Conch

Project Acronym: PREFORMA

Grant Agreement number: 619568

Project Title: PREservation FORMAts for culture information/e-archives

Prepared by: MediaArea.net SARL

- Jérôme Martinez
- Dave Rice
- Tessa Fallon
- Ashley Blewer
- Erik Piil
- Guillaume Roques

Prepared for:

Date: December 31, 2014

Summary: This reports serves as a snapshot of MediaArea's research, planning, and design work to develop a implementation checker, tentatively entitled Conch.

Project Introduction

The PreForma challenge illuminates and responds to a significant and real obstacle that faces the preservation community today. This report details MediaArea's design plans to create a toolset (tentatively entitled "Conch") including a implementation checker, policy checker, reporter, and fixer of a select list of formats.

As preservation workflows have incorporated digital technology, significant amounts of careful research have gone into the selection of file format recommendations, lists of codec specifications, and development of best practices; however, despite the existence of such recommendations, there remains a lack of assessment tools to verify and validate the implementation of such recommendations. A few validation tools (such as mkvalidator) are produced alongside the development of their associated standards; however, most file format specifications are not officially tied to any validation tool and are documented through human-readable narrative without equivalent computer-actionable code. Where a metadata standard may be described in both a data dictionary and a computer-usable XML Schema, file formats standards often lack a computer-usable verification method. The PreForma project recognizes this discrepancy and the resulting long-term impacts on archival communities and seeks to fill in the gaps necessary to provide memory institutions with levels of control to verify, validate, assess and repair digital collections.

MediaArea's approach to this challenge centers on FOSS (Free and Open Source Software), modular design, and interoperability and will rely strongly on MediaInfo (an open source MediaArea product) to meet this challenge. MediaInfo is often advised as the first tool to use when a media file is not playable, allowing the user to identify characteristics that would help find an appropriate playback or transcoding tools. MediaInfo's open licensing and agility in technical metadata reporting have encouraged its integration into several archival repository systems and OAIS-complaint workflows to assist archival with technical control of collections.

MediaArea sees community involvement as a key factor of evaluating the success of the project. To encourage this, during the prototype phase MediaArea will perform the development work for command line utilities, graphical user interfaces, and documentation in publicly accessible repositories at github.com. We will set up an online set of project resources such as public access to a corpus of test media, an IRC channel, and a responsive public issue tracker.

In order to foster and demonstrate a focus on interoperability throughout the project, MediaArea will work with Artefactual in order to facilitate integration of resulting project components into Archivematica, a digital repository focused on OAIS. This collaboration will bring the availability of additional OAIS and digital preservation expertise to the project and provide an additional means for the project deliverables to be made available to users.

Introduction of Featured Formats

During the development phases MediaArea will focus on one container format, Matroska, and two streams, LPCM and FFV1. The design work of MediaArea will address formats and codecs through a modular architecture so that other formats or codecs may easily be added alongside or after development.

Matroska, FFV1, and LPCM describe very unique concepts of information including:

- a container format, Matroska
- an audio stream, LPCM
- a video stream, FFV1

These three information concepts will inform distinct user interface and reporting design in order to process these concepts through differing strategies. For instance, reporting on the status of 100,000 of video frames within a video recording may be done more efficiently in a different interface as one designed to communicate the hierarchical structure of a file format.

Additionally other formats currently being addressed by PreForma fit within these three conceptual categories; for instance, PDF and TIFF are formats (containers) and JPEG2000 is a video stream. These three concepts affect the design of an overall application shell as conformance information for each category can have its own optimized user interface.

Matroska

Matroska is a open-licensed audiovisual container format with extensive and flexible features and an active user community. The format is supported by a set of core utilities for manipulating and assessing Matroska files, such as mkvtoolnix and mkvalidator.

Matroska is based on EBML, Extensible Binary Meta Language. An EBML file is comprised of one of many defined "Elements". Each element is comprised of an identifier, a value that notes the size of the element's data payload, and the data payload itself. The data payload can either be stored data or more nested elements. The top level elements of Matroska focus on track definition, chapters, attachment management, metadata tags, and encapsulation of audiovisual data. The Matroska element is analogous to QuickTime's atom and AVI's chunk.

Matroska integrates a flexible and semantically comprehensive hierarchical metadata structure as well as digital preservation features such as the ability to provide CRC checksums internally per selected elements. Because of its ability to use internal, regional CRC protection it is possible to update a Matroska file during OAIS events without any compromise to the fixity of its audiovisual payload.

Matroska has well written documentation and a draft specification but is not defined through an external standards organization although some drafts for such work have already been produced.

FFV1

FFV1 is an efficient lossless video stream that is designed in a manner responsive to the requirements of digital preservation. Version 3 of this lossless codec is highly self-descriptive and stores its own information regarding field dominance, aspect ratio, and colorspace so that it is not reliant on a container format to store this information (other streams that rely heavily on its container for technical description often face interoperability challenges).

FFV1 version 3 mandates storage of CRCs in frame headers to allow verification of the encoded data and stores error status messages. FFV1 version 3 is also a very flexible codec allowing adjustments to the encoding process based on different priorities such as size efficiency, data resilience, or encoding speed.

The specification documentation for FFv1 is partially complete and has recently been funded by vendors utiliziing FFv1 as a codec for audiovisual preservation and large-scale digitization efforts.

Linear PCM

Linear Pulse Code Modulation (LPCM) is a ubiquitous and simple audio stream. PCM audio streams may be comprised of signed or unsigned samples, arranged in little-endian or big-endian arrangements, in any number of audio channels. PCM is very flexible but is not self-descriptive. A raw PCM file can not properly be decoded without knowing the sample encoding, channel count, endianness, bit depth, and sample rate. LPCM is typically dependent on its container format (such as WAV) to store sufficient metadata for its decoding.

Because PCM streams contain only audio samples without any codec structure or metadata within the stream, any data by itself could be considered valid PCM and decoded as audio. Determining the conformity or technical health of PCM data requires the context of information provided by its container format.

Development of a conformance checker

MediaArea's design of the conformance checker is intended to allow interoperability between the conformance checkers of PreForma's other suppliers so that users may integrate multiple conformance checkers within a single 'shell'. The PreForma project is comprised of four components:

- implementation checker
- policy checker
- reporter
- metadata fixer

The PreForma project must document and associate implementation or policy rules with data types (such as formats, streams, frames, etc) and authorities (such as specifications, community practices, or the local rules of a memory institution). MediaArea recommends that communication between the implementation checker and the shell be performed through an API designed via collaboration of the PreForma suppliers.

Implementation Checker

For each supported formats (Matroska, FFV1, and LPCM), the implementation checker should assess compliance and/or deviation between files and a series of adherence checks which are written by dissecting rules and logic from each format's underlying specifications, including rules that may be deduced or inferred from a close reading of the specification or related authoritative documentation. MediaArea has drafted registries of conformance rules within the PreForma design phase and plans to collaborate with each format's specification communities to refine them. See the Conformance Check Registry.

For streams such as FFV1 some implementation checks may be performed frame-by-frame to discover frame-specific issues such as CRC mismatches, invalid frame headers, or incomplete frames. Frame-by-frame conformance assessments will naturally be time consuming as nearly the entire file must be read. In order to accommodate user's various time priorities the checker will use options to perform checks on the first few frames of a stream, a percentage of the frames, or all of the frames.

MediaArea has drafted a registry of metadata elements to be used in described an implementation check, which provides unique identifier, the scope, and underlying rationale and authority for the check. Code created to preform checks will be internally documented with references to conformance check's unique identifiers, so that MediaArea may create resources for each conformance check that relate the identity of the check, its underlying authority, sample files, and associated code.

