ORA4 objects in OCFL

Product description

An overview of how ORA4 objects will be stored in an OCFL-conformant file system. Alongside this document, the reader should have access to the ORA Data Model METS mapping document [ORA-METS], and the sample ORA4 OCFL storage root [ORA-OCFL-ROOT]. In addition, this document should be read alongside the OCFL specification [OCFL-SPEC].

About this document

Version: 0.1dev
Author: Tom Wrobel <thomas.wrobel@bodleian.ox.ac.uk>
Created: 27 March 2019
Last modified: 27 March 2019

Background

As part of the ORA4 Digital Preservation Solution, ORA4 will seek to store a human-readable, stable, digital preservation copy of an object. The most appropriate format for this is in a file system that conforms to the Oxford Common File System specification (OCFL).

OCFL is “an application-independent approach to the storage of digital objects in a structured, transparent, and predictable manner. It is designed to promote long-term access and management of digital objects within digital repositories.” [OCFL-SPEC]. The OCFL specification describes the way in which multiple versions of an object are stored, and provides a set of rules which enable an object to be validated. The history of an object, including its current state, is stored in an inventory.json file in the object’s root.

OCFL concepts

OCFL has the concept of a ‘storage root’ directory. This storage root contains the OCFL objects, each in their own directory.

OCFL does not prescribe where object directories are stored, only that any directories between the storage root and an OCFL object are empty. However, it is advised that whatever the logical structure of an OCFL storage root, that the path from the storage root to an OCFL object is intelligible from its OCFL object identifier [OCFL-IMPLEMENTATION-NOTES].

In addition, OCFL makes a series of important requirements: OCFL mandates that once a file is written it is immutable, and that once an object version is created, that this version is immutable.

ORA OCFL storage root and file system structure

ORA OCFL objects will be stored in a directory corresponding for their UUID. For an ORA object with the pid “ora_ abcdef01-abcd-abcd-abcd-abcdef013456", the ORA object directory will be “abcdef01-abcd-abcd-abcd-abcdef013456".

The path to this directory will be a pair-tree directory composed of the first four pairs of bytes in the UUID. For example (see also [ORA-OCFL-ROOT])

```
[storage_root]
          ├── 0=ocfl_1.0
      ├── ocfl_1.0.txt
      ├── ocfl_layout.json
      └── ab
```

```
└── cd
    └── ef
        └── 01
            └── abcdef01-abcd-abcd-abcd-abcdef013456
```

OCFL object structure

Each version of an OCFL object will contain a METS file describing the object, its metadata, the binary files associated with the object, and their metadata. This METS file will conform to the specification laid down in [ORA-METS].

The METS file will be named {UUID}.ora{datamodel-major-version}.mets.xml, e.g. abcdef01-abcd-abcd-abcd-abcdef013456.ora2.mets.xml

The METS file will list file locations relative to the OCFL version's 'logical file path' (see [OCFL SPEC] Terminology): the a METS file will use this as the value of the the Flocat[@xlink:href] property for that file. For example (see also sample METS files in objects in the [ORA-OCFL-ROOT]:

```
    <mets:fileGrp>
        <mets:file ID="FILENAME2" MIMETYPE="mimetype">
            <mets:FLocat xmlns:xlink="http://www.w3.org/1999/xlink"
LOCTYPE="URL" xlink:href="file:///test-file-3.txt"/>
        </mets:file>
    </mets:fileGrp>
```

A simple ORA OCFL object will therefore have the following structure:

```
abcdef01-abcd-abcd-abcd-abcdef013456
    ├── 0=ocfl_object_1.0
    ├── inventory.json
    ├── inventory.json.sha512
    └── v1
        ├── inventory.json
        ├── inventory.json.sha512
        └── content
            ├── abcdef01-abcd-abcd-abcd-abcdef013456.ora2.mets.xml
            └── test-file.txt
```

A multiple versioned ORA OCFL object will a version directory for each version. Each version MUST contain a METS file describing that version. In the following example:

1. An initial object was created with one file, *test-file.txt*
2. Two additional files were subsequently deposited, *test-file-2.txt* and *test-file-3.txt*
3. One of the files was found to be an incorrect file, and was removed from the object (the contents of an object in any given version are managed in the object's inventory.json file, see the [OCFL-SPEC].

```
abcdef01-abcd-abcd-abcd-abcdef013456
    ├── 0=ocfl_object_1.0
    ├── inventory.json
    ├── inventory.json.sha512
    └── v1
        ├── inventory.json
        ├── inventory.json.sha512
        └── content
            ├── abcdef01-abcd-abcd-abcd-abcdef013456.ora2.mets.xml
            └── test-file.txt
    └── v2
```

```
        ├── inventory.json
        ├── inventory.json.sha512
        └── content
            ├── abcdef01-abcd-abcd-abcd-abcdef013456.ora2.mets.xml
            ├── test-file-2.txt
            └── test-file-3.txt
└── v3
        ├── inventory.json
        ├── inventory.json.sha512
        └── content
            └── abcdef01-abcd-abcd-abcd-abcdef013456.ora2.mets.xml
```

Purging a file

In order to purge a file, it will be necessary to create a new object as if that file had never existed. This new object will replace the existing one, see [OCFL-IMPLEMENTATION-NOTES] File Purging.

Sample objects

A suite of sample objects will be maintained in the [ORA-OCFL ROOT].

References

[OCFL-IMPLEMENTATION-NOTES]
Hankinson, A. et al. (2019), *Implementation notes*. URL: https://ocfl.io/0.2/implementation-notes

[OCFL-SPEC]
Hankinson, A. et al. (2019), *Oxford Common File Layout Specification 0.2 - Candidate Recommendation, 20 March 2019*. URL: https://ocfl.io/0.2/spec/

[ORA-METS]
Wrobel, T. (2019), *The ORA Data Model in METS*, URL: https://github.com/tomwrobel/ora_data_model/blob/master/ora_datamodel_to_mets-2.0.xml

[ORA-OCFL-ROOT]
URL: https://github.com/tomwrobel/ora_data_model/OCFL/storage_root