GUIDELINES FOR THE PRESERVATION OF DIGITAL AUDIO-VISUAL ASSETS IN THE CLOUD

ETC Adaptive Production Archive Working Group

Andrea Kalas, Seth Levenson (Chairs)

October 3, 2019

Published by
The Entertainment Technology Center at The University of Southern California

Table of Contents

Revisions

Version	Status	Notes
v01	DRAFT	Initial draft.
v02	DRAFT	Revisions following ETC meeting on Thursday 28th February 2019, at which the initial draft and next steps were discussed. Expanded discussion of the term "digest", with synonyms and examples. Addition of a key to the data model diagram. Examples added, for clarification. Addition of notes relating to the C4 digest solution. Addition of a new section presenting initial cloud-related topics for further review and discussion. Other minor clarifications and corrections.
v03	DRAFT	Updates made following discussions with additional ETC committee participants. Notes in the Cloud section have been expanded.
v04	DRAFT	Updates based on further discussions with additional ETC committee participants. Addition of cloud-based process flows.
v05	DRAFT	Clarify the structure of the document (current state vs. cloud considerations). Augment the cloud solutions section, based on the ETC meeting on Wednesday 8th May 2019. Other miscellaneous updates and corrections.
v06	DRAFT	Re-work introduction to emphasize core focus on fixity. Re-work cloud section to remove background notes, and to add options chart, with narrative. Add "future opportunities" section at end.
v07	FINAL FOR REVIEW	Added executive summary.
v08	FINAL	Incorporated comments from final review.
v09	FINAL	Incorporated note on media migration.

Executive Summary

This paper's purpose is to help entertainment companies and others who have the responsibility to preserve valuable moving image assets and are considering cloud as part of this process. It focuses on the foundational issue of "Fixity," which is at the heart of digital preservation as it is a widely used process to prevent digital asset data loss. Ideally this document can be used in User Requirements and Request for Proposal documents. This document came out of a working group regarding cloud technology and archiving in the entertainment industry sponsored by the Entertainment Technology Center at the University of Southern California The group acknowledged at many times throughout that the issue of archiving of digital moving images is a very complex and multi-faceted project and the issue this document focuses on is one which is critical because it speaks to an essential requirement of digital preservation and the challenge that cloud environments present to that requirement. The group met for six months to discuss the issue. Targeted phone calls were also conducted with entertainment companies and cloud vendors to ensure as much participation and representation as possible. Andrea Kalas and Seth Levenson co-chaired the group.

The following ETC Member Organizations Participated:

- Disney
- Fox
- Iron Mountain Entertainment Services
- Microsoft Azure
- NBC Universal
- Paramount Pictures
- Sony
- Technicolor
- USC
- Warner Bros
- Western Digital

The following non-member organizations were consulted:

- Amazon Web Services
- Google Cloud
- MovieLabs
- Walden Pond
- Wasabi

Part One: Introduction

This document is divided into the following sections:

1. Introduction.
2. The "Current State" of Fixity. This section introduces fixity-related key concepts and terminology. It assumes no integration with cloud storage solutions, or only minimal integration with no special considerations given for cloud storage best practices.
3. Cloud Considerations. This section discusses the impact of cloud storage solutions on fixity processes.
4. Appendices - for additional background information and further reading.

Purpose and Scope

The audience for this document is any organization which needs to manage and preserve digital audio-visual assets, including the use of cloud-based storage systems such as Microsoft Azure, AWS, GCP and others. The associated metadata for preservation assets must also be managed and preserved - however this document focuses primarily on the assets themselves.

Asset preservation may cover many different steps and processes. This document focuses on one of those processes: *fixity* - how to show that a given preservation file has not changed over time. However, for fixity checking to be effective, it must be part of a larger archival and preservation context. This document will discuss that wider context by presenting a high-level process flow, and then drilling down into some of the options which might be considered when implementing a preservation solution.

Although a wider asset preservation context is referenced, this document focuses on the core process of fixity. This is because fixity represents the fundamental activity for bit-loss detection in preservation assets, and because that process can also be central to providing a mechanism for asset identification.

This document reflects a series of discussions between preservation/archive representatives from major Hollywood studios, and representatives from major cloud service providers.

Out of Scope

Some aspects which may be part of a comprehensive archival and preservation strategy are out of scope from this document, including the following:

- Provenance: tracking the physical journey of a digital asset from its point of creation through to ingest into a digital archive, as a guide to ensuring the asset's authenticity.
- Rights: managing rights under law to own, copy and store a particular asset.
- Security: ensuring sufficient protections are in place to control access to an asset, and to avoid theft or destruction of the asset.
- Longevity: ensuring that a preservation asset can continue to be accessed over the course of many years, even if the original tools for creating and/or reading the asset no longer exists.

Part Two: The Current State of Fixity

This section introduces fixity-related key concepts and terminology. It assumes no integration with cloud storage solutions, or only minimal integration with no special considerations given for cloud storage best practices.

Key Concepts

Asset: A coherent set of content representing a specific instance of a movie, TV show, or similar artifact (trailer, artwork, stills, soundtrack, etc.). An asset may be a single file; or it may be a **complex asset** consisting of multiple files and related materials (for example IMF - the interoperable mastering format[1]). In the context of digital preservation, it typically consists of the most detailed, highest resolution set of data - which, if lost, could not be replaced or reconstituted from other sources.

Preservation Registry: A tool for the in-house management of digital preservation assets. It may optionally store the assets themselves, but it must contain the relevant **preservation metadata**, needed to manage the preservation process. Such data might consist of file names and locations; groupings of files into assets; an audit trail of fixity checks; and so on. May be part of a Media Asset Management (MAM) application.