Policy Checker

For each format addressed through a implementation checker MediaArea will create a vocabulary list of technical metadata and contextual descriptions. Additionally MediaArea will define a list of operators to enable various comparators between the actual technical metadata and the user-provided expected metadata. Such defined language will allow users to make policy check expressions such as:

- FFV1.gop_size MUST_EQUAL "1"
- FFV1.slice_crc MUST_BE_ENABLED
- FFV1.version GREATER_THAN_OR_EQUAL "3"
- MKV.tag.BARCODE MUST_START_WITH "ABC"
- MKV.tag.DATE_DIGITZED IS_BEFORE "2014-01-01"
- MKV.tag.ISBN MATCHES_REGEX "$(?=[\text{-}0\text{-}9xX\ ]\{13\})(?:[0-9]+[-])3[0-9]*[xX0-9]$"

MediaArea proposes that PreForma suppliers collaborate to define a common expression for sets of policy checks via an XML Schema, associated data dictionary, and vocabulary of comparative operators. The collaboration would include agreement and definition on the operators ("Greater Than", "Starts With", etc) of the policy checks and attempts to normalize technical metadata between common formats where they have overlapping concepts. Each policy checker would produce a vocabulary of technical metadata specific to its format for policies to be checked against as well as inclusion within an API so that the shell can access the possible operators of any enabled implementation checker.

MediaArea will provide sample sets of policy checks based on interviews with memory institutions and community practice.

Reporter

MediaArea proposes that PreForma suppliers collaborate to create a common XML Schema to define the expression of PreForma reporting (referred to here as "PreFormaXML"). The schema should define methods to express technical metadata and relates checks to formats/streams (including components of formats and streams such as frames or attachments). The XML Schema should encompass not only information about the files being assessed but also the conditions and context of the particular use (which shell was used, with what policy sets, at what verbosity, etc). The XML Schema should be supported by a data dictionary that is also collaboratively written by PreForma suppliers. MediaArea anticipates that the implementations and features performed upon the basis of a common XML Schema may vary from supplier to supplier or per implementation checker, but that adherence to a common schema is essential to interoperability and consistent user experience amongst implementation checkers.

The PreFormaXML schema should accommodate the expression of results from multiple implementation checkers upon a single file. For instance a Matroska file that contains a JPEG2000 stream, a FFV1 stream, and a LPCM stream should be able to express one XML element about the file with sub-elements about each conformity check to reduce redundancy.

We can also add a JPEG2000 image as an attachment or a PDF as an attachment. Matroska is flexible.

CRC files can be added as attachments to reenforce OAIS.

MediaArea plans to include these features commonly within MKV, FFV1, and LPCM reporters:

- Export of a standardized PreFormaXML
- Export PreFormaXML with gzip compression (to reduce the impact of large and highly verbose XML files)
- Export of the same data within a semantically equivalent JSON format
- Other functions based on PreFormaXML (such as generation of PDF formats or summarization of multiple collections of PreFormaXML) will happen within the "Shell" component

Fixer

MediaArea will produce a fixer that allows for editing the file. Enabling this function will be performed with a substantial amount of caution as in some cases a user could use it to change a file considered a preservation master. The fixer will support assessing a file first to determine the risk of editing a structurally unhealthy file and provide suitable levels of warning to the user.

The metadata fixer shall support both direct editing on the input file (with warning) or producing a new output file as a copy that the metadata change as requested.

The metadata fixer will support comprehensive logging of the change and offer options to log the performance of the edit itself with the file if it has a means to accommodate it (such as Matroska).

In addition to metadata manipulation the fixer will accommodate structural fixes to improve the structural health of the file, such as repairing Matroska Element order if ordered incorrectly, or validating or adding Matroska CRC Elements at selected levels, or fixing EBML structures of truncated Matroska files.

Substantial care should be exercised to ensure that the implementation checker properly associates risk, user warnings, and assessments with each fix allowed. In order to allow a fix the software must properly understand and classify what may be fixed and be aware of how the result may be an improvement. Adjustments directly to a preservation file must be handled programmatically with great caution with diligent levels of information provided to the user.

An example of a fix that could be enabled in the RIFF format could be verifying that any odd-byte length chunk is properly followed by a null-filled byte. Odd-byte length chunks that do not adhere to this rule cause substantial interoperability issues with data in any chunk following the odd-byte length one (this is particularly found in 24 bit mono WAV files). If the odd-byte length chunk is not followed by a null-filled padding byte, then most typically the next chunk starts where the padding byte is and the padding byte may be inserted so that other following chunks increase their offset by one byte. This scenario can be verified by testing chunk id and size declaration of all following bytes so that the software may know beforehand if the fix (inserting the null-filled padding byte) will succeed in correcting the RIFF chunk structure's adherence to its specification.

Fixes for Matroska files could include fixing metadata tags that don't include a SimpleTag element or re-clustering frames if a cluster does not start on a keyframe.

Because many files focused on with FFV1 and Matroska implementation checkers will be quite large, MediaArea plans to provide options to either rewrite the original file with the check or edit the file in place so that the file is only changed according to the fix that is request. With the latter option is the user is 'fixing' the metadata in a 50 gigabyte Matroska file only the last few megabytes of the Matroska tagging element may be rewritten without a requirement to rewrite the non tagging elements at the beginning of the file (MediaArea deployed a similar feature within BWF MetaEdit).

Core

The shell will coordinate the actions of the implementation checker, policy checker, reporter and fixer. As PreForma seeks that the shell developed by each supplier supports each supplier's implementation checker(s), MediaArea encourages all suppliers to work collaboratively to negotiate API documentation to support not only our own interoperability but to support third-party development of additional implementation checkers to utilize the produced shells.

The development of the shell will strive to facilitate an intuitive and informed use by memory institutions at both expert and non-expert levels. The shell will include substantial internal documentation that mimics the online resources that we will provide so that the shell and implementation checker function well offline.

MediaArea will implement a scheduling service within the shell so that large tasks may be performed overnight or according to a defined schedule. MediaArea will enable the Shell to load queues of files from lists of filepaths or URLs. Because of the size of data involved in audiovisual checkers MediaArea will give priority to designing the shell and implementation checker to perform multi-threaded and optimized processing.

Implementation Checker (Shell)   The shell produced will support all functions and requirements of the implementation checker as described as an independent utility and also support:

- Allow the user to open one or many files at a time.
- Allow the user to queue simultaneous or consecutive file analysis.
- Allow the user to select how comprehensive or verbose an conformance check may be (for instance, samples frames or all frames of video).
- Enable the user to select sections of conformance checks or sets of conformance checks that they may wish to ignore.
- Enable the user to associate certain actions or warnings with the occurence of particular checks.
- Provide feedback and status information live during the file analysis.
- (For Matroska) Present a user interface that displays the hierarchical EBML structure of the file with the corresponding policy outcome for each policy check.
- (For FFV1) Present a user interface that displays frame bitstream in a table and enable the user to filter the presentation of frame bitstream according to warnings or coherency events (for example, discontinuous aspect ratio).

Policy Checker (Shell)   The shell produced will support all functions and requirements of the policy checker as described as an independent utility and also support:

- Allow PreForma-supported technical metadata vocabularies to be imported or synchronized against an online registry.
- Provide an interface for the user to import, create, edit, or export a valid set of policy checks.
- Implement selected set of policy checks on all open files or selected files.
- Present the outcome of policy checks in a manner that allows comparison and sorting of the policy status of many files.
- Allows particular sets of policy to be associated with particular sets of files, based on file identification or naming patterns.
- (For Matroska) Present a user interface that displays the hierarchical EBML structure of the file with the corresponding policy outcome for each policy check.
- Can display the FFV1 bitstream.

