UMID Applications in Practices

Yoshiaki Shibata and Jim Wilkinson

SMPTE Mot. Imag. J 2012, 121:58-67.
doi: 10.5594/j18149

UMID Applications in Practices

By Yoshiaki Shibata and Jim Wilkinson

The unique material identifier (UMID) is a SMPTE standard identifier that globally uniquely identifies audiovisual (AV) material. Because it is a core component of the Material Exchange Format and Advanced Authoring Format, it has been also handled by products claiming support for these formats. However, its intended use as a globally unique identifier to link AV material to its metadata has seldom been seen in practice. This paper aims to achieve its original intention by introducing the concept of a UMID-managed domain, where all AV materials are fully managed via their UMIDs, resulting in any AV material being unambiguously retrieved by its UMID. Another important aspect of the UMID-managed domain is that the domains from various products can be merged to produce a wider domain covering the entire system. To achieve this, however, the UMID resolution protocol language spoken by those products needs to be standardized. A couple of basic proposals are presented for further discussions.

INTRODUCTION

The unique material identifier (UMID) is a globally unique identifier for audiovisual (AV) material standardized in SMPTE ST330[1] and its application in SMPTE RP205.[2] More than a decade has passed since its initial standardization, and it has now become a core component of the Material Exchange Format (MXF) and Advanced Authoring Format (AAF) technologies. Thus, products claiming to support MXF or AAF are, by definition, also supposed to handle UMIDs.

According to SMPTE ST330, "The UMID provides a method of identification for instances of audiovisual material and thus enables the material to be linked with its associated metadata." This implies that when metadata associated with a desired AV material are available, the AV material will be obtained by tracking the link to the material. However, because UMID by nature is a dumb number that is not intended to tell anything about where the material is, even though the metadata with the UMID are at hand, you cannot access the material without the assistance of external functions to resolve the UMID into information about where the material is available.

In the media and entertainment (M&E) industry, AV material management by using unique identifiers is a common practice; AV products handling the AV materials within them assign a new identifier to material at its introduction to the products. Because the identifier is based on a proprietary identification scheme, mapping among those identifiers is required for AV material to be shared among products from different manufacturers. De Geyter et al.[3] show a couple of options to achieve this: One is to use the material identifiers in a particular product as a master and to make every other product to store the master identifier in their respective databases. Another is to provide a specialized identifier correlation service within a system, which manages the mappings of various identifiers to keep them up to date.

But a simple question arises: Why don't they use the UMID as a common identifier in the system? Because the UMID is a widely used standard identifier, it is supposed to be commonly used in products, even from different manufacturers. If it is, a system composed of such products would also have to treat the UMID as an identifier of the material within a system. However, the reality is that the originally intended use of UMID as a standard globally identifier has been seldom seen.

In this paper, we propose ways to use the UMID to achieve its original intention. After the UMID application principles are defined, the concept of a UMID-managed domain is introduced to embody the principles, including its applicability not only within a product but also to an entire systems and a distributed environment over the internet. Then, the UMID application in MXF is specifically discussed based on the principle. Because of the characteristics of MXF, the usefulness of UMIDs in an MXF file as more than just unique identifiers is demonstrated. The requirements and plausible solutions for the UMID resolution protocol for further discussions follow.

UMID OVERVIEW

Basic and Extended UMIDs

The UMID is a byte string of either 32 or 64 bytes. The shorter UMID is called the basic UMID, and the longer is the extended UMID, which is composed of a 32-byte basic UMID followed by the 32-byte source pack to store the information on when/where/who originally creates the AV material. The basic UMID is used to globally uniquely identify AV material; the extended UMID is used to identify it with finer granularity, such as by frame.

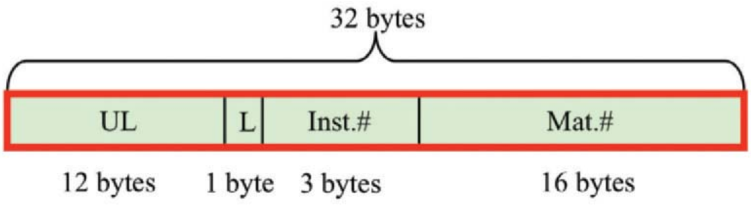

Figure 1. Basic UMID format.

