MXF — a progress report

P. Ferreira
MOG Solutions

The Material eXchange Format (MXF) is arguably the most successful file format in the broadcasting industry. Six years after its ratification as a SMPTE standard, this article descibes its roots, its main goals and its current adoption by the TV broadcasting industry.

This is the first of two articles on MXF and will be followed by an in-depth technical discussion on the format.

The origins of MXF

The origins of MXF can be traced back to the mid 1990s, when convergence between the IT and TV industries was taken for granted. The increasing computing power, faster networks, lower storage costs and more capable video-compression technology paved the way for the use of computer technology in the demanding world of Television. Non-linear editing was becoming commonplace, video servers were starting to be accepted as a reliable replacement for VTRs and, slowly but steadily, blue-screen-clad computers were being turned into reliable, flexible and highly efficient devices, either individually or embedded in specialist equipment.

As is so often the case, the upheaval didn't happen with a proper standardization backing, but rather based on impromptu solutions devised by manufacturers. This state of affairs led to a vast array of incompatible file formats, ad hoc workflows and different terminology; in short, a completely different scenario from tape-based and real-time streaming media interchange, where SDI had established itself as a reliable, interoperable link and workflows had long settled down.

With a good dose of foresight, the EBU and SMPTE joined forces to perform a seminal work aimed at *"producing a blueprint for the implementation of the new technologies looking forward a decade or more"* [1]. The outcome was a pair of documents: a report on user requirements and a detailed collection of the analyses and results attained by the task force [2].

This work, which has had a major impact on the industry, was full of relevant results, including the awareness of the blatant need to perform formal standardization work on Wrappers and Metadata. Metadata was finally glorified as a first-class citizen, on a par with audio and video, while a standard wrapper was pointed out as playing a fundamental hinge role in interoperability.

Why wrappers?

Wrappers serve two purposes mainly: (i) to gather programme material and related information, as well as (ii) to identify those pieces of information. In fact, a wrapper does not add much value per se – by the same token that a gift wrapper is not meant to take the part of the gift itself – but is invaluable in providing a unified way to structure content and to access it.

While some wrapper formats were in use at the time, none was widely accepted or provided the adequate characteristics (in terms of openness, extensibility, performance, etc.), and hence a movement was started to coalesce the requirements and ideas, in order to prepare a standardization process. The Pro-MPEG Forum [3] was the major backer of that effort, bringing together a strong list of vendors, end users and research institutions. They collaborated to produce a set of documents that drafted the Material eXchange Format (MXF) and which were eventually submitted to SMPTE for a formal standardization process.

This process reached a major milestone when, in 2004, MXF was ratified as SMPTE S377M.

What is MXF?

MXF is an open file format, conceived to act as a wrapper of audio-visual content, including associated data and metadata.

MXF is not a codec format: it will wrap any existing or future coding format, such as MPEG-2, AVC or MP3, in such a way that it allows generic decoders to access files in a consistent, format-agnostic manner. It can also carry uncompressed video or audio (PCM). Furthermore, wrapping the content in MXF does not entail any sort of format conversion: MXF is just a thin wrapper around the original encoding format. What does this mean? Unlike transcoding between, say MPEG and DV, the image quality is not altered at all, allowing the same content to be wrapped with various MXF flavours without the slightest impairment in quality.

MXF also defines ways of carrying ancillary data, such as timecode or closed-captions, as well as any metadata scheme. This is one of its major achievements as, for the first time, we have a standard way of bundling together all components of a programme, making it simple to interchange rich content.

MXF is extremely versatile, being useful for such distinct applications as storage of finalised works, just like a tape; streaming, such as edit-while-ingest; creation of cuts-only EDLs bundled with the source material or the definition of playlists/EDLs that point to external files, for the simple creation of multi-language versions of a programme.

Finally, MXF is efficient, as it was designed with a focus on maximizing the flexibility without compromising the efficiency. It defines several tools that can be used by vendors to construct files in ways that are optimized to the task in hand.

Quick technical overview

MXF is ultimately a collection of Packages of two kinds: *File Packages*, which represent the source material; and *Material Packages*, representing the desired output timeline. For example, if you want to compose a clip out of a pair of recordings, your MXF file will have two File Packages and a single Material Package, which instructs the decoder to play the two packages in sequence. Each of these packages is identified by a UMID [4].