Reporter (Shell)   The shell produced will support all functions and requirements of the reporter as described as an independent utility and also support:
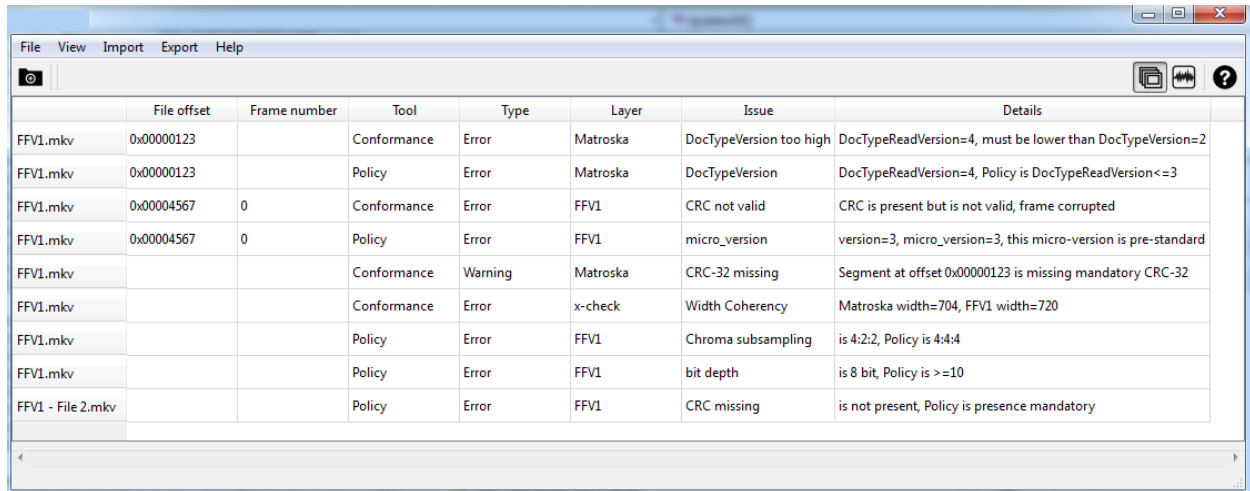
- Export of the PreFormaXML data at user-selected verbosity levels in a PDF format, which data visualizations supplied where helpful.
- Ability to read a collection of PreForma XML documents and provide a comprehensive summary and technical statistics of a collection to allow for prioritization, comparison, and planning.

Fixer (Shell)   The shell produced will support all functions and requirements of the reporter as described as an independent utility and also support:

- Allow for single file or batch editing of file format metadata.
- Allow for selected metadata to be copied from one file to another or from one file to all other open files.
- Allow for file format metadata to be exported and imported to CSV or XML to enable metadata manipulation in other programs to then be imported back into the Shell and applied to the associated files.
- (For Matroska) Present a user interface that displays the hierarchical EBML structure of the file and allows the user to create, edit, or remove (with warning) any EBML element and display the associated policy or implementation check that corresponds with such actions.

Interfaces The selected formats (MKV, FFV1, and LPCM) represent substantially distinct concepts: container, video, and audio. The optimization of a implementation checker should utilize distinct interfaces to address the conformance issues of these formats, but allow the resulting information to be summarized together.

Assessment of file conformance can be displayed via a graphical user interface or a command line interface.

Figure 1: GUI conformance output sample

Figure 2: CLI conformance output sample

An interface for assessing conformance of FFV1 video could enable review of the decoded FFV1 frames (via a plugin) in association with conformance data so that inconsistencies or conformity issues may be reviewed in association of the presentation issues it may cause.

MediaArea proposes an interface to present conformity issues for audio and video streams (FFV1 and LPCM) on a timeline, so that conformance events, such as error concealment or crc validation issues may be reviewed effectively according to presentation, parent Matroska block element, or video frame.

For deep analysis, a distinct interface that allows for its hierarchical structure to be reviewed and navigated. The presentation should allow for MKV and FFV1 elements to be expanded, condensed, or filtered according to element id or associated conformity issues.

