DIGITAL AV MEDIA DAMAGE
PREVENTION AND REPAIR

Recommendations and Techniques for Content in a 'Born Robust' Form

Deliverable D3.5

SEVENTH FRAMEWORK PROGRAMME

DAVID identifier:	DAVID-D3.5-CTI-Recommendations_for_Born_Robust_AV_Content.docx
Deliverable number:	D3.5
Author(s) and company:	Jörg Houpert, (Cube-Tec), Panos Melas (ITInnov), Werner Bailer (JRS), Paul Walland (ITInnov)
Abstract:	This deliverable describes the demand, the concept and suggests solutions for born robust audio-visual files. A list of tools fostering file robustness is presented as well as the current state of born robust attributes in the standardisation process media formats.
Internal reviewers:	Franz Höller (HSA)
Work package / task:	WP3/Task 3.3
Document status:	Final
Confidentiality:	Public
DOI:	10.7800/304DAVID35

Acknowledgement: The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 600827.

1 Table of Contents

2 List of Figures

3 List of Tables

4 Executive Summary

This deliverable identifies problems in accessing audiovisual files; presents approaches and tools to improve content robustness and give some insight to the current status in the related standardisation work. Modern audiovisual formats own a complexity and a hidden diversity of format variants that most users are not aware of. We focussed our research on the question: are technologies and procedures feasible or already available to harden an audiovisual (AV) file to make it more robust against damage and unwanted change. We extended the study on methods and procedures to keep media files robust and easily accessible for the whole media life-cycle. In addition we present a DAVID tool to losslessly recover MXF media files to make them fully standard compliant and highly interoperable.

5 Introduction

5.1 Purpose of this Document

This public deliverable summarises the findings of a research study performed within the DAVID project as Task 3.3 Recommendations and approaches for creating and maintaining content in a 'born robust' form.

5.2 Scope of this Document

Within this deliverable we are looking for technologies which can help to make new audiovisual content less vulnerable against defects, and provide means for facilitating maintaining content in robust form over time. We call this 'born robust' media content. This document summarises common problems affecting content robustness, and discusses different approaches for creating and maintaining content in robust form.

5.3 Status of this Document

This is the final version of the deliverable.

5.4 Related Documents

This deliverable makes use of the findings described in the DAVID deliverables

- D2.1 Data damage and its consequences[1]
- D2.2 Analysis of Loss Modes in Preservation Systems[2]
- D3.3 Final IT Strategies & Risk Framework[3].

5.5 Preserving content robustness

This subsection describes the genesis of "born robust" from the initial idea to the current definition.

5.5.1 Preserve to retrieve AV content in future

As it stands at the moment, we might archive a piece of AV for the simple reason that it seems important to keep our cultural, historical or artistic heritage. A little like that box of 'useful things' we keep in the garage 'just in case'. We never actually use the majority of what we keep, but there is always that nagging feeling that as soon as it is thrown away we will discover a use for it. Like a physical library, therefore, we might put content on the shelf because we feel it needs to be kept, but can't quite articulate why. Of course, the trouble with this approach is that it is very difficult to justify the ongoing expense of keeping the files. Just as in a library books and magazines take up valuable shelf space that could be used for other things, so files in a store are taking up valuable space and support resources that are hard to justify in any economic sense when we don't know when, or if, the files will ever be used again. But there is that continual nagging feeling that there is historical material that has been lost because its value wasn't recognised 50 years ago, and would now be incredibly interesting or enlightening.

[1] http://david-preservation.eu/wp-content/uploads/2013/10/DAVID-D2-1-INA-WP2-DamageAssessment_v1-20.pdf
[2] http://david-preservation.eu/wp-content/uploads/2013/01/DAVID-D2.2-Analysis-of-Loss-Modes-in-Preservation-Systems.pdf
[3] http://david-preservation.eu/wp-content/uploads/2013/01/DAVID-D3.3-Final-IT-Strategies-Risk-Framework.pdf

The trouble with this approach to archive is that there is no fixed point for recovering the files, and just like records kept in some archaic language, the means to read them might be lost by the time their value is realised, but with the amount of content it is possible to keep, the cost of maintaining all digital records in accessible condition 'just in case' is far higher than any archive budget could allow.

A second reason for archiving AV content is legislative - there are legal obligations on broadcasters to retain copies of what they transmit in case of legal or contractual challenges. This is a little different, because the files must be kept in a readable state for a specified period of time (anything from 6 months to 15 years), but do not have to be maintained in highest possible quality. The important issue is what was broadcast, not the quality in which it was broadcast (generally speaking), and any organisation that has these obligations will have a process in place to ensure it meets the letter of the law, and will be looking continually for reducing the cost of what it sees as an overhead burden. So this is a cost to the organisation, not a future benefit.

The third reason is a little more pragmatic, and is the most relevant to the issue of born robust. Any organisation that has access to, or creates, original AV material in whatever form has the opportunity to make the ownership of such material part of its business proposition. AV content has a value if it can be retrieved, re-formatted and sold to customers for documentaries, features (such as news or sport) or research. In order for this business model to work, the AV needs to be well recorded, along with all the metadata describing its attributes, it needs to be retrievable from storage and it needs to be re-playable. If this process is in place then the income derived from sales can cover the cost of maintaining the content in a playable state.

5.5.2 The idea behind “born robust” AV content

We focussed our research on the question:

Are technologies and procedures feasible or already available to harden an audio-visual file to make it more robust against damage and unwanted change?

We extended the study on how to keep media files robust and easy accessible for the whole media life-cycle.

5.5.3 The definition of “Born robust” AV content

The issue of 'born robust' therefore addresses the question of how we can create and maintain potentially valuable digital AV content in a way that we can guarantee being able to realise its value at any time in the future.

Thus we use the term of 'born robust' not only limited to the file creation, but to cover activities along the entire media life-cycle in order to ensure that the robustness of a file can be maintained and correction/repair is possible at any time during the AV file's life. The aspect of 'robust birth' concerns particularly approaches that need additional metadata to be created that can be used by tools later in the workflow, but it is a key insight of the work in DAVID that this is neither sufficient, nor is it the only option to improve robustness. As tempting as it might seem, a "create robust and forget" approach is not feasible in today's AV production and archiving environments.

In order to keep AV files in a robust, i.e., interoperable and playable state, ongoing actions and processes along the media life cycle are needed, enabled and support by appropriate metadata and tools. We don't know what format, storage or decoder changes in the future will affect our ability to decode a file, so whilst we can start by making a file robust to damage, by a combination of rugged encoding and redundancy, we also need to create a file maintenance process that ensures future readability, by identifying changes in the external environment that will affect the readability of a file, such as a change in decoder specification or media format, and instigate an ongoing migration of files that maintains their essential essence but transcodes to a supported format.

We are able, therefore, to recommend approaches and tools necessary to retain the essential future readability of a digital AV file from the moment of its creation through to its retrieval at some unknown time in the future.

