Synapse

Facilitating Large-Scale Data Management in Research Contexts

Daniel Andresen
Dept. of Computer Science
Kansas State University
Manhattan KS, USA
e-mail: dan@ksu.edu

Gerrick Teague
Software Development Dept.
Able Ant Design, Inc.
Manhattan KS, USA
e-mail: gerrick@ableant.com

***Abstract*— Research data management is becoming increasingly complex as the amount of data, metadata and code increases. Often, researchers must obtain multidisciplinary skills to acquire, transfer, share, and compute large datasets. In this paper, we present the results of an investigation into providing a familiar web-based experience for researchers to manage their data and code, leveraging popular, well-funded tools and services. We show how researchers can save time and avoid mistakes, and we provide a detailed discussion of our system architecture and implementation, and summarize the new capabilities, and time savings which can be achieved.**

Keywords- Globus; Dataverse; oid; metadata; research.

I. INTRODUCTION

Data management and processing increasingly consumes a modern researcher's time. Common tasks may include storing and sharing the data, generating metadata, developing codes to process the data, interfacing with High Performance Computing (HPC) clusters to process the data, access control, publishing and making the project discoverable to other researchers. Researchers often need to obtain significant knowledge outside of their domain in order to manage the data, often including command line interfaces, data transfer tools, and repositories, while other necessary tools, like data backup, and access control often go unutilized due to the hassle.

There currently exist tools to handle specific portions of the modern researcher's workflow in a generalized sense. For example, there are cloud data backup services, and data sharing tools such as Dropbox, but these services are not designed for a research context, so features, such as compliance with the Health Insurance Portability and Accountability Act (HIPAA), and fine-grained access control may be lacking. A further problem is the necessity to learn these different systems.

Technology is catching up to address the problems outlined above. Open OnDemand [2] provides a friendly graphical user interface when dealing with HPC clusters, but lacks robust data transfer. Harvard's Dataverse project [7] is an attempt to house, publish, and corroborate research, but lacks big data support. Ideally, there is a solution that handles most of the data flow for a research project to minimize the amount of out-of-wheelhouse technology a researcher must learn, but still be general purpose enough to provide benefit to multiple labs.

In this paper, we present Synapse, an open-source, web-based application. Synapse leverages both industry standards and emerging tools to provide a familiar, intuitive interface for researchers to handle common tasks. It greatly lowers the barrier of entry into many solutions including data transfer, HPC computing, data backup and housing, access controls, auto-metadata extraction, and research discovery [9].

Synapse achieves the above objectives by integrating three key technologies: 1) Dataverse handles the storage, providence, access control, and discoverability of research projects. 2) Globus [4] provides secure transfer of large amounts of data between given endpoints. 3) Open OnDemand provides a graphical user interface for processing researcher data on high performance computing clusters. We summarize our contributions as follows:

- We present current technologies that attempt to tackle this problem.
- We present the design, implementation and evaluation of a Synapse installation.
- We detail a modular metadata extraction system that can be extended to a particular lab's needs.
- We detail further work that can improve the system.

The rest of this paper first discusses related work in Section 2, and then describes our implementation in Section 3. Section 4 describes how we evaluated our system and the results. Section 5 presents our conclusions and describes future work.

II. RELATED WORK

Because the problem domain touches so many different areas, including data transfer, backup, storage, computing, discoverability, providence, and permissions, as well as the potentially significant resources required to implement such systems for data-large research projects, we find many excellent projects that handle a subset of the problem domain. The pace of improvement is very brisk in this space; technologies at the time of this writing may have overcome limitations reviewed here. Lastly, our search passed over many excellent products in favor of ease of integration, and market penetration. Our labs' requirements are specific: Storing and preservation of research data, Integration with our local HPC cluster, as well as auto metadata extraction from files. As such, we could not find a software solution that checked off all our requirements, but we found several excellent general-use solutions that came very close. Our

 ISBN: 978-1-61208-887-7

planning naturally went to discovering how to best integrate these projects into our solution.

The Center for Open Science produces the Open Science Framework (OSF) - an open-source web application that assists in research collaboration, document storage and archive, as well as registration of research projects. The application is very intuitive and supports a wide range of data sharing/transfer/storage technologies including Dropbox, and Google Drive. It handles persistent Uniform Resource Locators (URLs) for citing and sharing, access controls, version sharing, as well as publishing reports. In fact, in our research, OSF was a final candidate to incorporate into our system, since it fit most of our requirements. The OSF approach is to be the hub of data linked offsite in pre-existing storage shares, such as Dropbox, or Simple Storage Service (S3). Data can be hosted directly by OSF, but file sizes are limited to 5 gigabytes (GB). Overall, this system is very intuitive and flexible. Unfortunately, OSF is not currently designed to handle big data (neither very large data files, nor thousands of smaller files). Finally, market penetration seems low enough, with funding scarce enough, to warrant caution adopting this technology long term [6].

Dataverse is a project allowing users to store, discover, share, and analyze data. It is maintained by the Institute for Quantitative Social Science (IQSS) at Harvard University [3]. Dataverse strengths include wide adoption by academic research institutions, a vibrant community, rich documentation, constant funding, a robust permissions system, and a modest development pace. Its main weakness was its lack of big data support. But in recent months, it has made large strides to compensate. Dataverse's User Interface (UI) is a bit dated and slow, and the support for downloading many files is troublesome, but an active user base and development team can overcome these issues. Fig. 1 gives an example of the Dataverse UI. The structure of a Dataverse installation consists of a root Dataverse, which is a container of any combination of sub-Dataverses and datasets, the latter of which can only contain files and metadata. Due to its "batteries included" approach as well as its open and large community, we decided to leverage Dataverse for most of our needs, utilizing other software and connective code to flesh out our solution [7].

