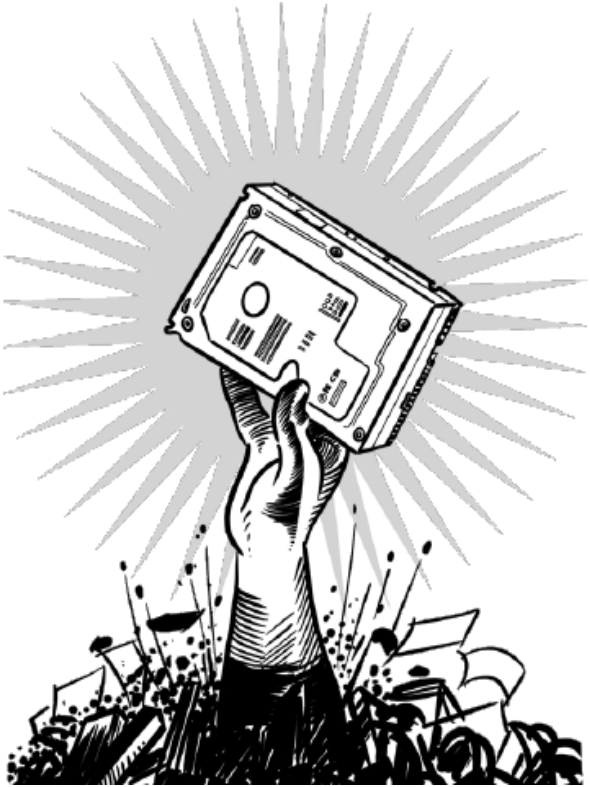
Enabling AIP Versioning and Delta AIPs with OCFL

Neil Jefferies – Open Preservation Foundation (OPF)
Sven Schlarb – AIT / eArchiving Initiative

eArchiving Initiative Training Webinar

Introduction to OCFL

Neil Jefferies
Executive Director of the Open Preservation Foundation
Innovation Specialist at the Bodleian Library at University of Oxford



European Commission

The Problem(s)

- OAIS, TDR, NDSA, etc., tell you *what* to do for digital preservation but not *how*.
  - These pseudo-standards also keep changing over time.
  - However, this is not desirable for technical Digital Preservation standards.
- OCFL aims to provide a stable (or at least backwards-compatible) mechanism for the *how*.
- Digital objects need to outlast the systems that create and preserve them. Digital preservation is therefore an exercise in risk management.
  - Migrations via export/re-ingest are risky.
  - Preservation should minimise dependence on any system-specific behaviours.
  - Backup and disaster recovery are not preservation.
- OCFL aims to limit the need for migration and aims for broad system compatibility.

The Problem(s)

- The reason for preserving objects is for their information content.
  - This depends on data, metadata and relationships to other objects (context and provenance).
  - These change over time – all need to be preserved and tracked equally.
- OCFL aims to handle complex digital objects with multiple versions (but does not define the composition of an object).
- ...FAIR (Findable, Accessible, Interoperable, Reusable) is very relevant even if it originates from another context.
  - Recognises the requirements and economic case for digital preservation.
  - Recognises that “metadata” can be as useful as “data” in the long term, perhaps more so in some cases!

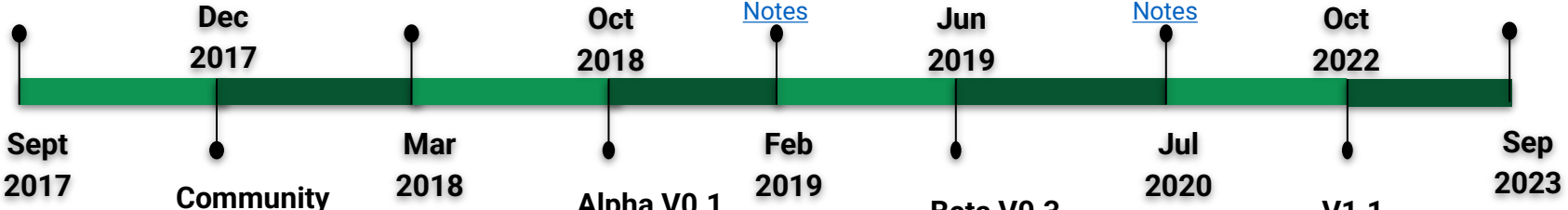
OCFL History

Origins

Discussions at Fedora/ Samvera Camp at Oxford (hence the “O”)

Recognition of the gap

Proposal: The OCFL



Community

Inaugural Community Meeting

47 attendees from 32 institutions

Collecting use cases, practice and ideas

Editorial Group

Formed from Community Members

Alpha V0.1

Specification

Implementation Notes

Alpha V0.2

Specification

Implementation Notes

Beta V0.3

Specification

Implementation Notes

V1.0

Specification

Implementation Notes

V1.1

Specification

Implementation Notes

V2.0 Cycle Begins

European Commission

OCFL Specifications

Specification – describes objects *at rest* in storage

- OCFL Object
- OCFL Storage Root

Implementation Notes – background and recommendations for objects *in motion* (not part of the formal standard)

- Implementation Guidance
- Operations on OCFL Objects

Oxford Common File Layout Specification

7 October 2022, updated 7 November 2024

This Version:

- https://ocfl.io/1.1/spec/

Latest Published Version:

- https://ocfl.io/latest/spec/

Editors:

- Neil Jefferies, Bodleian Libraries, University of Oxford
- Rosalyn Metz, Emory University
- Julian Morley, Stanford University
- Simeon Warner, Cornell University
- Andrew Woods, Harvard University

OCFL object

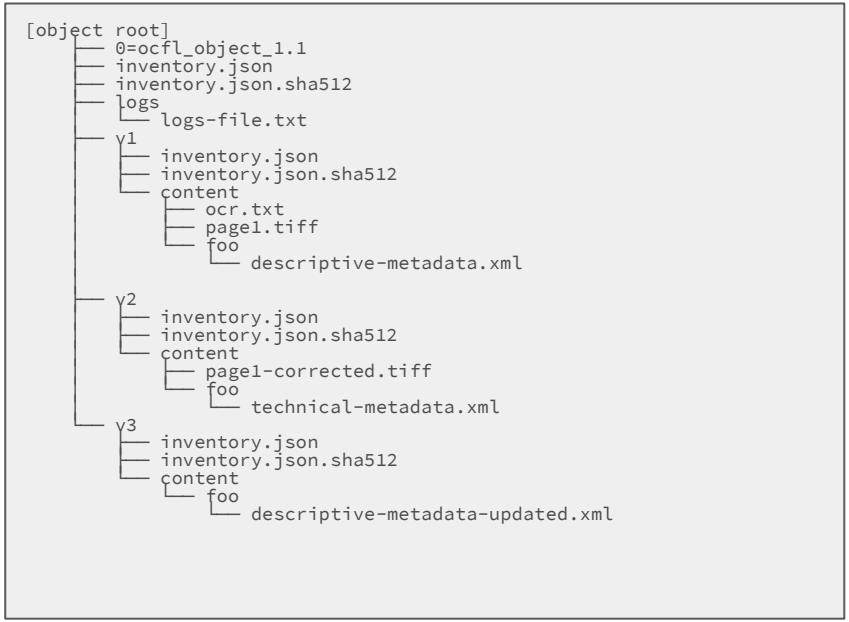
An ***OCFL Object*** is a group of one or more content files and administrative information that together have a ***unique identifier***.