1.) The file needs to be created robust to damage - this would include the use of checksums and error correction, the choice of rugged format, the use of simple metadata and the creation of helper descriptive information packages - meta-metadata about the file.
2.) The file needs to be stored in a rugged environment. This will affect the choice of storage (tape versus disk or optical for example) and the levels of duplication (need versus cost of first second and third tier storage).
3.) The file needs to be checked for bit corruption - on a regular basis, on a statistical sample basis or at retrieval time, and tools need to be available identifying how regular and how extensive these checks should be to provide a given statistical level of confidence in the integrity of the collection.
4.) A mechanism needs to be in place to identify when files need to be migrated from one format, which is becoming obsolete, into another which will be decodable for the next period of time. This will involve retaining a knowledge of the encoding conditions of the file and the ongoing changes in standards, formats (both physical media and encoding), and playout technology. The timing of such migration will be crucial - too early and the new standard is not stable, too late and the decoders are no longer available.
5.) The metadata describing the creation of the file must be retained and linked to the file - it will contain detailed information on how the encoding and wrapping of the file was done, to what standard and using what encoder products and environments. This information will be essential for informing how the file decode operation needs to be conducted for an accurate replay of the file.

5.6 How the document is organised

Within this deliverable we are looking for technologies which can help to make new content to be less vulnerable against defects. We call this 'born robust' media content.

Chapter 6 will discuss robustness problems of AV files. For this we have collected a set of candidate technologies which provides good potential to improve the robustness of content. Chapter 7 will introduce methods that improve the content robustness to secure the content robustness over the whole media life-cycle. Chapter 8 discusses standardisation activities related to content robustness, and Chapter 9 presents a summary and recommendations.

6 Identified problems in AV files

According to D3.1 digital damage is defined as: "any degradation of the value of the AV content with respect to its intended use by a designated community that arises from the process of ingesting, storing, migrating, transferring or accessing the content". In this section we try to classify specific problems of media files, and their ecosystem, which are already identified in DAVID WP2 and WP3, and which can effectively render them un-playable.

6.1 A classification for AV file problem types

The media file, as Figure 1 shows, consists of a structured stream of bits organized in different layers of the logical structure of the media file. These logical layers are the essence (baseband), the coded bit stream (encoded audio/video files) and the wrapper (container) layer.

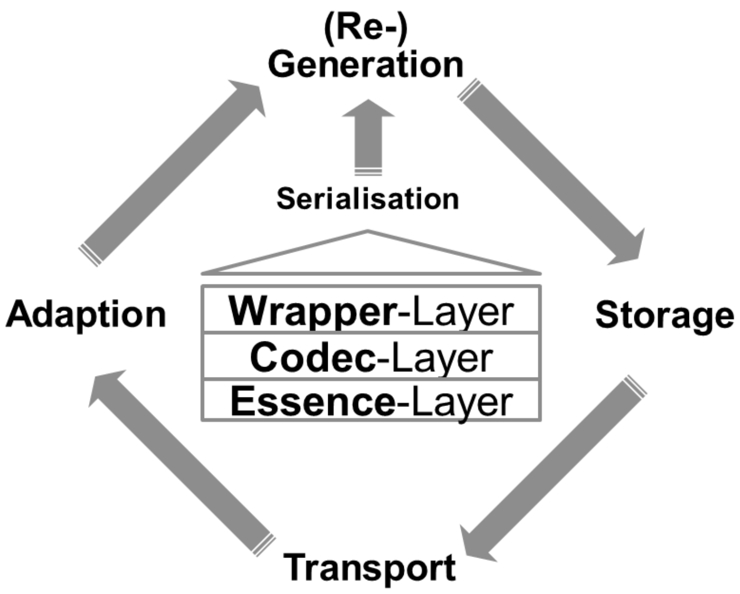

Figure 1 AV file main components and stages

The media life cycle starts with the file generation, typically by a serialisation of the data structure representation within the recording device. Typical stages in the media file life cycle are storage and transportation of the file and planned modifications, like partial restore, up to modifications which generate a new manifestation of the media file in the form of a re-generation, such as happens in a transcoding process. All the steps in the surrounding ecosystem can harm the usability of the media file.

The identified AV file problems can be grouped in two main categories as Figure 2 shows. The first one is interoperability issues caused by the media file structure itself. The second one is access issues caused by external circumstances during the workflows handling the media item.

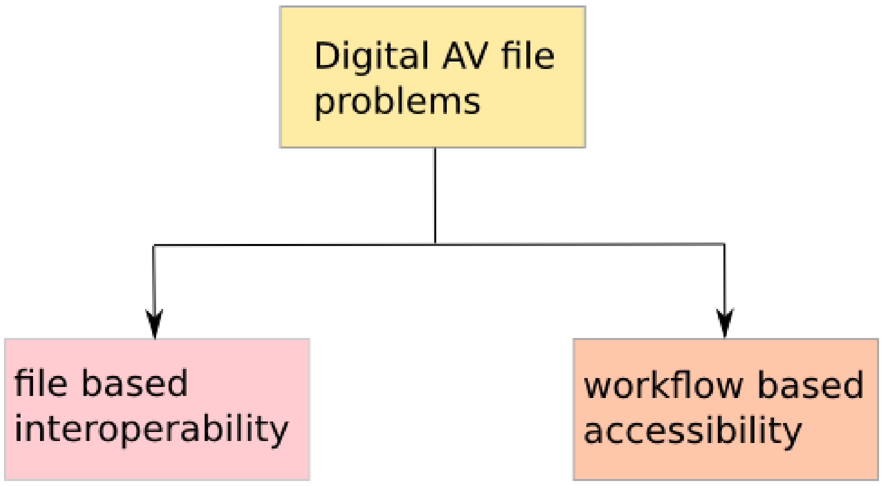

Figure 2 Digital media issues taxonomy

6.2 AV file based interoperability issues

These are problems referring to media file format structure incompatibilities. The file integrity is intact and accessible, however the content structure is either wrong, or the format itself has become obsolete and cannot be decoded. Format compatibility issues, analysed in D2.2, can occur in all logical layers of a media file, i.e. essence, codec, wrapper as Figure 3 shows.

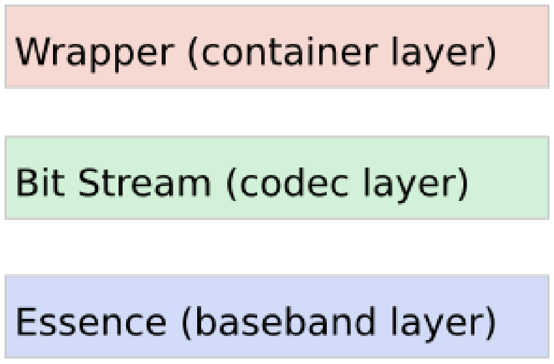

Figure 3 Media file logical view layered structure

Damage within each of these layers can have a different impact on the re-playability of the media file, e.g. cause audible or visual artefacts or make the file undecodable with regular tools when vital structural metadata is affected.