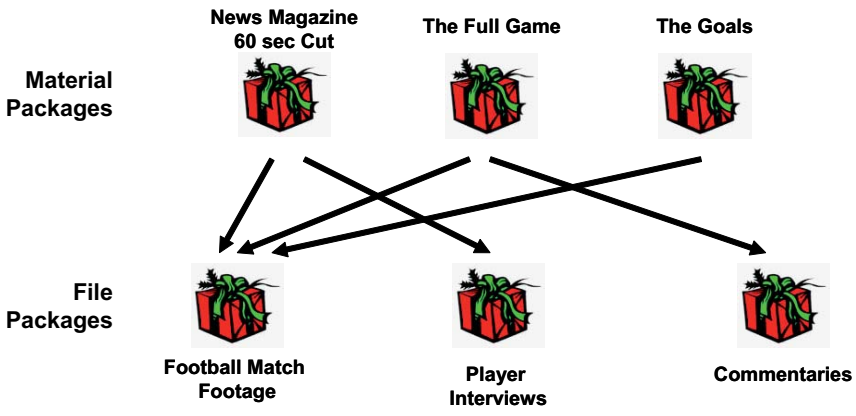

Figure 1
Kinds of packages

This concept of Packages is part of MXF's Structural Metadata (SM). SM is key in MXF, as not only does it define the output timeline, but also it provides fundamental information such as technical charac-

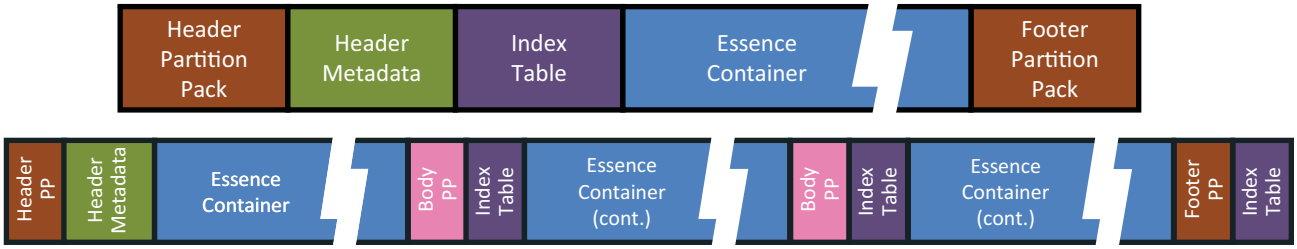

Figure 2
(Upper) **Structure of a basic MXF file.** *(Lower)* **Structure of a multi-partition MXF file**

teristics of the media, identification of the packages, hooks to insert Descriptive Metadata, etc. This metadata is transported in the MXF file's header and can be repeated throughout the files, possibly with updates.

This brings us to the structure of an MXF file *(Fig. 2)*. MXF files are divided into at least two partitions, namely the Header and the Footer. Additional partitions can be inserted in the middle of the file and are called Body Partitions. This is the macro level.

At the micro level, MXF is composed of Key Length Values (KLVs) [5]. A KLV is a triplet composed of a key which identifies what comes next, a length and the actual data. This mechanism is the cornerstone of MXF's extensibility, allowing a decoder to skip data it does not understand.

And what about Essence? MXF carries it inside Essence Containers, which are a structured way of organizing the picture, audio and data elements. Essence Containers can be frame-wrapped, clip-wrapped or custom-wrapped. The first kind wraps each frame and corresponding audio inside its own KLV. The second lumps all the samples inside a single KLV. The latter is used scarcely and is application-defined.

The last major component of an MXF file is the Index Table, which is invaluable to facilitate random access into any frame of the file. Index tables are basically a way to map a frame number into a byte offset and are therefore quite dependent on the essence format, namely whether frames have constant or variable sizes. For example, while an index table for a DV clip is straightforward, one for Long GoP MPEG must index each frame independently.