Digest Repository: A tool for the in-house management of preservation asset digests and fixity checks. May be a part of the Preservation Registry – but is called out here as a separate tool due to its central role in the fixity checking process.

Archive Location: Separate from the preservation registry, there will be one or more archive locations, where clones of preservation files can be stored. Such clones are assumed to be byte-for-byte copies of the original files, to support fixity checking.

Fixity: The property of a digital file that indicates its contents have not changed over time. This typically means that none of the bytes in a file have changed (for example, due to a failure of the storage medium, bit rot, or tampering). The term is also used when referring to **fixity checking** or **health checking** - the process of determining whether a file's contents have changed over time. The term can also be applied to an asset (e.g. a movie) which may be a collection of multiple files.

Digest (or message digest): A sequence of characters created by a hashing algorithm operating on the contents of a digital file[2]. A digest may also be referred to as a **hash**, or **checksum**. Used to implement fixity checking.

What to Preserve?

This document assumes that preservation efforts will focus on assets which are original, unique, or expensive (or potentially impossible) to recreate. The document assumes that metadata relating to such

[1] For an overview of IMF see the following presentation: https://www.smpte.org/sites/default/files/section-files/BBTB2017-W04%20Dominic%20Johnson%20-%20IMF.pdf.

[2] See Appendix A for a more detailed discussion of digests.

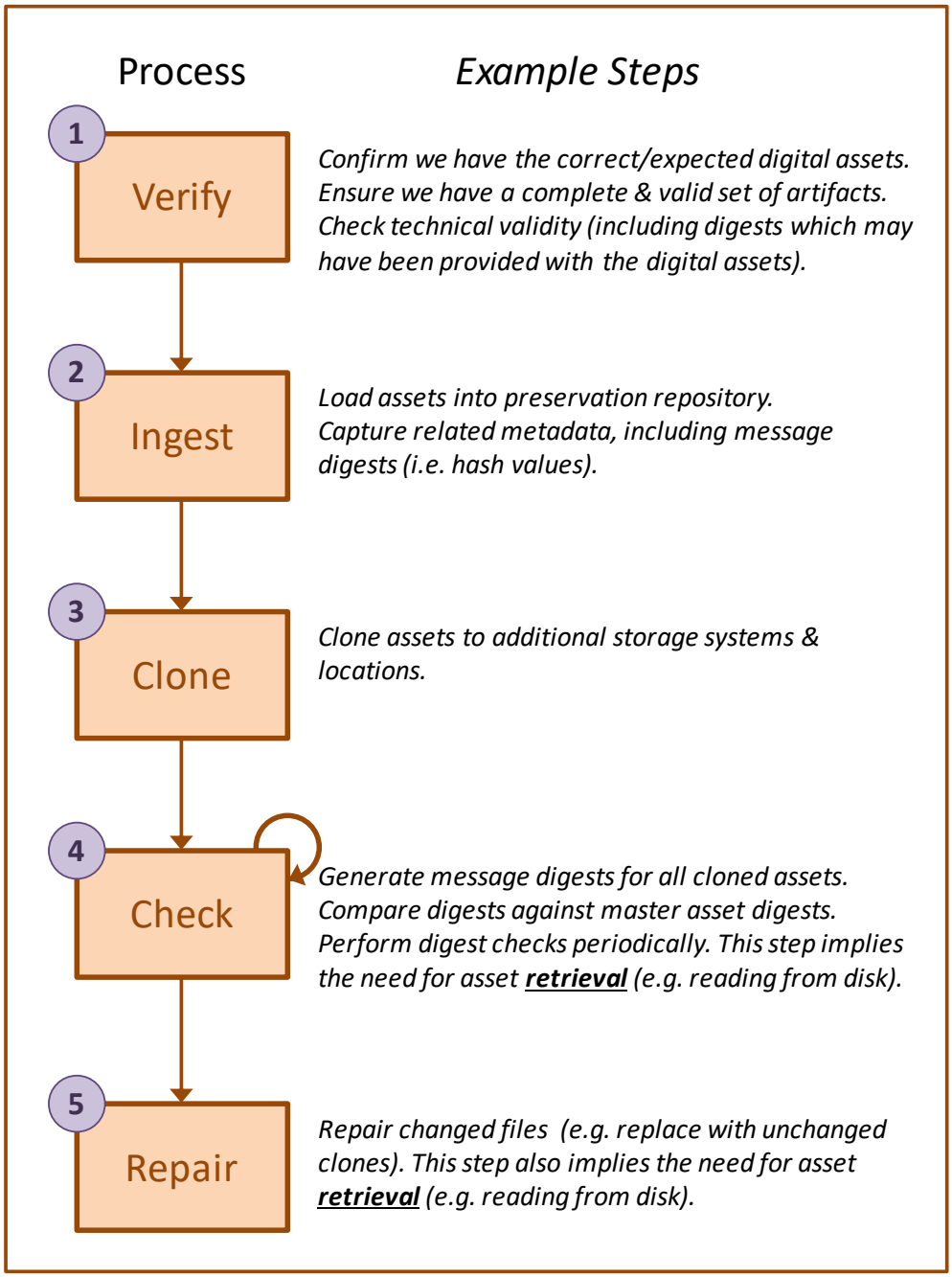
assets will also need to be preserved - however our focus is primarily on preservation of the assets themselves.

Assets which may not fall into any of the above preservation categories include those which are created from preservation assets, to be used for mastering and distribution purposes.

Any assets which are selected for preservation must be testable for fixity at any point in time. The results of fixity tests must be accessible, and unambiguous. Actions taken to resolve fixity failures must be automated, and captured for reporting purposes.