A summary of the file properties can also be displayed via a command line interface for quick reference or export.
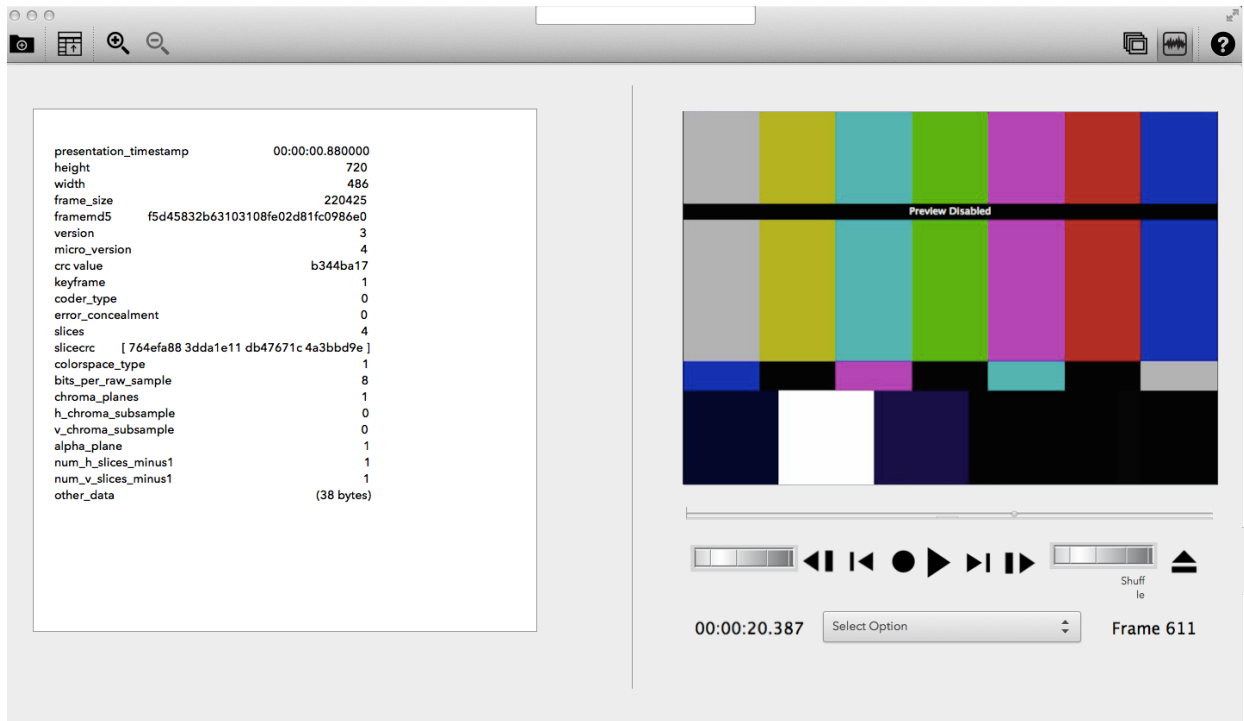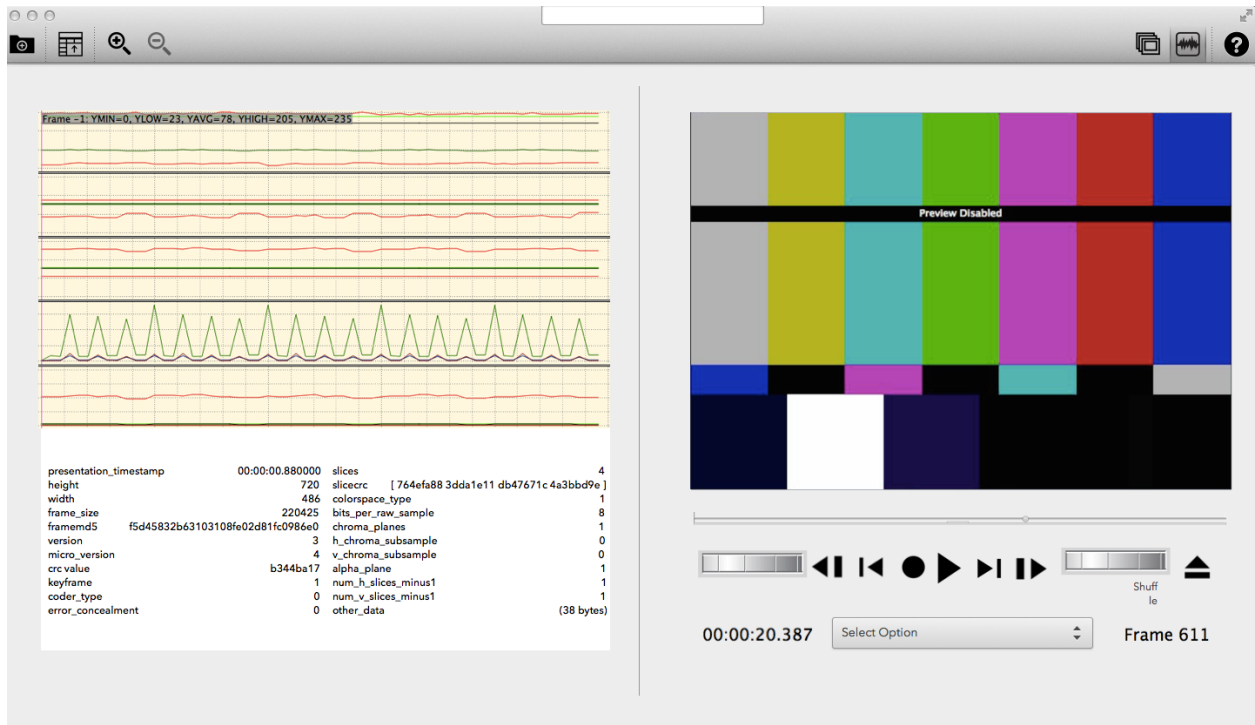
Figure 3: frame view mockup
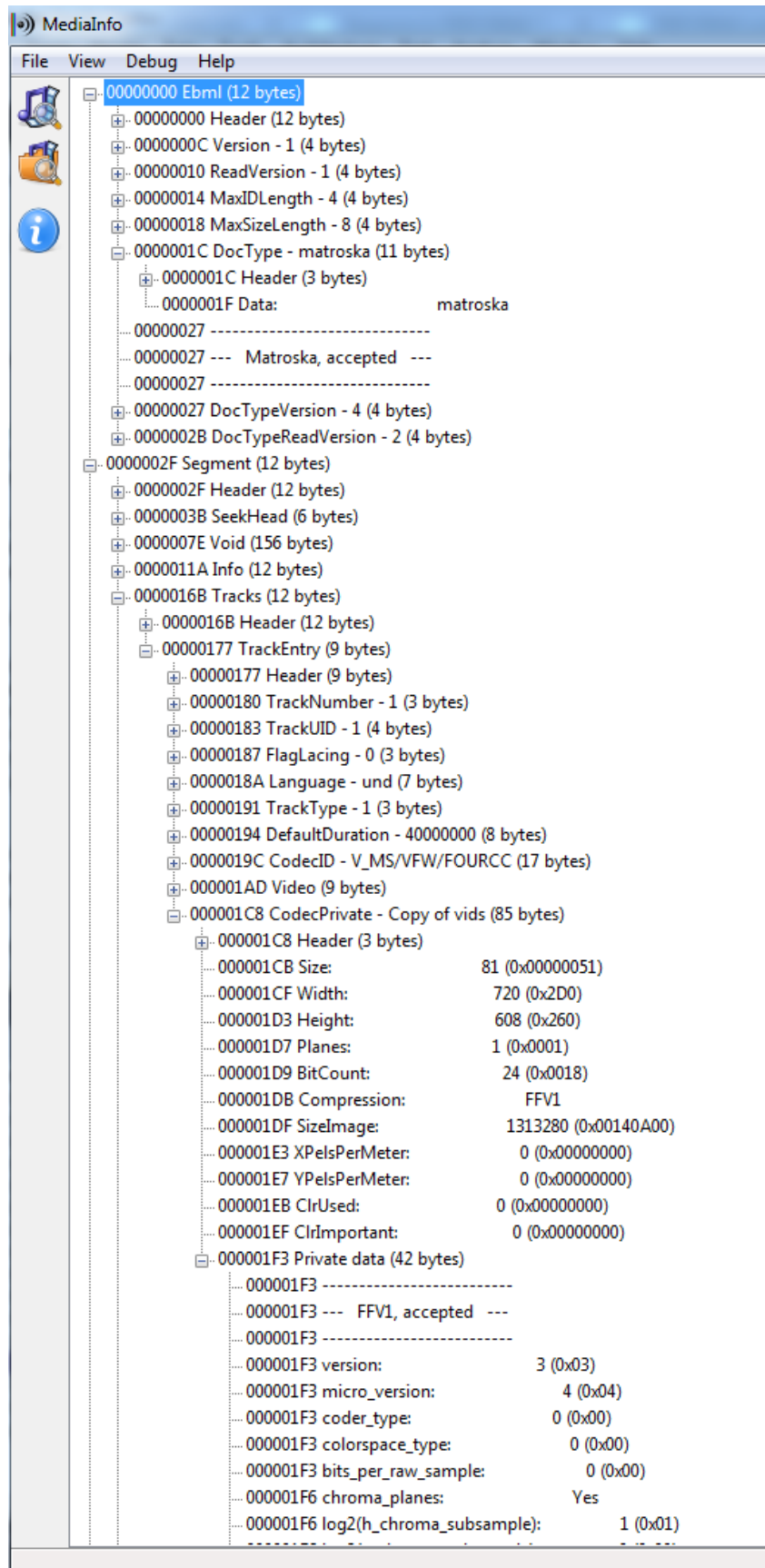

Figure 4: frame scrolling mockup

Figure 5: MediaInfo Windows

```
General
Unique ID                                : 20526629856631539204993159942613942988 (0x9A6CD9E193016EB230C85E0E7E23793C)
Complete name                            : /Users/ashley/Downloads/pres_metadata_sample_20110608.mkv
Format                                   : Matroska
Format version                           : Version 2
File size                                : 69.2 MiB
Duration                                 : 6s 439ms
Overall bit rate                         : 90.1 Mbps
Encoded by                               : {Name of the technician or organization responsible for the encoding and file creation process}
Encoded date                             : UTC 2011-05-03 02:12:03
Mastered date                            : UTC {Date and time of the file creation process. Enter in ISO 8601 format.}
Writing application                      : mkclean 0.8.2 r from libebml v1.2.0 + libmatroska v1.1.0 + mkvmerge v4.7.0 ('I Got The...') built on Jun  6 2011 16:40:20
Writing library                          : Lavf53.2.0
Original source form                     : {Physical format of the source tape. Vocabulary?}
OriginalSourceForm/BarCode               : {Barcode or other identifier from the tape.}
OriginalSourceForm/BarCode/source        : {name of barcode authority or creator; example "XYZ Archive Barcode"}
OriginalSourceForm/LABEL                 : {The record label or imprint on the source media object.}
OriginalSourceForm/DATE_RECORDED         : {The time that the recording began. This is akin to the TDRC tag in ID3.}
OriginalSourceForm/condition             : {Comments on the condition of the source, any preparation for playback of the source tape (if relevant).}
OriginalSourceForm/initial_source_timeco : 01:02:15;10
OriginalSourceForm/initial_source_timeco : DropFrame=Yes / 24HourMax=No / IsVisual=No
EncodedBy/Url                            : {URL for the technician or organization listed in ENCODED_BY.}
EncodedBy/processing_actions             : {Description of any actions performed during processing, such as trimming silence.}
EncodedBy/capture_software               : Blackmagic Media Express
EncodedBy/capture_software/version       : 2.0.3
EncodedBy/capture_operating_system       : Ubuntu
EncodedBy/capture_operating_system/versi : 10.10
EncodedBy/capture_device                 : Decklink Studio SDI
EncodedBy/capture_device/manufacturer    : Blackmagic-Design
EncodedBy/capture_device/serial_no       : xyz6789
EncodedBy/capture_device/settings        : {notes on adjustments or settings}
EncodedBy/playback_device                : SVO-5800
EncodedBy/playback_device/manufacturer   : Sony
EncodedBy/playback_device/serial_no      : abc1234
EncodedBy/playback_device/settings       : {notes on adjustments or settings}
EncodedBy/playback_device/playback_signa : {Protocols used to transfer audiovisual data between the playback deck and the capture device. example: Component}
Attachment                               : Yes / Yes

Video
ID                                       : 1
Format                                   : FFV1
Codec ID                                 : V_MS/VFW/FOURCC / FFV1
Duration                                 : 6s 440ms
Width                                    : 720 pixels
Height                                   : 486 pixels
Display aspect ratio                     : 4:3
Frame rate mode                          : Constant
Frame rate                               : 29.970 fps
Standard                                 : NTSC
Compression mode                         : Lossless
Language                                 : English
Default                                  : Yes
Forced                                   : No
Encoded date                             : UTC 2011-05-03 02:12:03
```

Figure 6: CLI output sample

Optimization for Large File Size

Design of a implementation checker and shell should be considerate of the large file sizes associated with video. For instance, an hour-long PAL FFV1 file (which contains 90,000 frames per hour) should provide efficient access if cases where one FFV1 frame contains a CRC validation error.