GitHub is a popular cloud-based version control repository. It extends Git, the de facto version control system for software code, which provides providence, collaboration, versioning, and powerful merging workflows to manage data. GitHub uses Git as its core, but also provides a powerful web-based UI to hide - yet complement - many of Git's powerful features. Being a web-first technology, it allows code to be discoverable by search engines, as well as providing gateway functionality with authentication and permissions. It also provides a highly programmable infrastructure to send and receive data. In a lot of ways, GitHub provides most everything needed for researchers to utilize. Unfortunately, due to technical limitations neither Git nor GitHub can handle files of any size. GitHub recommends repositories to be less than 1 GB, with a hard file size limit of no more than 100 megabytes (MB) per file [1].

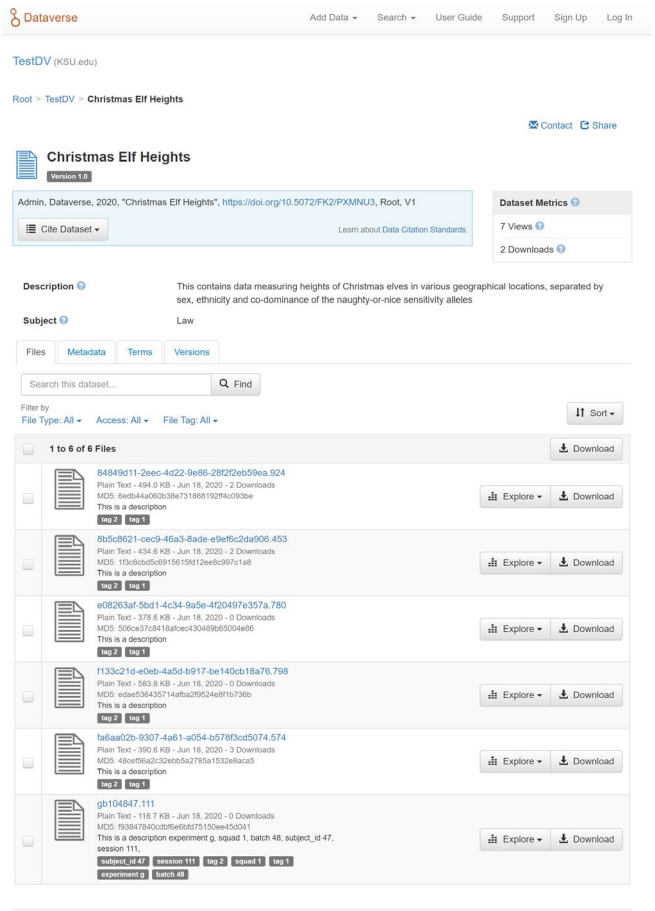

Figure 1. View of data inside a Dataverse data set.

Soon after the proof-of-concept for Synapse was developed, Scholars Portal based in Toronto Canada, showcased a Globus integration with their large Dataverse installation. They have at least 55 Canadian institutions utilizing their research Dataverse installation. They have received a grant in part to solve the large data problem, utilizing Globus's transfer capabilities. Their initial attempt leveraged the Dataverse and Globus UI's to allow the data to be imported. Due to the size of their installation, they opted for a S3 storage implementation. After a major design iteration, they have arrived at a solution similar to Synapse, but more general purpose. The Scholars Portal tool looks to fulfill all the requirements set forth for Synapse with the exceptions of HPC integration, and auto metadata extraction [5].

III. SYNAPSE DESIGN

In this section, we present the Synapse architecture in a high-level view to demonstrate how the major pieces work together as a system. Fig. 2 illustrates a broad overview of the system. Synapse is composed of four major components: Dataverse [7], which is the data storage, discovery and access control system, Globus, the data transfer system [8],

 ISBN: 978-1-61208-887-7

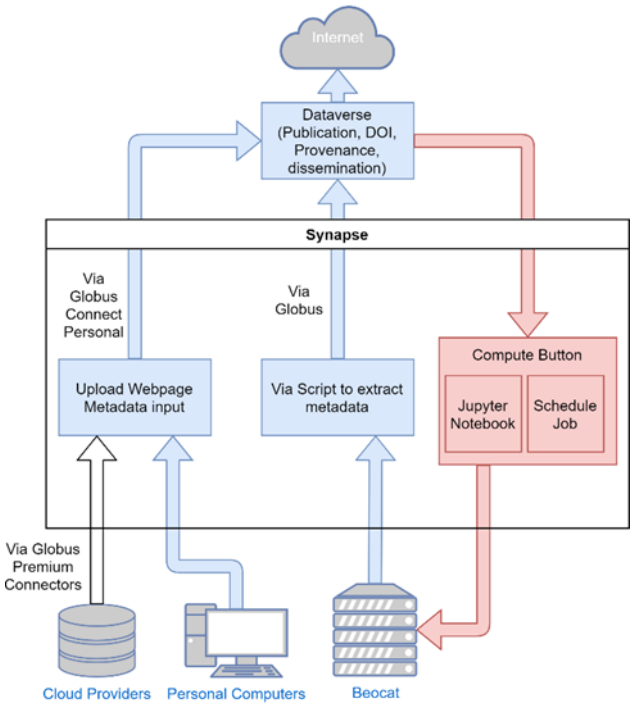

Figure 2. Principal elements of the Synapse system.