While an application of the extended UMID and/or the source pack is an interesting topic, we do not go into the details in this paper. More information about them is available elsewhere.[4]

Basic UMID Format

The basic UMID is a single 32-byte entity composed of 4 fields, as shown in **Fig. 1**:

- *SMPTE Universal Label (UL)*. The first 12 bytes of the basic UMID constitute the SMPTE UL. The first 10 bytes are fixed values based on the registered International Organization for Standardization label administered by SMPTE.[5] The 11th and 12th bytes indicate the type of material this UMID identifies and the creation methods of values in the fields that follow this label, respectively. SMPTE ULs are 16 bytes long, but the last 4 bytes of the UMID UL are zero and can therefore be omitted when used in the UMID.
- *Length (L)*. This 1-byte field specifies the length of the byte string that follows. Because 19 bytes follow in the case of basic UMID, this field is fixed to 13_h.
- *Instance Number (Inst.#)*. This 3-byte field specifies whether the material number, the field that immediately follows this, is a newly created value or the inherited one from another UMID already existing elsewhere. For the newly created basic UMID at the introduction of new AV material, this field is fixed to 00_h 00_h 00_h. As a result, if a given UMID is revealed to have its instance number zero-filled, the UMID may be regarded as identifying the original material with a newly created value.
- *Material Number (Mat.#)*. This 16-byte field accommodates a globally unique value, which makes the UMID a globally unique identifier. Several creation methods of the value for this field are specified in SMPTE ST330.[1] An example is given by the combination of the network node number of a device that creates AV material, together with the time snap at which the material is created. Because any single network device (with a single network port) can be globally uniquely identified by its network node number, AV material with such a UMID will be also globally uniquely identified under the assumption that only one material is to be created at a certain time snap by the device.

Two Distinct Uses of UMIDs

As originally intended, the primary use of UMID is as a globally unique AV material identifier. For example, when new AV material is created from scratch by acquisition, a UMID is issued and attached to the material automatically so that it can be globally uniquely identified. In this UMID use, the instance number of the UMID is zero filled, as shown in **Fig. 2 (a)**.

Another use of UMID is as a linking tool. Suppose new AV material is created from an existing source material by partial retrieval. Then, a UMID, with its material number being inherited from that of the source material while its instance number is set to nonzero, is attached as shown in **Fig. 2 (b)**. In this case, the resulting "truncated" material is associated with the source material via its material number; that is, the UMID of the source material may be easily obtained by just masking the instance number field of the resulting material to zero.

Although two uses of UMID have been introduced here, they are exclusive. If a UMID is used as a linking tool, as shown in **Fig. 2 (b)**, its global uniqueness cannot be guaranteed because the value space of the 3-byte instance number is far from sufficient to accommodate a globally unique value. Although it would be possible for an application to carefully control the instance number so that the UMID remains unique, the uniqueness is guaranteed only within a predefined closed domain that the application can control; that is, though there is a small chance of this, two independent applications could create the truncated materials of different durations but assign UMIDs whose instance numbers happen to be same.

Consequently, when AV material with a UMID of a nonzero instance number is at hand, the material cannot be managed by the UMID because its global uniqueness is not guaranteed. If the material is also desired to be managed globally, the UMID needs to be replaced with a newly created value with a zero instance number (thus as a globally unique identifier).