Controlling complexity

Just like most "large" standards, MXF leaves some room for manufacturers to "play with". Although this is a good thing, it usually requires that users and manufacturers agree on functionality subsets; otherwise it is too difficult to achieve interoperability. One good example is the definition of MPEG Operating Points that were made for D-10/IMX.

In MXF *(Fig. 3)*, this constriction can be made by using Operational Patterns (OP). MXF defines the Generalized Operational Patterns, OP1a up to OP3c. These OPs constrain the complexity of a MXF file by controlling the relationship

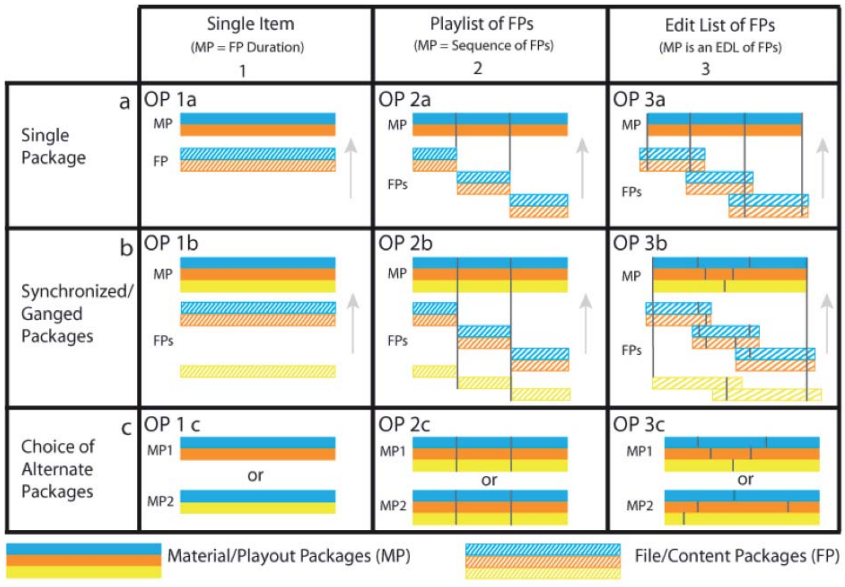

Figure 3
Generalized operational patterns

between File and Material Packages.

For example, an OP1a file has a single File Package and a single Material Package that "points" to the whole File Package. That is, the result of playing back the file is the same of playing all the essence it contains, from the beginning to the end. Another common OP is OPAtom, which requires that an MXF file contains essence of a single track, video, audio or data.

Structure of the standard documents

With extensibility being one of the basic tenets of MXF, it would be unwise to standardize it as a single document, as any extension would require that the standard had to be revised. Therefore, it was decided to split MXF into an open set of documents, allowing evolution to be achieved by defining new standard extensions to MXF.

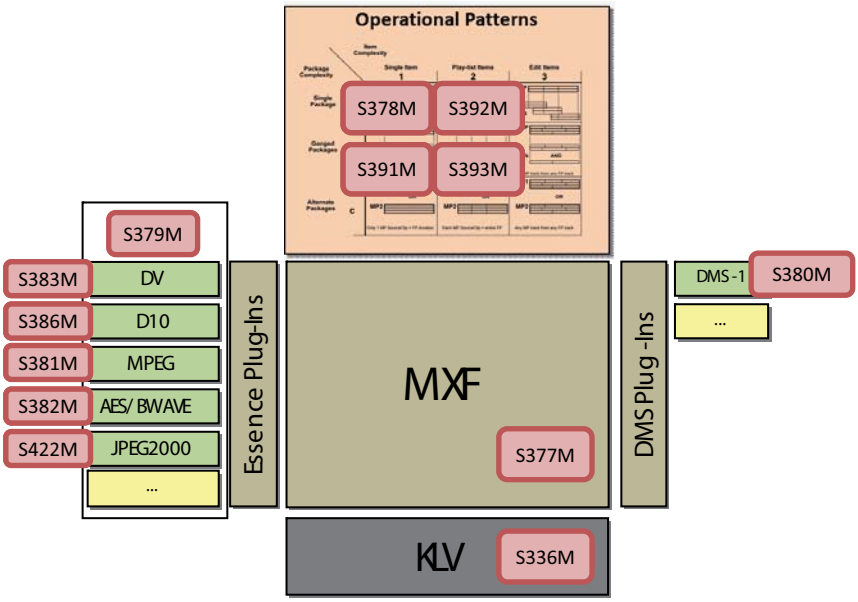

Figure 4
Extensibility via new standards

The core of the MXF standard is the File Format document, S377M [6]. It is by far the most important document, as it defines all common aspects of MXF files, irrespective of the essence format or metadata schemes they carry, as well as any constraints imposed by operational patterns.