Open OnDemand, a graphical user interface for high performance computing clusters [2], and the Synapse web application, which provides metadata extraction and glues these pieces together with a web-based user interface.

We will describe the two main use cases of the Synapse website, importing data into Dataverse with metadata and exporting data out of Dataverse to be used in a HPC cluster. Importing data into Dataverse can be described in a sequence of steps involving 4 systems: a researcher's computer, the Synapse web server (herein referred to simply as 'Synapse'), the Dataverse server, and Globus. Before the flow starts, a user must have a Dataverse account, a Globus account, and a valid Globus endpoint located on the researcher's computer with access to the source files to be transferred. The Globus endpoint could be either the free Globus Connect Personal service to be used on workstations, or a paid Globus Connect server. First, we will go over the of each use case, then explain in more detail interesting steps.

The first use case is importing data into Dataverse. Fig. 3 depicts the process. Step 1 involves the user navigating to the Synapse website. If the user's Globus credentials have expired, a Globus login is required using Open Authorization (OAuth), redirecting the user to the Globus authentication webpage. After a successful login, the user is redirected back to the Synapse page. The user can then select any endpoints they have access to, in this case the endpoint linked to the source files. The user may then drag-and-drop the files onto the panel in the webpage, and Synapse will query the selected Globus endpoint with the limited file information the drag and drop operation gives us to get the absolute path of the files to be uploaded which Globus will then use. Once one or more paths are found, they are displayed to the user for confirmation. The user then selects which dataset in Dataverse to put the files into, what lab they belong to, as well providing any additional metadata to describe what is being uploaded. Once finished, the user will submit the job to be processed.

Step 2 is fully automated by Synapse and performs the following tasks sequentially:

1. *Store a manifest file* detailing the transfer job, source, destination, metadata, etc.
2. *Extract metadata found in the filenames or data files themselves.*
3. *Prepare a custom Globus endpoint* on the Dataverse instance as the destination. With access controls specific to the Globus user logged in.
4. *Tell Globus to initiate a transfer* of the data, capturing the task id so we can query the Globus system for progress updates. At this point, the task is handed off to Globus. Globus will eventually transfer the files to the destination if possible or fail.

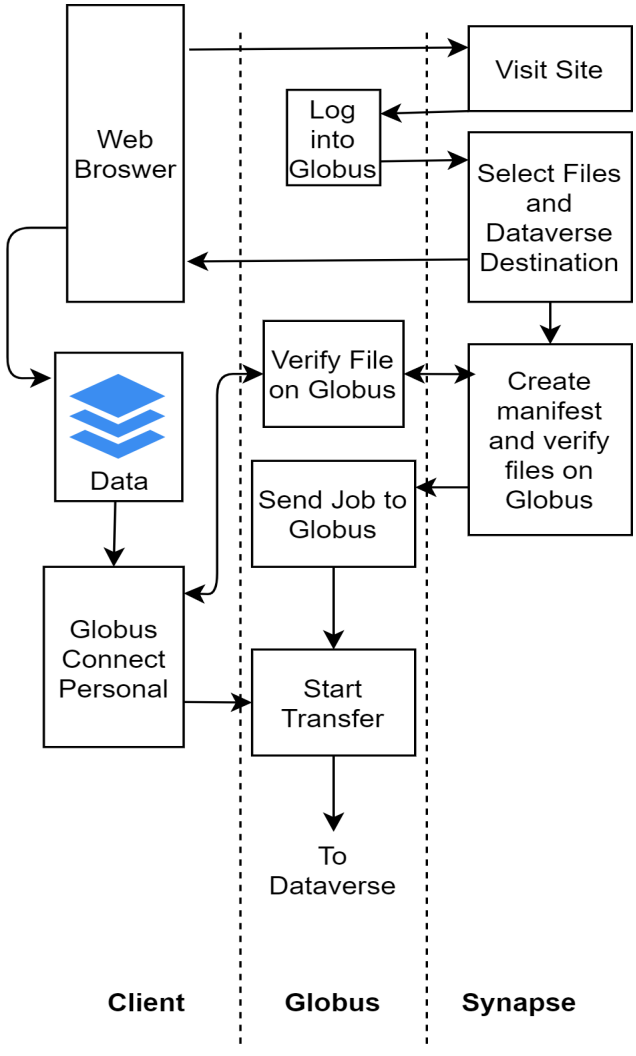

Figure 3. Synapse operation flow when importing into Dataverse.

Step 3 consists of a script running on the Dataverse installation that also has access to the Globus destination endpoint. This script will monitor the destination endpoint for new directories that represent the jobs mentioned in Step 2. When a new directory is found, the script will retrieve the manifest file described in Step 2 and add it to a processing list. Then the script will iterate over the processing list and query Globus to see if the files are still being transferred. Synapse is then updated with the new status, to inform the user. If the Globus transfer has successfully completed the job, then the files are imported into the specified Dataverse dataset using the Dataverse Application Programming Interface (API), given the user's Dataverse API key, which details that user's access permissions. During the importing of files into Dataverse, the Synapse web server is called to post updates to the user. Once the importing finishes, the user will be notified. Fig. 5 displays the main Synapse page that provides the UI for the import process. The "Globus Source" field indicates where Synapse should pull the data from, the "Your Lab" field handles the automatic metadata extraction. The "Upload to Dataset" lets the user select a Dataverse dataset from a list of datasets the user has access to. The "Description" and "Keyword" fields allow extra metadata to be set for all files uploaded.

