Always on the Move: Transient Software and Data Migrations

D. Wilcox
DuraSpace
9450 SW Gemini Drive #79059
Beaverton, OR 97008
USA
dwilcox@duraspace.org

ABSTRACT

Software is transient: it's the data that matter. Anyone who works with software understands that applications come and go, and even those that last many years go through numerous upgrades and re-architectures as they try to keep up with the latest technological advances. This flux creates a problem: how do we efficiently move our data through changing software and version upgrades as losslessly as possible? The Fedora community recently confronted this problem, which led to the development of a tool that could have broad utility for data migrations.

Fedora is a flexible, extensible, open source repository platform for managing, preserving, and providing access to digital content. Fedora has gone through several major version upgrades since its initial conception almost 20 years ago, most recently with the move to Fedora 4. Fedora 4 is a complete software re-architecture, which necessitates a data migration from previous versions of Fedora. This experience led the Fedora community to focus on making it as easy as possible to get data into and out of Fedora in standard formats, an effort that culminated in the creation of an import/export utility that standardizes on RDF and the BagIt specification to transfer data between versions of Fedora as well as external preservation systems. This effort coincides with a recently chartered Research Data Alliance working group with a mandate to explore technologies and standards for interoperability between repository platforms and make recommendations on this topic. The group is exploring the Fedora import/export utility as a possible basis for broader interoperability between diverse repository platforms by building on this standard import/export functionality. This paper will provide an overview of the import/export efforts that have taken place so far, and discuss how we might achieve broader interoperability and ease data migrations through the work of the RDA Research Data Repository Interoperability working group.

KEYWORDS

Fedora, repository, open source, migration, data

1 INTRODUCTION

Fedora – the flexible, extensible, durable object repository architecture – is both a concept and a software implementation. The concept of Fedora was first proposed in a research paper by Sandy Payette and Carol Lagoze in 1997. This paper lays out a model for digital objects in an open repository architecture that would later be implemented as software: "A fundamental requirement of an open architecture for digital libraries is a reliable and secure means to store and access digital content. FEDORA is a digital object and repository architecture designed to achieve these requirements, while at the same time providing extensibility and interoperability." [3] Fedora was envisaged as a flexible and extensible architecture that could interoperate with new and existing systems and services. These ideas were present in the original version of the Fedora software, and they continue to be represented in the current version to this day.

However, like any other software application, Fedora will not be around forever – at least not in its current form. Fedora has already gone through a major re-architecture from version 3 to 4, necessitating a data migration for any existing users. This highlights a problem that memory institutions will continue to face over the years: how to maintain the resources and support that will be required for inevitable data migrations as software applications change or are phased out.

2 TRANSIENT SOFTWARE

If there is one thing all memory institutions should be able to agree on it is this: the data, not the software, are what matter most. Of course, institutions need software to manage, preserve, and provide access to their digital collections, but it is the collections themselves that matter; the software will inevitably need to be replaced, continually, over time. The Fedora community grappled with this basic truth recently as we made the change from Fedora 3 to 4, and determined that an in-place upgrade would not be possible due to deep architectural changes in the codebase. Thus, the community embarked on a quest to migrate their data from Fedora 3 to 4, but this turned out to be more difficult than we originally imagined.

Anyone who has undertaken a data migration will tell you that they are always more difficult than originally planned. There are several reasons for this: different data models and formats from the old system to the new one, different metadata standards, the time it takes to move large amounts of data, and, perhaps most difficult, accounting for all the inconsistencies in the source data. Over the years, through different curators and standards, data inevitably becomes inconsistent; new metadata standards are adopted, custom metadata fields are created when the current standard doesn't seem to support a given need, names are misspelled, etc. Just figuring out what you actually have and normalizing (to the extent possible) all the data is probably the most time-consuming part of a migration.

3 MIGRATING DATA

Once your data is normalized you need some way to get it out of the source system and into the new system. This can be challenging as different systems store data in different ways, and there are no universal tools for doing this kind of work. This migration use case was one of the primary drivers behind the import/export tool that the Fedora community has developed.

As a community-supported, open source project, Fedora is focused on adopting widely used standards and making data easy to get in and easy to get out in a standardized way. This goes back to the earlier point about the primacy of the data – Fedora will not be around, at least in its current form, forever, so adopters should have a relatively easy way to get their data out in a standard format that doesn't require Fedora or any proprietary software to access and understand.

New Fedora features are developed by and for the Fedora community; as such, each new feature starts with an expression of interest, followed by meetings where stakeholders gather to discuss potential use cases and functionality. In the case of the import/export tool, a set of use cases were added to the Fedora wiki and vetted by stakeholders. These use cases include the following [1]:

1. Transfer between Fedora and external preservation systems, such as APTrust, MetaArchive, LOCKSS, DPN, Archivematica, etc
2. Package [Export] the content of a single Fedora container and all its descendant resources
3. Transfer between fedora instances or (more generally) from Fedora to an LDP archive
4. Load [Import] the contents of a package into a specified container.
5. Round-tripping resources in Fedora in support of backup/restore
6. Round-tripping resources in Fedora in support of Fedora repository version upgrades
7. Batch loading arbitrary sets of resources from metadata spreadsheet and binaries
8. Import or export containers or binaries using add, overwrite, or delete operations.

Once the use cases were agreed upon, functional requirements were derived and organized by priority. These requirements were then assigned to code sprints and community developers signed up to work on each sprint. This process allowed the community to break the effort down into manageable chunks and produce milestone releases that could be tested and verified by stakeholders. The testing results could then be rolled back into the development process to be worked on during the next sprint. After following this process for several cycles, the development team completed an initial release of the tool [4].

