Abstract

Colorado State University, along with seven other academic institutions and a university press, migrated out of DigiTool to DSpace in the fall of 2015. This article will analyze the customizations made to the application, including assigning multiple handles; automating ingest of electronic theses and dissertations; mapping of multiple schema (METS, MODS, MARC) to Dublin Core; modifying the Simple Archive Format for ingest; Kaltura for ingesting bulk video submissions; and harvesting the OAI feed for CSU's web site, discovery layer, and other repositories. A number of processes for ingest have been automated or made available to non-system staff, resulting in significant cost and time savings.

1 Introduction

Colorado State University (CSU) deployed DigiTool in 2007 after several years on OCLC's CONTENTdm application. A committee benchmarked digital repository options at that time, which included open-source applications. CSU selected DigiTool as a mature out of the box application that would require relatively less time to implement and a smaller long-term overhead. The system served its purpose well over eight years of service and attracted the attention of other Colorado institutions who joined CSU to form a digital repository consortium known as the Digital Collections of Colorado (DCC). Towards the end of DigiTool's lifecycle at CSU, the system had not been receiving major system updates and received instead only the occasional bug patch. The advances in other products available on the market began to become apparent at this point, for features such as improved system management, full Open Archives Initiative support for harvesting metadata, Simple Web-service Offering Repository Deposit protocol, modern user interfaces, smarter design and improved software stacks, and simple REST (REpresentational State Transfer) architecture.

2 Migration

CSU's approach to selecting software systems begins with prioritized consideration for open source products, resorting to paid commercial offerings if open source options do not seem viable for the particular use cases. Cost and improved access to modify and extend the code base of these applications heavily influences purchasing decisions. For our digital repository replacement, we generally sought products that offered capabilities comparable to DigiTool, while being widely-supported and still in active development. We compared a number of products, such as Fedora and CONTENTdm before selecting DSpace. While not the most advanced product offering, it does have a very large community of users — far more than any other system — as well as a very active development community. Its extensive documentation is noteworthy, it follows metadata standards, and most importantly is an integrated application as opposed to other digital repository frameworks such as Fedora, which requires Hydra or Islandora for the user interface, and is a much more rigorous initial implementation and customization.

To export the data out of the DigiTool, we used its API and direct calls to the system database. Our original collection structure was built using regular expressions instead of direct assignment of items, meaning that items were tied to no collection directly, but were associated at time of generating the index for the front-end interface. A script needed to be written that retrieved collections from the DigiTool database and created an XML representation in the format of DSpace's Community and Collection Structure Importer tool to recreate the original collection hierarchy using DSpace's communities, subcommunities and collections.

Another script had to be developed to export all items from DigiTool. The script iterated from zero to our largest item identifier (we selected 1,000,000 to ensure capturing all items) and also generated an XML record containing the source MARC, Dublin Core, MODS, METS, or Complex Dublin Core formatted metadata. To get a complete package for each item in a record, including its persistent identifier (handles, in our case), we tied the metadata to identifiers of the files on the file system. This allowed us to create a directory and flat file based representation of our digital repository.

Since DigiTool is based upon regular expressions, we wrote a script to crawl the DigiTool interface and "screen scrape" the regular expressions used for each collection, then recreated them using PHP's regular expression structure to scan a record's metadata, using the same logic as DigiTool to mark what collection(s) an item corresponds to. We mapped those collections to the DSpace handles for each collection, using information that was retrieved from the success log returned by DSpace when using the Community and Collection Structure Importer tool. To streamline management of the database, we stopped using a multitude of different metadata formats and used qualified Dublin Core to represent all content. With the help of ImageMagick, Perl:MARC, PERL:MARCXML, and a few slightly modified XSL transforms from the Library of Congress, our script transformed the various metadata formats to qualified Dublin Core, and converted JPEG2000 images to JPG images for display and end use.

Thus, the script we had written iterated through each item record, converted its metadata and any possible JPEG2000 images to JPEG, and packaged it and its content and collection association in DSpace's simple archive format (SAF) folder format layout in a separate location. It is worth noting that the process only copied the converted JPEG2000 files. All other files were symbolically linked to the location of the DigiTool export on DSpace's filesystem to save space and time in this process. From here, we simply had to run DSpace's batch ingest command line utility, specifying the folder containing the data to be imported and wait for it to process a few hundred-thousand records (which was surprisingly quick on our local system). This final step left us with a DSpace installation with the same collection structure and contents as our DigiTool system, including imported item handles and new collection/community handles.