One point of interest is the fact that we rely on the operating system hosting both the Dataverse installation and the Globus endpoint to temporarily store the file, and then import it into Dataverse. Since the Dataverse importing process can take a considerable amount of time, docking the file on the same portable operating system interface (POSIX) file system allows for some optimizations, such as the ability to import a dummy file with the same metadata as a very large file then update the dummy file pointer to point to the actual large file. Negating both the lengthy importing process of a large file but also negating the necessity of copying the data from one location to another. Using POSIX has a significant disadvantage as well: Dataverse's future seems to be moving toward a simple storage system architecture (commonly referred to as S3). With S3 a different methodology must be used leveraging recent Dataverse updates.

Another subsystem worth discussing is the *automatic metadata extraction*. We have found that labs can have quite consistent internal conventions categorizing data produced from experiments, but data categorization may not be consistent between labs of a department. We decided upon a flexible plug-in approach to extract metadata from the data. A user can write a plug-in for a lab that inherits from a metadata class that extracts the data specific for that lab, or a set of experiments conforming to the format the plug-in can parse. Synapse will auto-detect the plug-in and display it amongst the list of metadata extractors available in the data import page of the Synapse system. All files will be exposed to the selected metadata extractor, which will decide if the file is a match for this extractor. If so, it will extract the data and will associate the extracted data with the file, then store the information in the manifest file described in step 2. After transfer, the script outlined in Step 3 will apply the metadata to a file's tags and description in Dataverse during import. *The tag and description fields of Dataverse are searchable, allowing for quick data discovery.*

Metadata parsed the way mentioned above and imported using the Synapse system auto-applies the metadata specific to that file for each file. Fig. 4 depicts a comparison of the Synapse metadata extraction vs a standard Dataverse upload. The top file is imported into Dataverse using the Dataverse interface, the bottom entry contains auto-populated metadata imported utilizing Synapse. The extra data is file specific; there is a proportional time savings based on the number of files to import into Dataverse utilizing Synapse vs the standard import method; the latter method needing to manually input the metadata per file after import. Both the batch-specific metadata, as well as the file-specific metadata can be searched upon using the Dataverse search tool as well as the data harvesting / sharing engine, if enabled.

Figure 4. Dataverse data with and without metadata.

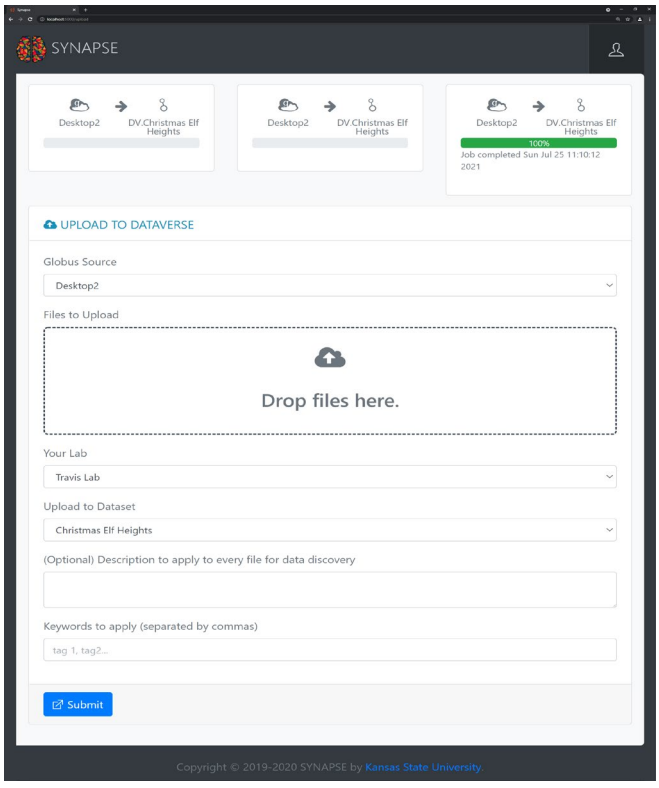

Figure 5. Synapse data import page.

The second use case of Synapse use is to move data from Dataverse into a HPC cluster for processing. This starts at the Dataverse site. If a user has access to the data, they may select the desired file(s) and select “Explore”, then “Send to Beocat”. This makes a call to Synapse. Synapse ensures the user is logged into Globus, grabs the Globus destination ID of the HPC cluster endpoint, and pulls the selected data out of Dataverse via the API, and submits the job to Globus to transfer to the HPC cluster’s endpoint. Note that it will transfer to the logged in user’s Globus HPC cluster endpoint. The Dataverse data plugin system was utilized to enable stored files to be forwarded to Synapse for transfer. In order to transfer files back to Dataverse from the HPC cluster, then one can use Synapse and just select the HPC cluster endpoint for transferring back to Dataverse, for another import.