Then, there are three main kinds of documents:

- Operational Patterns, e.g. OP1a [7];
- Descriptive Metadata plug-ins, e.g. DMS-1 [8];
- Essence Containers and Essence Container Mappings, e.g. MXF Generic Container [9] and Mapping of MPEG to the Generic Container [10].

Finally, there is a set of related documents, for example, the mapping of MXF Metadata to XML [11].

By definition, this set is live: new documents are defined as required, for instance to support new Essence formats. An up-to-date list of all relevant documents can be found on the IRT's MXF site [12].

Filling in the gaps

Operational Patterns are usually enough to constrain complexity but, on some occasions, other approaches work better or are needed in order to clarify some points.

For example, some manufacturers publicly disclose the exact format of the MXF files they generate or accept, as did Sony and Panasonic via SMPTE [13][14][15]. Another example is EBU R122 [16], which is the outcome of an important work aimed at harmonizing the representation of timecode in MXF files. Finally, the Advanced Media Workflow Association is doing a very relevant work on Application Specifications, namely "AS-02 – MXF Versioning Application Specification", which intends to create a "multi-lingual, multi-versioned, web services friendly constrained version of MXF

for use in facilities and systems" [17].

MXF adoption

The successful adoption of MXF as the wrapper of choice for the broadcasting industry has been anticipated, given the strong support from both manufacturers and end-users. In fact, as early as in 2003, even before the standard was published, the first MXF-based software and hardware products were released: MOG Solutions MXF::SDK® and Sony e-VTR®. The latter was an important bridge from the tape-based to the file-based paradigm, at the same time demonstrating actual commitment from one of the major vendors; the former gave all players in the industry a simpler path to integrate MXF support into their products. This sent a strong message both to manufacturers and end-users; soon after, several manufacturers declared their commitment to MXF or even announced MXF-aware products.

MXF was quickly adopted by most camera manufacturers, which inevitably led to increasing support from editor manufacturers. The snowball effect continued, joined by transcoding engines, video servers, MAM systems, etc. Today, MXF is pervasive and well supported by most manufacturers and by all sorts of software and equipment.

Major applications

As stated above, MXF is used in a broad range of applications, essentially covering all steps of the workflows taking place in broadcasting environments. The next sections will highlight some of these applications and discuss which MXF variants are most used.

Cameras

Most professional cameras record content in MXF. As the change from tape- to file-based took place, manufacturers selected different kinds of storage media and different variants of MXF. The choice of the MXF variant was based mainly on two aspects: the integration with the post-production processes and the characteristics of the storage media – for instance, optical media does not perform well when accessing multiple files at the same time.

The choices, in terms of MXF flavours, basically fell into one of two camps: single-file-per-clip, frame-wrapped OP1a; multiple-files-per-clip, clip-wrapped OPAtom.

Abbreviations

AAF	Advanced Authoring Format
AMWA	Advanced Media Workflow Association http://www.amwa.tv/
ANC	ANCillary data
AVC	(MPEG-4) Advanced Video Coding, part 10 (aka H.264)
DM	Descriptive Metadata
DMS	Descriptive Metadata Scheme
DV	(Sony) Digital Video compression format
EDL	Edit Decision List
GoP	Group of Pictures
GPS	Global Positioning System
IRT	*Institut für Rundfunktechnik GmbH* (German broadcast technology research centre) http://www.irt.de/
IT	Information Technology
KLV	(SMPTE) Key Length Value
MAM	Media Asset Management
MPEG	Moving Picture Experts Group http://www.chiariglione.org/mpeg/
MXF	Material eXchange Format
NATO	North Atlantic Treaty Organization
OP	(MXF) Operational Pattern
PCM	Pulse Code Modulation
SM	(MXF) Structural Metadata
SMPTE	Society of Motion Picture and Television Engineers (USA) http://www.smpte.org/
UMID	(SMPTE) Unique Material Identifier
VBI	Vertical Blanking Interval
VTR	Video Tape Recorder
XML	eXtensible Markup Language

The first choice has the advantages of cohesion, no need for cross-referencing among files and single-file access. The second choice was deemed more suitable for post-production, as accessing audio media does not require unwrapping an interleaved MXF file.