High Level Process Flow

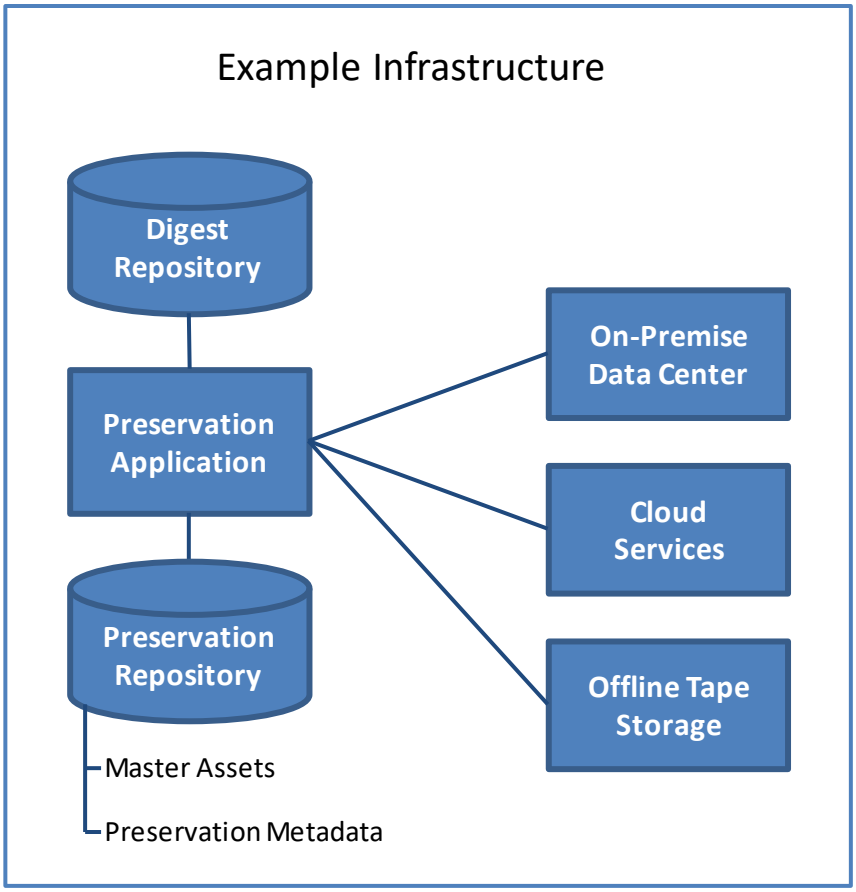
The following diagram shows one possible high level process flow for managing preservation assets.



The following sections describe the above steps in more detail. In any specific implementation, detailed steps might be performed in a different order, or combined into different groupings.

For simplicity, we make the following assumptions:

1. There is a single centralized preservation registry, and a related digest repository.
2. Each preservation asset is stored in the preservation registry.
3. All asset digests and related fixity check results are stored in the digest repository.
4. There are one or more additional, separate archive locations.
5. A clone of each preservation asset is stored in at least one of these archive locations (and ideally in more than one of these locations).



1. Verify

Before an asset is placed into a preservation archive, the business may need to perform some or all of the following activities:

- Decide whether the asset is a suitable candidate for preservation archiving. Not all assets may need to be placed under preservation control. Factors such as cultural significance and financial risk may play a part in this decision process.
- Decide what level of preservation is appropriate (assuming different preservation strategies for different levels of preservation criticality).
- Confirm that an asset is correctly and completely represented by its set of files and other artifacts.

- Verify that the file formats of each artifact are valid (e.g. conform to the file format) and readable.
- Check that the asset is in fact what it purports to be (e.g. the correct edit of the given movie).
- If a digest was provided with the asset, recalculate the digest to ensure it matches what was provided.

Some of the above steps are technical in nature and could be performed in an automated way (such as file format checking); others would require business processes with human intervention and manual curation.

2. Ingest

This involves loading assets into the preservation registry, and capturing the related metadata for the assets. See the data model for more details on candidate metadata items.

For new items being ingested into the preservation registry, a digest is generated and is stored in the repository as metadata, along with the file identifier, and details about the algorithm used to generate the digest (for example, the C4 digest presented in Appendix A).

For updates to existing files, a **versioning strategy** may be required: Does the new file simply replace its previous version; or should both versions be preserved (and the relationship between the versions captured as metadata)?

A process may also be required defining the permanent removal of assets from the preservation registry.

3. Clone

As noted above there may be one or more **preservation strategies** defined. A preservation strategy may specify attributes such as:

- How many clones should be created.
- Which different types of storage should be considered (on-premise; 3rd-party cloud; offline; disk; tape; etc.).
- How frequently a fixity check needs to be performed on each clone.

The preservation registry should support the definition and enforcement of preservation strategies.

For each new clone created, ensure that its digest matches that of the original/master asset.

4. Check

This is the **fixity check** activity which lies at the heart of the preservation process. For each clone of a file in each archive location, this involves generating a new digest, and comparing this digest to the corresponding digest generated by the previous fixity check for this specific file. Fixity checks are managed by the digest repository, and the results of fixity checks are stored in the digest repository.

Assuming the Verify, Ingest and Clone steps were performed correctly (and the relevant digests were verified), this step is sufficient to demonstrate fixity: A file, which was previously known to be valid, continues to generate the expected digest.

Each fixity check is executed as per the schedule defined in the relevant preservation strategy. Fixity checks should also be performed on the master preservation asset, stored in the preservation registry.

The digest repository records the results of all such checks.