The import/export tool is a command line utility that can be supplied with parameters to both export content from Fedora and import content back into the repository. The default export format is RDF, with serialization options that include JSON-LD, XML, n-triples, N3, Turtle, and plaintext. Resources can also optionally be exported with any binary files they may be associated with to achieve a complete representation of the repository contents. Additionally, the BagIt [2] packaging format is supported for both exports and imports. This format is supported by several long-term digital preservation systems, making it easier to move content from Fedora to these services. Since BagIt is a fairly loose standard, BagIt Profiles [6] are also supported to provide more information on the contents and structure of the Bag.

Once content has been exported from Fedora, it can be reimported to the same repository or a different Fedora instance using the same tool. This supports the migration use cases listed in the previous section, and lays the groundwork for more robust migration scenarios in the future. Additionally, by leveraging BagIt profiles, the import/export utility can account for differences in data modeling between repository instances; the Bag manifest contains relevant details that can be provided to the tool when importing content to the destination repository, which ensures important structural information is not lost between systems.

While this initial work is important in its own right, the real value comes from the foundation it lays for future work. As an external tool, the import/export utility could be made to work with a variety of repository systems without the need to modify the core code of these other systems. The next section lays out the potential for leveraging this opportunity more broadly.

4 ACHIEVING INTEROPERABILITY

The import/export utility was deliberately designed as a separate module; it interacts with Fedora via the REST-API [5], and therefore does not modify any of Fedora's core code. This design decision opens the door for broader interoperability between repository systems by providing a means to interact with other systems without modifying core code. The main barrier in this case is adoption; developers and maintainers of other repository systems would need to adopt the utility as a means of getting data into and out of their systems.

This need for broad adoption may be addressed by a recently formed Research Data Alliance working group. The Research Data Repository Interoperability working group was struck to establish a standard by which content can be moved between different repository platforms. The group has created a primer document outlining candidate standards and technologies, of which the Fedora import/export utility is one. The group has also considered specifying a generic API for this purpose, but this would present a problem for users of existing platforms who do not have the resources to modify or upgrade their existing installations. Many users customize their repository platforms in ways that make it impossible to upgrade to a new version without significant code rewrites, thus setting the burden for adoption of a generic API very high. The import/export utility presents a much more attainable solution precisely because it does not require core repository code modifications in order to use.

The Research Data Repository Interoperability working group will meet at the 9th plenary meeting of the Research Data Alliance on April 6th, 2017 to finalize the primer document and decide on which technology/standard to pursue. Should the Fedora import/export utility be selected, the group will proceed to work with representatives of other repository platforms (who have already been identified) to determine how to integrate the utility with as many repository platforms as possible. This work will in turn feed into a published specification document that will outline the standards implemented by the utility and how it can be integrated with other platforms that may not be on our initial list. The overall goal of this work is to facilitate data migrations between as many repository platforms as possible, both in order to ease the burden of system migrations and to make it easier to share data objects between different platforms.

4 CONCLUSIONS

Experience has taught us that no software applications last forever; even long-lasting applications go through significant changes and rewrites that often require data migrations from one version to another. By recognizing and acknowledging this fact, we can plan for data migrations by adopting standards and practices that make the data more transparent, allocating resources for migration work, and developing tools to aid in the migration process.

The transition from Fedora 3 to 4 presented an opportunity for the Fedora community to better align with well-recognized standards and develop tooling to support data migrations, at least in the case of moving data between different versions and instances of Fedora. A group of community stakeholders drafted use cases, gathered requirements, and allocated developer resources to designing and building an import/export utility that can be used by anyone wishing to move data between Fedora instances or from Fedora to an external preservation system using the BagIt packaging standard.

While this work represents a significant milestone for the Fedora community, it has even broader implications for repository data migrations more generally. The Research Data Alliance Research Data Repository Interoperability working group will evaluate the import/export utility as a starting point for migrating data between different repository platforms. Should this utility be selected, group members will work to draft a specification document outlining how repositories can adapt the import/export utility to work with their repository platform to get data in and out in standardized formats. As this specification becomes more widely adopted, we will ease the burden of data migrations and make it easier to both upgrade existing repository platforms and migrate to new platforms in the future.

ACKNOWLEDGMENTS

This work would not be possible without the generous support of the Fedora community, particularly the DuraSpace members who allocate funding to the Fedora project on an annual basis. The import/export utility was designed, built, and tested by a diverse group of Fedora community stakeholders, all of whom deserve great thanks for their dedication to the project and our goals.

REFERENCES

[1] Design – Import – Export. 2017. Retrieved March 31, 2017, from http://wiki.duraspace.org/display/FF/Design+-+Import+-+Export.

[2] J. Kunze, J. Littman, L. Madden, E. Summers, A. Boyko, and B. Vargas. 2016. The BagIt File Packaging Format (V0.97). Retrieved March 31, 2017 from https://tools.ietf.org/html/draft-kunze-bagit-14.

[3] S. Payette and C. Lagoze. 1998. Flexible and Extensible Digital Object and Repository Architecture (FEDORA). Proceedings of the Second European Conference on Research and Advanced Technology for Digital Libraries in Lecture Notes in Computer Science 1513, 41: 41–59. https://doi.org/10.1007/3-540-49653-X_4.

[4] Release fcrepo-import-export-0.1.0. 2016. Retrieved from https://github.com/fcrepo4-labs/fcrepo-import-export/releases/tag/fcrepo-import-export-0.1.0.

[5] RESTful HTTP API. 2016. Retrieved from https://wiki.duraspace.org/display/FEDORA4x/RESTful+HTTP+API.

[6] N. Ruest and M. Jordan. 2015. BagIt Profiles Specification. Retrieved from https://github.com/ruebot/bagit-profiles.