A video implementation checker should be well optimized and multi-threaded to allow for multiple simultaneous processes on video files. Additionally the implementation checker should allow a file to be reviewed even as it is being processed by the implementation checker and allow assessment of files even as they are being written.

Focus on Fixity

Both FFV1 and Matroska provide fixity features that serve the objectives of digital preservation by allow data to be independently validated without the requirement of managing an external checksumming process. FFV1 version 3 mandates CRCs on each frame. Matroska documents methods to embed checksums (MD5) in Matroska elements to allow for future validation of any content.

Although the Matroska specification states that "All level 1 elements should include a CRC-32" this is not the practice of most Matroska multiplexers. As part of the Fixer aspect of this project, MediaArea proposes to develop a implementation checker that allows users to add CRC-32 to selected elements.

The advantages of embedded fixity in preservation media files are significant. The use of traditional external checksums does not scale fairly for audiovisual files, because since the file sizes are larger than non-audiovisual files there are less checksums per byte, which creates challenges in addressing corruption. By utilizing many checksums to protect smaller amounts of data within a preservation file format, the impact of any corruption may be associated to a much smaller digital area than the entire file (as the case with most external checksum workflows).

Reference and Test Files

MediaArea anticipates creating a large corpus of reference and test files highlighting many of the issues documented in our conformance check registry. After an intital clearing of any associated rights, such files will be published under PreForma's selected open license agreements and disseminated for public consumption. Curated references to other relevant reference and test files in sample libraries will also be made available, which will include but not be limited to the following:

- Selections from http://samples.ffmpeg.org
- Selections from http://archive.org
- Selections from http://multimedia.cx
- PDF/A files buggy files: http://www.pdfa.org/2011/08/isartor-test-suite/
- JPEG 2000 files: https://github.com/openplanets/jpylyzer-test-files
- Matroska buggy files: Homemade + request to Matroska mailing list
- FFV1 buggy files: Homemade + request to FFmpeg mailing list
- LPCM files: Homemade

Together these references examinine a diverse set of file format expressions created by a variety of unique software platforms.

Intended Behavior by Use Case

Overview

The following use cases are presented to describe intended behaviors of the implementation checker:

Conformance Checking in an Open Archival Information System (OAIS) PreForma acknowledges the recommendations described in Consultative Committee for Space Data Systems' Open Archival Information System (OAIS) reference model intended for the long term preservation of digital information (CCSDS 650.0.-B-1/ISO 14721:2003). PreForma MediaArea aims to identify all relevant areas of the OAIS reference model in relation to its proposed conformance checker toolset. An additional examination of other OAIS-related standards (PREMIS, for example) will further minimize any incompatibility with the PreForma project.

The OAIS reference model calls for the formation of conceptual containers known as Information Packages. Submission Information Packages (SIPs) contain data objects created by Producers that are received by Archive Managers for subsequent ingest into the Archive. During the ingest process, functional entities such as Quality Assurance (QA) are performed on SIPs, and proved well-formed and valid, are transformed into Archival Information Packages (AIP) for long-term retention. Finally, generated Dissemination Information Packages (DIPs) are retrieved for access by Consumers and/or Designated Community.

Conformance checking services play a major role in the OAIS reference model. The Ingest QA function in particular validates SIPs in the temporary storage area prior to AIP generation. The PreForma conformance checker and its associated toolset would verify SIPs through implementation checking and policy checking with rules and specifications defined by the Archive. The resulting report would be submitted as associated Preservation Description Information (PDI).

Checking file integrity and conformance is also needed in the forming of an Archival Information Packages. As Archival Information Packages are generated for the Archive, a conformance checker would map all transformations through the collection of associated Representation, Content, and PDI information. Upon a dissemination request, reports created by the conformance checker would be used as descriptive information needed for the processing of objects for the Dissemination Information Package (DIP). For Matroska, individual sub-element CRCs can be submitted as Fixity Preservation Description Information and packaged with the AIP for long-term preservation.

Policy check expressions are useful in various functions of OAIS workflows, including Archival Storage where format migration of AIPs is periodiclally undertaken. Here the comparators of technical metadata between source and target content data objects is key. For example, if the relationship between an archive's Producer and Management has deteriorated over time and previously ingested SIPs (now AIPs) have now been identified as containing incorrect metadata concerning an object's pixel aspect ratio, a fixer could in effect produce a new AIP with the corrected pixel aspect ratio while while retaining the Producer's original object.

Like OAIS itself, the PreForma conformance checker aims to serve as a framework for standards-building activities. PreForma MediaArea's project produces a complex Representation Net in the OAIS Archive, providing information needed to adequately understand the proper playback of an associated data object.

Conformance Checking at Digitization Time

Verification of Digitization Policy Archival digitization workflows are generally highly defined and consistent so that various analog source objects are associated with particularly digitization requirements. Generally digitization specifications are selected in order to reduce alterations to the significant characteristics of the analog source material. Example of such digitization scenarios may be:

- A PAL Betacam SX tape is digitized to a Matroska/FFV1 file at PAL specifications with YUV 4:2:2 8 bit video and 4 channels of 24 bit LPCM audio
- A NTSC U-Matic tape is digitized to a Matroska/FFV1 file at NTSC specifications with YUV 4:2:2 10 bit video and 2 channels of 24 bit LPCM audio
- A 1/4" audio reel is digitized to a 2 channel 96000 Hz, 24 bit audio LPCM file
- A CD-R is ripped to a 44100 Hz, 16 bit, 2 channel LPCM file
- A DAT tape is ripped to either a 32000, 44100, or 48000 Hz 16 bit file

Such digitization requirements may be expressed into a policy checker set through the shell or policy checker to verify that the results of digitization are consistent with the archive's specifications. This includes both

sets of technical metadata and specification as well as anticipated embedded descriptive, preservation, or administrative metadata.

Verification of Lossless Digitization   Until recently audiovisual digitization required a fairly inflexible set of hardware requirements and extremely limited possibilities for an open source approach to video digitization. Due to the bandwidth and processing requirements for the digitization of standard definition video required the installation of PCI cards and often the use of hardware encoders that were designed to encode video as fast as the video was being received to codecs like MPEG2 or JPEG2000. With modern connectivity options such as USB 3 and Thunderbolt it is easier to add video digitization capabilities to modern computers and more archive are performing this internally. Additionally modern computer processers can now transcode video losslessly in software from a video input without the need to rely on proprietary hardware-based encoders. Open source solutions such as DVA Profession, bmdcapture, and FFmpeg along with the open provision of video digitization software development kits, such as the Blackmagic SDK are facilitating new open development projects for archival video digitization.

As vendors and memory institutions are increasing considering and implementing digitization workflows that encode video directly to lossless codecs without the use of an intermediate file-based uncompressed audiovisual data, it is increasingly crucial to assess this lossless file soon after creation to detect any flaws within the digitization process.

For those digitizing video through processes that incorporate libav or FFmpeg such as bmdcapture of FFmpeg's decklink integration, a separate framemd5 may be written alongside the encoded ffv1 data. The resulting ffv1 data may then be verified against the framemd5 to verify that the correct bits were written to disk.