5. Repair

If a fixity check fails (that is, if there is a difference between the current and previous digests), then the file has changed in some way and is no longer valid. In this case, the file needs to be repaired - typically by replacing it with a file which generates the correct digest.

The preservation registry should support and control the repair process.

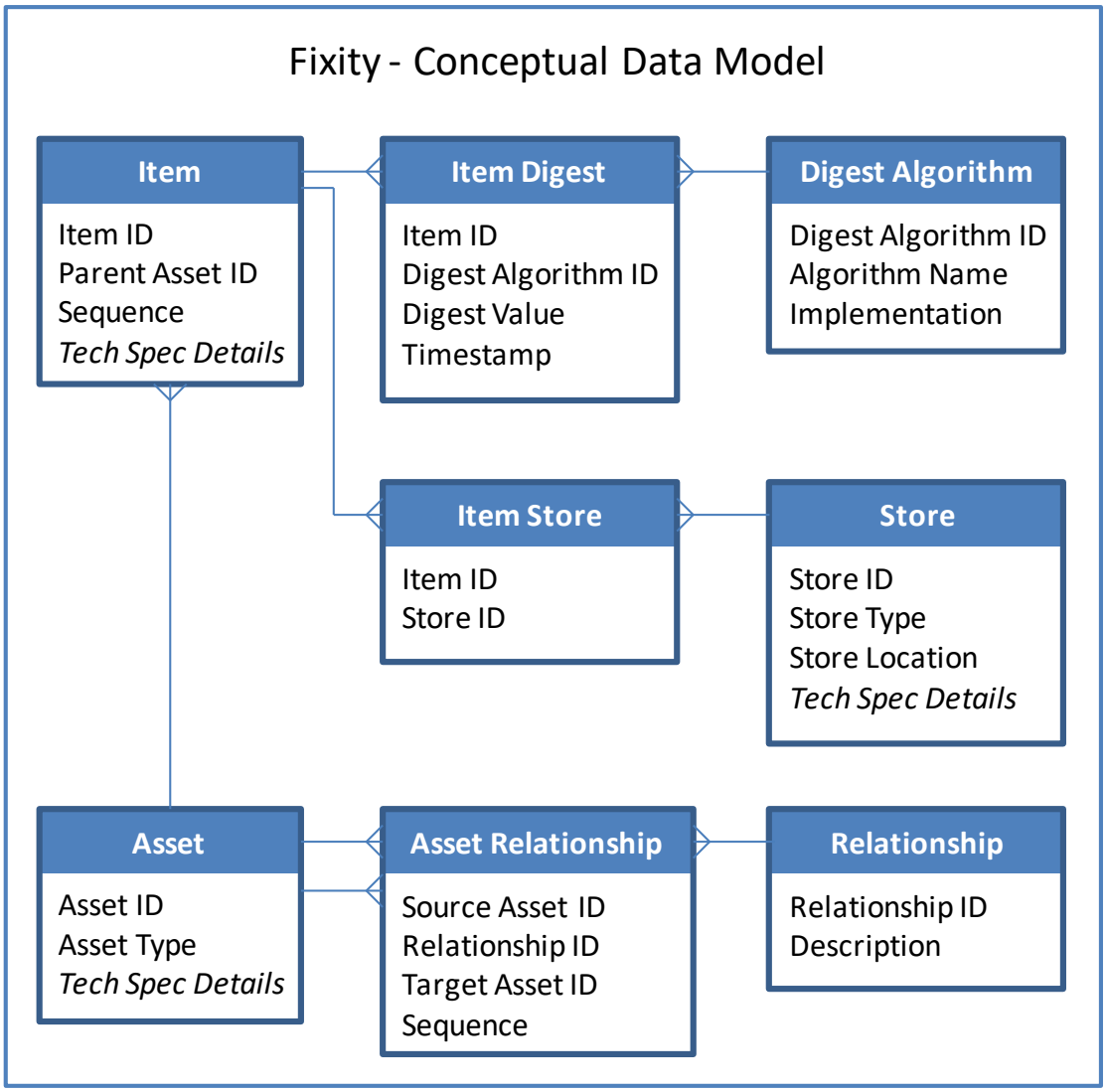
Policy Review

In parallel with the above high level process, a policy review should be performed. On a periodic basis, the chosen preservation strategies should be reviewed, including:

- Choice of archive locations: Are these still appropriate? Are there newer, more cost-effective options available?
- File formats: Are these still actively supported, deprecated, or no longer maintained?
- Preservation strategies: Do they need to be changed?

Conceptual Data Model

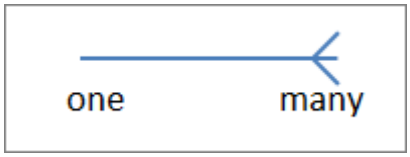
A high-level conceptual data model to support fixity is shown below:



Key to diagram:

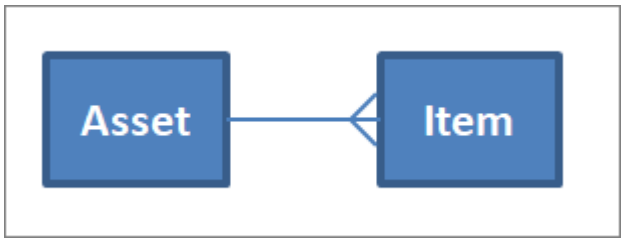
Boxes represent entities - see the table below for details.