The object may contain a sequence of ***versions*** organised into ***version directories*** that represent the evolution of the object's contents.

Object described by an Inventory (**inventory.json**) file.

OCFl does not define what constitutes an object or version.

Options for local and community extensions.



OCFL storage root

The ***base directory*** of an OCFL storage layout.

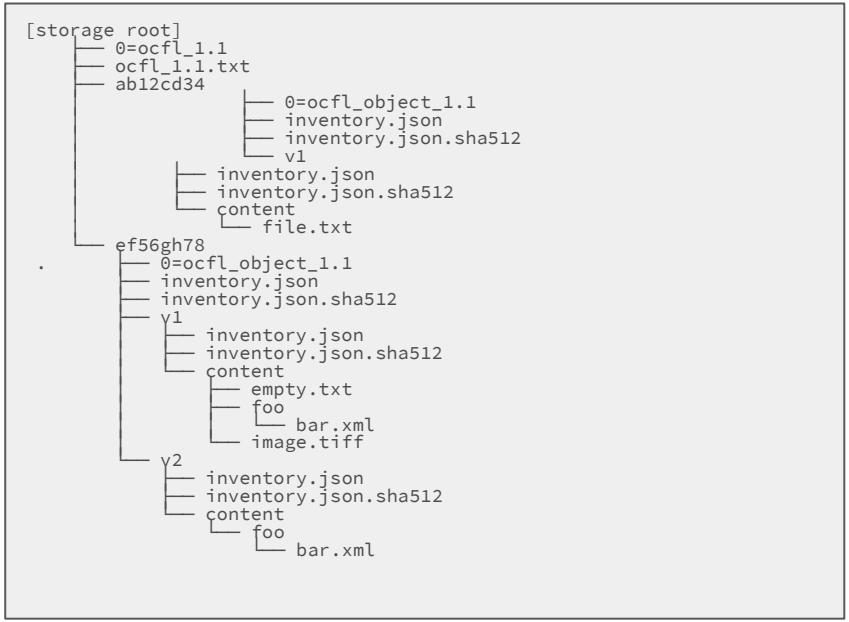
Should also ***contain the OCFL specification*** in human-readable plain-text format.

MUST contain the ***root conformance declaration.***

OCFL Objects may conform to the ***same or earlier*** version of the specification.

The ***storage hierarchy*** must terminate with an OCFL Object Root.

Options for local and community extensions.



European Commission

Benefits of using OCFL

- **Completeness** – A repository can be rebuilt purely from the files it stores.
- **Parsability** – By humans and machines, to ensure content can be understood in the absence of original software.
- **Robustness** – Against errors, corruption (accidental or deliberate), and migration between storage technologies.
- **Versioning** – Repositories can make changes to objects, but their history persists, to allow referential integrity and recoverability.
- **Storage diversity** – Content can be stored on diverse storage infrastructures including cloud object stores.
- **Efficiency** – Many design decisions are made with a view to computational, bandwidth and storage efficiency in the light of real-world experience.

Benefits: completeness

A repository can be rebuilt purely from the files it stores

- An intellectual object is stored as a single entity combining data, metadata, and history. Any previous version can be accessed with minimal overhead.
- Falls in line with Trusted Digital Repositories (TDR, ISO 16363), NDSA Levels of Preservation, and Open Archival Information Systems (OAIS).
- Allows ease of mapping from one system to another, with the ability to keep a picture of the past.

Many of these standards talk about what you should do, but not how. OCFL provides the how.

Benefits: parsability

By humans and machines, to ensure content can be understood in the absence of original software

- In disaster recovery situations, humans should be able to understand the content using basic OS tools.
- Machine readability allows for simple access applications to be built placed on top of an existing OCFL storage root.
- The specification explicitly allows for the inclusion of human-readable documentation in the file layout.

```
[object root]
    ├── 0=ocfl_object_1.0
    ├── inventory.json
    ├── inventory.json.sha512
    ├── v1
    │   ├── inventory.json
    │   ├── inventory.json.sha512
    │   └── content
    │       ├── metadata
    │       │   ├── descriptive.xml
    │       │   ├── technical.xml
    │       │   └── rights.xml
    │       ├── page-1-OCR.txt
    │       ├── page-1.tiff
    │       ├── page-2-OCR.txt
    │       └── page-2.tiff
    └── v2
        ├── inventory.json
        ├── inventory.json.sha512
        └── content
            └── metadata
                └── descriptive.xml
```

Benefits: robustness

Against errors, corruption (accidental or deliberate), and migration between storage technologies

- Fixity is built into the OCFL as part of content-addressing.
  - Multiple fixity algorithms supported – can change over an object's lifetime.
- Content can easily be validated using the inventory.json.
  - An inventory exists for every version.
  - Top-level inventory is duplicated. Improves recovery chances.
- Objects can be completely self-contained. Migration and replication are simple (efficient) file copying operations.
- The implementation notes document robust approaches to operations on OCFL objects and failure recovery procedures.

Benefits: versioning

Repositories can make changes to objects, but their history persists – to allow referential integrity and recoverability.

- Changes to objects are tracked over time. Previous object versions are considered immutable.
- Forward delta differencing is employed to reduce the amount of content stored.
- Previous versions of objects can be reconstructed using the inventory.json file.


```
},
"type": "https://ocfl.io/1.0/spec/#inventory",
"versions": {
  "v1": {
    "created": "2018-01-01T01:01:01Z",
    "message": "Initial import",
    "state": {
      "7dcc35...c31": [ "v1/content/metadata/descriptive.xml" ],
      "cf83e1...a3e": [ "v1/content/metadata/technical.xml"],
      "ffccf6...62e": [ "v1/content/metadata/rights.xml"],
      "ge72e1...d6e": [ "v1/content/page-1.tiff" ],
      "jdferd...56d": [ "v1/content/page-1-OCR.txt" ],
      "gk4er6...57d": [ "v1/content/page-2.tiff"],
      "7adjhe...4ad": [ "v1/content/page-2-OCR.txt"]
    },
    "user": {
      "address": "alice@example.com",
      "name": "Alice"
    }
  },
  "v2": {
    "created": "2018-02-02T02:02:02Z",
    "message": "Fix descriptive.xml",
    "state": {
      "4d27c8...b53": [ "v2/content/metadata/descriptive.xml" ],
    }
```

Benefits of OCFL: storage diversity

Content can be stored on diverse storage infrastructures including cloud object stores

- Supports conventional filesystem metaphor but…
  - Assume only basic file and folder functionality.
  - Links (of any sort) are avoided because implementations vary.
- Separates the logical file path in an object's structure from its actual path on storage
  - This allows the OCFL to work with various storage infrastructures, including object stores prevalent in cloud offerings (e.g. Amazon S3).
  - This also allows object structures that do not match filesystems limitations to be stored – for example, very long pathnames or unusual character sets.
  - Mapping is held in the inventory.

