MXF Supporting the Integration of Media Applications and Broadcasting Products

Vitor Rodrigues[1], Mário Cordeiro[1], Paula Viana[1,2], José Ruela[1,3]
{vgr,mcordeiro,pviana,jruela}@inescporto.pt

[1]INESC Porto, Campus da FEUP, Rua Dr Roberto Frias 378, Porto, Portugal
[2]ISEP, Rua Dr António Bernardino de Almeida, Porto, Portugal
[3]FEUP, Rua Dr Roberto Frias 378, Porto, Portugal

Abstract

The ASSET project, partially funded by the European IST programme, is defining and developing a universal and unified software architecture that will be made available to manufacturers for allowing easy interfacing between digital television equipment and applications. To guarantee interoperability between manufacturers, a standard to share/exchange content files between broadcast facilities was chosen. This paper gives an overview of the Material eXchange Format (MXF), currently under joint development by the Pro-MPEG Forum and the AAF (Advanced Authoring Format) Association, and describes its use as a content exchange format for the ASSET project.

I - Introduction

The introduction of IT concepts and technologies is opening the possibility for new approaches on the implementation of digital television facilities covering the whole workflow: acquisition, creation, editing, control, storage, broadcasting, publishing and archiving of digital TV content.

This approach has not yet solved the problems that Broadcasters and System Integrators face due to lack of connectivity and interoperability between equipment and applications: solutions available in the market are still vertically integrated or proprietary, requiring specific and costly development, relying typically on a single manufacturer or system integrator.

The main goal of the ASSET project [1] (IST-2001-37379 Architectural Solutions for Services Enhancing digital Television) is to overcome these problems by creating a universal and unified system architecture that will provide a set of software tools and APIs that shall make the integration of different systems independent from the device manufacturer, programming language and underlying middleware platform.

The project is exploiting open standards and emerging technologies (like MXF [3], standard data models for describing essence, XML [4] and distributed systems technologies) for defining the concept of an Asset Middleware that will wrap the standard software layered architecture into a software middleware, which proposes:

- The abstraction of broadcast software and hardware devices as logical resources;
- Generic, openly defined and simple interfaces to control devices and data distribution;
- Added value for system logic: decisions to configure the devices and to convert/move data.

II - ASSET System Architecture

The ASSET architecture [2] is based upon a software Framework - the *ASSET Framework* - composed by a set of standard interfaces and protocols for applications and products working together in an integrated environment.

Products from different manufacturers can be natively connected to the ASSET Framework or through a software adaptor – the ASSET agent or ASSET proxy – enabling their control and management by any ASSET compliant application connected to the Framework.

The ASSET architecture is implemented through a software bus called the Media ASSET Bus or MAB. The goal of the MAB is to provide support for integrating the widest range of applications and products within the ASSET framework, independently from the underlying protocols or the operating system environment.

III - Content Exchange

The requirement to share/exchange content files between broadcast facilities, using non-proprietary formats, has been emphasised by user organizations like EBU (European Broadcasting Union). Aspects like multiple users to simultaneously and independently access the same content; various and adaptable speed transfer across Local and/or Wide Area Networks; common "container" for data, metadata and essence organised in data models; simple and direct access to the content through standardised network protocols and interfaces and unified formats for manipulating, managing, sharing, storing and distributing essences and metadata are priorities that must be satisfied by emerging content exchange technologies.

The Pro-MPEG Forum [5] and the Advanced Authoring Format (AAF) Association [6] developed an open standard that ensures the interoperability among the different vendor systems involved in production environments.

IV - MXF - MATERIAL EXCHANGE FORMAT

MXF [7][3][11] has been designed to meet user demands. It is being put forward as an Open Standard that is not compression-scheme-specific, simplifying the integration of systems using MPEG and DV as well as future, yet unspecified, compression strategies. Transportation of these different files is then independent of content, and does not dictate the use of specific manufacturers' equipment. Any required processing can simply be achieved by automatically invoking the appropriate hardware or software codec.