Most of these files have a very simple structure, with two or three partitions and simple index tables. The lone exception is when MPEG Long GoP is used: since each frame must be indexed independently, the index table is not known beforehand. Thus, the index segments are flushed periodically and a new partition is created each time.

Timecode is typically handled by recording the start timecode both in the File Package and in the Material Package. Some frame-wrapped formats include system items with timecode.

There has been increasing interest in adding metadata to the MXF files, both static like titles, and dynamic such as GPS coordinates or camera parameters.

Editors

All major video editors are able to import the MXF files created by professional cameras. Most of them edit natively in the original media or, at least, without requiring a conversion to another wrapper.

Some editors are able to edit directly in "generic" MXF files, that is, files that are not specific to some vendor.

Editors which integrate tightly with storage systems typically import media into an OPAtom-based MXF format, thus simplifying the access to separate tracks.

Video Servers and Capture Systems

The majority of video servers has an excellent support for MXF, mostly OP1a and OPAtom, but also increasingly AS-02. Most of them support several codecs wrapped in MXF, irrespective of the actual MXF variant, and can create and play back files which are compatible with those generated by the major cameras.

Some servers support the creation of files that can be edited or played by external devices during the recording.

Timecode is usually recorded in system items and the support for VBI and ANC packets, typically as data items in frame-wrapped files, is becoming commonplace.

Archive Systems and Media Asset Management Systems

There is increasing support for MXF, especially the variants defined by the major cameras. Some systems support "generic" MXF files.

One of the most important operations in an archive is the extraction of subclips, also known as partial restore. MXF gives excellent support for such a feature and most systems support it.

Other industries

MXF has found usage outside of the broadcast industry. One such example is NATO, which standardized the use of MXF for surveillance applications. Another one, arguably the most famous, is Digital Cinema: in 2005, Digital Cinema Initiatives, a consortium comprising Hollywood's major studios, released the first version of their "Digital Cinema System Specification". In conjunction with several SMPTE standards, it proclaims MXF as the wrapper of choice for distribution of digital cinema.

Interoperability

Today, MXF is pervasive, being used everywhere: production, post-production, archive, play-out, mostly with seamless interoperability. However, until recently, MXF was gaining some reputation as a failed attempt for interoperability. Why was that?

It is well-known that most successful new technologies endure three phases throughout their life cycle *(Fig. 5)*.

We are now clearly into the "Real Work" phase. However, during the "Euphoria" phase, most people assumed that the mere fact of having a file with a ".mxf" extension would mean it would work in any MXF-compliant equipment. As soon as people started to find out that this was not quite true, we quickly slid into "Disillusionment". Why was MXF not always MXF? There are many reasons for that, some accidental, some deliberate.

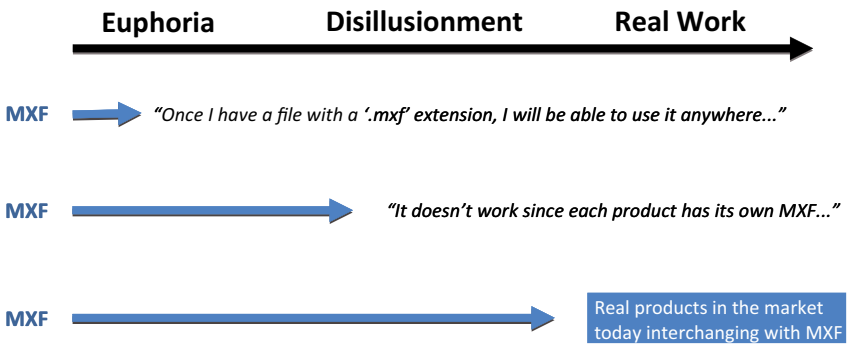

Figure 5
The three phases of a new technology

The accidental reasons are simple to explain: MXF is quite a complex standard and many of its concepts were quite new for the industry. The sheer complexity of the undertaking inevitably led to some lack of precision in the language of the standard, some corner cases that had not been foreseen, some workflows that had not been properly addressed, some implementations that were not flawless, etc. All of this required some time to settle down but, thanks to the excellent collaboration among manufacturers, end users and institutions like the EBU and SMPTE, most of these problems have been addressed and were ratified in the 2009 version of the standard.