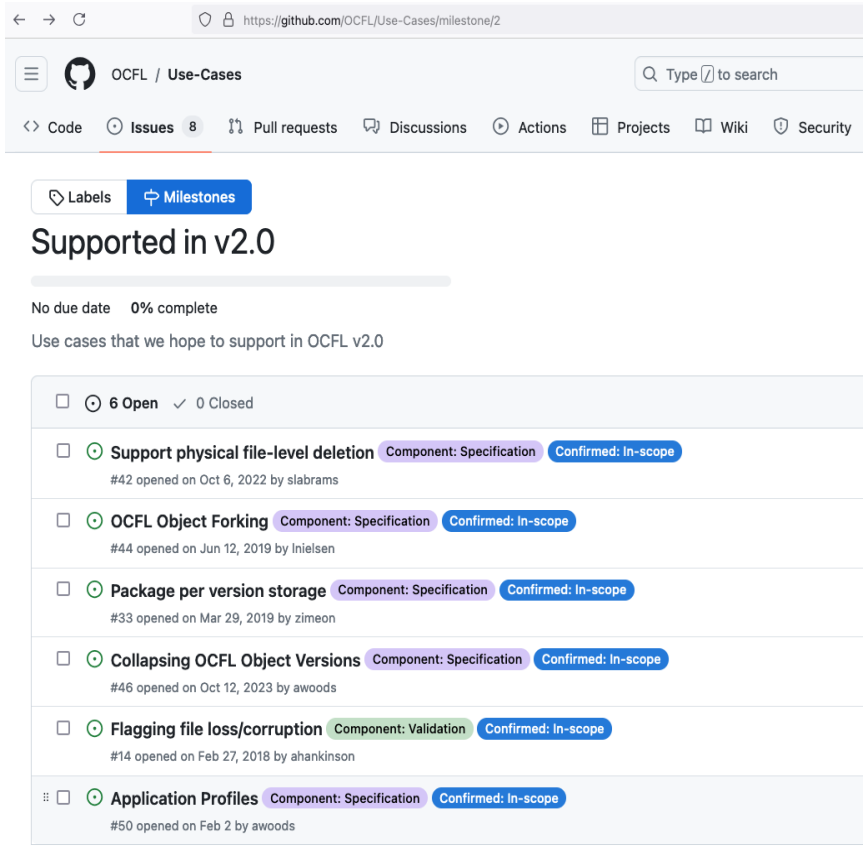
Benefits of OCFL: efficiency

Design decisions are made with a view to computational, bandwidth and storage efficiency

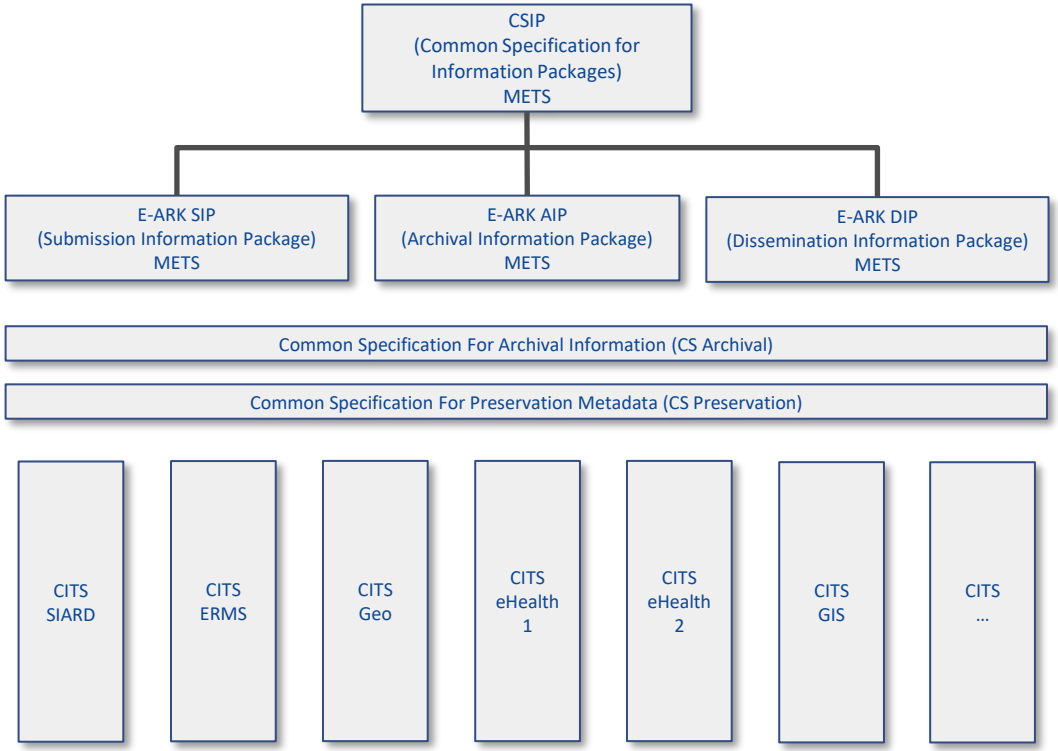
- Deduplication – between versions of an object, through content addressing
  - File level – simpler to access an object version and handles corruption well
  - Hard/soft links not a standard filesystem feature
- Forward delta deduplication of content lowers overall storage costs.
  - Also improves efficiency of object duplication
  - Optional since some infrastructures do this automatically
- JSON format for inventory.
  - Easy to parse and implement securely

OCFL v2.0 use cases

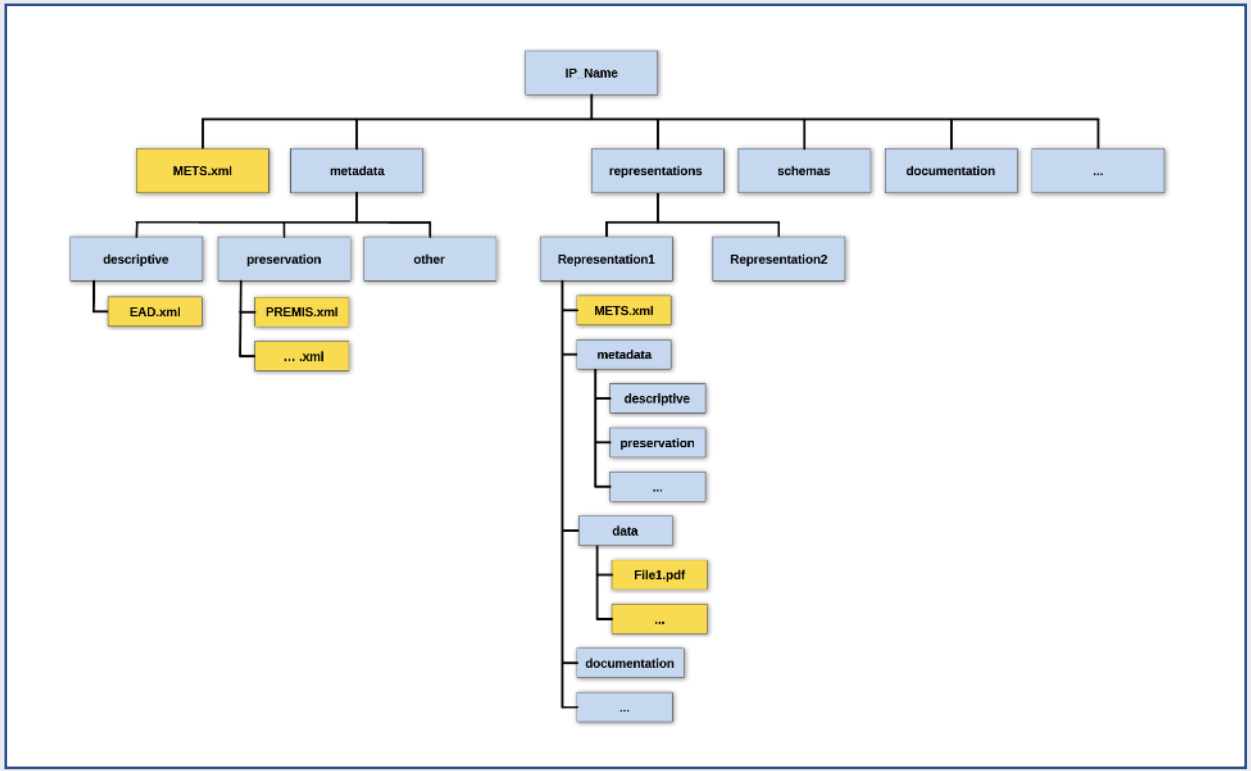
- Focusing on six use cases for v2
- Decisions based on community and editor support, feasibility, and compatibility with OCFL premises
- YOUR input welcome
- Expect a draft in the next year