As is most often the case in significant software migrations, new challenges arose. In this particular migration, metadata character encoding became our most difficult issue. While much of the content came from DigiTool, we had existing legacy records from CONTENTdm. The migration discovered that even though DigiTool had been representing much of the metadata in UTF-8, in reality, much of the metadata was in other character encoding formats. Our investigation revealed inconsistent methods of inputting metadata, such as copying data from web browsers or various text editors that do not necessarily use UTF-8 as their default format. Either by DigiTool's design or our implementation, the system did not strictly check encoding of the data before inserting into the system. To handle this issue we had to write a number of scripts that, during the conversion process to DSpace SAF format, would analyze the character strings contained in the metadata to determine the true encoding of the text and convert it to the correct UTF-8 format.

3 Customizations

Out of the box, DSpace supports resolving multiple handle prefixes (likely to support a migration path from other digital repository solutions) but only supports one default handle prefix for creating new communities, sub-communities, collections, and items. In our DigiTool system, each institution had their own handle prefix. Thus we were left with the option of either implementing separate instances of DSpace for each institution in our consortium, or modifying the DSpace codebase to handle new submissions to different handle spaces. In order to retain cross-community searchability, we did not want to pursue the former, albeit easier, option.

Extended research lead us to LINDAT, specifically their GitHub page. LINDAT added the capability to fully support multiple handle prefixes in their DSpace implementation. These changes resulted from considerable modification to their code, relative to our philosophy to avoid heavily modifying products, when feasible, to lessen the risk of these modifications breaking due to future updates. As a result, CSU opted to implement DSpace on behalf of its consortium and support multiple handle prefixes.

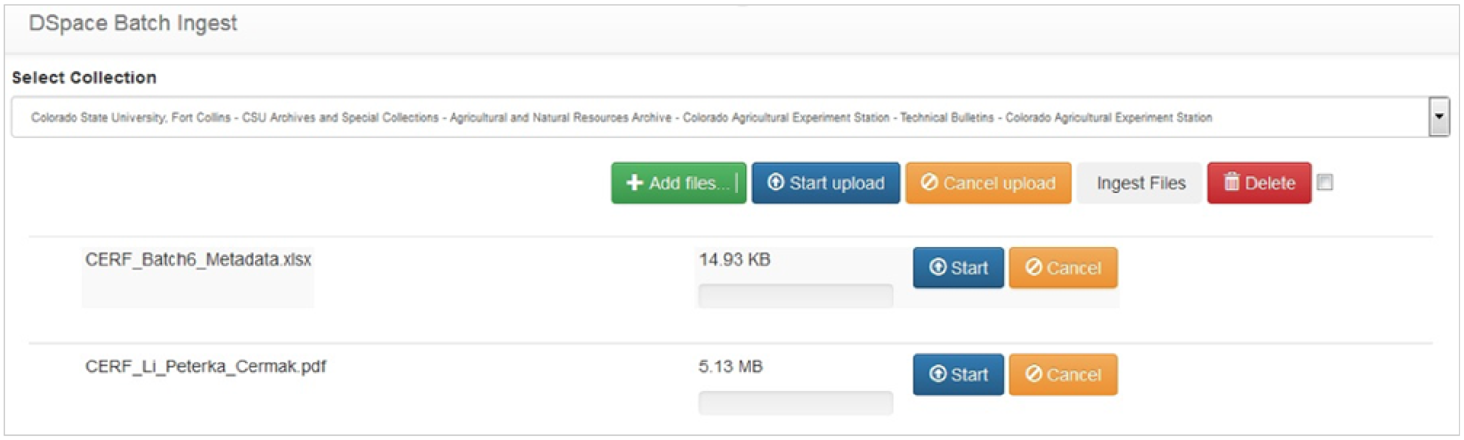
When creating our collection structure in DSpace, we set up top-level communities corresponding to each institution in our system with a different handle prefix. We changed the default prefix of each community while importing each group's collection structure in a way that would preserve their existing prefixes and avoid suffix collisions by setting the appropriate incrementer in the database outside of the range that was already in use (a consequence of DigTtool's lack of persistent identifiers for collections). The change to DSpace's code redirects generation of handles to a custom function we created. Instead of using the incrementer from the database and prepending that number with the default system prefix to create a handle (`{prefix}/{increment number}`), our function references the parent of a sub-community, collection or item, and retrieves its handle prefix. The function then selects all items in the system database with a matching handle prefix, sorts them in reverse order and select the first item (the one with the greatest handle suffix). The next increment of this item's suffix is used for the newly inserted resource. This ensures ease-of-use for submitters, as the items automatically assume an identifier in the appropriate handle space. Any additions or changes to top-level communities still require small, but manual, database modifications.

```
/*
Files Modified
[DSpace-src]/dspace-api/src/main/java/org/dspace/handle/HandleManager.java
[DSpace-src]/dspace-api/src/main/java/org/dspace/content/Community.java
[DSpace-src]/dspace-api/src/main/java/org/dspace/identifier/VersionedHandleIdentifierProvider.java
This Modification redirects the handle creation for a community, subcommunity or collection to the added
"generateHandle" function added in the HandleManager.java file.

Line 190, Community.java::create(Community parent, Context context, String handle)
*/

if(handle == null && parent != null)
{
handle = HandleManager.generateHandle(parent, context);
}

/*
This imports the HandleManager.java class into VersionedHandleIdentifierProvider.java import on top of
VersionedHandleIdentifierProvider.java.
*/

import org.dspace.handle.HandleManager;

/*
The following code replacement changes the default handle generate for items added to DSpace, to a custom one that
we added to the HandleManager.java file.

replace in VersionedHandleIdentifierProvider.java::createNewIdentifier(Context context, dspaceObject dso, String
handleId)
*/

if(handleId==null){
handleId = createId(handle.getIntColumn("handle_id"));
}
```

```
// with...

if(handleId==null){
        handleId = HandleManager.generateHandle(dso, context);
}
```