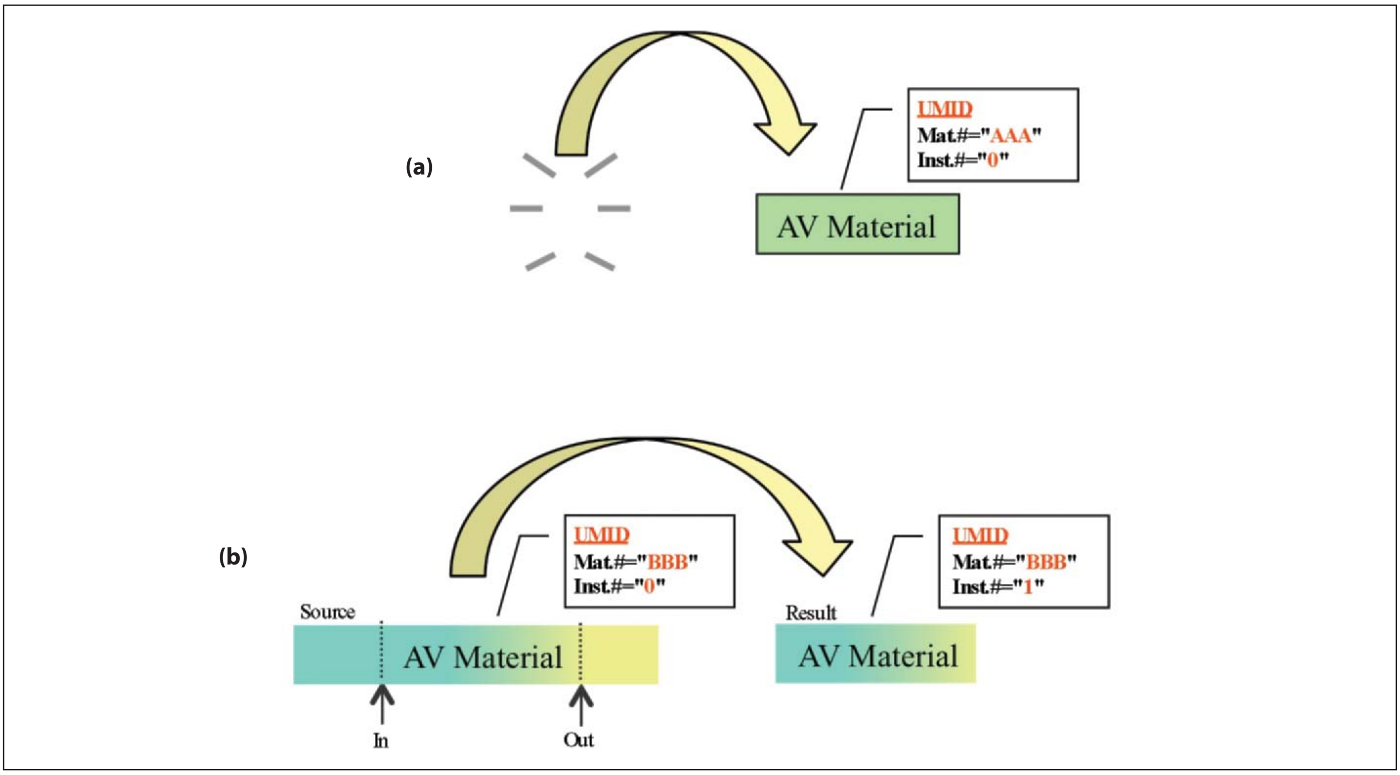

Figure 2. Two distinct uses of UMID: (a) UMID as a unique identifier; (b) UMID as a linking tool.

UMID APPLICATION AS A GLOBALLY UNIQUE IDENTIFIER

Goal of UMID Application as an Identifier

While the primary use of a UMID is as a globally unique identifier for any given item of AV material, the creation and attachment of a unique value to the material is just the first step. Another step is that the desired material must be made accessible when its identifier is known.

A typical example of such a scenario is the UMID-based AV material search, as demonstrated in **Fig. 3**. In this scenario, AV materials are stored in various kinds of servers connected to the network,