Lines connecting boxes represent "one-to-many" relationships between entities:



Entity	Description
Item	A specific preservation file upon which a fixity test can be performed. Items may be sequenced - for example, the ordering of frames in a movie. Items will have additional technical specification details, describing the items in more detail.
Digest Algorithm	Identifies the specific algorithm used to generate a digest (this document assumes the use of C4 - see Appendix A - but any suitably robust digest may be used). Identification includes implementation details (such as the specific tool/platform used to execute the algorithm).
Item Digest	The result of computing a message digest for a specific item, using a specific algorithm. A history of all such digests can be captured.
Store	A specific location in which an item can be preserved, and where fixity checks can be performed. The location can be identified by geographic location, storage medium, and whatever additional technical specifications may be required.
Item Store	Identifies which items are preserved in which locations.
Asset	An asset contains one or more items, in order to group those items into logical collections. For example, an asset may represent reel one of a movie, and will contain all of the items in that reel.
Relationship	Assets can be related to other assets. Some relationships may be hierarchical - for example, an asset representing a movie may be related to three other assets, each representing a reel within the movie. This can be captured by the "*is a reel of*" relationship. Other non-hierarchical relationships can also be supported, as needed (e.g. "*is a composite of*", "*is a replacement for*").
Asset Relationship	Captures a specific relationship between two assets, using a "subject-predicate-object" construct, where the source ID is the subject, the relationship ID is the predicate, and the target ID is the object. In the above movie reel example, the relationship would be "*is a reel of*", and would be used to relate "reel" assets to the related "movie" asset. One such asset relationship would therefore be captured as: *Asset A is a reel of Asset B.*

Example:



Reading from left to right, a specific asset (e.g. a movie) is a collection of many items (e.g. files).

Reading from right to left, a specific item belongs to one asset[3].

In the main diagram, the Asset Relationships entity has two lines connecting it to the Asset entity because the Asset Relationships entity contains two separate Asset ID fields (source ID and target ID).

Tech Spec Details - Additional File, Asset, and Storage Metadata

The fixity conceptual data model diagram refers to "tech spec details" in various places (Item, Asset, and Store). These refer to implementation-specific metadata attributes which could vary greatly across different preservation applications. For example, one organization will likely have different file naming standards, different directory hierarchies, and so on, from another organization.

See "Appendix B - Asset Technical Specifications" for some specific examples of additional data which could be captured in a fixity data model.

The following points are noted:

- A digest is generated from the internal contents of a file and is not affected by external attributes such as the file name, file type, last updated timestamp, and so on. You can rename a file, without changing its digest.
- Organizations may need to track these types of external attribute in their preservation applications, as well as tracking digests.
- Organizations may also have other ways to identify assets, such as IDs generated by the organization, or assigned by third parties (for example EIDR IDs[4]).
- Digests can be used as the primary way to uniquely and unambiguously identify a given file, as well as being used to support fixity checking.
- A set of related digests can be fed into a hashing algorithm to create a new "digest of digests", to represent one asset (e.g. a TV show or movie) comprising a well-defined set of files. In this way, a hierarchy of digests can be created, mirroring the hierarchy of assets and items.

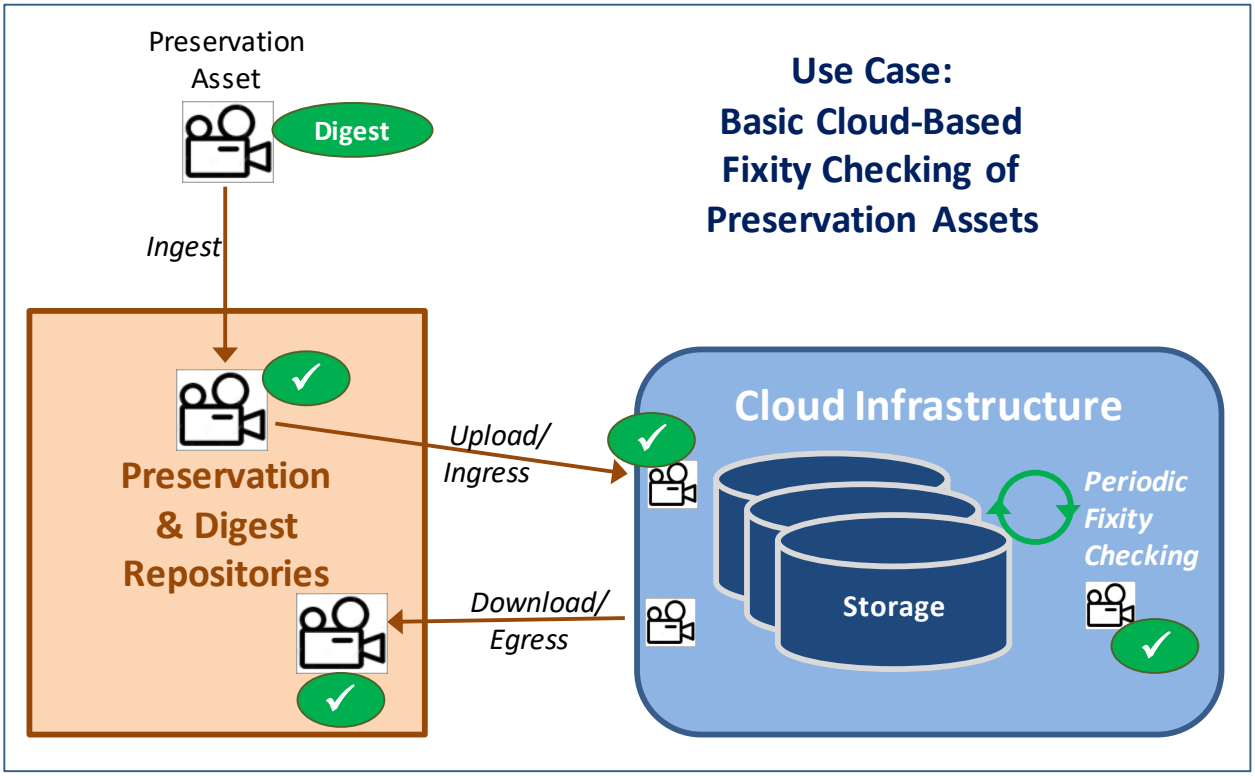
[3] The relationship symbols used in this document do not attempt to capture more detailed cardinality (one-to-one, one-to-many, etc.), or optionality ("must", "may") - for example, where an asset must consist of at least one item, may consist of zero or one item, and so on.