Figure 1: Code to Modify the Handle Generation

Most of the DCC partners make their Electronic Theses and Dissertations (ETDs) available through DSpace. Each institution receives its ETDs from ProQuest in zipped packages. Previously, in the DigiTool system, these items would need to be processed manually at each member institution and converted to the correct format, potentially having the PDFs optimized for file and delivery purposes. Many ETDs also included a zipped file of supplemental content. The DCC partners also received their files from ProQuest individually instead of in batches, resulting in a very time-intensive process that we believed could be automated. To address this situation we set up an SFTP destination directory on the DSpace server for each institution where ProQuest could upload the ETD files. Once new files are received, they are processed through an XSL transform file that converts the metadata to the correct Dublin Core format required in our DSpace implementation (with the exception of Networked Digital Library of Theses and Dissertations metadata,) automatically optimizes the ETD PDF file using GhostScript and compresses the supplemental material in the DSpace Simple Archive Format folder. The compressed folder then runs through DSpace's batch ingest command line utility and the extracted items are placed in designated ETD collections for each institution. A log file for the process is emailed to the appropriate institution's defined contact. This eliminates all human interaction with the processing of ETD files (and possible errors that humans might make,) while greatly reducing the amount of time required to process them.


DSpace Batch Ingest

Select Collection

Colorado State University, Fort Collins - CSU Archives and Special Collections - Agricultural and Natural Resources Archive - Colorado Agricultural Experiment Station - Technical Bulletins - Colorado Agricultural Experiment Station

CERF_Batch6_Metadata.xlsx 14.93 KB

CERF_Li_Peterka_Cermak.pdf 5.13 MB

Figure 2: Screen shot of the batch utility program

CSU is actively uploading new content into DSpace, and takes advantage of the application's batch ingest capabilities rather than pushing content through the single item ingest form. Unfortunately, DSpace's batch ingest tools within the XMLUI and JSPUI web interfaces are limited to users who are global administrators in the system, which is undesirable for our consortial model. CSU also wanted to preserve the ability to manage permissions granularly within an institution's community. To circumvent these limitations, we created our own batch ingest utility. DSpace, being a Java web application running on a Java web server (Jetty, in our implementation) sits behind an Apache web server acting as a reverse proxy to allow Shibboleth integration. Within the scope of the Apache web server (in order to operate under the same domain name) we created a simple PHP-based web application, inspired by the jQuery-File-Upload project. We chose this approach in particular because it imposed no file size limits and uploaded multiple files in parallel for greater efficiency. Thus, end users can login using their DSpace local accounts or institutional Shibboleth accounts (LDAP, X.509, and IP options are also available,) select the collection they want the content to be ingested into (limited based on administrative privileges,) and upload digital files and a CSV file with a format very similar to DSpace's Simple Archive Packager.