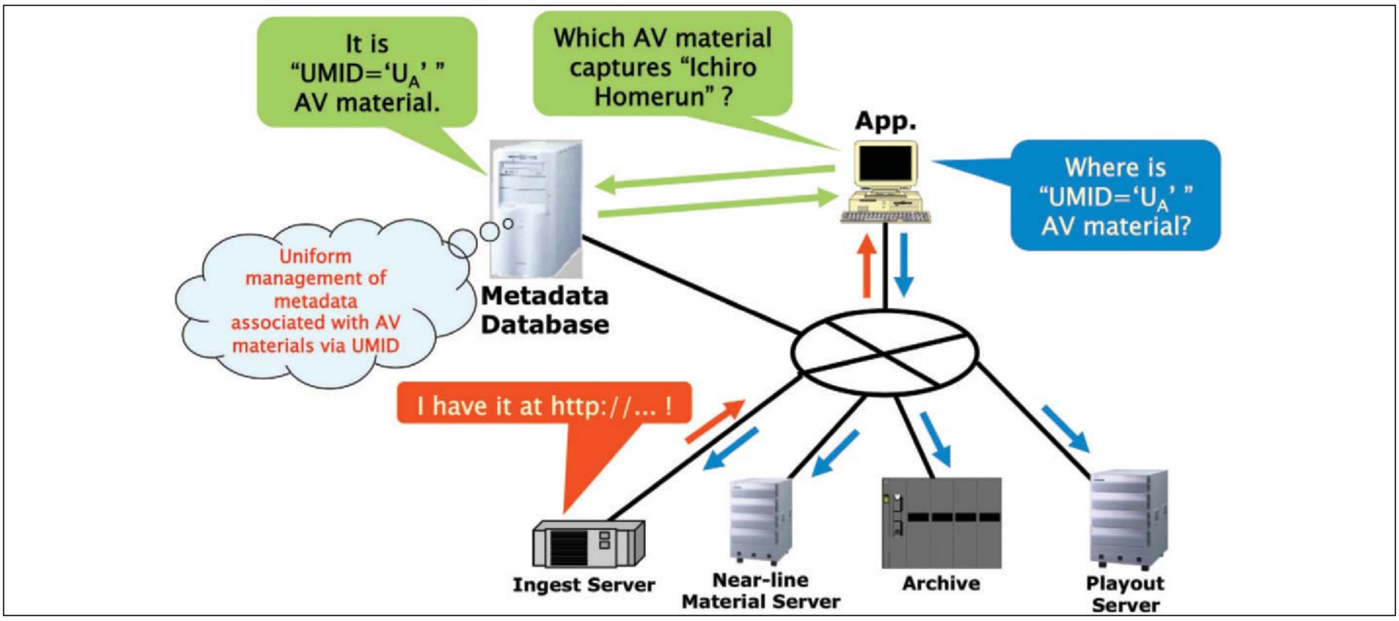

Figure 3. UMID-based AV material search.

and metadata associated with some AV material via its UMID are collected and stored altogether in the dedicated metadata database for their uniform management.

When an external application needs to obtain AV material that captures an "Ichiro Homerun" shot, for example, it then gives the request to the metadata database accordingly. The metadata database then replies to the application with the requested AV material by its UMID.

Because the UMID cannot indicate anything about where to access the AV material, the application needs to resolve the UMID—that is, to translate the UMID into its corresponding uniform resource locator (URL). In this scenario, the application distributes a query asking, "Where is 'UMID = U_A' AV material?" and the ingest server responds to it with the URL for the requested AV material, which is then used by the application to access the material.

Because the application would have difficulty handling a specific query and response to the UMID resolution request, depending on a particular device, the UMID resolution protocol needs to be standardized so that any application can talk consistently to any device.

UMID Application Principles

UMID Integrity

Behind a scenario described previously, each server is assumed to appropriately manage its own AV materials with their valid UMIDs. Although several designs would achieve appropriate UMID-based material management, we should identify the UMID application principles, the most fundamental rules all products must strictly follow regardless of their implementations, to realize standardized UMID-based material management.

First is UMID integrity, which is expressed as follows: "Different AV materials shall be identified by different UMIDs." Otherwise, a UMID cannot be a useful material identifier. With this principle, however, it is important to understand precisely what a valid UMID is.

The UMID integrity cannot be determined only by its value. Some UMID value might be syntactically wrong—for example, starting with bytes other than 06_h $0A_h$ (which may be easily detected by a syntax checker). But even if it is syntactically correct, in some cases the UMID is not valid. For example, another AV material might exist that has the same UMID but a different duration. This could happen if a clone material is created and then truncated by cutting off the last 5 min of the material.

So a question arises: How can we obtain material with a valid UMID? There are only two methods to do this. One is to create a new UMID by yourself (using a properly implemented UMID creation function) and attach it to the material (if it is not yet attached) or replace the existing UMID with it (if one is already attached). This is because when you create a UMID, the SMPTE ST330–based UMID creation method[1] ensures that you will have a new globally unique UMID value.

Another method is to obtain the AV material from a UMID-managed domain, where the integrity of UMIDs is always guaranteed. This ensures that AV materials in the domain are carefully managed, based on their valid UMIDs. Further discussion on the UMID-managed domain is in the next section.

UMID Identification

Another rule that should be made part of the UMID application principles is regarding what the UMID precisely identifies. This is equivalent to the strict definition of "AV material" used in this paper so far, or the precise meanings of "identical" or "different" for plural AV materials.

