OR2015 | 10th International Conference on Open Repositories
June 8-11, 2015, Indianapolis, Indiana, USA

Evaluating the Suitability and Sustainability of “Local Dropbox” Solutions to Complement a Research Data Repository

Alex Garnett, Data Curation and Digital Preservation Specialist, Simon Fraser University, garnett@sfu.ca
Carla Graebner, Liaison Librarian: Economics and Government Information; Data Curator and Digital Preservation Project Manager, cgraebne@sfu.ca

Session Type

Presentation

Abstract

Integration with cloud synchronization services such as Dropbox has often been touted as a means of facilitating deposit into university library repositories, but there are relatively few production examples and no turnkey plugins currently available to do this. We discuss our integration of the “local-Dropbox” solution *Pydio* with Islandora, noting the particular benefits and challenges associated with our approach, and provide guidance for any sites looking to undertake a similar implementation.

Conference Themes

Supporting Open Scholarship, Open Science, and Cultural Heritage
Managing Research (and Open) Data
Integrating with External Systems
Building the Perfect Repository

Keywords

Local data repositories, desktop sync, sustainability

Audience

This presentation should appeal to repository manager, librarians, and technical architects.

Presentation content

In some jurisdictions, including British Columbia, academic faculty are subject to rules which prevent public employees from using cloud services such as Dropbox which may store data in the United States. During a survey of SFU faculty during which we sought to establish their primary wants and needs for a local research data repository, some of the most commonly heard feedback was that faculty simply wanted to be able to use Dropbox, or something like it. Obviously, the decision to try to “complete” in any sense with Dropbox (in terms of scale, interface, and so forth) was not one that could be made lightly, but as we were at that time in the pilot phase of using our repository stack, we decided to evaluate potential solutions.

The two most viable open-source Dropbox-style synchronization solutions as of early 2013 when we conducted this evaluation, and still now in 2015, are *Pydio* (http://pyd.io) and *Owncloud* (http://owncloud.org/). Pydio was chosen over Owncloud for two reasons: unlike Owncloud, we were able to find evidence of it being used in production at scale (by the commercial hosting provider *Dreamhost,* as an in-browser file manager), and also unlike Owncloud, it had a functioning CAS plugin, which would support the single-sign-on method currently used across campus. Unlike Owncloud, it did not have a desktop synchronization client[1]; this was a good excuse to write our own: https://github.com/axfelix/datasync. We were able to get an instance of Pydio running on our VM server architecture with minimal difficulty in late 2013, and it has been relatively easy to maintain. So far, scalability has not been an issue (with individual collections running up into TB numbers, spanning thousands of small files), and upstream updates have all been improvements, with no feature deprecation that we have observed.

The solution was ideally to kill two birds with one stone: Pydio would lessen the overhead for deposit into Islandora while providing more storage than the “free” subscription tier of third-party cloud services without the policy conflict. Through a contract with DiscoveryGarden Inc., we created an Islandora plugin (https://github.com/axfelix/sfudora) to accept deposits initiated through Pydio, as well as a very small piece of code to add an interface hook on the Pydio side (figure 1). This allows users to individually select files from within the Pydio web interface, synchronized from a local disk, which they intend to deposit into Islandora. On doing so, they are automatically redirected to an Islandora deposit form of our creation, and upon submitting the form, the objects are transferred from Pydio to Islandora in the background. Islandora “Solution Pack” selection (determining how files of similar types are converted to more web-friendly and/or preservation-friendly formats) is bypassed, as files are automatically identified by MIME type.

In this way, Pydio provides a nice workaround for some of Islandora's shortcomings around the deposit interface. Specifically, while Islandora's customizable deposit forms and derivative creation hooks are generally excellent (and still utilized by deposits originating from Pydio, using our plugin), directly uploading large files to Islandora can be problematic due to the way the PHP thread handles them in the browser form. “Batch” deposit of multiple files into Islandora, which are intended to share the same metadata at the collection level, is also currently not well supported by the existing interface. While these problems are lessened for sites where deposit into Islandora is exclusively or near-exclusively performed by a repository manager that can develop against the Islandora API manually, they are particularly acute for implementations like ours that permit and encourage end-user deposit, making Pydio especially valuable.

[1] Pydio has since released both a Python and a Java desktop synchronization client, but currently these do not yet work in tandem with the CAS plugin, so it is good that we wrote our own.

So far, uptake has been relatively slow but steady. Participants in our research data repository project have been largely self-selecting, though we have succeeded in beginning to attract researchers from across campus who do not currently have a perpetual backup or file sharing solution in place. It is likely that many faculty continue to use Dropbox and Google Drive as a matter of course and will continue to do so, even when forced to cover those costs out of pocket. We are in no position to facilitate a large-scale migration, nor are we particularly incentivized to help enforce data privacy legislation; however, we are happy to be able to provide services to faculty who trust the university library more highly than external cloud providers, and for whom this is an important issue. The Islandora functionality “workarounds” have actually been of much greater use than expected, arguably moreso than our being able to offer Pydio itself, as smooth batch deposit has proven essential to the smooth operation of our respository services. To that end, we will very likely continue to support Pydio alongside the rest of our respository stack, and though there is no urgency to do so, we would be fairly confident in recommending it as a provincial or national-level solution if a larger-scale provider approached us with the means.