An inspiration for the use of framemd5 reports within a digitization workflow is inspired by the verify option with the flac utility available at http://flac.sourceforge.net/. The '-V' or –verify command is used to decode the encoded stream in parallel to the encoding process to double-check the losslessness of the transcoding. With this method any discrepancy between what data is read and transcoded versus what data is written to disk could be identified in a subsequent verification process. The use of framemd5 data within a digitization workflow enables verification in cases where an option similar to flac's –verify argument isn't available.

Assessment of Vendor/Producer Deliverables   For archives that clarify specifications for audiovisual digitization projects, the implementation checker should facilitate a workflow for the archivist to express those specifications and verify received material against them. In addition to testing for the presence and order of required metadata tags the implementation checker should also be able to verify that they adhere to particular patterns as expressed through regular expressions.

The implementation checker should be able to verify that files were transferred completely and that the delivered material does not contain any partial files from an incomplete or aborted transfer.

The implementation and policy checker's reporting on deliverables will enable the user to provide specific feedback to the vendor or producer to create files with greater compliance or coherency.

Conformance Checking at Migration Time

Fixity Verification   Migration of large amounts of data introduces risk for digital corruption and/or sector loss. Ongoing data migration is essential for digital preservation but can require a time consuming verification process. Both Matroska and FFV1 contain features for internal fixity so that a file copied from point A to point B can be assessed at point B alone to verify the data integrity of the frames. MediaArea recommends using Matroska's CRC features for use in digital preservation to allow for fixity verification to be more stable and achievable with the file alone without necessarily depending on external databases or records of checksums.

Obsolescence Monitoring   Migration is typically an ideal time to perform obsolescence monitoring and preparing actions to limit complications in obsolescence status. Just as memory institutions must maintain the technology that their physical collections are dependent upon, this is equally true for digital collections. As this maintenance becomes more complex, costly, or unlikely archives will typically reformat material (with as little compromise to the content and characteristics of the source as possible) to a format that has more sustainable characteristics.

To counteract arising obsolescence challenges it is critical to have access to thorough sets of technical metadata in order to associate certain codecs, formats, or technologies with sustainability risks or to identify what one format should be superseded by another in a particular digital preservation. For instance an institution that utilized FFV1 version 0 as a lossless preservation codec may wish to identify such files to reformat them to FFV1 version 3 (now that it is non-experimental) in order to take advantage of version 3's additional advantages. In our research one archive found that some digitized material received from a vendor was missing technical metadata about field dominance and had to identify exactly which materials were affected to order to rectify the issue.

The team and roles

- Jérôme Martinez (Digital Media Specialist): technical design, implementation of the bytestream/bitstream analyzer, extraction of metadata.
- Guillaume Roques (Back end / Front end developer): database management, automation, performance optimization, shell.
- Name to be confirmed (Junior developer): GUI development, reporting.
- Dave Rice (Archivist): communication with memory institutions, definition of tests, documentation.
- Ashley Blewer (Archivist): technical writing and documentation, design and user experience optimization
- Tessa Fallon (Archivist): technical writing and documentation, community outreach, standards organization
- Erik Piil (Archivist): technical writing and documentation, OAIS compliance support

Community

Artefactual Systems and Archivematica

Artefactual Systems is a privately owned company incorporated in the Province of British Columbia with expertise in open-source, open-standard technologies for archival collections and digital repositories. Artefactual is best known for its two open-source software tools, the Archivematica digital preservation system and the AtoM online access system.

MediaArea proposes to include within its project a component focused on implementing parts of the MediaArea conformance checker within an independent and OAIS-focused repository system. As many of Archivematica's development philsophies (modularity, OAIS, open source, and focus on memeory institutions) align well with the spirit of PREFORMA's Challenge Brief and Archivematica was an early adopter of Matroska/ffv1/LPCM within a preservation context, we have selected Artefactual as an ideal collaborator to ensure that the results of our work are well-prepared for integration within existing open source repository solutions. Archivematica can benefit from PREFORMA's development of conformance checkers to strengthen that step of the OAIS processs. Additionally the incorporation of the conformance checker into Archivematica shall allow memory institutions with a new means to access the results of the project.

We believe that incorporating Artefactual into our phase 2 proposal provides a meaningful deliverable, the incorporation of a PREFORMA conformance checker into a key OAIS solution. The collaboration also provides the team with a strategic and indenpendent test case implementation. Although Artefactual will not take part directly in the development of the conformance checker, our project proposal includes funding to sponsor Artefactual to test, provide feedback upon, and implementation selections of the conformance

checker. Additionally we anticipate that Artefactual's existing work on the Format Policy Registry may provide a positive influence on our work on the policy checker.

Project Advisors

MediaArea has approached several individual and institutional partners from varying types of organizations to provide expertise, testing, and feedback to the project as it develops. The intent of this aspect of our proposal is to facilitate an efficient and responsive mechanism for feedback with specialized areas of expertise relevent to the project. This initiative also kickstarts the relationship between the project, open source development communities, and PreForma's target users. Such partners would receive a fixed stipend amount from our proposed in exchange for participation in the projects mailing list, occasional requested meetings, and commenting on the project.

- Moritz Bunkus (Matroska main developer): validation of Matroska tests, Matroska specific technical support, review of standardization efforts
- Michael Niedermayer (FFmpeg maintainer and FFv1 primary author): validation of FFV1 tests, FFV1 specific technical support, review of standardization efforts
- Luca Barbato (Libav maintainer): validation of FFV1 tests, FFV1 specific technical support, review of standardization efforts
- Ian Hendersohn (User of ffv1/mkv/lpcm in a national archive): Provide testing and feedback of project tools
- Peter Bubestinger (User of ffv1/lpcm in a national archive): Provide testing and feedback of project tools
- Artefactual Systems (consulting and development company): Provide testing and feedback of project tools, implement select conformance checker tools into Archivematica
- Kieran Kuhnya (encoding strategist): Advisor on implementation and standardization efforts

Through real-time feedback and specialized review these participants will provide crucial evaluation of the project throughout the second phase.

Open Source Ecosystem

Cross Platform Support

MediaArea excels in open source development for cross-platform support and chooses development frameworks and tools that enable cross-platform support to be maintained. Several applications developed by MediaArea such as QCTools, MediaInfo, and DVAnalyzer are available under nearly all major operating systems. To achieve this we will program in C++ and use the Qt application framework (only for the GUI, pending licensing).

For an impression of MediaArea's focus on cross platform usability please see our download pages:

- http://mediaarea.net/en/MediaInfo/Download
- http://bavc.org/qctools-downloads

MediaInfo is also officially provided by multiple open source distributions:

- Debian: https://packages.debian.org/wheezy/mediainfo
- Ubuntu: http://packages.ubuntu.com/utopic/mediainfo
- RedHat / Fedora: https://apps.fedoraproject.org/packages/mediainfo
- OpenSuse: http://packman.links2linux.org/package/mediainfo

- Arch Linux: https://www.archlinux.org/packages/?q=mediainfo
- FreeBSD: http://www.freshports.org/multimedia/mediainfo/
- Homebrew (open source software package management system for Mac): http://brewformulas.org/MediaInfo

Online Resources

MediaArea will utilize GitHub as a social and development center for Conch development and uses GitHub's issue tracker and wiki features alongside development.

For communication MediaArea will establish public mailing lists and an IRC channel for foster support and involvement from memory institutions.

MediaArea will solicit, create, and accept test files and reference files that highlight various features of the implementation checker and illustrate likely preservation issues that may occur within the selected formats.

Community Interviews

In December 2014, MediaArea started conducting interviews with FFV1, Matroska, and LPCM stakeholders in order to collect feedback and insights from the archives community. To date, interviews have been conducted with:

- Hermann Lewetz, Peter Bubestinger; Österreichische Mediathek
- Ian Henderson; UK National Archives
- Christophe Kummer; NOA
- George Blood; George Blood, L.P.

