**Tibetan Buddhist Resource Center Report**

2/25/2016

**Staff Interviewed**

Chris Tomlinson, Senior Technologist

**Brief Summary of DRS Use**

The Tibetan Buddhist Resource Center (TBRC) is currently testing a workflow to deposit to the DRS2. Once the deposit agreement is smoothed out and they begin actively depositing, one person will be operating the workflow. Chris Tomlinson will not be the person doing the actual depositing, but Chris has been designing the workflow and doing the test depositing. When active depositing is initiated, they anticipate that it will be a full-time activity because they have about 13.51 terabytes of processed images, source image data, and derivative artifacts such as PDFs.

In the future, they plan on depositing all of their content, but presently they are focused on processed images comprised of JPEGs and TIFFs. Their archive is organized by works, which contain three to four volumes on average. Each work has a TBRC resource identifier (RID), and in late 2014, Andrea Goethals provided a mapping of these works to Hollis IDs. Though they would like to pursue bulk deposits, the maximum amount the DRS will accept per batch will be around 50 gigs – which they estimate will be around 30 volumes – per batch. Each imaged volume of work is approximately 500 pages. If they estimate two batches a day, they predict this will still take them up to nine months.

For the workflow Chris has created, BatchBuilder is not directly utilized. Instead, Chris has composed a command line script that calls BatchBuilder. The GUI is employed if there are any complications with running the script.

**General Workflow for Depositing**

* A one-time activity employs the BatchBuilder GUI to create a dummy project that will be used to create the project instances for individual deposit batches
* A list of work RIDs and the corresponding Hollis IDs are prepared as input to a deposit task for DRS
* The master script processes the works list, creating a BatchBuilder batch for every 30 volumes (this parameter was determined in testing during 2014 and early 2015 so as to limit the amount of data that would be submitted in a batch based on limitations of DRS import at the time)
* The script uses an XSLT processor to add Hollis IDs to the project files on a per volume basis
* The script calls BatchBuilder to build the template and build actions for each batch
* Once the batches have been generated, each batch is transferred to DRS dropbox via FileZilla
* A small script is then run that triggers the DRS import process via a rename of the batch.xml file
* Email fail and success reports are then examined to determine any needed corrections

**Inventory of Scripts and Tools Utilized for Deposit**

Scripts

* Shell script to perform the work of creating the batches
* Small script that triggers the import
* Small XSL script that performs the Hollis ID substitutions

Tools and systems used

* BatchBuilder
* Hollis
* FileZilla

**Summary of Primary Challenges and Desired Aspects**

There have been a couple of primary challenges to creating a smooth and efficient workflow. Timing and connectivity were issues early on, where they had problems triggering the deposit once it was Ftp’d. They circumvented this by transferring the batch then running a script to rename the batch.xml file. Chris did say that figuring out BatchBuilder would have been a lot more difficult without the help of Vitaly.

The main drawback, however, is the inability to do bulk deposits. Chris acknowledged some of the DRS features that make size a limitation, such as the creation of a METS file per image. For their purposes, a METS file makes more sense per-volume that they are depositing, and having so much duplication - 600 METS files for 600 images – is overkill for them even though it compresses nicely. This makes more sense from a per-artwork perspective, for instance. Overall, the DRS being oriented more towards accommodating large numbers of small batches has not been helpful for their needs of wanting to crack down on their backlog in a timely manner.

Chris referenced Amazon Web Services “Snowball” as a potential good example for data transport. They offer a “petabyte-scale data transport solution that uses secure appliances to transfer large amounts of data.” It has been a solution AWS implemented to bypass time and security issues common in large-scale data transfers. In their model, when the client creates a job, Amazon ships a Snowball appliance to the client, and the client attaches it their local network and selects the directories that they want to transfer. Then the files are encrypted and transferred to the appliance, and the appliance is shipped back.