6.2.1 Wrapper layer issues

The Material eXchange Format (MXF) is considered today as the de-facto media container format for file interchange and interoperability in professional AV production workflows. The main cause of problems with the MXF wrapper is the complexity and the wide dissemination of the standard itself with a plethora of allowed possibilities and implementations. D2.2 has identified the following causes within MXF that lead to interoperability problems:

- complexity of standards, i.e. comprehensibility and unambiguousness
- incompleteness of standards definition and vendor implementations
- version conflicts with revisions of single standards documents within the suite of standards
- variability in metadata
- diversity of adaption

Typical problems identified in the wrapper layer include:

- non-KLV data
- invalid BER lengths
- incorrect data definitions for time code tracks
- invalid SMPTE unique material identifiers
- invalid SMPTE universal labels
- partition pack does not start at KLV grid line of the previous partition

Interoperability issues with MXF can cause playback problems such as:

- software crash
- no playback
- playback stops
- playback is jerky, or slower than real-time
- colour flashes
- contradictions between MXF descriptors and the actual encoding
- wrong aspect ratio at playback
- squashed picture caused by field wrapping options incompatibilities
- no audio playback
- flashes due to wrong frame rates

6.2.2 Codec layer issues

Problems on the coded essence stream layer issues can introduce:

- inter-dependences
- double definitions
- redundant recordings
- non-consistent Presentation Time Stamps (PTS)
- wrong/inconsistent colorimetric spaces
- wrong/inconsistent field/frame wrapping, field dominance
- valid but unusual resolution

6.2.3 Essence layer issues

Essence damage (essence layer) caused by non-standard compliance or missing interoperability can result in corrupted or missing data. This will cause artefacts on playback/transcoding operations.

Interoperability and conformance problems are potentially damaging and can result in systematic failure. They cannot be detected from generic checksum tools, but they become apparent at later stages, or during transcoding and migration operations. Since standards and technologies keep evolving constantly, interoperability and conformance issues become a challenging problem for the born-robust concept.

6.3 Workflow based access issues

A media file has to go through several operations/workflows during its life-cycle in which digital damage can occur:

6.3.1 Play-out problems

Playout problems can occur that either block the playback operation or there are artefacts during playback. The cause of the problem can be format incompatibility with the playback system, or system malfunction, or artefacts introduced during storage or encoding.

6.3.2 Ingest operation

During an ingest operation there might be problems, typically they might be caused by an incorrect format identification, or metadata misinterpretation, e.g. when it is missing or incomplete.

6.3.3 Migration

A migration operation can fail when tools fail to access the media file (file corruption issue) or the migration tool cannot support the source/destination formats.

6.3.4 File corruption issues

File corruption issues can occur as physical damage to a file, e.g. during storage or transportation operations. This damage in the AV file can be random changes of bit values (bit rot), or it can be more extensive damage that makes the file unreadable, or even a complete loss of the file content. Although this category of problems affects a file, they are not file interoperability issues, but are related to workflow issues, as ensuring fixity and integrity hinges crucially on an appropriate workflow.

6.3.4.1 Bit rot

Although the frequency of such corruption is extremely low, there are several issues for which bit rot should still be considered an issue. Bit rot does not only happen during data storage operations.

- use of non-error corrected subsystems (e.g. computer memory),
- the ever-increasing amounts of stored AV data,
- encoder's higher data compression rates

6.3.4.2 File crash, corruption

Describes situations when a file is corrupted and cannot be read. Main causes of file crash are:

- hardware failure,
- malicious software,
- system crash

File corruption can cause damage on all parts of a media file:

- corruption of essence causing visual/audible artefacts during playback
- corruption of wrapper preventing playback
- corruption of metadata affecting identity of the asset

6.3.4.3 Full file loss

Describes files with extensive corruption, or files that cannot be found, and there are no backups.

File corruption in any form can be easily detected with periodic fixity checks. Although limited bit rot damaged files can be fixed, a replacement from a backup file is often a more practical solution. Analysis on loss modes in WP2 has shown that although file corruption can cause significant problems, they rarely occur.

6.3.4.4 Scrubbing failure

During storage, integrity checks fail on scrubbing, this is usually caused by file corruption problems.

6.3.5 Other problems

These are problems that lie usually outside an AV file, but still affect indirectly operations and workflows on AV files. Such issues can arise from the OS environment, file system and subsystems operations, or even humans that handle AV files, and which can indirectly cause serious corruption or incompatibility problems to AV files.

- Rights metadata is an important issue in any AV material, failure to access and decode correctly IPR metadata, can substantially degrade the actual value of that asset.
- Human errors are a major cause of concern, since they can lead to severe file corruption or file loss.
- Hardware incompatibilities
- Incompatibilities between subsystems

- Failure of central control systems
- User authentication, software licences, server h/w failures.
- Problems arising from migration to a new format

Changes in hardware technologies are frequent as well as system and sub-system software updates. Handling such changes is not always straightforward and should be carefully planned in order to eliminate digital damage to AV files.

6.4 Lessons learnt from media file preservation

Changes in hardware technologies are frequent as well as system and sub-system software updates. Handling such changes is not always straightforward and should be carefully planned in order to eliminate digital damage to AV files. The cost of migration and maintenance of non-robust media files is very high. Early identified problems are usually easier to fix with bespoke tools.

Experiments with media files as in D2.2 have shown that different parts of a media file have different sensitivity to physical damage. Physical corruption to a media file does not always translate to visual damage.

Successful playout of an MXF file is not an indication that this file is not corrupted. Playback devices might be highly tolerant today, but do not guarantee future playability. On the other hand different behaviour of the same media file against a set of different decoders is an indication of a non-standard compliant media file.

7 Approaches for ‘born robust’ content workflows

Although ensuring playability in future of an AV file is the key concept of born robust, in practical terms the AV file robustness can be approached only partially. The identified causes of AV files digital damage, as classified in Section 6.1, are file access, and content format interoperability. Therefore, robustness of AV files can be improved by increasing the resilience of workflows against file access issues and by addressing format interoperability issues on the different logical levels of an AV file.

AV file corruption resilience can be improved indirectly by:

- Improved error detection
- Improved error correction

AV file format interoperability resilience can be improved indirectly by:

- Adherence to format standards, build well-formed AV content format
- Maintain format compatibility, as formats evolve, or become obsolete

In this section we provide recommendations that can improve resilience of an AV file against digital damage in practice.

7.1 A process enabling 'born robust' AV content

A born robust process can be defined as a set of tools and operations, within an environment, that directly or indirectly improve the resilience of an AV file. Such an environment can be an archival storage system with robust capabilities that enhances AV file re-playability.

Figure 4 shows such a process within an archival system with the necessary robust operations.