E-ARK Common Specification for Information Packages



E-ARK CSIP – example with files



- IP level
  - Structural metadata: METS.xml
  - Descriptive metadata: EAD.xml
  - Preservation metadata: PREMIS.xml
- Representation level
  - Structural metadata: METS.xml
  - Content files: e.g. File1.pdf

Use Cases

- Incremental updates in dynamic collections:
  - In scenarios where collections are actively updated (e.g., records management systems or scientific datasets).
- Recording changes and creating lightweight delta AIPs:
  - Preservation actions often involve changes to metadata (e.g., reformatting, update of descriptions) without altering content.
- Enhanced traceability and provenance:
  - Traceability on file/storage level.
- Support for fixity verification and recovery:
  - Additional fixity layer.
- Optimised synchronisation and replication:
  - Delta AIPs can be transmitted more efficiently across systems and geographic locations.

A set of files to be archived

Original submission

file1.odt

file2.odt

file3.odt

PDF Migration on Ingest

file1.pdf

file2. pdf

file3. pdf

Policy: PDF-normalisation

PDF/A Migration

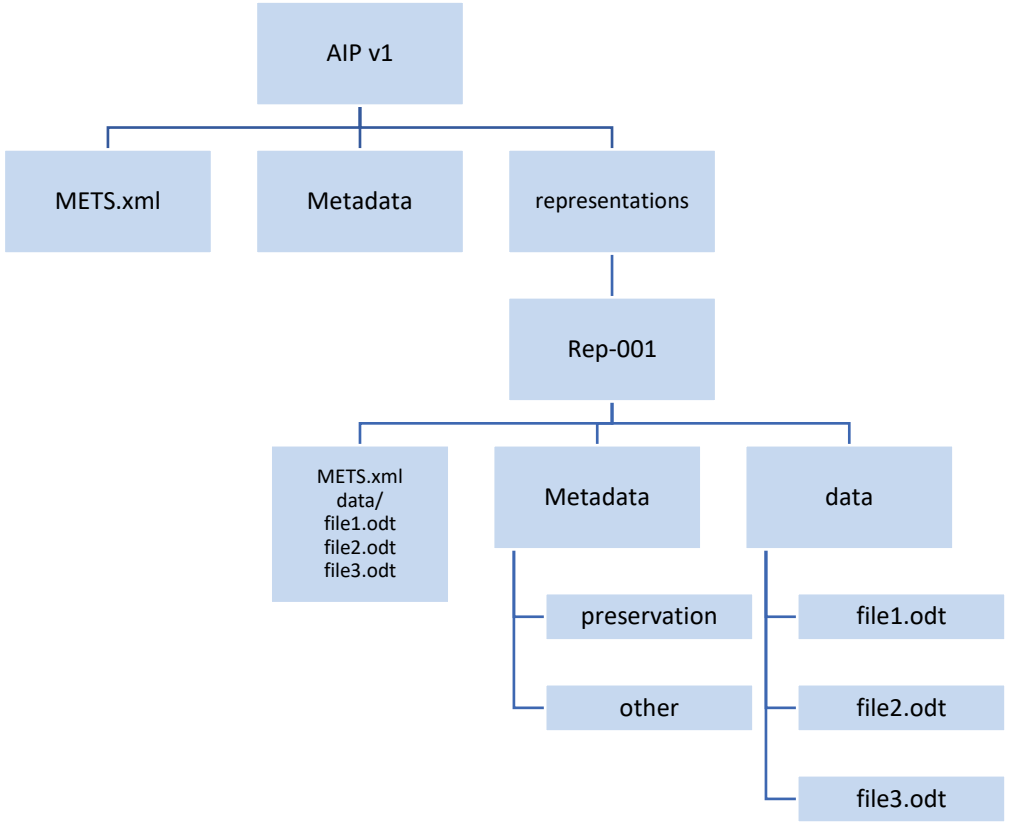
file1.pdfa

file2. pdfa

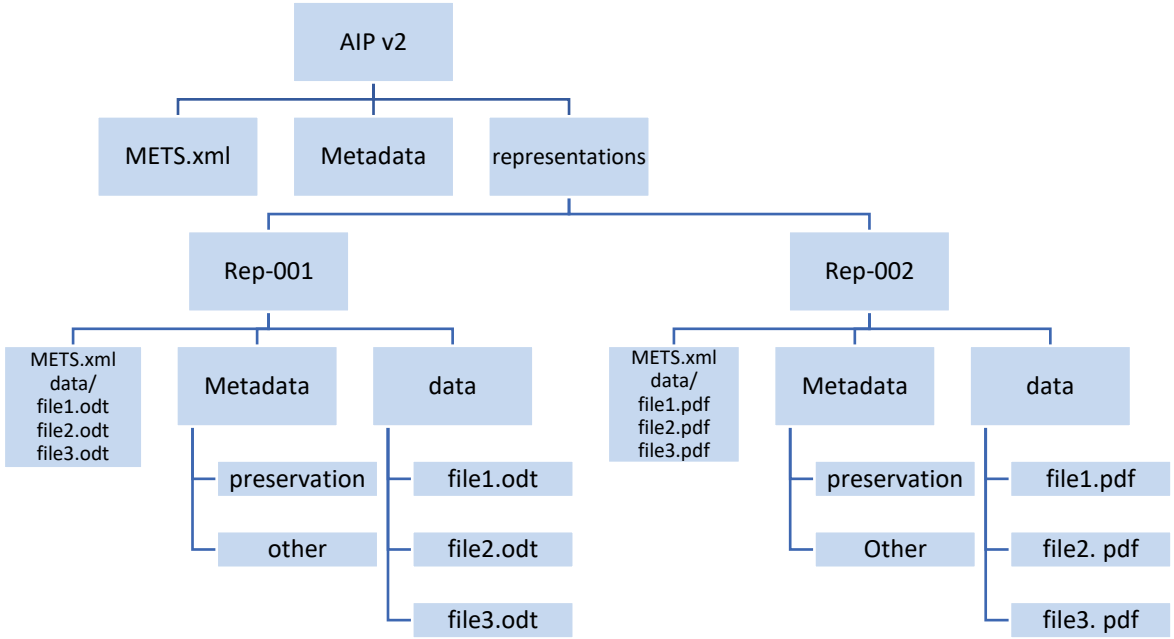
file3. pdfa

Policy: PDF/A

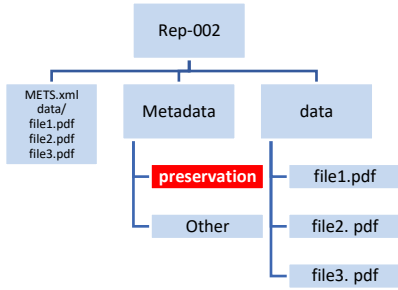
AIP: original submission



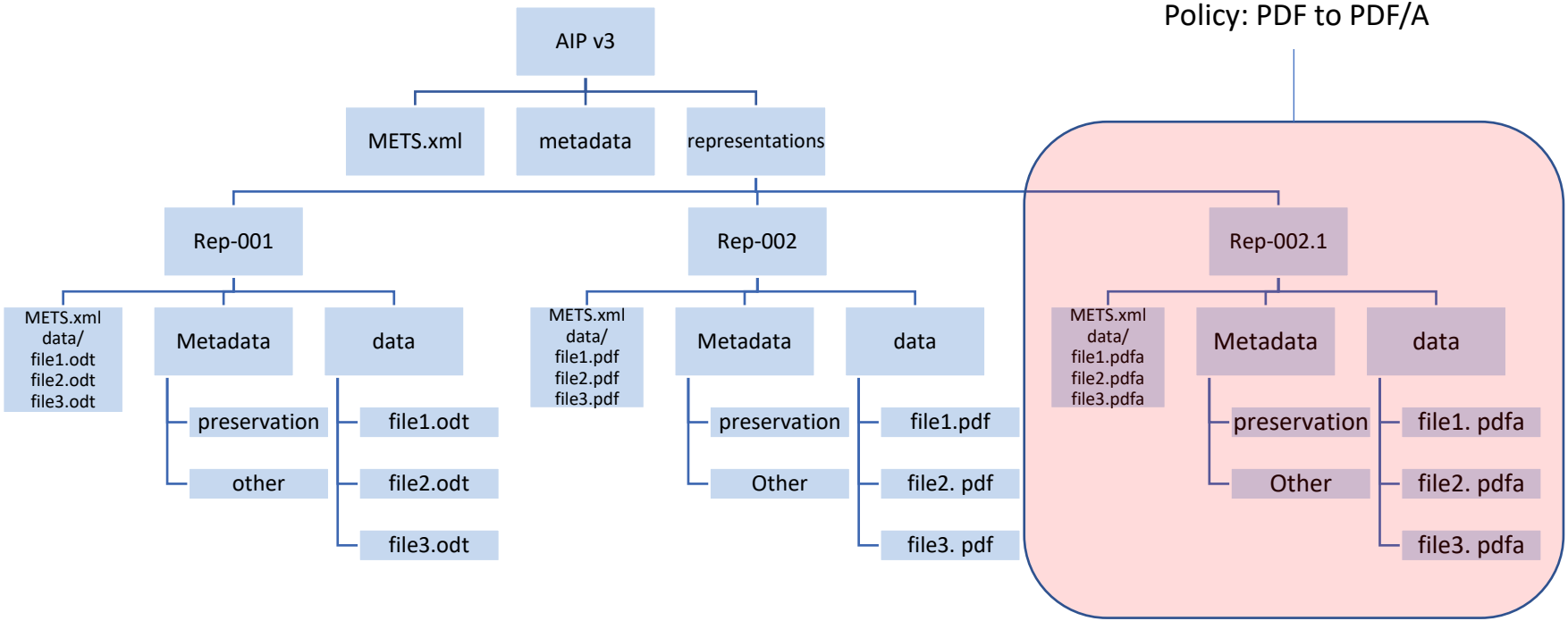
AIP: format migration due to normalisation policy



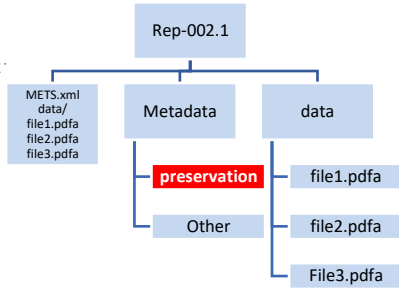
```
<premis:event xmlns:premis="http://www.loc.gov/premis/v3">
    <premis:eventIdentifier>
        <premis:eventIdentifierType>local</premis:eventIdentifierType>
        <premis:eventIdentifierValue>event-20240715-001</premis:eventIdentifierValue>