[4] See https://eidr.org/.

Part Three: Cloud Considerations

Cloud-Based Fixity Checking of Preservation Assets

The following diagram presents a basic scenario for performing fixity checking in the cloud. In this scenario, an asset originates on-premise, and is then transferred to a cloud infrastructure for long-term preservation storage. This will be used as a baseline scenario for discussing other cloud-based fixity options and variations.



Assumptions for this basic scenario:

a) The asset to be preserved is a single file, which does not yet exist in the preservation archive.

b) As part of the metadata associated with the asset, a digest (such as a C4 ID or similar) has been provided.

c) Prior to the start of this scenario, the asset has been checked to verify that it is the correct and complete asset, with no flaws, and with the correct digest value.

The steps in this scenario are:

1. The asset is ingested into the preservation registry. This step assumes the file is physically copied.

2. As part of the ingest process, the asset's digest is recalculated and compared to the original digest value, to ensure there have been no data changes.

3. A clone of the asset is uploaded to cloud storage, for long-term preservation. Note that this scenario makes no technical assumptions about how the clone is physically stored in the cloud, or what data preservation strategies are implemented by the cloud provider.

4. As part of the cloud upload process, the asset's digest is recalculated and compared to the original digest value once again, to ensure there have been no data changes.

5. Periodic fixity checks are then performed on the cloud-based asset. These may take one of two forms:

    a. A "traditional" re-calculation of the hash similar to the process performed when the asset was first uploaded to the cloud. Such a process could be controlled and performed by the asset's owners.

    b. Some other "non-traditional" process, as implemented by the cloud provider, as part of its proprietary solution for long-term data preservation.

6. At some point the cloud-based copy of the asset may need to be downloaded from its cloud storage, back to the on-premise preservation registry.

7. The downloaded asset's digest is recalculated and compared to the original digest value once again, to ensure there have been no data changes during the download.

The Fixity Gap

There is a gap between the "traditional" on-premise approach to demonstrating fixity (re-computation of digests), and potentially different approaches to file integrity management, as implemented by cloud vendors:

- In the cloud, a file may be held in deep storage, with relatively expensive access costs.
- In the cloud, a file may be split across different storage devices (and even geographical locations) in ways which may make traditional digests more cumbersome to compute.
- Cloud vendors may be checking and repairing data at a higher frequency than performed by traditional preservation processes (using proprietary technologies) – but those checks may be hidden from asset owners and may not be meaningful to asset owners.
- A brute-force approach to fixity (i.e. the constant re-computation of large volumes of digests) may be perfectly feasible for on-premise storage solutions but may incur significant cost penalties when translated to cloud infrastructure.
- How can cloud vendors demonstrate that their data preservation solutions provide the same level of robustness and auditability as "traditional" fixity checking?

- Is there a way to provide inter-operability between digest-based fixity and cloud-based data durability solutions?

Bridging the Gap

The "basic cloud-based fixity checking" scenario presented above represents a relatively straightforward but somewhat naïve approach to adopting fixity in the cloud: it follows the same process as for traditional (on-premise) fixity checking.

The following grid presents this scenario, followed by other approaches which may be more adapted to cloud-based preservation, and which attempt to bridge the gap between current preservation approaches and future cloud-based processes.

No.	Approach	Description	Implications
1	Traditional Fixity	Archive team is responsible for periodically generating and checking all digests for cloud-stored assets. Archive team is responsible for tracking results and performing any repairs which may be needed.	Requires repeated storage access, just to support digest calculation. May limit the ability to leverage long-term storage options provided by cloud vendors. May require custom development to support spinning up *ad hoc* compute resources for hash generation. Potentially large cost implications.
2	In-Production Archiving	Archiving is one part of a fully integrated workflow for production assets. IDs (such as C4) are ubiquitously used to co-ordinate workflows.	Can provide opportunities for more efficient/economical fixity checking. May not provide full fixity coverage for long-term preservation assets.
3	Custom Cloud Solutions	Cloud provider develops solution for a client's choice of digest, frequency of fixity etc.	Cloud fixity achieved at potentially reduced cost. Point-solutions, rather than industry-wide best practices.
4	Object-Level Data Integrity Services	Cloud vendors provide new industry-agnostic services/APIs.	Potential for demonstrable and trusted reporting of long-term asset integrity.

Traditional Fixity

As described above, this approach involves re-implementing the same on-premise fixity process, but targeted at cloud-stored assets.

It implies frequent and repeated reading of files from cloud storage, with the costs associated with doing so.

Such costs may be prohibitively high.

In-Production Archiving

As noted in the above summary table, this is not a preservation fixity solution *per se*. Rather, it is a way to exploit production workflows for the benefit of preservation activities.

For example, if a digest such as a C4 value is used consistently throughout the production process, then this identifier may support in-flight fixity checking for cloud-based assets which are already being used for some other production activity.

The longer-term checking of preservation assets may not be covered in this scenario.