The other major impairments to interoperability that most people stumble upon are actually design decisions. Let's see why.

One of those problems is quite obvious: MXF is not an encoding format, rather a wrapper around existing formats. Therefore, an MXF file containing AVC essence will not be validated by a decoder that, although perfectly capable of reading MXF, only understands MPEG-2. This one is pretty much accepted by everyone. What usually causes more complaints is the fact that there are many "MXF flavours". But what is an MXF flavour?

MXF tries to address such a large set of requirements and to be useful in such a broad range of workflows that, right from the start, it was decided that there had to be a means of constraining MXF into simpler profiles. The MXF moniker for profile is "Operational Pattern". It does make a lot of sense: MXF allows one to define very complex EDLs and various ways of interleaving or multiplexing essence inside a file. Would it be cost-effective and efficient to require all decoders to support that? Certainly not. That was the right way to go but, unfortunately, most incompatibility problems we find today are caused by the use of different operational patterns.

Planning for interoperability

Obviously, the best way to achieve interoperability in a facility is to always use the same codec and MXF flavour but, of course, this is not possible most of the time. What can then be done in order to minimize interfacing problems? One should always bear in mind that: transcoding is slow and impairs image and audio quality; MXF rewrapping is fast and does not change the essence. Therefore, as discussed in [18], one should define "islands" where formats are well defined. Then, when transferring media into those "islands", it should be rewrapped into the appropriate container.

For example, when ingesting material from cameras, one can rewrap to the required container, on-the-fly. Or, when exporting from an editing system, one can prepare the MXF container for the playout server. Is this ideal? No, but certainly it is optimal, as there is no better way.

Metadata in MXF

Real-time Metadata

A peculiar kind of metadata is commonly referred to as RT Metadata. A good example is the insertion of camera parameters or even GPS data into the MXF stream. This kind of application is becoming more common and will certainly get a huge boost with stereoscopic 3D, with items such as depth maps being embedded into MXF.

MXF includes mechanisms to convey such kinds of information, interleaving or multiplexing it with video and audio essence in efficient ways.

Descriptive Metadata

Descriptive Metadata (DM) is widely used, not only in the broadcasting industry, but horizontally across several businesses, being an extremely powerful tool for finding content. MXF provides excellent support for DM, being able to carry whatever DM scheme is chosen for an application and providing mechanisms to attach DM both to clips or segments thereof.

Just like with the essence, MXF is a mere container for DM. Hence, the definition of data schemes falls outside its scope. However, SMPTE defined a generic scheme, suitable for most applications in the broadcasting industry, the Descriptive Metadata Scheme-1 [8]. Still, MXF is capable of carrying other DM schemes, such as EBU P/Meta or any other custom scheme, both open and private.

Until now, the use of DM inside MXF containers has not been widespread. Some cameras record titles and similar information as DMS-1, some editors are able to interpret it and, for example, some archive systems are able to partially restore an MXF file with the appropriate cuts in DM. Still, most applications resort to basic key/value pairs or store DM in ancillary XML files.

MXF and other technologies

MXF has a clear focus on the interchange of raw material, and finished or almost finished programmes. Still, there are applications with requirements that go well beyond this scope, as well as applications that have a small intersection with the media, just needing access to the metadata, both technical and descriptive.

MXF has an added bonus, important for these two ends of the complexity spectrum, which is simple integration with AAF and XML.

The promoters of AAF (formerly known as the AAF Foundation, now AMWA [17]) were co-developers of MXF and shared the goal of having a coherent object model for both MXF and AAF. This coherence makes MXF and AAF perfectly complementary, ideal for storing raw media in MXF and complex EDLs in AAF.

