The Oxford Common File Layout

Andrew Hankinson (Bodleian Libraries, University of Oxford), Simeon Warner (Cornell University), David Wilcox (DuraSpace)

Abstract

The Oxford Common File Layout (OCFL) initiative is an effort to define a shared approach to file hierarchy for long-term preservation. What began as a discussion at the Fedora and Samvera Camp held in Oxford, UK in September of 2017 has grown into a focused community effort. Many repository systems store large amounts of data that are difficult to migrate, and such migrations risk data loss. Within a storage system, digital objects should be designed under the assumption that they will be accessed and managed by a variety of applications. This effort is an attempt to decouple the structure of the persisted files from the software that might manage it by creating an expectation of file hierarchy to which software applications must conform, whether implemented in a conventional filesystem with files and directories, or in an object store. Thus the file hierarchy functions as a storage Application Programming Interface (API). This effort addresses three primary requirements: 1) completeness, so that a repository can be rebuilt from the files, 2) parsability, both by humans and machines in the absence of original software, and 3) robustness against errors, corruption and migration between storage technologies. This paper describes motivations, principles and anticipated features of the proposed file layout. The OCFL initiative aims to produce draft specifications for trial use by Fall 2018.

Themes

- Collaboration and capacity-building
- Technological infrastructure
- Sustainable digital preservation approaches and communities

Keywords

Preservation, File, Filesystem, Object Store, API

Introduction

Most existing repository systems store digital objects in application-specific ways. For example, a DSpace [4] filesystem hierarchy cannot be managed by Fedora [5], and vice-versa. Some institutions, however, choose to implement file structures designed to be independent of specific digital repository systems. The two most significant efforts for this are the d-Flat, ReDD, and associated specifications from the California Digital Library [3], and the Moab efforts at Stanford [1]. Both of these efforts attempted to describe software-agnostic approaches to storing digital

contents on a filesystem. Other efforts include “RecordSilo” as part of the University of Oxford’s DataBank system [9] and the University of Notre Dame’s “Bendo” adaptation of the Moab format [2]. The BagIt specification [6], adopted by Library of Congress and Research Data Alliance [7], is also notable in this context for providing inspiration for several file layouts in use, despite being designed primarily as a format for object transfer and not object structure. Such software-agnostic approaches are the motivation for this work.

The Oxford Common File Layout (OCFL) initiative began as a discussion at the Fedora and Samvera Camp at Oxford University in September of 2017 - and it is from this city that the initiative derives its name. It quickly became clear that the use cases under discussion would be of interest to a much broader community than the Fedora and Samvera [11] groups, so an initial conference call was advertised widely to the community of digital preservation practitioners. The result was a call with 49 participants, which spoke to the interest in this topic. Since the initial call, a discussion paper has been drafted and circulated, an editors group has been established, community mailing lists and a Slack channel have been created, and several more conference calls have taken place.

Within the context of this paper, a digital repository software application is understood to be software that manages digital objects. We consider these digital objects to function as Archival Information Packages [10]; each is a *“container that contains all the necessary information to allow long term preservation and access to archival holdings”*. This discussion will focus primarily on the file contents; secondary consideration will be given to the metadata as it relates to file management within a digital object.

In the coming months, the OCFL initiative will produce a specification for a file-and-folder hierarchy for digital object storage. It is hoped that the community creating this specification will then develop re-usable tools meeting their shared needs. This paper details the history, goals, and scope of the initiative, as well as summarizes the current progress and timeline for finalizing the specification.

Motivations

1. Many repository systems store large amounts of data (hundreds of terabytes or petabytes) that are time consuming and/or expensive to migrate or reorganize. Thus stable storage organization has significant value in an environment where software applications change more rapidly.
2. Storage migrations place the underlying content at risk and thus should be minimized. Mapping from one system to another may involve changing semantics (“what constitutes a file?”) or be subject to built-in optimizations (e.g., de-duplication based on checksum) that may not be immediately obvious post-migration. While lengthy and comprehensive testing procedures can help identify and isolate these problems, it will not entirely eliminate them.

3. Digital objects within a storage system should be designed with the assumption that they will be managed by many different applications. Application-specific needs should be separable from the contents that the application is asked to manage. Access and digital preservation is a process and not a fixed objective, and no single application will encapsulate all necessary actions.
4. File and directory hierarchies are pervasive organizational metaphors across most computing systems, and as such can persist across CPU architectures, disk formats, and operating systems. Object storage systems, such as Amazon S3, also support this metaphor. As such, it is a fundamentally stable "technology" on which to build a digital repository.