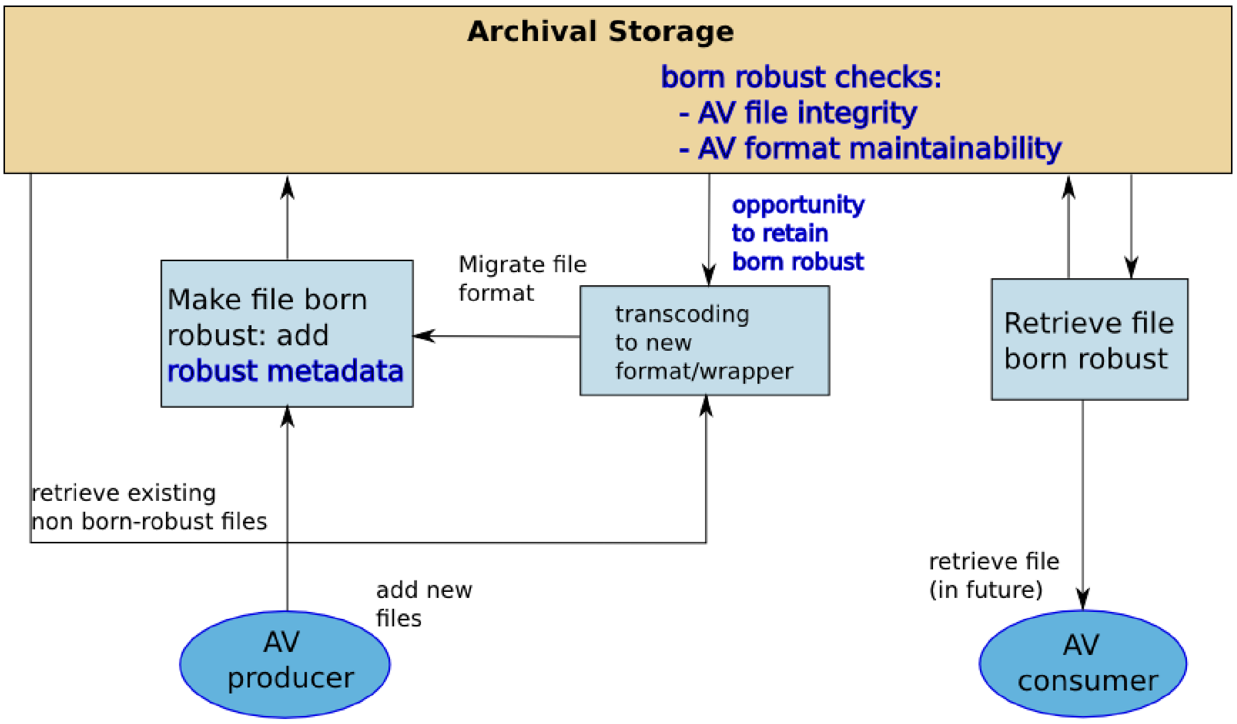
Archival Storage
born robust checks:
- AV file integrity
- AV format maintainability
opportunity to retain born robust
Migrate file format
Make file born robust: add robust metadata
transcoding to new format/wrapper
Retrieve file born robust
retrieve existing non born-robust files
add new files
retrieve file (in future)
AV producer
AV consumer

Figure 4 Born-robust process incorporates metadata for increased file integrity, reparability, and format compatibility

7.1.1 *Make a file born robust*

A born robust AV file includes a set of born-robust metadata that enhance AV file robustness in terms of integrity, reparability and wrapper compatibility. Section 7.2 discusses in more detail ways to increase the robustness AV files using robust metadata.

7.1.2 *Born robust operations in Archival Storage*

Any archival storage can be used to store/retrieve born-robust AV files, providing it can support born-robust archival operations and checks:

- maintain integrity of AV files
    - fixity checks that maintain and monitor AV file intactness
    - enable verifying the completeness of a representation of the AV content
    - archival system that understand born-robust metadata
- maintain content format interoperability
    - monitor recommended set of checks that will identify opportunity windows to migrate existing AV material to newer formats in order to maintain future robustness
    - include tools in the workflow that enable verification of format compatibility and consistency across the layers of the representation, and provide means for correction

7.1.3 *Maintain born robust AV files format compatibility*

Maintaining AV file integrity is not sufficient to maintain robustness of the AV file essence, and hence content robustness. Technology changes in wrapper formats are frequent and format standards evolve. In order to avoid wrapper and format obsolescence (incompatibility) problems, the AV file essence should migrate to newer formats and standards as they emerge.

There is an overlapping window where old technologies/standards are phased out, and at the same time new ones are adopted. The born robust process should identify such "opportunity windows" and migrate existing born robust AV file essence to newer formats. It is also important to mention that ideally, during migration, the essence of the AV file should be fully retained and unaltered. A well-formed AV file with an enhanced set of robust metadata will support the correct interpretation of the essence from the old format to the new one along the migration change path.

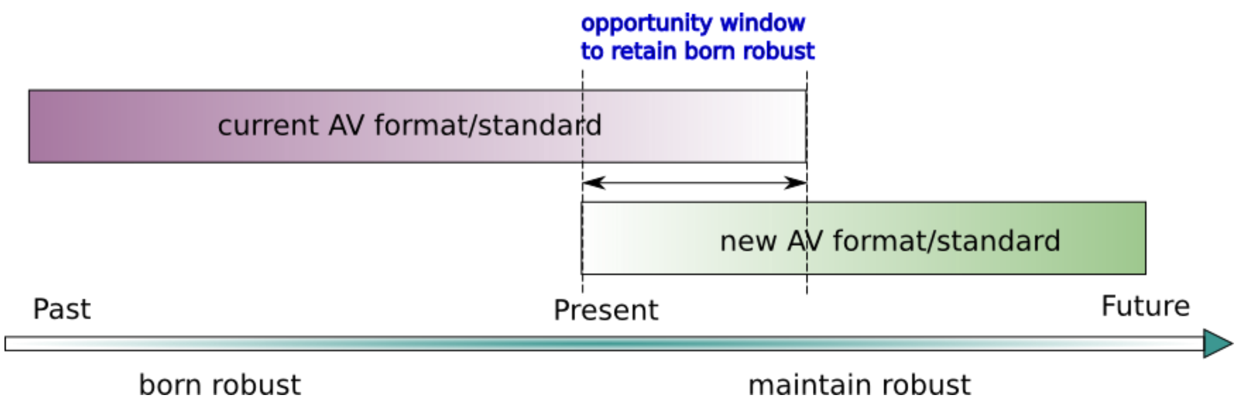

Figure 5 Migration opportunity window

7.1.4 *Convert existing non born robust files to born robust*

The concept of born-robust AV files should not be restricted to new AV files only. Other existing non born robust healthy AV files should be converted/migrated to a born robust format and stored back into the archive. After that transformation, these files should become equivalent to born robust files and retain the born robust status.

7.1.5 Retrieve file born robust

Retrieve a born robust file from archival for use by an AV consumer. The archiving system, before delivery to consumer, should check and verify robust metadata validity for that AV file.

7.2 Approaches for improving content robustness

This section discusses the options for using metadata in order to improve robustness of AV content, either by measures working on AV files or by tools integrated in the workflow. There are different approaches to improve robustness of AV content. Table 1 classifies these approaches into six groups, using two main criteria: where additional metadata to improve robustness resides, and whether the approach needs to be predefined in the formats used or is open, in the sense that it can be applied later, even if the format does not provide (sufficient) support for robustness.