As for XML, SMPTE standardized a formal mapping between MXF and XML [11], which makes it straightforward to extract or update metadata in XML, easing the integration with applications that don't require access to the internals of MXF.

Pedro Ferreira was born in 1973, in Guimarães, Portugal. He has an MSc. in Telecommunications and Computer Science from the University of Porto. After graduating, he worked at INESC Porto as a researcher on the use of Distributed Systems technology in Digital Television, collaborating as well in projects such as ACTS ATLANTIC. He also led the Distributed Systems and Essence Processing area in the BBC ORBIT project, was responsible for INESC Porto's participation in some EU IST projects and was engaged in standardization activities in SMPTE and Pro-MPEG.

In 2002, Mr Ferreira left INESC to co-found MOG Solutions, where he became responsible for R&D, leading the development of award-winning products such as MXF::SDK, bespoke projects such as the collaboration with NBC Olympics, as well as the participation in several EU projects, such as Worldscreen and EDCine.

Pedro Ferreira is now Director of Product Marketing and is striving to make MOG's products easier to use and more effective in streamlining file-based workflows. He is also responsible for MOG's training department, collaborating with the EBU and several other customers.

The future of MXF

The SMPTE process requires a revision of each active standard every 5 years. MXF underwent this revision and a new version of the standard was published in 2009.

The lessons learned during the first years of MXF led to some important decisions, particularly to clarify a number of issues and to create simpler profiles for MXF. This new version will certainly improve interoperability in situations where it was previously impaired by different interpretations of the standard.

SMPTE continues addressing the needs raised by new technologies or new workflows. Over the next few years, we will certainly witness new MXF-related standards in areas like 3D, multi-channel audio, new codecs, etc. This constant focus on enlarging the scope of MXF and its inherent flexibility will make sure MXF is here to stay.

References

[1] **The EBU-SMPTE Joint Task Force for Harmonized Standards for the Exchange of Programme Material as Bit Streams – First Report: User Requirements, April 1997**.

[2] T**he EBU-SMPTE Joint Task Force for Harmonized Standards for the Exchange of Programme Material as Bit Streams – Final Report: Analyses and Results, July 1998**.

[3] ProMPEG: **http://www.pro-mpeg.org/**

[4] SMPTE 330M-2004: **Unique Material Identifier (UMID)**.

[5] SMPTE 336M-2004: **Data Encoding Protocol Using Key-Length-Value**.

[6] SMPTE 377-1M-2009: **Material Exchange Format (MXF) – File Format Specification (Standard)**.

[7] SMPTE 378M-2004: **Material Exchange Format (MXF) – Operational pattern 1A (Single Item, Single Package)**.

[8] SMPTE 380M-2004: **Material Exchange Format (MXF) – Descriptive Metadata Scheme-1 (Standard, Dynamic)**.

[9] SMPTE 379-1M-2009: **Material Exchange Format (MXF) – MXF Generic Container**.

[10] SMPTE 381M-2005: **Material Exchange Format (MXF) – Mapping MPEG Streams into the MXF Generic Container**.

[11] SMPTE 434-2006: **Material Exchange Format – XML Encoding for Metadata and File Structure Information**.

[12] IRT/MXF: **http://ftp.irt.de/IRT/mxf/information/specification/index.php**.

[13] SMPTE RDD 9-2006: **MXF Interoperability Specification of Sony MPEG Long GOP Products**.

[14] SMPTE RP 2002-2006: **Content Specification on Solid State Media Card for DV/DV-Based Essence**.

[15] SMPTE RP 2027-2007: **AVC Intra-Frame Coding Specification for SSM Card Applications**.

[16] EBU Recommendation R122: **Material Exchange Format Timecode Implementation**.

[17] AMWA: **http://www.amwa.tv/**.

[18] Pedro Ferreira: **Material eXchange Format Interoperability**
EBU tech-i, Issue 3, March 2010.

This version: 23 July 2010

Published by the European Broadcasting Union, Geneva, Switzerland

ISSN: 1609-1469

Editeur Responsable: Lieven Vermaele

Editor: Mike Meyer

E-mail: **tech@ebu.ch**

The responsibility for views expressed in this article rests solely with the author