The final component of the Synapse system is Open OnDemand maintained by the Ohio Supercomputer Center (OSC), which is an open-source software suite that allows visual access to supercomputing resources via a web browser. It has three main features: 1) Scheduling HPC jobs with a graphical interface. 2) use a Virtual Network Computing (VNC) desktop on the HPC cluster, and 3) run specific applications such as Jupyter, and MATLAB directly on the HPC cluster. OSC’s approach is like Synapse’s, using a web browser in a familiar way to interface with disparate technologies to minimize the learning curve. Fig. 6 shows an example of RStudio running on the Beocat HPC cluster at Kansas State University.

Open OnDemand includes a visual file manager, like a researcher’s personal computer file systems to manipulate files. OSC has found that using this file manager is sufficient for managing approximately nine gigabytes of data; for anything greater Globus is recommended [2]. Once a file is moved from Dataverse to the HPC cluster, then Open OnDemand can be utilized for processing.

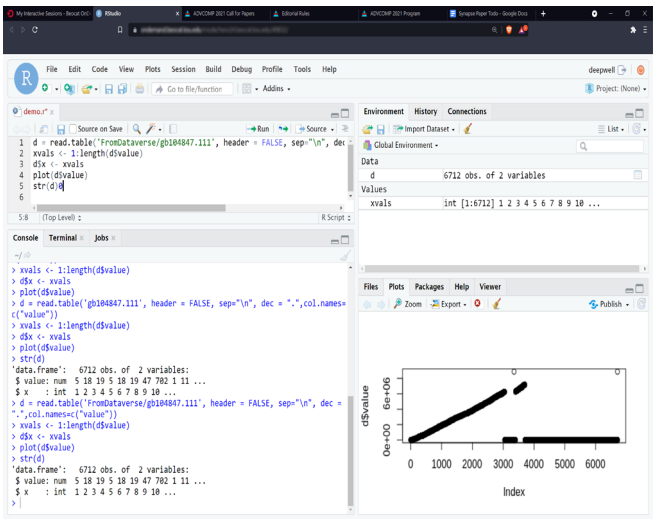

Figure 6. Open OnDemand running RStudio on a HPC cluster.

IV. EVALUATION

In this section we evaluate the time Synapse takes to complete various data importing jobs into Dataverse. We will compare this against the upload feature found in Dataverse itself. There are three machines that we will use for evaluation: 1) The client machine where the data is originally stored. 2) The server hosting the Synapse web server, and 3) The Dataverse installation server. For all tests described below, the client machine (1 in Fig. 7) is a MacBook Pro i9-9980HK CPU running Windows 10 20H2, with 32GB random access memory (RAM), and a 1 terabyte (TB) solid state drive (SSD). The Synapse web server (2 in Fig. 7) is also a MacBook Pro i9-9980HK CPU, 32 GB RAM, 1TB SSD running Ubuntu 20.04.1 under a virtual machine. Finally, the Dataverse server (3 in Fig. 7) is also a virtual machine running CentOS 7.7.1908 quad core with 32GB of Ram running on a local HPC cluster. The Dataverse version we tested with was 4.18.1. This setup is depicted in Fig. 7.

All tests were conducted using the same dataset. This dataset consisted of a series of random generated files, with random data to allow minimal compression which Hypertext Transfer Protocol (HTTP) and Globus might employ for transit. There are 5 datasets used:

- 1 Kilobyte (KB) files
- 10 KB files
- 100 KB files
- 1 MB Files
- 10 MB Files

In each set, we recorded imports using one, 10, 100, and 1000 files. The smallest test transferred 1KB of data (1 file * 1KB File size), while the largest test imported 10GB of data (1000 10MB files).

In each run, we record 5 data points:

- Total import time
- Time to process metadata
- Time to transfer data via Globus
- Time to import data to Dataverse after the Globus transfer
- The amount of time it takes to import data without metadata via the Dataverse website.

We group the data by the 5 data points, with plots for each file size tested.

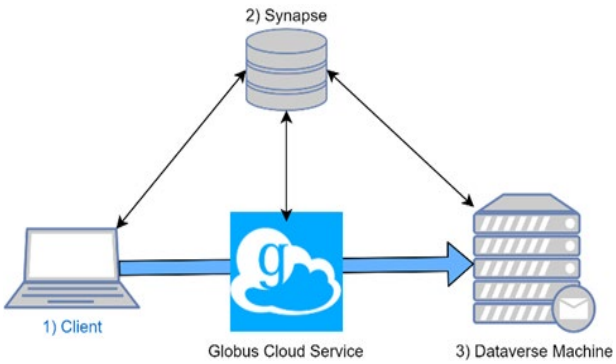

Figure 7. Components of the evaluation system.

 ISBN: 978-1-61208-887-7

First, we look at the overall time it takes to import data into Dataverse utilizing Synapse. As we can see from Fig. 8, it is only until we get to 10MB files do we see much of an increase in processing time from smaller file sizes. This suggests the overhead of the various processes in the Synapse system is taking up the lion's share of the compute time, and it is only when our files sizes get substantial, do we see bandwidth limiting factors come into play. To examine this further, Fig. 9 charts the amount of 'lost time' not accounted for by the major steps the Synapse system takes in importing a file described by our 5 data points. This non-account for time includes user interaction after a user drops files to be processed but before they press the submit button, polling time waiting for the service running on the Dataverse server looking for new jobs, network latency between communication between the various machines, etc. We can see this 'lost time' appears to be influenced by the number of files to be processed but is a much greater percentage of the overall time for small number of files, rather than file sizes, bearing out the reasoning of overhead time. Now, we will break out the individual components of the import.