Table 1: Classification of approaches for born robust AV content

	1) No additional robustness data required	**2) Additional robustness data within the content**	**3) Additional robustness data outside the content**
	no information is needed in addition to the content representation or external tools	*metadata for robustness improvement is fully contained within the media file*	*metadata for robustness uses additional born robust information located outside the media file*
a) Pre-defined robustness method	1a) improved entropy *Example:* *Robust encoding*	2a) added redundancy *Example:* *Internal checksum(s)*	3a) added redundancy *Example:* *External checksum(s)*
b) Open robustness method	1b) format redundancy and format heuristics *Example:* *Format compatibility tools*	2b) added redundancy *Example:* *Born robust recovery-pack*	3b) added redundancy *Example:* *Born robust recovery-pack add-on file*

The different types of approaches are able to address different subsets of the problems discussed in Section 6, and have their respective advantages and disadvantages. Table 2 provides an overview of these properties, which may be useful for choosing an appropriate approach for a specific process or workflow.

Table 2: Comparison of the properties of different approaches for born robust content

Approach	**Problems addressed**	**Advantages**	**Disadvantages**
1) No additional robustness data required			
1a) Pre-defined robustness method	Interoperability/ essence, codec (partly),	no additional tools needed; no need to manage	the pre-defined method is based on assumptions about the types of threats on fixity

Approach	Problems addressed	Advantages	Disadvantages
	Access/file corruption (up to a certain extent)	robustness metadata	and integrity and their expected impact. Robustness may be at risk if underlying assumptions change (e.g., storage and transmission technology, workflows) or if technology is used in other domains than originally intended; assumes that encoders/wrappers create consistent files and are not a source of error (in practice, this does not hold, in particular for complex formats)

Approach	Problems addressed	Advantages	Disadvantages
1b) Open robustness method	Interoperability/codec, wrapper; Access/subsystem incompatibilities;	can be inserted into existing workflows; can be applied at interfaces where content from untrusted sources arrives	can only make statement about current condition; may not be able to determine that damage/modification of the baseband content has occurred
2) Additional robustness data <u>within</u> the content			
2a) Pre-defined robustness method	Access/subsystem incompatibilities; Access/file corruption (up to a certain extent)	detect damage/ modification with regard to an earlier stage considered correct; metadata cannot get lost, or mixed up	extracting checksums/hashes needs precise knowledge of file/stream structure; needs support from encoder/wrapper
2b) Open robustness method	Access/file corruption (up to a certain extent) Interoperability/ essence, codec, wrapper	link between content and metadata is implicit; can be added at later workflow stage	additional metadata may be ignored or even removed by tools processing the file; extracting checksums/hashes needs some knowledge of file/stream structure; increased file size
3) Additional robustness data <u>outside</u> the content			
3a) Pre-defined robustness method	Access/file corruption (up to a certain extent) Interoperability/ essence, codec, wrapper	can be agnostic to type of file/stream; can be added at any time to existing content	link between content and metadata needs to be maintained
3b) Open robustness method	Access/file corruption (up to a certain extent) Interoperability/ essence, codec, wrapper; Access/subsystem incompatibilities	can be added at any time to existing content	link between content and robustness metadata needs to be maintained; increased file size

7.3 Tool support for improved robustness

This section discusses selected tools that support improving content robustness, using different approaches and types of metadata. As discussed above, improving robustness requires that many stages in a workflow address robustness by extracting, adding and validating metadata. The tools discussed in the following cover specific tasks in such a workflow. Tools for workflow management and service orchestration are required in addition, but are more general tools, and thus not discussed in this document. Most of the tools mentioned are freeware and open source.

Apart from the tools discussed in this section, there are further sources for tool information:

- PrestoCentre Tools Catalogue[4]
- COPTR Tool Registry[5]
- APARSEN tool repository[6]
- eCult Tech catalogue[7]

7.3.1 Tools for checking fixity and integrity

These tools deal with creating and verifying fixity and integrity information. As fixity is typically agnostic of the file structure, many of these tools are applicable to any file type. Thus a wide range of tools are available; tools developed outside the AV domain can also be used.

7.3.1.1 Fixity

Fixity creates a manifest of files stored in directories identified by the user, documenting file names, locations, and checksums. The user can then schedule regular automated scans of the directories to monitor for any changes to files. Fixity is ideal for monitoring of files in long term storage, complimenting tools such as Bagger and the BagIt specification that can be used to check fixity at points of transition.

Version 0.5 of Fixity, the free and open source fixity monitoring tool developed by AVPreserve, has been officially released for download.

http://www.avpreserve.com/tools/fixity/

7.3.1.2 ACE (Audit Control Environment)

ACE is a system that incorporates a new methodology to address the integrity of long term archives using rigorous cryptographic techniques. ACE continuously audits the contents of the various objects according to the policy set by the archive, and provides mechanisms for an independent third-party auditor to certify the integrity of any object.

https://wiki.umiacs.umd.edu/adapt/index.php/Ace

7.3.1.3 cksum

A Unix/Linux command which computes a cyclic redundancy check (CRC) checksum for each given file, or standard input if none are given

http://www.gnu.org/software/coreutils/manual/html_node/cksum-invocation.html

7.3.1.4 md5sum

A Unix/Linux command which computes a 128-bit checksum (or fingerprint or message-digest) for each specified file.

http://www.gnu.org/software/coreutils/manual/html_node/md5sum-invocation.html

7.3.1.5 sha15sum

A Unix/Linux command which computes a 160-bit checksum (or fingerprint or message-digest) for each specified file.

http://www.gnu.org/software/coreutils/manual/html_node/sha1sum-invocation.html

[4] https://www.prestocentre.org/tools-catalogue (access for PrestoCentre members only)

[5] http://coptr.digipres.org/Main_Page

[6] http://www.alliancepermanentaccess.org/index.php/tools/tools-for-preservation/

[7] http://www.ecultobservatory.eu/content/tech-catalogue

7.3.1.6 MD5summer

An application for Microsoft Windows 9x, NT, ME, 2000 and XP which generates and verifies md5 checksums. It provides a GUI for batch creation and verification of checksums.

http://www.md5summer.org/

7.3.1.7 D10SumChecker

D10SumChecker is intended for ensuring the integrity of MXF D10 Files. File integrity verified through the computation of a checksum for the whole file is a weak strategy in the case of the large 30GB/hour MXF/D10 files. An error limited to a single bit would give a checksum failure and trigger a recovery process from a backup copy for quite a large amount of data. In addition, if the second copy is found corrupted the file might be declared “lost”. This tool supports an integrity check based on data units which match content usable elements (“edit units”), such as video frames.

http://www.crit.rai.it/EN/attivita/opensource/

7.3.2 Tools for format validation

This section contains tools that validate the correctness and compliance of a file or wrapper wrt. the respective specification.