Notes and partial transcripts (in English) from the interviews are available in the MediaInfo PreForma GitHub repository. Public release of interviews is pending complete transcriptions and review of transcriptions by all participants in order to ensure accuracy and compliance with Creative Commons CC-BY 4.0. The interviewees' feedback will help inform MediaArea's approach to development in all areas, and especially reinforced our plans to standardize the FFV1 specification through an open source standards organization

Advance Improvement of Standard Specification

FFV1 Specification   Efforts to create an FFV1 specification began in April 2012, continuing through the August 2013 release of FFV1 version 3. Currently the specification remains in development at http://github.com/ffmpeg/ffv1. Ideally a specification should fully inform the development of a decoder or parser without the need to reference existing implementations (such as the ffv1 implementations within ffmpeg and libav); however MediaArea's initial research and prototyping efforts with FFV1 found the current specification insufficient to create a decoder. As a result MediaArea utilized ffmpeg's FFV1 implementation to fully interpret the specification. Several threads on the ffmpeg-devel and libav-devel listserv reference discussions about the development of the FFV1 specification and consideration of efforts to standardize the specification through a standards organization, such as IETF (Internet Engineering Task Force) [1].

In consideration of FFV1's utilization within preservation contexts, the standardization of the codec through an open standards organization would better establish FFV1 as a trustworthy, stable, and documented option. In MediaArea's interviews with FFV1 adopters, interviewees noted that FFV1's current status proved problematic in gaining organizational buy-in for adoption of FFV1. Additionally, standardization of FFV1 would increase awareness of and interest in FFV1. This increased visibility is vital to engaging an overly cautious archives community. At the moment FFV1 can be seen at a tipping point in its use within preservation context. Its speed, accessibility, and digital preservation features make it an increasingly attractive option for lossless video encoding that can be found in more and more large scale projects; the standardization of FFV1 through an open standards organization would be of broad interest to digital

preservation communities and facilitate greater accessibility of lossless encoding options that are both efficient and standardized.

MediaArea proposes working closely with the lead authors of the FFV1 specification in order to update the current FFV1 specification to increase its self-reliance and clarity. Development of the FFV1 specification early within the PreForma project will generate substantial feedback to the authors of the specification which could then be offered through the specification's github page via pull requests or the issue tracker. MediaArea proposes at a later stage of development that the Preforma project serve as a catalyst to organize, facilitate, and sponsor the IETF standardization process for FFV1.

Considering the two-year timeline of the PreForma project and usual pace of IETF standardization projects, we propose at least submitting FFV1 as an Independent Submission to IETF that could provide workable timeline, encourage a detailed review process, and assign a formal RFC number to the specification.

Matroska Specification Both the Matroska specification and its underlying specification for EBML are at mature and stable stage with thorough documentation and existing validators, but several efforts of the PreForma project can serve as contributions to this specifications. The underlying EBML specification 2 has already been drafted into RFC format but is has not yet been submitted to IETF as an Independent Submission or otherwise. MediaArea recommends that PreForma play a similar catalyst role for further standardization with Matroska as well, helping enable the refinement of the current RFC draft and coordinating an IETF process.

Matroska has a detailed metadata specification at http://www.matroska.org/technical/specs/tagging/index.html. Each tag has an official name and description while provides rules and recommendations for use. Many of these tags could be associated with validation rules, such as expressed by regular expression to assure that the content of the tag conforms to expectations. For instance tags such as URL, EMAIL, or ISBN have specific allowable patterns for what may be contained. As part of build a conformance tool for Matroska, MediaArea will generate conformance tests for individual tags and these tests may be contributed back to the Matroska specification in a list of regex values, an XML schematron file, or other acceptable contribution method.

Other Suggested Improvements or Contributions to Standard Specifications

- Register an official mime type via IETF for Matroska.

- Register dedicated FFV1 codecid with Matroska (current use is via fourcc)*.

- Proposal of a tagging extension to Matroska based on the requirements of the digital preservation community.

- Feedback for features and functions of FFV1 version 4, which is currently under development.

- Creation of metadata translators to convert common descriptive metadata formats within memory institution. For instance convert EBUCore into the XML representation of the Matroska tagging specification so that such metadata may be easily imported and exported between EBUCore and Matroska.

- fourcc is AVI-style with some bitstreams not having a clear license due to coming from Microsoft

Sustainable Open Source Business Ecosystem

MediaArea has long been an open source native and has an open source business model based on sponsored support (bug correction and feature requests), application support, and branched customization based on an institution's specific needs since 2007. Previously existing in a non-business capacity since 2002.

MediaArea's long term goal is to merge previous open source standalone products designed specifically for broadcasting and memory institutions into its flagship product, MediaInfo. These products include the WAV

implementation checker, professional metadata editor and fixer BWF MetaEdit; the AVI implementation checker, professional metadata editor and fixer AVI MetaEdit; and the baseband analyzer for quality assurance, QCTools. Each piece of aforementioned software, designed by MediaArea, has a strong focus on individual areas of digital preservation based on the specific sponsor's needs. Thanks to our discussions with memory institutions, we strongly believe that an integrated environment for conformance checking is sorely needed in the field. By sponsoring the Matroska/FFV1/LPCM + shell/Implementation Checker/Policy Checker/Reporter/Metadata fixer parts of this project, Preforma plays a major role in the creation of a fully integrated and open source implementation checker.

MediaArea plans to build this stable, integrated solution over the course of the Preforma project phase, which will include the current team investigations of Matroska, FFV1, and LPCM, as well as other Preforma investigations such as TIFF and JPEG-2000. This will ensure that proper feedback from Preforma developers and stakeholders is provided in a meaningful timeframe. After the Preforma project is completed, MediaArea anticipates offering access to an integrated solution in two ways: as a ready-to-use environment with a subscription business model (SaaS), and as a ready-to-download version of the integrated solution. This is based on MediaArea's future business model, which consists of a combination of subscriptions and paid punctual support, such as bug corrections and new feature requests. With this long term business model approach in mind, MediaArea will be able to continue offering a Preforma-specific version, free of non-Preforma related layers, as a subset of our own integrated solution.

Project Management Strategy

Goal:

To ensure a vital project, the MediaArea.net team will track processes through an open issue tracker, allowing for consistent and detailed reports with an emphasis on feedback and transparent communication throughout various iterations of the project.

Method:

It is through daily task distribution, management and reflection that project advancements, as well as risks, are addressed. Such open, community-based interaction in the management and implementation of the project allows evolution to occur in all components of the endeavor to include: management, administration and documentation with the allowance for other opportunities for discussion and change to emerge throughout the whole of the project. In the process, the MediaArea team will provide detailed project reports that encourage open, constructive feedback during the testing process that will shape and influence discussions during project meetings.

Justification/Purpose:

Such assessments, occurring daily, aim to build a sense of community that can freely and continually address all deviations and uncertainties, highlighting changes and assessing the benefits, as well as problems with each shift in the projects testing and implementation. With daily communication, evaluations, critiques and suggestions can be addressed with quickly and efficiently.

Intended Result:

This approach to risk management is well-established within The MediaArea team's previous work, with openness between all involved being strongly encouraged. The MediaArea hopes for and invites all relevant communities involvement in the strategy and management of the project, developing and including use of public

tools (github.com, wikis, forums, etc.) and other communication during the planning and implementation of the project timeline. Such interactions and testing of software are welcome at any point during the implementation stage. These engagements will help immensely in addressing and evaluation of priorities and outcomes of the project. Through a process of constant assessment and open access the MediaArea team can respond to changes in software and address performance and usability of the software in its various forms.