An MXF file is a "wrapper" for containing audiovisual material in a playable format. Each file contains a comprehensive metadata structure together with component parts that enable MXF files to be written and read directly also by AAF-compliant tools. The entire data structure is based on approved SMPTE standards relating to metadata coding using the KLV (Key Length Value) syntax [8] and standardized metadata dictionary items for interchange [9].

File Structure

The MXF file structure follows the usual scheme of many other file formats, having the basic header, body and footer components as shown in Figure 1.

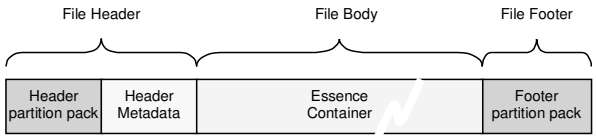

Figure 1 - Core Components of a simple MXF file

The File Header, which provides information about the file as a whole, must always be present at the start of every MXF file. It includes an optional Run-In, a Header Partition Pack, Header Metadata and optionally an Index Table. At least the first part of the header must be consistent for all implementations.

The File Body provides a mechanism for embedding essence data within MXF files and may have one or more Essence Containers, multiplexed using partitions. The associated MXF Essence Container specifications define how each essence element in the container will be KLV encoded and the format of the Essence Descriptors that are required to describe each essence element.

The *File Footer,* located at the end of the file, includes a Footer Partition Pack, an optional repetition of the Header Metadata sets and an optional Random Index Pack. It may also include optional Index Table Segments.

Header Metadata

The *Header Metadata* is split into two categories: Structural and Descriptive. Both categories must be a sequence of KLV coded packets.

The MXF specification defines the *Structural Metadata* packages and sets as a single scheme. There shall be no other Structural Metadata schemes in MXF. The Structural Metadata scheme must occupy the first part of the Header Metadata and defines the capabilities of the file and how it is constructed.

Any *Descriptive Metadata* is defined as a 'plug-in' in order to accommodate one or more Descriptive Metadata Schemes. Where present, any Descriptive Metadata Scheme will use the plug-in mechanism provided by the Structural Metadata Scheme. The Descriptive Metadata is used to define various editorial aspects of the file, for example Production and Clip Information.

Essence Containers

The MXF Generic Container is the native Essence Container of the MXF File Body. The MXF Generic Container is a streamable data container that can be placed on any suitable transport and potentially stored. The concept of this container was based on the work done by the EBU/SMPTE Task Force in the Wrappers and Metadata sub-group [10].

The premise for the MXF Generic Container format is that of a general-purpose essence data and metadata container of many different kinds of essence and metadata elements into a single entity by interleaving the data streams in a defined and time-synchronous manner.

Essence container specifications are written as a plug-in to allow the inclusion of any standardized container. They can be based on one of several basic types, including MPEG, DV and uncompressed.

Partitions

An MXF File may be divided into a number of *Partitions* that logically divide the file to allow easier parsing, to help streaming and to manage the creation of Index Tables (which, in turn, make random access in a storage system easier). An MXF file may contain many different Essence Containers and partitions help managing them.

The *Header Partition* must be located at the start of the file and starts with a Header Partition Pack, followed by the Header Metadata, optional Index Table segments and optionally by the whole, or the first part, of an Essence Container.

If a file has *Body Partitions*, then each shall comprise a Body Partition Pack followed optionally by a repeat of the Header Metadata, optional Index Table segments and optionally by a whole or part of an Essence Container.

When present, a *Footer Partition* must be located at the end of the file and must comprise a Footer Partition Pack followed optionally by the Header Metadata, and optionally by Index Table segments. An *Index Table* is an optional component that allows quick access to any component in an essence container through a byte offset value from a defined address. The Footer Partition may optionally be followed by a Random Index Pack.