7.3.2.1 BWF MetaEdit

BWF MetaEdit permits embedding, validating, and exporting of metadata in Broadcast WAVE Format (BWF) files.

http://sourceforge.net/projects/bwfmetaedit/

7.3.2.2 MP3val

MP3val is a small, high-speed, free software tool for checking the integrity of MPEG audio files. It can be useful for finding corrupted files, e.g. incompletely downloaded, truncated, or containing garbage. MP3val is also able to fix most of the problems. Being a multiplatform application, MP3val can be run both under Windows and under Linux (or BSD).

http://mp3val.sourceforge.net/

7.3.2.3 Jpylyzer

Jpylyzer is a validator and feature extractor for JP2 images. JP2 is the still image format that is defined by Part 1 of the JPEG 2000 image compression standard (ISO/IEC 15444-1).

http://jpylyzer.openpreservation.org/

7.3.2.4 MXF Analyser Professional

The commercial MXF Analyser Professional is based on the MXF::SDK developed by IRT and MOG Solutions. The analyser is being implemented to support in-depth analysis of:

KLV layer

- Partition multiplex
- Metadata (decoding and analysis)
- Index Tables
- Essence Containers and their payload

The total structure of the MXF file (including the contents of the Header Metadata for each partition) is exported as an instance of the XML Schema and can be further validated using XML tools.

http://mxf.irt.de/tools/analyzer/

7.3.2.5 MXF File Test Engine

This BBC R&D project provides a system that allows various tests to be performed on MXF files. It is an open source project written in C++ and compiles to a Windows DLL. Tests are also in DLLs and are controlled by a basic scripting language. This allows new tests to be added without recompilation.

http://www.freemxf.org/

7.3.3 Tools for format identification

There are many format identification tools available, however, only a small part support AV files. Due to the nature of containers with streams using different codecs, format identification is related to technical metadata extraction for AV content.

7.3.3.1 DROID

DROID (Digital Record Object Identification) is a software tool developed by the UK National Archives to perform automated batch identification of file formats. Developed by the UKNA Digital Preservation department as part of its broader digital preservation activities, DROID is designed to meet the fundamental requirement of any digital repository to be able to identify the precise format of all stored digital objects, and to link that identification to a central registry of technical information about that format and its dependencies.

http://www.nationalarchives.gov.uk/information-management/manage-information/preserving-digital-records/droid/

7.3.3.2 MediaInfo

MediaInfo is a metadata extraction tool for AV content that supports a wide range of containers and codecs.

http://mediaarea.net/en/MediaInfo

7.3.4 Tools for AV file correction

This section lists tools for correcting issues detected by validators, such as violations of format specifications, inconsistencies between different components in a file, etc.

7.3.4.1 MXF Legalizer

MXF Legalizer is a commercial tool partly developed within the DAVID project by the DAVID partner Cube-Tec. The aim of the tool is to correct damaged broadcast MXF Files autonomously on a large-scale. It can be used as a compliance gateway into an AV archive to secure full standards conformity and long-term usability of the preserved AV collections.

http://www.cube-tec.com/products/mxf-legalizer/mxf-legalizer-text/

7.3.5 Approaches supported

Table 3 relates the tools discussed in the previous sections to the different classes of approaches for improving robustness. This implies also which types of problems can be addressed by using these tools.

Table 3: Robustness approaches supported by different tools.

Tool	**Pre-defined**			**Open**		
	None	**Int.**	**Ext.**	**None**	**Int.**	**Ext.**
Fixity			x			x
ACE						x
cksum			x			x
md5sum			x			x
sha1sum			x			x
Md5summer			x			x

Tool	Pre-defined			Open		
	None	Int.	Ext.	None	Int.	Ext.
D10SumChecker			x			x
BWF MetaEdit				x		
MP3val				x		
Jpylyzer				x		
MXF Analyser Professional	x			x		
MXF File Test Engine	x			x		
DROID				x		
MediaInfo				x		
MXF Legalizer	x			x		

8 Standardisation activities

Good concepts on methods for “born robust content” alone will not provide much benefit to content users. To generate a real impact in media technology these technologies must be implemented in widely spread media tools.

There are different approaches to target born robust content and processes, which relate to the different types of robustness approaches described in Section 7.2:

- Specifying fixity and integrity metadata (additional robustness data)
- Profiling (simplifying format compatibility without requiring additional data)
- Archive container formats (packing content and robustness metadata)

8.1 Fixity metadata

There are several standards in the preservation and AV media domains that provide at least basic support for checksums, including among others PREMIS [PREMIS], AudioMD [AES57,AES60]/VideoMD [VideoMD], MPEG-7 [MPEG-7] and EBU Core [EBUCore]. The EBU has also published a quite comprehensive controlled vocabulary of checksum algorithms in the context of TV Anytime[8].

However, these standards support one or more checksums per file or bitstream, but do not allow fixity metadata on a finer granularity. The MPEG Multimedia Preservation Application Format [MP-AF], which is currently being finalised, has added support for component and fragment level checksums. Its fixity descriptor provides either a single checksum, or a list of fixity checks for segments. The fixity description for a segment contains a specification of the fragment by one or more of the following means: a byte range, start and span with the respective unit (e.g., time) and stream/track identification.

In the context of media processes, the Framework for Interoperable Media Services (FIMS[9]) specifies a range of services for ingest, transfer, transform, storage and analysis of media. The metadata structures include the option to specify different types of identifiers and checksums for the media resources, in order to verify their identity and fixity in each step of the processing chain.

8.2 Integrity metadata

While fixity can provide additional metadata for handling robustness of single files/containers, it cannot ensure the completeness of digital content in multi-file cases. In digital preservation, the integrity is used to denote that digital content is both complete and unmodified.

Explicit support for integrity metadata is missing in many standards. Thus the MPEG Multimedia Preservation Application Format [MP-AF] has added a descriptor with a list of entities to check for completeness (and subsequently verify their fixity). In order to address migration scenarios that discard some representations after a period for validation, these lists may also include deprecated entities or entities with an expiry date.

8.3 Format profiling

Many formats for AV content, in particular container formats such as MXF, are designed to cover a wide range of different applications. This results in comprehensive and complex specifications, which leave many choices to implementers and increase the risk of creating incompatible implementations or producing incoherent files. Defining reduced profiles which limit the variability of the format and require

[8] http://www.ebu.ch/metadata/cs/tva_ChecksumAlgorithmCS.xml

[9] http://www.fims.tv

basic metadata are one approach to reduce these risks and increase the robustness of files conforming to the profile specification.

8.3.1 AMWA AS-07: MXF Archiving & Preservation [AS07]

AS-07 defines a vendor neutral sub-set of MXF for long-term archiving and preservation of moving image essence and associated materials including audio, still images, captions and metadata. A further simplification is the AS-07 Baseband Shim, which has been created to carry a single rendition of a single source item.

8.3.2 JPEG2000 Archiving Profiles

JPEG2000 [J2K] has been adopted as a coding format for visual essence in digital cinema, supporting both lossy and lossless compression. Part 3 of the standard [MJ2K] defines a file format for image sequences, including a simple profile. Amendment 1 of part 3 [MJ2K-A1] defines two profiles targeting archival application, the *Archive Preservation Format Profile* and the *Archive Access Format Profile*.

8.4 Archive containers

Different container formats for packing content (including multiple essence files that constitute a representation) and metadata (including metadata supporting robustness) have been standardised. The Interoperable Master Format [IMF] is not discussed here in detail, as it has not been defined for archival applications. However, metadata for robustness could be included as metadata tracks in the container.

8.4.1 MPEG Professional Archive Application Format (PA-AF)

PA-AF [PA-AF] specifies the following: a metadata format to describe the original structure of digital files archived in a PA-AF file; a metadata format to describe context information related to a PA-AF file and digital files archived in it; a metadata format to describe necessary information to reverse the pre-processing processes applied to digital files prior to archiving them in a PA-AF file; and a file format for carriage of the metadata formats and digital files.