Fig. 10 describes the time took to process the metadata in milliseconds. Note that the file size had virtually no effect on the metadata processing and was solely dependent upon the number of files to process. This is due to the metadata extracted was all contained in the file name, rather than the file contents. While Synapse's metadata system is flexible enough to handle even file contents, if the extractors use the file names, the system could conceivably handle millions of files.

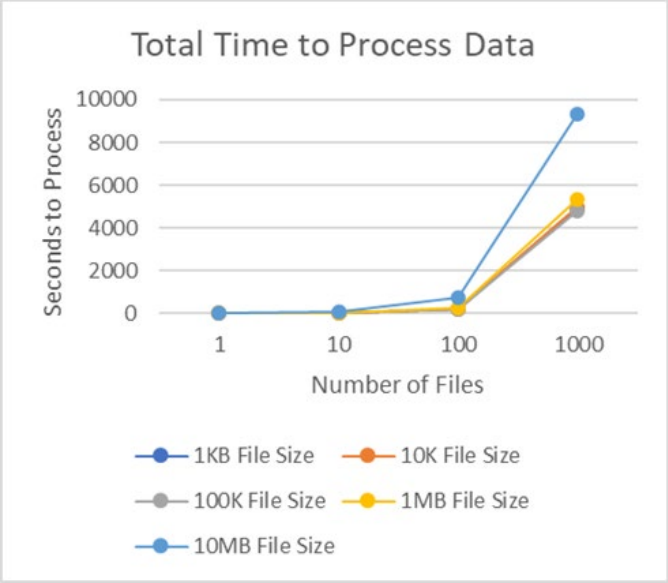

Figure 8. Total length of time to import data into Dataverse via Synapse.

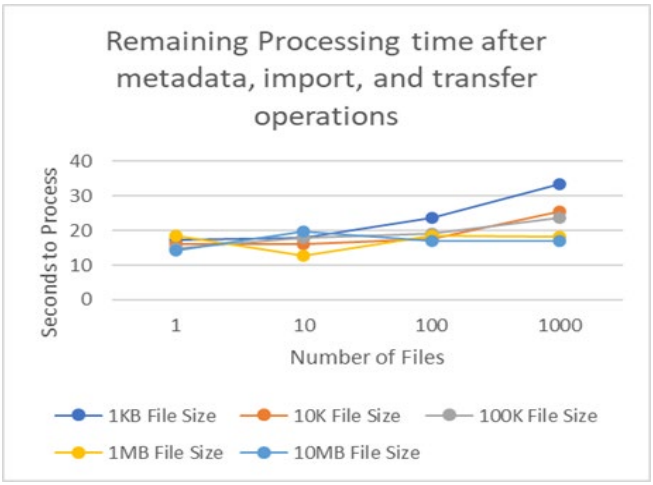

Figure 9. Time not accounted for by data points.

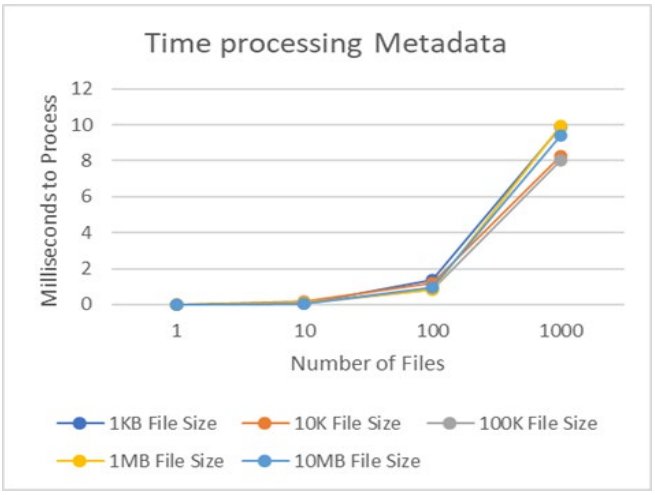

Figure 10. Milliseconds took to process metadata.

Fig. 11 plots the time it takes for the data, once described to Globus, to be transferred to the Dataverse endpoint. Not surprisingly, it is dependent upon total bytes to be transferred. File transfer systems have a per-file overhead, but in Globus this appears to be small.

Fig. 12 shows the time it takes to import the file into Dataverse once it arrives on the Dataverse machine via Globus. As opposed to Globus, it appears the main differentiating factor is the per-file overhead. Even for small file sizes, it takes quite a while to process many files. Interestingly, it is not a linear progression. With a given file size the import time is not proportional to the number of files processed but appears to be exponential. For example, importing 100 1KB files took 163.45 seconds, but 1,000 1KB files took 4885.52 seconds, a 29.9x increase rather than expected 10x. As the graph shows, this holds true regardless of the file size. This phenomenon may be due to design deficiencies in the Dataverse import API process, or due to the Synapse importing residing on the Dataverse machine.

 ISBN: 978-1-61208-887-7

This importing process after the files have already been transferred to the Dataverse machine is where the greatest amount of efficiency gains can be realized and will likely be a focus of any future work. This will be elaborated in Section V.