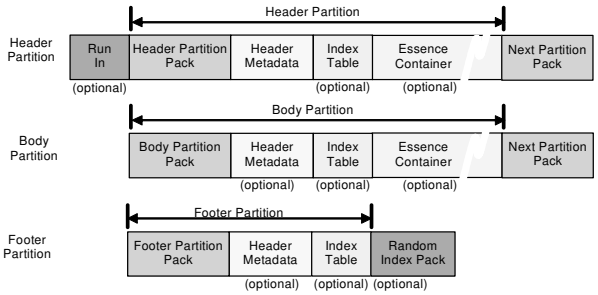

Figure 2- Required order of file components in each partition type

Operational Patterns

To manage the complexity of MXF, Operational Patterns (OP) have been defined to limit the features which can be used in different applications, thus allowing software and hardware developers to create products with different levels of functionality. It can be seen as a grid that is divided vertically depending on the timeline complexity within the file and horizontally depending on the number of different packages within the file.

For timeline complexity (Item Complexity) the following items are considered:

1. Single Item – the file contains only one item;
2. Play List Items – several items that are butted one against the other;
3. Edit Items – the file contains several items with one or more cut edits.

For the number of different packages (Package Complexity) the following option are considered:

a. Single Package – the Material Package can only access a single Source Package at a time;
b. Ganged Packages – the Material Package can access one or more Source Packages at a time;
c. Alternate Packages – two or more Material Packages, which can access one or more File Packages at a time.

Item complexity and Package Complexity can be chosen to create the desired Operation Pattern, e.g. OP 2.b - Play List Items with Ganged Packages.

V - MXF IN ASSET FRAMEWORK

According to the ASSET Framework, content exchange (audio-visual material and associated metadata) relies on a file format that provides full interoperability between different equipment and applications. The exchange format should be open and standardized, compression-independent, cross-platform and support streaming/transfer bridging.

Due to its characteristics, MXF is being used as the ASSET solution for Content Exchange. Two different scenarios were defined:

- the native interchange file within the same ASSET framework;
- the import/export of both Essence and Metadata between different ASSET architecture based systems or simply external ASSET MXF compliant products.

The difference between these two approaches does not reside in the file format being used (both use MXF) but in the content of the MXF file:

- for the Exchange between ASSET compliant products connected to the same ASSET framework, only the essence and the structural metadata will be included in the file. Associated descriptive metadata are indeed stored and shared by both products, in one (or eventually several) common Media Asset Management system(s).
- for Importing/exporting to/from outside ASSET, descriptive metadata can be included in the MXF file. The import process must extract the descriptive metadata and update Media Asset Management system(s). The export process must build MXF files with both structural and descriptive metadata (available from the from a Media Asset Management system(s)) allowing external applications and systems to access essence and metadata (structural and descriptive) in just one file transfer.

MXF Import/Export module

As referred to above, all content exchanges in the ASSET framework are made always using MXF files. All ASSET compliant products provide MXF capabilities for import and export of MXF files. In some third party applications these MXF functionalities might not be present, so it is necessary to provide external MXF functionality to those products.

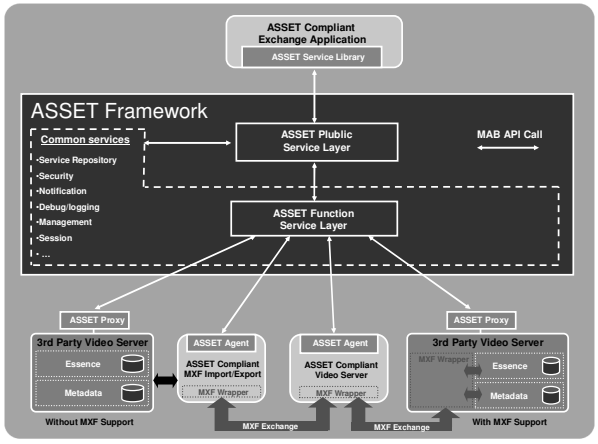

Figure 3 - Integration of third party products using the MXF Import/Export module

To provide the adaptation to the framework of third party products, ASSET provides external adaptors (ASSET proxies) that make the mapping between proprietary API's and the ASSET generic API. This concept is valid only for command and control adaptations. The content exchange adaptation using MXF support and compatible ASSET transfer capabilities to the third party products can be provided either inside the products, implemented internally by product manufacturers, or externally, using an additional

ASSET framework module, the MXF Import/Export. This module allows products or applications to integrate MXF content handling and transport without a previous knowledge of how to do that inside the system.