The OCFL initiative, in following the example of Moab and the CDL family of specifications, seeks to promote the filesystem hierarchy as separable from a management application, and a fundamentally stable and loosely coupled component of a digital repository. To this end, it will:

1. Provide a specification of the file layout independent of repository software implementations;
2. Enable the preservation of the object, and versions of the object;
3. Maintain a record of actions on the object;
4. Permit the object, and the collection of objects, to be validated against a versioned specification;
5. Not require special software to provide basic functions of viewing, moving, copying, renaming, and deleting, other than those shipped with a given operating system;
6. Work with a wide variety of storage systems, including disk (local and network), tape, and cloud object storage services;
7. Permit the full restoration of digital objects within a digital repository using only data stored on the filesystem.

Principles and Features

Use cases are still being collected and reviewed [8] but initial work suggests that the following principles and features should be adopted:

- An OCFL digital object is a collection of files and metadata. OCFL should be agnostic to content and metadata types (for all but minimal administrative metadata). Indeed, the OCFL is unaffected by local choices of descriptive and preservation metadata, so we expect institutions to continue to leverage community standards such as PREMIS.
- OCFL digital objects should be organized based on their identifiers. OCFL should be agnostic to the identifier type, treating it simply as a unique key identifying the object.
- Content versioning should be implemented within OCFL digital objects. It is expected that a particular version will be immutable once created, and hash-based file identification within an object will avoid storing multiple copies of unchanged files.

- OCFL digital objects should support the storage of records (logs) of actions taken on an object - particularly those actions that create new versions of objects. The exact nature of these records will be governed by local considerations but their location within the object will be defined, and they will be outside of the content versioning structure (e.g. recording a fixity check should not trigger version creation).
- OCFL digital objects should be amenable to validation which will verify fixity of content and consistent internal structure in accord with the specification.

Implementation

The OCFL defines a very specific layout of files, directories, and system metadata. With that information, simple clients can be written to interact with resources in OCFL or emergent libraries can be used for such interaction. Project participants intend to implement reusable software libraries allowing easy and efficient interaction with OCFL objects on disk or in object stores. Since the OCFL specifies persistence layout, existing systems that will wish to natively interact with OCFL resources will have to rewrite their persistence abstraction. One of the primary goals of the OCFL is to specify a preservation-focused layout that recognizes that persisted resources will outlast any application that manages those resources. Institutions and applications that do not want to migrate their data with each application migration will therefore be motivated to implement the OCFL.

Conclusion

There has been significant community interest in the OCFL initiative since its inception in September 2017. This interest validates the need for a shared file layout and has resulted in a working group that will draft a trial specification. There is strong agreement on key ideas that are based on previous work, combined with the desire to make improvements based on lessons learned. This is a tightly scoped effort with a short timeframe to produce a draft specification for trial and iterative development. These conditions create a promising environment for progress to meet a real community need.

Acknowledgments

This work is the result of community discussions with many participants

References

[1] Richard Anderson. 2013. The Moab Design for Digital Object Versioning. *The Code4Lib Journal* 21 (July 2013). Retrieved from http://journal.code4lib.org/articles/8482.

[2] Bendo. (December 2016). Retrieved from https://github.com/ndlib/bendo/blob/master/architecture/bundle.md.

[3] D-flat: A Simple File System Convention for Digital Object Storage. (September 2013). Retrieved from https://confluence.ucop.edu/display/Curation/D-flat.

[4] DSpace | DSpace is a turnkey institutional repository application. Retrieved from http://www.dspace.org/.
[5] Fedora Repository. Retrieved from https://fedorarepository.org/.
[6] J. Kunze, J. Littman, L. Madden, E. Summers, A. Boyko, B. Vargas. The BagIt File Packaging Format (V0.97) draft-kunze-bagit-14. (October 2016). Retrieved from https://tools.ietf.org/html/draft-kunze-bagit-14.
[7] Eoghan Ó Carragáin. Approaches to Research Data Packaging - RDA 11th Plenary BoF meeting. Retrieved from https://rd-alliance.org/approaches-research-data-packaging-rda-11th-plenary-bof-meeting.
[8] OCFL Use-Cases. Retrieved from https://github.com/OCFL/Use-Cases/issues.
[9] RecordSilo. (October 2014). Retrieved from https://github.com/anusharanganathan/RecordSilo.
[10] Reference Model for an Open Archival Information System (OAIS). 2012. *Recommended Practice*, 2. Retrieved from https://public.ccsds.org/pubs/650x0m2.pdf.
[11] Samvera - an open source repository solution for digital content. 2018. Retrieved from http://samvera.org.