    </premis:eventIdentifier>
    <premis:eventType>migration</premis:eventType>
    <premis:eventDateTime>2024-07-15T10:30:00Z</premis:eventDateTime>
    <premis:eventDetail>Migration of Office document formats (DOC, DOCX, ODT) to PDF for long-term preservation.</premis:eventDetail>
    <premis:eventOutcomeInformation>
        <premis:eventOutcome>Successful</premis:eventOutcome>
        <premis:eventOutcomeDetail>
            <premis:eventOutcomeDetailNote>Migration completed successfully.</premis:eventOutcomeDetailNote>
        </premis:eventOutcomeDetail>
    </premis:eventOutcomeInformation>
    <premis:eventPurpose>
        Migration in accordance with institutional digital preservation policy mandating standardization to PDF for textual documents.
    </premis:eventPurpose>
    <premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifierType>local</premis:linkingAgentIdentifierType>
        <premis:linkingAgentIdentifierValue>policy-policy-2024</premis:linkingAgentIdentifierValue>
        <premis:linkingAgentRole>implementer</premis:linkingAgentRole>
    </premis:linkingAgentIdentifier>
    <premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifierType>software</premis:linkingAgentIdentifierType>
        <premis:linkingAgentIdentifierValue>LibreOffice-7.6</premis:linkingAgentIdentifierValue>
        <premis:linkingAgentRole>converter</premis:linkingAgentRole>
    </premis:linkingAgentIdentifier>
    <premis:linkingObjectIdentifier>
        <premis:linkingObjectIdentifierType>URN</premis:linkingObjectIdentifierType>
        <premis:linkingObjectIdentifierValue>urn:uuid:3f2c0d92-748d-4e6a-9c1e-91c58f1e2a97/Rep-002</premis:linkingObjectIdentifierValue>
    </premis:linkingObjectIdentifier>
</premis:event>
```