Using both ASSET proxies for adaptation of internal product API and the MXF Import/Export module for MXF and content exchange adaptation, third party products or applications (MXF compliant or not) can be easily integrated into an ASSET framework based systems (Figure 3).

MXF Import/Export XML based API

The ASSET MXF Import/Export module uses the XML based Message Exchange Format (MEF) defined for providing a common way for adapter implementations to describe their function and parameters (Figure 4). Different message types were defined to allow, for example, registering, initialising, processing a command, etc.

```
<mab-message type="type of message">
  <mab-header>
    <!– Generic header information –->
  </mab-header>
  <mab-data>
    <!– The message data comes here –->
  </mab-data>
</mab-message>
```

Figure 4 – Generic MAB Message Structure

Requesting the MXF Import/Export services requires a message type "executeCmd" as illustrated in Figure 5. The body part (Figure 6) identifies the metadata and essence repository as well as the final destination of the MXF file.

```
<mab-message type="executeCmd">
  <mab-header>
    <mab-header-block>
      <header name="session-id" value="8CE983D392"/>
    </mab-header-block>
    <mab-header-block>
      <header name="security" value="user1/xxxx"/>
    </mab-header-block>
  </mab-header>
</mab-message>
```

Figure 5 – ExecuteCmd header message

```
<mab-data>
  <export service="MXFImportExport">
    <input>
      <metadata>
        <descriptive>
          <file>
//metadata_common_service/demo/descrip_metadata.xml
          </file>
        </descriptive>
      </metadata>
      <essence>
        <url>
ftp://user:xx@essence_repository:21/demo/essence.dv
        </url>
      </essence>
    </input>
    <output>
        <mxf>
          <file>
//localhost/mxf_repository/demo_clip.mxf
          </file>
        </mxf>
      </output>
    </export>
  </mab-data>
</mab-message>
```

Figure 6 – MXF Import/Export XML body message

CONCLUSIONS

This paper gives an overview of the project approach and describes the work under development in the IST ASSET project. The basic architecture is outlined and the content exchange approach is detailed. An MXF introduction is presented to illustrate the importance for the definition of common file formats in digital TV environments. The application of the ASSET concepts is illustrated with the presentation of an ASSET compliant MXF Import/Export module.

The project has already defined the software architecture, concepts and demonstration scenario and partners are now working towards the development of the prototype that shall demonstrate the effectiveness of the ASSET solution.

REFERENCES

[1] ASSET web site – http://www.ist-asset.com

[2] Paula Viana et all, "A Unified Solutions for the Integration of Media Applications and Products in Broadcaster Environments – The ASSET Architecture", Proceedings of NAB2003, USA, April 2003

[3] Pro-MPEG Forum, Material eXchange Format (MXF) 10b, http://www.pro-mpeg.org/mxf.htm, Oct. 16, 2002

[4] W3C, Extensible Markup Language (XML) 1.0, http://www.w3c.org/TR/REC-xml, Oct. 6, 2000

[5] Pro-MPEG Forum site - http://www.pro-mpeg.org

[6] AFF Association site – http://www.aafassociation.org

[7] Bruce Devlin, "MXF - the Material exchange Format", EBU Technical Review, July 2002

[8] SMPTE 336M-2000, "Television - Data Encoding Protocol Using Key-Length-Value", 2000

[9] SMPTE RP210-2001, "Metadata Dictionary Contents", 2001

[10] EBU/SMPTE, "EBU/SMPTE Task Force for Harmonized Standards for the Exchange of Program Material as Bit-streams", Final Report, Sept 1998

[11] Jim Wilkinson and Bruce Devlin, "The Material Exchange Format (MXF) and its Application", SMPTE Journal, Sept 2002

ACKNOWLEDGEMENTS

The authors would like to thank the support of the European IST programme and partners on the ASSET project: HP, THOMSON, Dalet a.n.n, INESC Porto, INRIA, IRT, FPDI, SHS Multimedia.