While a general archival process may include processes ranging from creation and delivery to the archival system, to dissemination to consumers, PA-AF is limited in scope as follows: PA-AF specifies neither how to create input content nor any agreement on how the content should be handled and delivered to the archiving process; PA-AF assumes that input content for the archiving process is available in an appropriate digital format; PA-AF specifies the format of a digital archive produced by the archival process; PA-AF does not specify how the archive output by the archival process is disseminated to end-users.

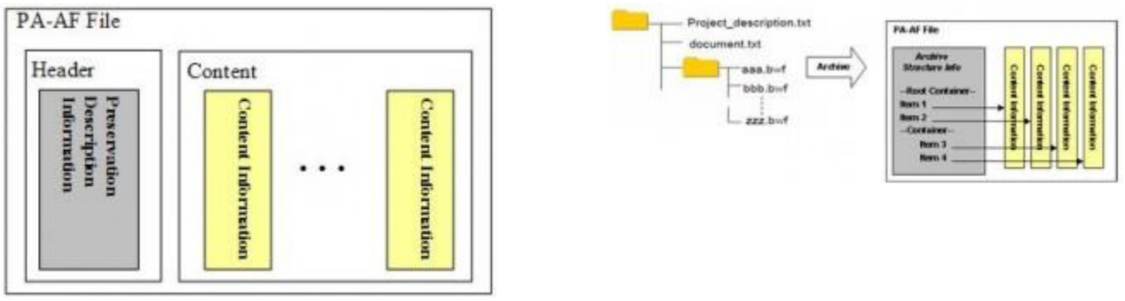

Figure 6: Logical view and structure of a PA-AF file.

The metadata document providing the entry point and which contains or references further metadata can be specified using MPEG-21 Digital Item Declaration Language (DIDL).

8.4.2 Archive eXchange Format (AXF)

AXF [AXF] is a device and technology independent format that supports interoperability among disparate content storage systems and ensures the content's long-term availability no matter how storage or file system technologies evolve. At the most basic level, AXF is an IT-centric file container that can encapsulate any number and any type of files in a fully self-contained and self-describing package. This encapsulated AXF package – or object – actually contains its own file system, which abstracts the underlying operating system, storage technology, and the original file system from the AXF Object and its valuable payload. In addition to its storage and preservation characteristics, AXF can also be used for the fully authenticated and tracked transport of file-based assets via any network topology.

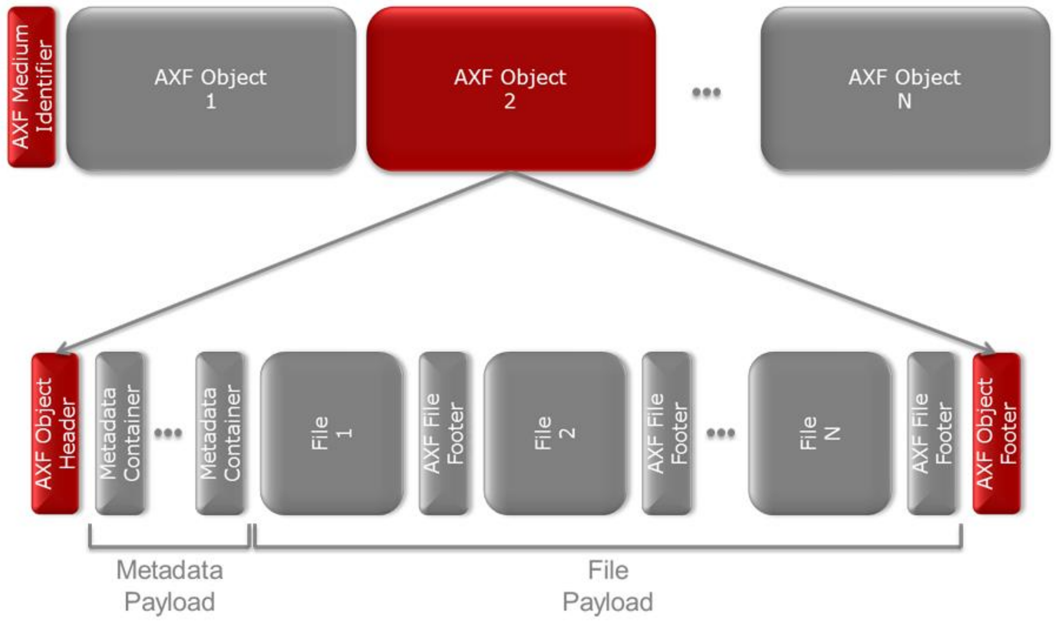

Figure 7: Structure of an AXF container.

Like PA-AF, AXF supports a virtual directory tree of the content inside the container. AXF defines some structural metadata in order to enable parsing the container. Other metadata can be included as part of the payload of the container.

9 Summary and Recommendations

This deliverable identified issues arising from malfunctions in accessing audiovisual files, presents approaches and tools to improve content robustness and gives some insight to the current status in the related standardisation work.

There are two principal approaches to improve AV media ‘robustness’; *file-oriented* and *workflow-oriented* methods. Born robust *file-based* approaches such as error correction codes can reduce media vulnerability at the cost of additional space for the redundant information for the correction code. Hash codes for fixity and sub-file fixity can support early detection of unintentional file modifications helping analyse the fault causes. Additionally stored format describing meta-information to access the file in the case of damage (recovery-pack) can help to recover files, but adds one layer with redundant stored format metadata, which can be in conflict with equivalent metadata in other format layers. The complexity of todays layered AV formats have already exceeded a level where even advanced users can’t predict the interdependence of a single parameter change. Most users, as well as programmers, are not aware of this hidden complexity as they alter file attributes. This is one of the many reasons why today’s media collections carry a wide spread of diverse hidden format inconsistencies and variations. For this reason most of the media playback devices use their own techniques to access the content as good as they can. This product specific behaviour conceals the underlying problem and can cause the same AV file to behave differently in different products. This makes today’s playability tests a poor indicator of file integrity and file standard conformity as they under-represent the scope of the problem. Standard media conformity checkers in the form of advanced QC tools are the better tool-set but still have not reached a fully mature state with standardised measurement variables and result representation methods. The EBU QC group is currently doing important standardisation work in this field.

We have to accept that there are only very limited possibilities for file hardening on a file-level to improve robustness and to warrant soundness - as even a highly “robust” media file will hardly survive in non-robust workflows. On the other side a file without any additional robustness attributes will easily survive in a “robust” workflow. Based on this insight the DAVID team has also studied loss prevention workflow techniques which promise to improve media longevity, like secure media migration strategies, computational intelligence for fault diagnosis and for preventive maintenance and (self-)healing methods to support recovery from loss. Initial results are promising, but call for a future maturation process.

There are a few lessons learnt which can be taken away as general recommendations.