Dataverse has a webpage where one can drag and drop files to be imported into a dataset much like Synapse, but without the metadata, and large file support. For comparison, Fig. 13 depicts the time it took to import via the Dataverse webpage. Fig. 14 adds an arbitrary 5 seconds per file to emulate metadata input customized per file. As Fig. 14 illustrates, if metadata is not a requirement, and one has a good connection to the Dataverse server, and they are dealing with a relatively small number of smaller files, the Dataverse website upload tool can yield significant time savings in the import process.

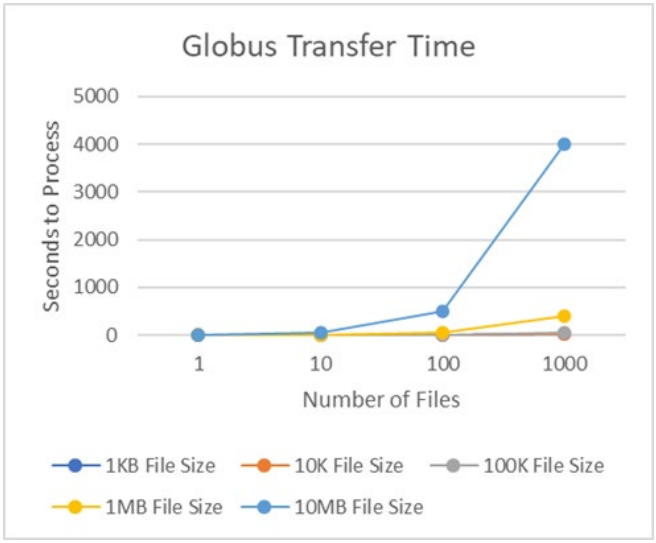

Figure 11. Globus transfer time.

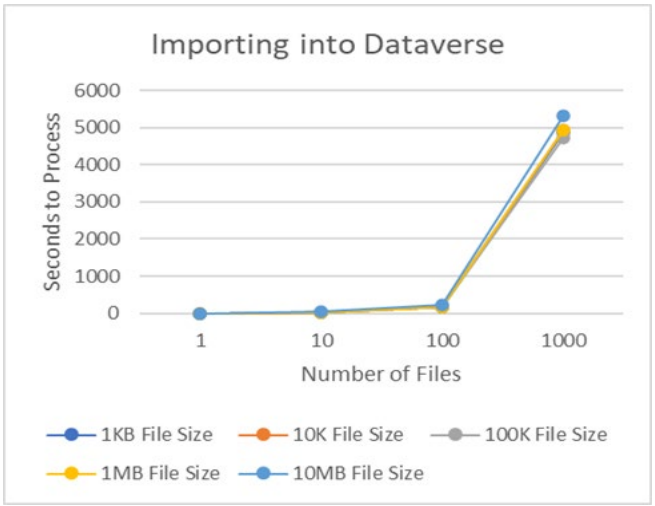

Figure 12. Seconds to import into Dataverse after data is transferred.

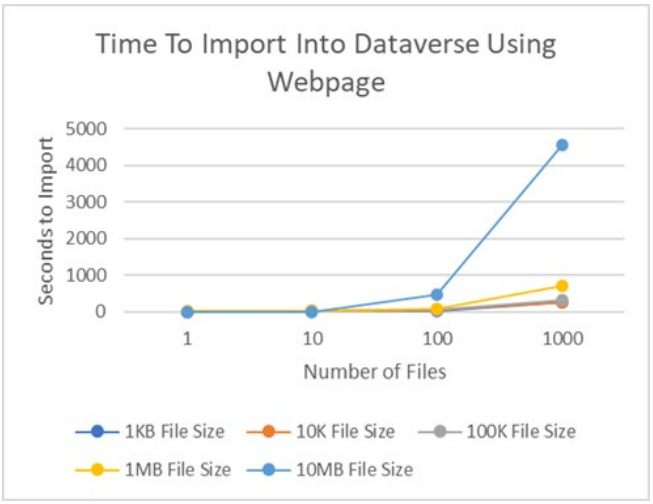

Figure 13. Time to import data using Dataverse website.

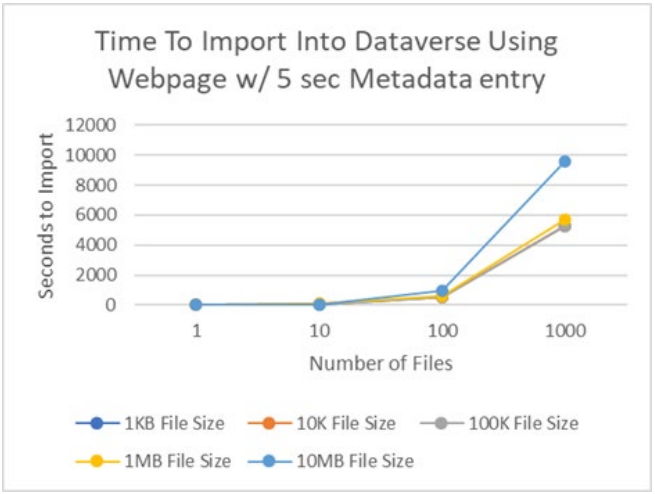

Figure 14. Dataverse website entry with metadata time allowance.

One of the clear advantages of Synapse is the ability to automate repetitive metadata entry over multiple files. Another advantage is the ability to frontload all the effort in saving the data to the repository, so the user can walk away after data entry, since error recovery is built into the data transfer process. Even if the import process takes days or weeks, the user does not need to watch the import or wait for it to finish to then describe the data after import.