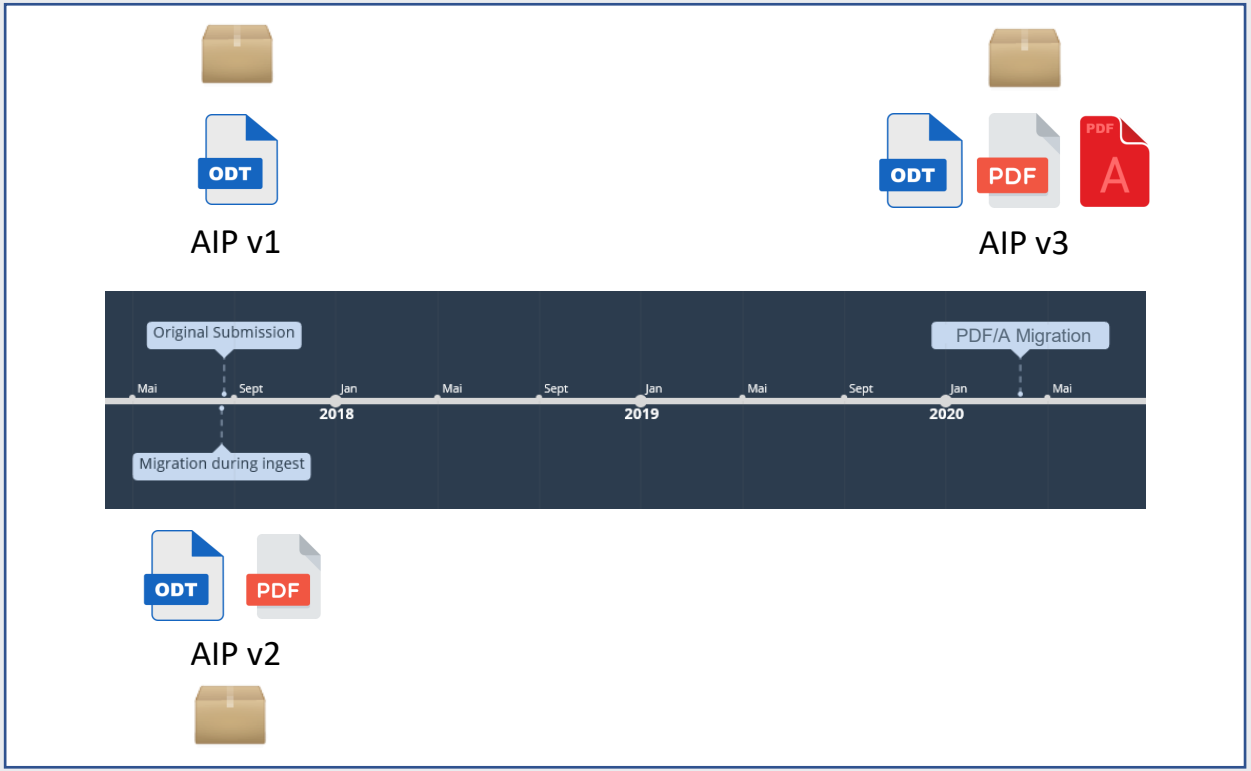
AIP: format migration due to policy decision



```
<premis:event xmlns:premis="http://www.loc.gov/premis/v3">
    <premis:eventIdentifier>
        <premis:eventIdentifierType>local</premis:eventIdentifierType>
        <premis:eventIdentifierValue>event-20250715-001</premis:eventIdentifierValue>
    </premis:eventIdentifier>
    <premis:eventType>migration</premis:eventType>
    <premis:eventDateTime>2025-07-15T10:30:00Z</premis:eventDateTime>
    <premis:eventDetail>Migration of PDF files to PDF/A-2b to meet archival standards and ensure long-term accessibility.</premis:eventDetail>
    <premis:eventOutcomeInformation>
        <premis:eventOutcome>Successful</premis:eventOutcome>
        <premis:eventOutcomeDetail>
            <premis:eventOutcomeDetailNote>Migration verified using veraPDF; all files compliant with PDF/A-2b.</premis:eventOutcomeDetailNote>
        </premis:eventOutcomeDetail>
    </premis:eventOutcomeInformation>
    <premis:eventPurpose>
        Compliance with institutional digital preservation policy requiring all PDFs be normalized to PDF/A for long-term archival storage.
    </premis:eventPurpose>
    <premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifierType>local</premis:linkingAgentIdentifierType>
        <premis:linkingAgentIdentifierValue>preservation-policy-2025</premis:linkingAgentIdentifierValue>
        <premis:linkingAgentRole>implementer</premis:linkingAgentRole>
    </premis:linkingAgentIdentifier>
    <premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifierType>software</premis:linkingAgentIdentifierType>
        <premis:linkingAgentIdentifierValue>Ghostscript-10.02</premis:linkingAgentIdentifierValue>
        <premis:linkingAgentRole>converter</premis:linkingAgentRole>
    </premis:linkingAgentIdentifier>
    <premis:linkingAgentIdentifier>
        <premis:linkingAgentIdentifierType>software</premis:linkingAgentIdentifierType>
        <premis:linkingAgentIdentifierValue>veraPDF-1.24</premis:linkingAgentIdentifierValue>
        <premis:linkingAgentRole>validator</premis:linkingAgentRole>
    </premis:linkingAgentIdentifier>
    <premis:linkingObjectIdentifier>
        <premis:linkingObjectIdentifierType>URN</premis:linkingObjectIdentifierType>
        <premis:linkingObjectIdentifierValue>urn:uuid:3f2c0d92-748d-4e6a-9c1e-91c58f1e2a97/Rep-002.1</premis:linkingObjectIdent
    </premis:linkingObjectIdentifier>
</premis:event>
```