- Open standard formats continuously supported by an established standardisation body are a good basis for long-term media availability. There is a tendency for new AV file formats to be standardised in a stricter - more constrained way, allowing less room for format variations and should be preferred.
- Insist on full standard compliance as a prerequisite for any of your tool acquisition processes, and point out the value of improved standard compliance to vendors.
- Be careful using media tools for which no one takes responsibility in case of failure.
- Automatic file QC for checking standard compliance is a must; use recent releases of advanced QC tools as QC tools themselves need continuous refinement.
- Establish best practices for quality management in the curation of your file collections with a high amount of stable automated workflows.
- The use of fixity hash codes to identify the origins of failure early is a must.

10 Conclusions

There is no single point in time in the life-cycle of an audiovisual media to secure its robustness. There is no way to build an auto-immune media format. The media accessibility needs to be maintained constantly by using only accepted, highly constrained media standards and automated checks with the latest versions of advanced QC tools. In future automation technology in quality controlled processes should create less unintended format variations and interoperability issues to safeguard the media life-cycle.

Existing broadcast file collections with a wide spread of diverse hidden format variations can strongly compromise the future re-use of AV files. Within the DAVID project a repair tool (called MXF Legalizer) has been developed for format unification using a lossless correction procedure. It is optimised to produce fully standards-compliant and interoperable output files, which can be seen as the starting point for the new life-cycle as born robust media.

11 References

[AS07]	AS-07 MXF Archive and Preservation Format, AMWA Draft specification, 2014, http://www.amwa.tv/projects/AS-07.shtml
[AES57]	AES57-2011, AES Standard for Audio Metadata - Audio Object Structures for Preservation and Restoration
[AES60]	AES60-2011, AES Standard for Audio Metadata - Core Audio Metadata
[AXF]	Archive eXchange Format (AXF) — Part 1: Structure & Semantics, SMPTE ST 2034-1:2014
[EBUCore]	EBU Tech 3293, EBU Core Metadata Set v.1.5, 2014
[IMF]	Interoperable Master Format, ST 2067:2014.
[J2K]	ISO/IEC 15444, Information technology -- JPEG 2000 image coding system
[MJ2K]	ISO/IEC 15444-3:2007, Information technology -- JPEG 2000 image coding system: Motion JPEG 2000
[MJ2K-A1]	ISO/IEC 15444-3:2007/Amd 1:2010, Additional profiles for archiving applications
[MP-AF]	ISO/IEC DIS 23000-15, Information Technology — Multimedia application format (MPEG-A) — Part 15: Multimedia Preservation Application Format (Draft International Standard), 2015.
[MPEG-7]	ISO/IEC 15938-5, Information technology – Multimedia content description interface (MPEG-7) — Part 5: Multimedia description schemes
[PA-AF]	ISO/IEC 23000-6:2012, Information technology — Multimedia application format (MPEG-A) — Part 6: Professional archival application format
[PREMIS]	PREMIS Data Dictionary for Preservation Metadata, Library of Congress Standard. Available online: http://www.loc.gov/standards/premis/
[VideoMD]	Library of Congress, VideoMD, http://www.loc.gov/standards/amdvmd/audiovideoMDschemas.html
[DAVID-D2.2]	DAVID, Deliverable 2.2, Analysis of Loss Modes in Preservation Systems
[DAVID-D2.1]	DAVID, Deliverable 2.1, Data damage and its consequences on usability
[DAVID-D3.1]	DAVID, Deliverable 3.1, Initial IT-based strategies for avoiding, mitigation and recovering from digital AV loss & Initial conceptual risk management framework and tools for AV preservation

12 Glossary

Abbreviations used within the DAVID project, sorted alphabetically.

AAF	Advanced Authoring Format
ACE	Audit Control Environment
ADF	Ancillary Data Packet
AESxx	Advanced Encryption Standard xx
AMWA	Advanced Media Workflow Association
ANC	Ancillary data
APARSEN	Alliance Permanent Access to the Records of Science in Europe Network
AV	Audio-Visual
AXF	Archive eXchange Format
BER	Basic Encoding Rules
BSD	Berkeley Software Distribution
BWF	Broadcast WAVE Format
CABAC	Context-based Adaptive Binary Arithmetic Coding
CAVLC	Context-based Adaptive Variable Length Coding
CBR	Constant Bit Rate
CPB	Coded Picture Buffer
COPTR	Community Owned digital Preservation Tool Registry
CP	Content Package
CRC	Cyclic Redundancy Check
Dx.x	Deliverable x.x
DAVID	Digital AV Media Damage Prevention and Repair
DIDL	Digital Item Declaration Language
DLL	Dynamic-Link Library
DM	Descriptive Metadata
DMS	Descriptive Metadata Scheme
DROID	Digital Record Object Identification
EBU	European Broadcasting Union
FP	File Package
FIMS	Framework for Interoperable Media Services
GC	Generic Container
GOP	Group of Pictures
HANC	Horizontal Ancillary Data
HRD	Hypothetical Reference Decoder
HSS	Hypothetical Stream Scheduler
IDR	Instantaneous Decoding Refresh

IEC	International Electrotechnical Commission
IMF	Interoperable Master Format
ISO	International Organization for Standardization
IT	Information Technology
J2K	JPEG 2000
JP2	JPEG 2000 part 1core file extension
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation
KAG	KLV Alignment Grid
KLV	Key Length Value
LSB	Least Significant Bit
MB	Macroblock
MBAFF	Macroblock-Adaptive Frame-Field Coding
MD5	Message Digest function 5
MP	Material Package
MP-AF	Multimedia Preservation Application Format
MPEG	Moving Picture Expert Group
MSB	Most Significant Bit
MTTF	Mean Time To Failure
MXF	Material eXchange Format
NAL	Network Abstraction Layer
NIE	Number of Index Entries
NPE	Number of Pos Table Entries
OID	Object Identifier
OP	Operational Pattern
OS	Operating System
PA-AF	Professional Archive Application Format
PREMIS	Preservation Metadata: Implementation Strategies
PTS	Presentation Time Stamps
QC	Quality Check
RBSP	Raw Byte Sequence Payload
REST	Representational State Transfer
RDD	Registered Disclosure Document
RIP	Random Index Pack
RP	Recommended Practice
SEI	Supplemental Enhancement Information
SHA-1	Secure Hash Algorithm 1
SID	Stream Identifier
SMPTE	Society of Motion Picture and Television Engineers

SOAP	Simple Object Access protocol
SODB	String of Data Bits
SP	Source Package
UID	Unique Identifier
UL	Universal Labels
UMID	Unique Material Identifier
UUID	Universal(ly) Unique Identifier
VANC	Vertical Ancillary Data
VBR	Variable Bit Rate
VCL	Video Coding Layer
VLC	Variable Length Coding
VUI	Video Usability Information
WPx.x	Work Package x.x
WSDL	Web Services Description Language
XML	Extensible Markup Language

Partner Acronyms

Cube-Tec/CTI	Cube-Tec International GmbH, GE
HSA	HS-ART Digital Service GmbH, AT
INA	Institut National de l'Audiovisuel, FR
ITInnov	University of Southampton - IT Innovation Centre, UK
JRS	JOANNEUM RESEARCH Forschungsgesellschaft mbH, AT
ORF	Österreichischer Rundfunk, AT

Acknowlegdement: The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 600827.