Object-Level Data Integrity Services

This scenario involves cloud vendors providing a potentially new or augmented set of services. Such services would not be targeted specifically at the entertainment industry. Instead they could be of potential value to any organization needing highly trustworthy and demonstrable proof of long-term data integrity. This could include government agencies, museums, financial institutions, and so on.

In this context, an object is assumed to be one file representing one preservation asset. For clients with complex assets represented by multiple files, mapping from one complex asset to multiple objects (and verifying the reporting per object) would also be required.

Future Opportunities

This document deliberately focuses on the core activity of fixity checking: the fundamental activity for bit-loss detection in preservation assets. However, there are other areas which are critical to ensuring a robust infrastructure capable of supporting long-term asset preservation. Some key areas are noted below:

- **Provenance**: Tracking the physical journey of a digital asset from its point of creation through to ingest into a digital archive, as a guide to ensuring the asset's authenticity.
- **Rights**: How preservation strategies are aligned with intellectual property issues including: Duty of Care, Fixity in support of verifiable IP, Access and Security.
- **Security**: Ensuring sufficient protections are in place to control access to an asset, and to avoid theft or destruction of the asset; also how encryption and fixity interact.
- **Longevity**: Ensuring that a preservation asset can continue to be accessed over the course of many years, even if the original tools for creating and/or reading the asset no longer exists.
    - Immutable physical copies of digital assets may provide alternative ways to achieve longer-term preservation.
    - Software emulation may allow for non-ubiquitous file formats.
- **Migration**: Media migration is a standard part of data infrastructures. How does it relate to ongoing data integrity? What models for automation can media migration and data integrity be developed in the cloud? In on-prem infrastructures? What are the basic points of information about media migration that preservationists need to know deprecated/updated.
- **Models for 0% data loss in cloud environments:** Research on distributed architectures and defined data sets to determine how a no-bit-loss approach can be achieved.

For core fixity checking and the above additional areas, a real-world case study should be performed.

Appendices

Appendix A - Introduction to Digests

This section provides a basic introduction to digests and their usage in supporting fixity checks, and other preservation activities. A much more detailed discussion can be found in the **C4 white paper and web site**, both referenced in Appendix D.

A hashing algorithm, or "hash function", takes an input (e.g. the contents of a file) and generates an output value, referred to as a digest or hash. A well-designed hash function has several useful properties. Some of the most basic are:

- Regardless of the size of the input, the output value is always the same length. For example, the SHA-512 algorithm generates a 64-byte output - typically displayed to users as a value which is 128-characters long[5].

- It should be *effectively* impossible to recreate the input from the output value (except by brute force trial-and-error).

- It should be *effectively* impossible for two different inputs to generate the same output value (e.g. collisions should not occur).

- Any change to the input data, however small, will result in a different output value.

The word "effectively" implies that the probabilities of a collision or a successful brute-force attack are so infinitesimally small that they can be ignored.

Digests generated by suitably robust hash functions (sometimes referred to as *cryptographic* hash functions) can therefore be useful in ensuring integrity of transmitted data: Any change to the underlying data will cause a different digest to be generated. By comparing digests at different points in time, data changes can be detected.

These same properties also make such hash functions ideal candidates for fixity checking.

Examples of hash functions are:

- **MD5**: Generates a 128-bit output value. It is widely used as a data integrity function.

- **SHA-512**: A more robust cryptographic hash function than MD5. Part of the "SHA-2" family of hash functions.

[5] Assuming the value is displayed using hexadecimal notation.

- **C4**: A hash function based on SHA-512, but with some modifications and additional properties which make it more suitable to be used for digital asset tracking, fixity checking, etc.

A fixity check process which is based on message digests can use any suitably robust hash function. C4 is recommended in this document because of the additional benefits it brings when it is used in a wider asset management context. But there is no technical reason why other suitable hash functions cannot be used to support fixity checking - including cloud-based fixity checking.

The following examples show c4 output values for two inputs:

Input One:
```
Hello world!
```

c4 hash:
```
c43xNRh9Ax5QUuWtf3Pe9ofPQYLXgsNsoW6RNcxg5nXvxHjU4TBH6uxAhcH5TCwDwHBUkP
k1N6Sjpy6To5FaBGKfjC
```

Input Two:
```
Hello world?
```

c4 hash:
```
c422wfSpE93Bc4W4aykCtuQ6THmmv9V3PDMiKoNV6TrdgTxcUEqfDXU2RunJpFa363P9Tt
unZuLckZUPuagVXYQeXY
```

A small change to the input data results in a completely different output value.

As well as fixity checking, digests can be used for other purposes - for example:

- To support file de-duplication. Two or more files with the same digest are duplicates, even if the file names are different.
- As universal unique identifiers. Teams collaborating across departments or organizations can use digests to ensure they are referring to the correct file.
- To provide security - for example, to ensure that no information about a file can leak out from the file's digest.
- To provide one overall digest for a group of files. A set of digests can themselves be sorted, concatenated, and then fed into a hash function, to create a "digest of digests". In this way, a hierarchy of digests can be created, as needed.

Full-strength cryptographic hash functions may not be required to support fixity checking. An organization could implement a successful fixity process using MD5 digests. However, by using a more robust hash function, an organization can take advantage of the extra security features it may provide.

An organization should consider the computing effort needed to re-generate a large number of digests over time, for fixity checking. Stronger cryptographic hash functions can be more computationally intensive (i.e. slower) and therefore more expensive.