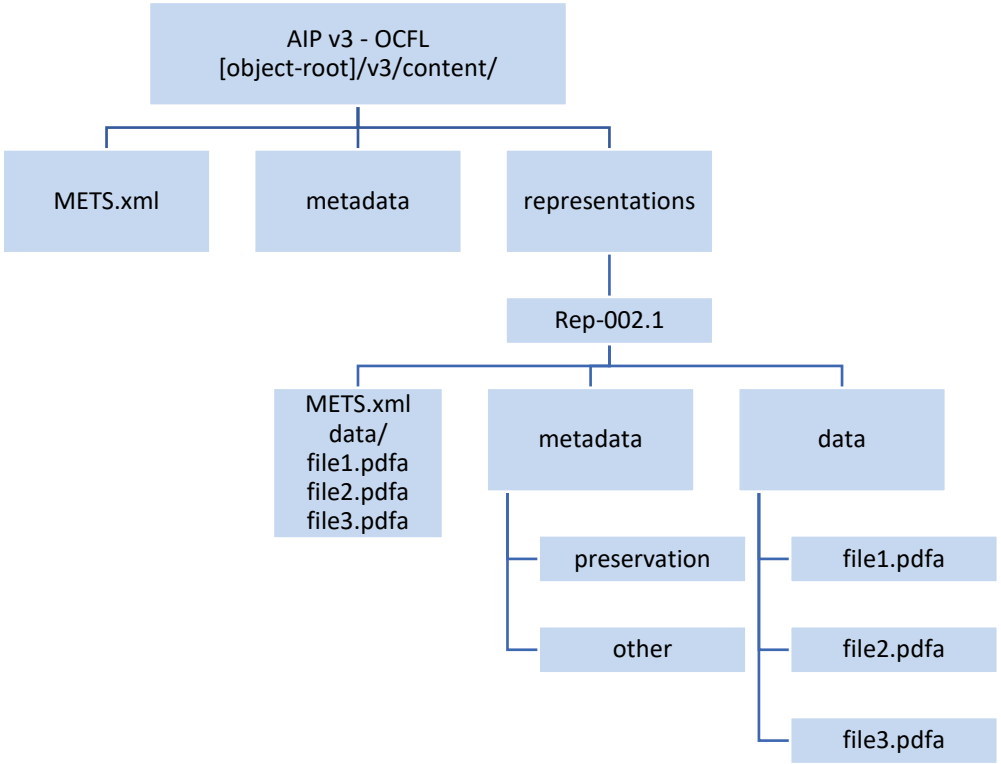
AIP – IP lifecycle / AIP versions

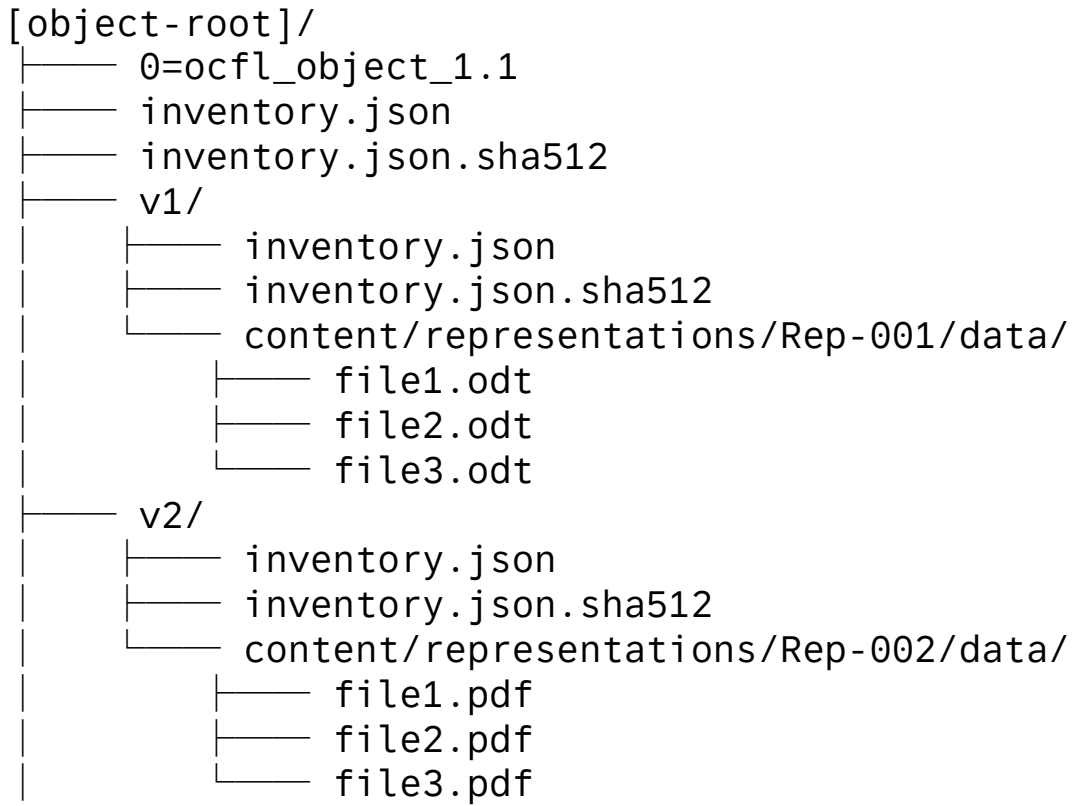


- How to manage versions over the lifetime of the AIP?
- Packaging complete information packages for each version would leads to redundancy
- Problematic especially for archives with large files (databases, videos, etc.)

file cons created by Dimitry Miroliubov - Flaticon

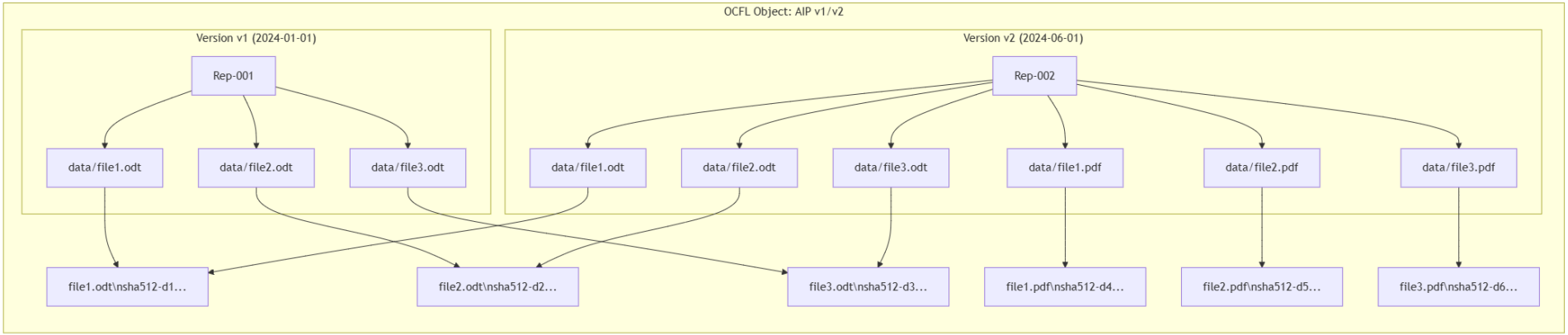
AIP – Delta-Version of the AIP (OCFL)




[object-root]/
0=ocfl_object_1.1
inventory.json
inventory.json.sha512
v1/
inventory.json
inventory.json.sha512
content/representations/Rep-001/data/
file1.odt
file2.odt
file3.odt
v2/
inventory.json
inventory.json.sha512
content/representations/Rep-002/data/
file1.pdf
file2.pdf
file3.pdf

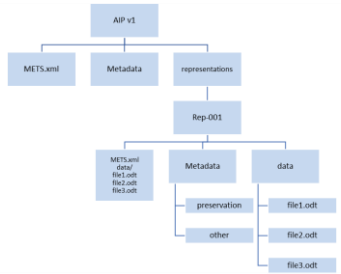
```
{
  "id": "example-object",
  "type": "https://ocfl.io/1.1/spec/#inventory",
  "digestAlgorithm": "sha512",
  "manifest": {
    "sha512-d1...": ["v1/content/representations/Rep-001/data/file1.odt"],
    "sha512-d2...": ["v1/content/representations/Rep-001/data/file2.odt"],
    "sha512-d3...": ["v1/content/representations/Rep-001/data/file3.odt"],
    "sha512-d4...": ["v2/content/representations/Rep-001/data/file1.pdf"],
    "sha512-d5...": ["v2/content/representations/Rep-001/data/file2.pdf"],
    "sha512-d6...": ["v2/content/representations/Rep-001/data/file3.pdf"]
  },
  "versions": {
    "v1": {
      "created": "2024-01-01T00:00:00Z",
      "message": „Original submission",
      "state": {
        "sha512-d1...": ["representations/Rep-001/data/file1.odt"],
        "sha512-d2...": ["representations/Rep-001/data/file2.odt"],
        "sha512-d3...": ["representations/Rep-001/data/file3.odt"]
      }
    },
    "v2": {
      "created": "2024-06-01T00:00:00Z",
      "message": „PDF representation created during ingest",
      "state": {
        "sha512-d4...": ["representations/Rep-002/data/file1.pdf"],
        "sha512-d5...": ["representations/Rep-002/data/file2.pdf"],
        "sha512-d6...": ["representations/Rep-002/data/file3.pdf"]
      }
    }
  }
}
```