Risk Analysis Model

- Define Risk
- Establish date of risk
- Define risk (factors, causes and possible effects)
- Include other contextual information related to risk
- Note individual who discovered risk
- Establish risks probability and impact
- Hypothesize time when risk could occur
- Hypothesize the impact of risk
- Prioritize risk based on established effects of risk
- Classify risk
- Establish point-person for addressing risk
- Create method to reduce likelihood of risk or avoid risk entirely
- Establish alternative plan should risk be unavoidable
- Mark last emergence of possible risk
- Declare risk concluded (Responsibility of Project Manager)
- Note date of the ending of risk
- Contextualize in writing
- Share with pertinent parties ** status and date when the status was last recorded, ** person who accepted the risk, e.g. project manager, ** conclusion date and ** reason for conclusion.

Internal Risk Assessment

Internal Management: To assure the flow of the project, the team will establish a team leader, as well as a junior leader, for the major components of the project. In the case of any event in which the team leader is unable to retain their duties, the junior leader will take their place. Though established as a junior leader, the leaders will function as joint team leaders through the entirety of the project, barring the stepping down of either team leader for unforeseen reasons.

In the event of a major change to staff (whether through avoidance of risk to project vitality, or external life events), the remaining team members will communicate about the best method for reallocation of resources to either replace lost staff, or consider changes in distribution of work among the remaining members. All changes will occur openly, with the participation upon relevant groups and stakeholders. In the event of a drastic shift in leadership, the newly appointed person will affirm the roles of leadership newly assigned to them and work to adhere to the standards of the original contract. If changes to the contract are required, alterations will also occur openly, with the participation of relevant groups and stakeholders.

Addressing Unrealistic Schedule   In the event that the project appears as though it will move beyond the original scope of time, the team will communicate and establish the reasons for the delay. If these reasons can be addressed and changed to fall within the original schedule necessary changes will be made. Such changes and discussions will remain open to relevant stakeholders. If the team communicates and changes cannot be made to address the deadline, then a decision will occur within the team as to the plausibility of establishing a new schedule. This discussion will remain open and include all pertinent parties. If it is found

that a new schedule can be implemented such changes will be made, at which point all contracts, funding schedules and likewise paperwork will be altered.

In turn, should the team complete work before the established date a meeting will occur to discuss what additional work might remain, or how the deliverable products might be further improved. If such improvements are established, the team will continue work within the schedule time. Such decisions will include input from all communities involved. Alternatively, should the team find that their work is at the point of ideal completion, they will meet to discuss how best to end before the scheduled date. This discussion will include all relevant stakeholders, with changes to contracts, financing and other factors included as necessary.

Addressing Unrealistic Budget: Based from the experience of MediaArea's recent software developments projects of similar scale to PreForma we are confident that our budget, objectives, and allocation of resources are appropriate and realistic.

In the event that the project appears as though it will exceed the proposed budget, the team will communicate and establish the reason for the change in financial expectations. If these issues can be addressed and changed to stay within the original budgeted amount, such changes will be implemented. The decisions involving these changes will be shared with pertinent parties. If the team meets and decides that the work will exceed the budget, an establishment of the dollar amount by which the project will exceed its original amount will occur. This amount will be shared with all vital stakeholders.

After the team establishes the amount with which the project will run over budget, the possibility for addressing this funding change will occur, such considerations will include all relevant communities. If the new budget is attainable, work will be redistributed with all contracts, payments and financial elements in need of alteration occurring.

Should the team be unable to attain the necessary funding to continue the project. The deliverables on the project at the point of the cessation will be handed over to pertinent projects. Considerations about continuing the project will be returned to if financing opportunities emerge.

In turn, should the team complete the project under the established budget, a discussion will occur as to how the remaining monetary commitments might be used to improve or expand upon the projects goals. Such discussions will be transparent and include all parties involved. If new budgeting occurs, changes to all necessary contractings and financial paperwork will occur accordingly.

Ceasing of File Format: Should any file format become incompatible the team has the ability to analyze and address any existing format and the changes that occur with the respective format. The project software is designed based on past project experiences, to allow for movement between different formats. The team has already dedicated research and development efforts towards standardizing formats to specifically assist in the archival sustainability of the MKV and FFV1 formats.

Additionally, the software framework can easily be applied to analyze any file format with minimal change. The software is developed so that it is not difficult to swap in other formats and policies, and the MediaArea team has experience doing such with MXF or MOV file formats, should the Matroska format no longer continue development.

Software Incompatibility: Should the software become incompatible due to major operating system updates, API conflicts, or other unforeseeable technical issues, the team has the ability to analyze and address any existing software and the changes that will occur with the respective software. The project software is designed based on past project experiences, to allow for movement between different software structures. For example, if a major operating system such as Windows changes intrinsically and does not allow for the software to run at its basic level, crucial updates will be patched as needed. The modularity of the project software allows for key components to work independent of each other, making it even more unlikely that a system-wide failure should occur based on software compatibility. The software is expressly written in C++ due to its longstanding compatibility standards and optimal portability between systems. Notable changes to software compatibility will be shared with pertinent stakeholders.

Open Source Compliance: Should the supporting software chosen as foundational software for the creation of this project become unavailable due to open source licensing complications, an alternative open source solution will either be chosen to work in its place, or an in-house solution will be developed, depending on if an alternative solution exists or if budget constraints allow for the creation of in-house software components. Timing and budget priorities can be assessed, reviewed, and implemented according to need. All changes will be discussed and considered with the input of relevant communities.

Example of usage in European memory institutions

The National Library of Wales (NLW) has used MediaInfo in their digital audiovisual preservation workflow for several years.

Former Chief Technical Officer of the National Library of Wales said:
"As a National Library incorporating the National Screen and Sound Archive of Wales, we have to preserve digital audiovisual material in perpetuity. Part of this work is characterising AV files and extracting technical metadata. We found no better tool at this job than MediaInfo, and the support and response from MediaArea SARL has always been excellent."

Vicky Phillips, Digital Standard Manager, said: "The National Library of Wales has been using MediaInfo as a technical metadata extractor tool since we started preserving Off air (television and radio) recordings in 2009. From 2009 up until last year we were using an internally developed transformation stylesheet (xslt) in order to map the output from MediaInfo to the Library of Congress audioMD and videoMD metadata standard. This metadata was then stored alongside other administrative and descriptive metadata for that object within a METS document within our Digital Repository. At the time this worked quite well for us, although we had to fix any issues with the xslt when changes were made to MediaInfo output, which caused our transformation to fail or produce errors. However, we did feel that this localised schema mapping implementation wasn't a very sustainable solution. So when we were looking at updating the system which captures our Off-air recordings last year and were looking at reviewing workflows and metadata etc. we decided that this would be a good opportunity to try and get the technical metadata we required for preservation purposes to be output directly from MediaInfo. The Library of Congress audioMD and videoMD were always seen as just a stop gap until other metadata schemas were developed. We therefore started looking at both PBCore and EBUCore. EBUCore's elements seemed a lot more specific than PBCore and being an European standard which is utilised by a number of European projects it was felt that this was the best option for us. Discussion between MediaInfo, EBUCore and myself then commenced in order to produce EBUCore directly from MediaInfo. With the aim being of developing a mapping which is developed and maintained by those who are involved with the software and metadata standard (developers and users). This would then enable the audio visual community to be able to save technical metadata to a globally recognised metadata standard without having to do any mapping themselves.
We also use MediaInfo as a validation tool too (similar to JHove). For example we use it to check that the duration of the digital file isn't empty. If it's empty we know that there is an issue with the file and somebody takes a look at it and sometimes the item is re-processed.
We have been running MediaInfo on MP2-TS files from 2009 and are planning on running it on DPX, ProRes and IMX files very soon."

MediaArea demonstrated his capability to understand the need of the memory institution, to "translate" it in a technical implementation, and to discuss the project with a standardization institution.

Participation at Open Source conferences

The MediaArea team is active in the open source community and has presented the work on Preforma at two conferences during Phase 1:

- Dave Rice presented our work on Preforma at FOSDEM on January 31st https://fosdem.org/2015/schedule/event/enabling
- Ashley Blewer presented our work at Code4Lib on February 11th http://wiki.code4lib.org/2015_Lightning_Talks