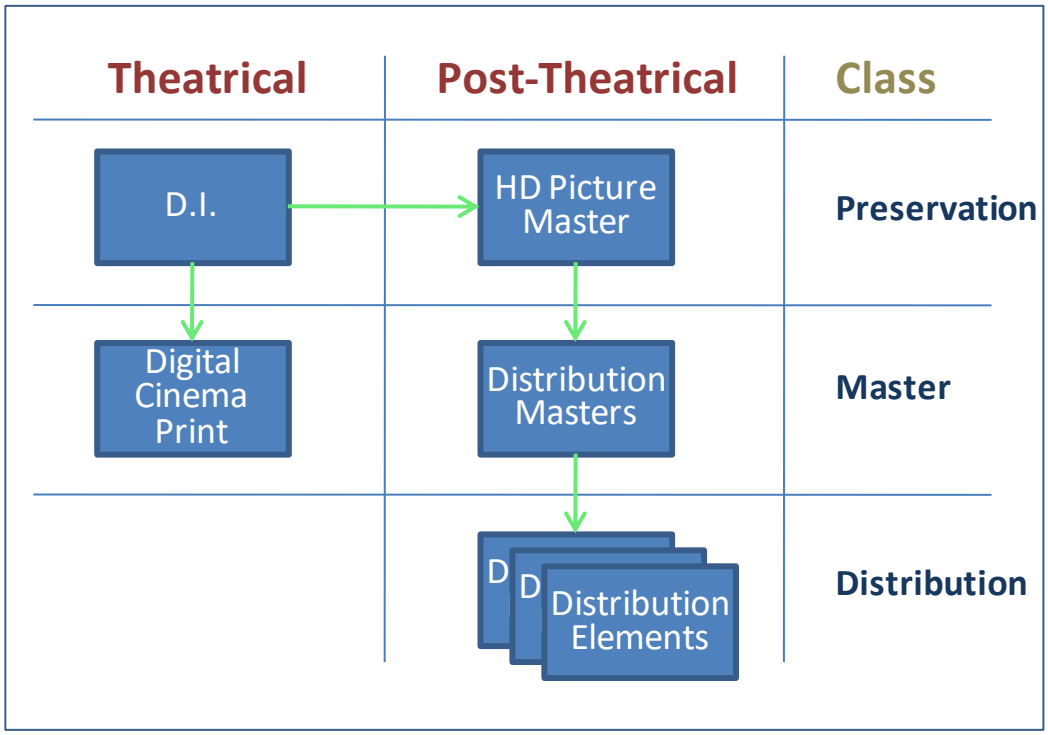
Appendix B - Asset Technical Specifications

The following list provides some examples of technical attributes which could be captured by a fixity data model, to describe preservation assets (see the Conceptual Data Model in Part One of this document):

- picture type
- picture format
- picture language code
- audio configuration
- audio language code
- file date created
- file size
- file format
- file run time
- audio bit depth
- audio bitrate
- audio codec
- audio sample rate
- audio stream length
- picture bitrate
- picture codec
- picture display aspect ratio
- picture frame height
- picture frame width
- picture frame rate
- picture stream length

Appendix C - Asset Types (Preservation & Non-Preservation)

Assets are classified in order to simplify the definition of storage policies and service level requirements:



Example categorization:

Asset Class	Use	Expected Lifespan	Clones	Health Check	Access SLA	Geographic Separation
Preservation	**Archival assets which represent the most original and unique version, in picture and sound, of the theatrical film or highest level HD picture files**	**Unlimited**	**3 times**	**Yearly**	**Hours/Days**	**1 Replicate**
Master	**Assets used to generate Post Theatrical Distribution Elements**	**5-10 years**	**3 times**	**Bi-Annually**	**Minutes/Hours**	**1 Replicate**
Distribution	**Assets used to service Business Units and deliver to distribution partners.**	**<5 years**	**3 times**	**Bi-Annually**	**Minutes/Hours**	**1 Replicate**

Appendix D - Further Reading

C4 White Paper

The original C4 white paper, including a detailed discussion of fixity, its challenges, and how C4 provides a solution to those challenges - and additional benefits in the wider context of asset management workflows. C4 is a SMPTE Standard (ST2114:2017):

http://www.etcentric.org/wp-content/uploads/2015/09/C4-ID-ETC-Whitepaper.pdf

Further information can be found on the C4 web site:

http://www.cccc.io

Digital Preservation Handbook

The Digital Preservation Coalition (DPC) has published a Digital Preservation Handbook:

https://dpconline.org/handbook

The handbook includes a section on cloud services:

https://www.dpconline.org/handbook/technical-solutions-and-tools/cloud-services

It also includes a glossary of terms:

https://www.dpconline.org/handbook/glossary

Cloud Storage and Digital Preservation

The UK National Archives has published a report on cloud storage and digital preservation in 2015:

http://www.nationalarchives.gov.uk/documents/CloudStorage-Guidance_March-2015.pdf

PREMIS Data Dictionary

The PREMIS Data Dictionary for Preservation Metadata, published by the U.S. Library of Congress:

http://www.loc.gov/standards/premis/

Definitions of Digital Preservation

The American Library Association (ALA) published a white paper "to promote an understanding of digital preservation within the library community":

http://www.ala.org/alcts/sites/ala.org.alcts/files/content/resources/preserv/defdigpres0408.pdf

PBCore

PBCore is a cataloging standard for the description of audiovisual content. The PBCore Advisory Sub-Committee is part of the Association of Moving Image Archivists (AMIA):

http://pbcore.org/

MPAA Best Practices for Content Security

A presentation by the MPAA describing general content security expectations and current industry best practices:

https://www.mpaa.org/wp-content/uploads/2018/10/MPAA-Best-Practices-Common-Guidelines-V4.04-Final.pdf

ETC@USC Member Companies