Simplified logical path mapping from inventory.json

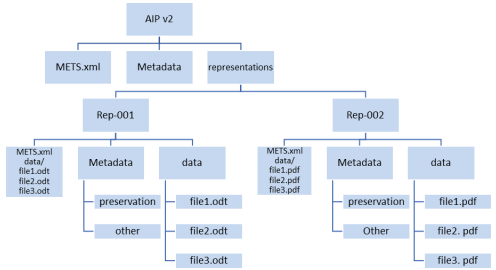


Serialisation: consolidated AIP versions

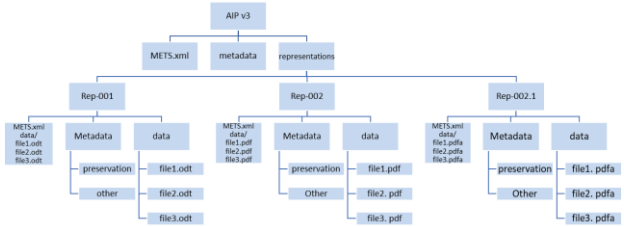
v1



v2

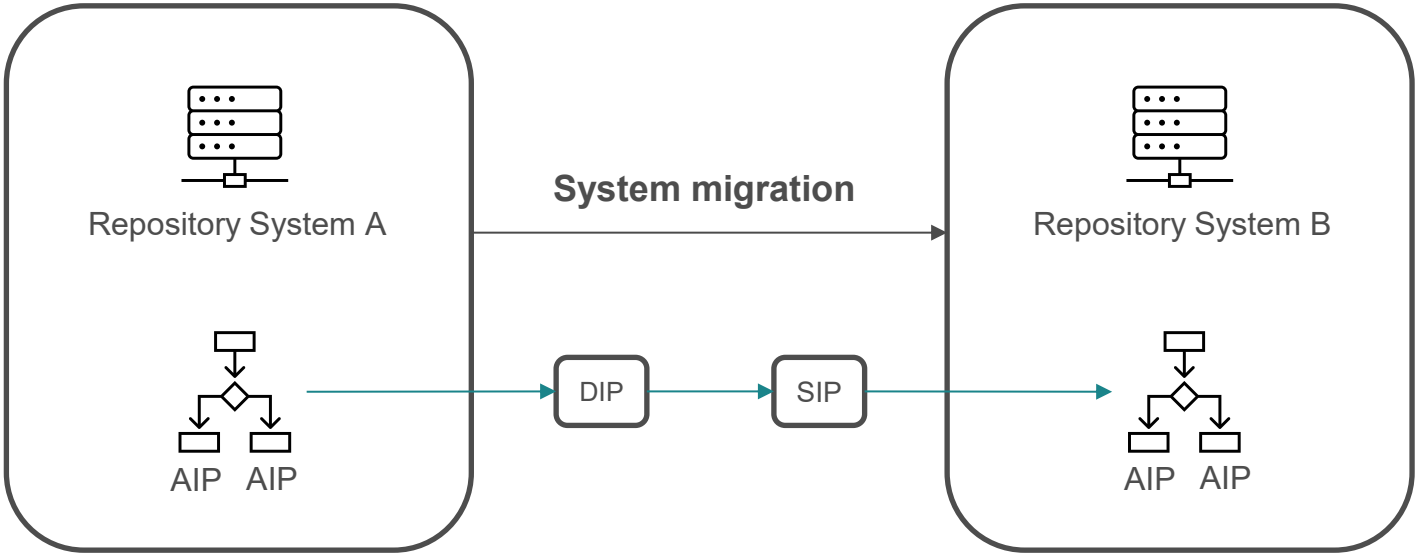


v3



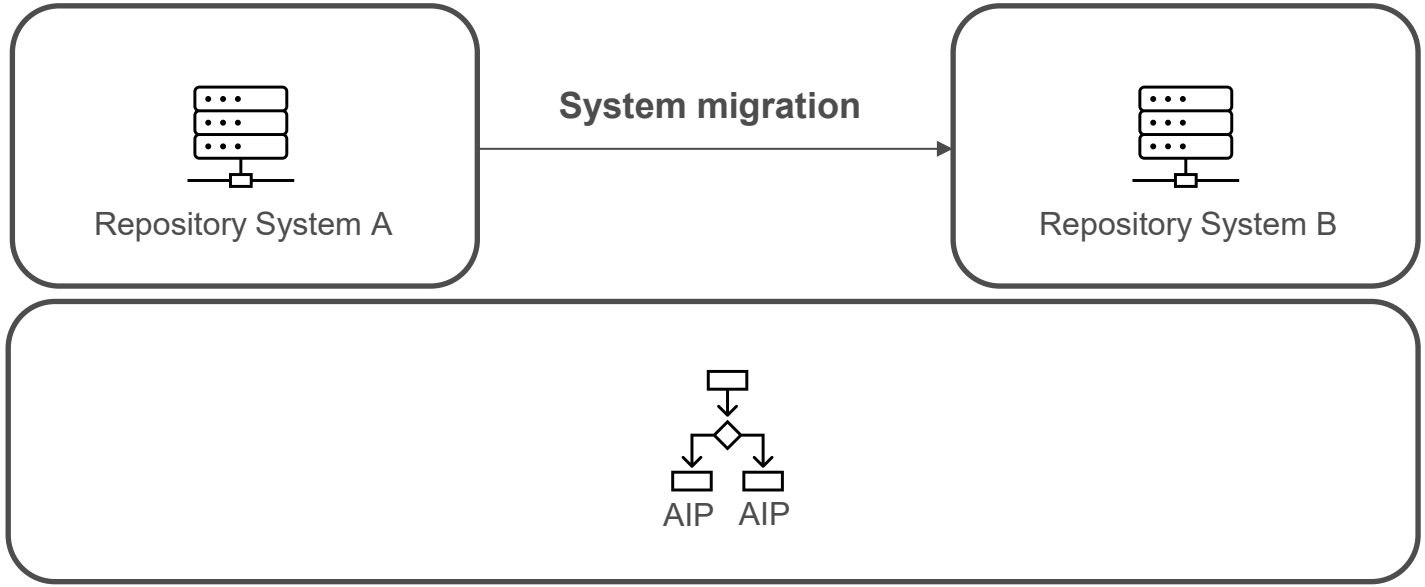
- Any version can be serialised based on the OCFL object
- Packaged containers can be created
  - for long-term storage on tape drives, for example
- Access/delivery can be provisioned based on OCFL storage

System migration – export/ingest



A stored AIP from the source repository is transformed into a DIP which is sent to the receiving repository where it is treated as, or transformed into, a SIP and ingested.

System migration – read from storage



Remember that the amount of data is growing at a fast pace!

Remember that moving data without need is creating an unnecessary digital preservation risk!

Conclusions

- Remember that the amount of data is growing at a fast pace!
  - Strategies for efficient storage and transfer are needed!
- Consolidated AIP means serialising all content and metadata
  - e.g., in BagIt or ZIP form, into a single package.
  - Ensures completeness and portability – but it can also result in large and rapidly growing package sizes.
- Consolidated AIPs are good for transfer, but be aware that they:
  - can be storage-intensive (redundancy).
  - cause load and bottlenecks for ingest pipelines.
  - scale poorly without deduplication strategies.
- Avoid unnecessary data movement because it increases preservation risk!

Thank you

Contact

https://www.e-ark-foundation.eu

support@e-ark-foundation.eu

https://www.linkedin.com/groups/8343650

https://bsky.app/profile/eu-earchiving.bsky.social

https://www.youtube.com/@e-ark