V. FUTURE WORK

There are many features that can be implemented to make Synapse a more valuable tool. The most beneficial upgrade would be moving away from a traditional POSIX storage system into a modern S3 compatible storage system. S3 adoption among the Dataverse installations around the world is growing and will likely be the de-facto method of data storage within Dataverse. The Institute for Quantitative Social Science (IQSS), which develops and maintains the

 ISBN: 978-1-61208-887-7

Dataverse project recognizes this fact and is in talks with key institutions such as Scholars' Portal, to facilitate both a more robust S3 storage implementation, as well as tighter Globus integration. We should be able to leverage such advances to improve the performance of the Synapse system.

There can also be further optimization of importing speeds with newer versions of Dataverse, which allows for optimization of the import process. Currently Dataverse inspects and processes each file to - among other things - determine the file contents and produce a preview of the data. This is computationally expensive, and for large files, prohibits importing at all. Third party workarounds have obviated this pre-processing. The IQSS team has recognized this need, and is planning on facilitating this trend, possibly leveraging parallel importing.

Dataverse also supports a rich method of metadata harvesting and sharing utilizing the Open Archives Initiative Protocol for Metadata Harvesting (OAI-PMH) which provides interoperability between data repositories to increase discoverability of research. Additionally, we can vastly decrease the amount of time to import very large files by creating a small dummy file representing the actual file and import that data in with the metadata associated with the dummy file. After importing, we can find the POSIX storage path in the Dataverse database, and then do a POSIX 'move' command, moving the original file that landed on the Dataverse server via Globus, overwriting the destination of the dummy file that Dataverse imported. Finally, we need to update the database for the new file size. This method would prevent copying the data, which is input/output (IO) intensive for very large files and would also minimize the amount of processing Dataverse does to files being imported.

The version of Dataverse we tested provided limited API support for moving arbitrary files out of Dataverse. Either a single file could be moved, or the entire dataset. In the future, with collaboration with Dataverse, we could provide additional features to pull arbitrary files out of Dataverse. For example, the results of a file search could be selected and sent via Synapse to a HPC endpoint.

The Open OnDemand portion residing on KSU's local HPC cluster has been used in a production environment, with positive feedback. Synapse as a whole needs to be evaluated by research groups. Since many of the main goals have been achieved, moving the project into an Agile methodology to close the feedback loop now would be appropriate [10].

VI. CONCLUSION

We have presented Synapse, a web-based tool to enable researchers to store, manage, process and publish data and results using familiar technologies. We have evaluated Synapse with various datasets and shown its usefulness in Metadata extraction, and resilience to network faults. If metadata description is required on a per-file basis, Synapse can greatly decrease the import time into a Dataverse repository.

The main limitation observed is the relative slowness of the import process, compared to the native Dataverse method. The second sizable limitation is our current approach utilizing an antiquated POSIX file system. Despite these issues, Synapse allows for researchers to manage automated metadata extraction, data storage, transfer, computing and publication all wrapped in a familiar web browser experience so researchers can focus on the research.

ACKNOWLEDGEMENTS

This work was supported by the Cognitive and Neurobiological Approaches to Plasticity (CNAP) Center of Biomedical Research Excellence (COBRE) of the National Institutes of Health under grant number P20GM113109. The content is solely the responsibility of the authors and does not necessarily represent the official views of the National Institutes of Health.

REFERENCES

[1] GitHub Development Platform Size Limitations. Available from: https://docs.github.com/en/github/managing-large-files/working-with-large-files/what-is-my-disk-quota#file-and-repository-size-limitations 2021.8.30

[2] D. Hudak, D. Johnson, A. Chalker, J. Nicklas, E. Franz, T. Dockendorf, and B. McMichael, "Open OnDemand: A web-based client portal for HPC centers." *Journal of Open Source Software*, 3(25), 622, 2018. https://doi.org/10.21105/joss.00622

[3] Institute for Quantitative Social Science (IQSS), available from: https://www.iq.harvard.edu/ 2021.8.31

[4] I. Foster, "Globus Online: Accelerating and Democratizing Science through Cloud-Based Services," *Internet Computing*, IEEE , vol. 15, no. 3, pp. 70,73, May-June 2011

[5] Scholars Portal Dataverse, available from: https://learn.scholarsportal.info/all-guides/dataverse/ 2021.8.30

[6] Center for Open Science's Open Science Foundation, available from: https://osf.io, 2021.8.30

[7] Harvard's Dataverse, available from: https://dataverse.org 2021.8.30

[8] B. Allen, J. Bresnahan, L. Childers, I. Foster, G. Kandaswamy, R. Kettimuthu, J. Kordas, M. Link, S. Martin, K. Pickett, S. Tuecke, "Software as a service for data scientists," *Communications of the ACM*, vol. 55, no. 2, pp. 81,88, February 2012 doi:10.1145/2076450.2076468

[9] Synapse open-source code, available from: https://github.com/cnap-cobre/synapse-globus 2021.8.31

[10] A. Srivastava, S. Bhardwaj and S. Saraswat, "SCRUM model for agile methodology," *Proceedings of the 2017 International Conference on Computing, Communication and Automation (ICCCA)*, pp. 864-869, 2017. doi: 10.1109/CCAA.2017.8229928