Our approach has additional capabilities, such as adding to multiple collections and setting labels on bitstreams to detail the metadata for each uploaded file. The web application will then return the output from the batch ingest process indicating success or failure. If successful, the system will clear the uploaded files and execute the "filter-media" process of DSpace to generate the thumbnails and full-text indexes. If unsuccessful, the system will retain the files so any errors can be fixed and the items re-submitted.

CSU makes use of the open-source Kaltura Streaming Media application to manage its video content. When a video is uploaded as part of our in-house batch ingest process, the source file is ingested using Kaltura's API. The resulting link (which can represent various video qualities and formats) is wrapped in a redirecting HTML file that is ingested into DSpace as the primary bitstream for the item. Thus, when a user clicks on the primary bitstream for a video file, they are redirected to the corresponding stream. A copy of the original video is retained as a private bitstream in DSpace.

In addition to making use of DSpace's OAI feed for use in our VuFind discovery layer, and with regional partners such as the Western Waters Digital Library, CSU has re-purposed the metadata and images into a series of dynamic websites. Roughly matching CSU's collection structure, the sites provide additional sorting and display options beyond those available through the DSpace interface. Developed for entry-level as well as international scholars, the sites see significant web traffic, such as the Garst Wildlife photograph collection which received more than 6 million visits in 2015.

4 New Areas for Development

Since its migration to DSpace, CSU has been engaged in discussions with an open educational resource library to provide a stable platform for the materials. The library is hosting basic metadata, in some cases the objects, in an SQL database and providing public access from their website. Institutional members of the library have requested MARC records of the resources for inclusion in their own local catalogs.

CSU has offered to migrate the database to a closed collection in DSpace for management and ongoing access. In addition to being able to provide persistent URLs via the library's own handles for individual items, educational resources published in other repositories will have their metadata and objects harvested into the CSU digital repository. Each resource with multiple versions will have distinctive links to each format type, e.g. a PDF or an ePub. CSU will be able to export MARC records for local catalogs from the DSpace collection into OCLC with the handles, and, via a Rest API, deliver the content to the library website for general access. The library will be provided administrative access to this private, non-indexed collection for adding new content as it becomes available.

Like many academic institutions, CSU has been digitizing its collections since the mid-1990s, and has been involved with establishment of best practices for scanning, for Dublin Core metadata, and for conversion of audio to digital formats. [1] However, its readiness for assuring long-term access to its digital content has been hindered by the lack of technical metadata for the objects, and not having geographically-dispersed storage. CSU is piloting Archivematica [2] to meet these digital preservation challenges. Our review of available products in the marketplace, spurred by the Digital POWRR project [3], made our selection of Archivematica the logical choice. Archivematica is an open-source application with an active development community which is fully integrated with DSpace. The application conforms with the Open Archives Information System model to create Archival Information Packages with the normalized files and metadata. These AIPs can then be stored in local or remote locations or with service providers.

5 Conclusion

Other articles have provided descriptions of DSpace's ingest capabilities for both ETDs [4] and single items for biographic or archival collections [5]. This case study, however, has analyzed the modifications required in order to make a successful multi-institution collaboration successful. Notably, a number of processes were improved to accommodate ingesting materials in bulk by non-system staff using a web utility, along with automating ETD processing to avoid human errors, speeding up accessibility for our users, and accommodating a widely variable ingest schedule.

Notes & References

[1] The Colorado (later the Collaborative) Digitization Program, developed best practices documents in a multi-state, multi-institution initiative in the early to mid-2000s. The best practices documents and some of the CDP's services have been absorbed by Lyrasis and are available from Lyrasis here.

[2] More information on CSU's Archivematica product and its micro-services is available here.

[3] The Digital POWRR (Preserving digital Objects with Restricted Resources) Project evaluated a variety of open-source and proprietary digital preservation options. See the Tool grid and detailed report here.

[4] Averkamp, S., & Lee, J. (2009). Repurposing ProQuest Metadata for Batch Ingesting ETDs into an Institutional Repository. *The Code4Lib Journal*, (7).

[5] Nash, J. & Wheeler, J. (2016). Desktop Batch Import Workflow for Ingesting Heterogeneous Collections: A Case Study with DSpace 5. *D-Lib Magazine* 22 (1-2). http://doi.org/10.1045/january2016-nash

About the Authors

Youssef Benchouaf is Software Developer, and Systems Administrator, with Colorado State University's Academic Computing and Networking Services.

Daniel Hamp is Systems Engineer, Software Developer, and High Performance Computing Administrator with Colorado State University's Academic Computing and Networking Services.

Mark Shelstad is Colorado State University Library's Coordinator for Digital Collection Services.