The most important requirement to be fulfilled is a clear boundary for everyone between identical and different, without ambiguity. In this sense, "video codec kinds," for example, cannot be used, because their classification boundaries are often obscure; someone might want to distinguish between the H.264/MPEG-4 AVC and the MPEG-2 codecs while another person may want to distinguish the variable and the constant bitrates within the MPEG-2 codec.

Base on those considerations, we propose the following: "What the UMID identifies is the essence representation at its playout."

With this principle, we can judge whether two given AV materials are identical or not without ambiguity. For example, if they are clones except for their titles, they are identical. Furthermore, even if they have essences compressed in different kinds but mathematically lossless video codecs, they are identical because the baseband signals at their playout are identical.

On the other hand, if they have essences compressed in different kinds but "visually lossless" video codecs, they are different because even though no visual distinction is detected between them by human eyes, they are mathematically different baseband signals at playout.

This rule does not exclude attempts to assign different UMIDs to materials sharing essences but having different titles, codecs, and so on. Furthermore, together with the first rule in the previous subsection, even the assignment of different UMIDs to clone materials is not ruled out. The UMID application principles we propose here define one point of reference as the bottom line, which we call the logical approach. Another point of reference, defined as the top line and called the physical approach, requires all physical entities to be distinguished by different UMIDs; this is done by assigning new UMIDs to even clone materials.

Any UMID-based material distinction exists between those two extreme cases, the logical and the physical approaches. Because the logical approach constitutes the bottom line of such a material identification scheme, it is suitable for the specification as a technology standard that all products can and must follow.

UMID-Managed Domain

What Is the UMID-Managed Domain?

Based on the UMID application principles proposed previously, we introduce a concept called the UMID-managed domain, which

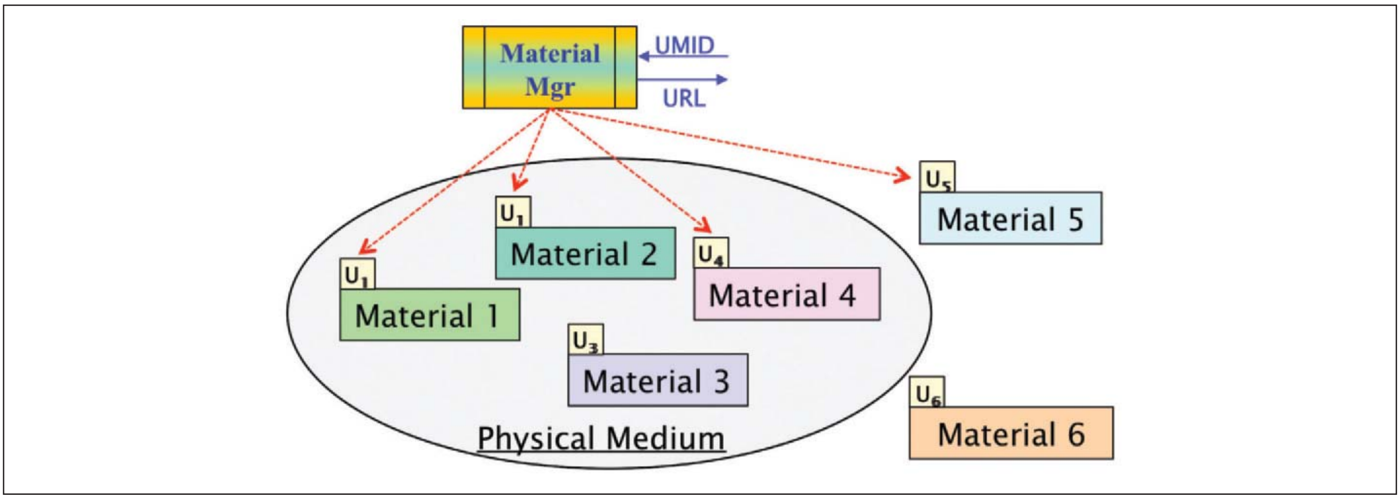

Figure 4. UMID-managed domain.

is composed of AV materials appropriately managed according to their UMIDs. In this domain, all UMIDs used to manage their attached materials are treated as valid globally unique identifiers. In addition, the latest locations of the AV materials are always maintained, together with their respective UMIDs, so that any of them can be resolved to their respective locations (typically as URLs) for a given UMID without ambiguity.

Figure 4 schematically illustrates the concept of the UMID-managed domain. In this figure, the Materials 1 to 4 are recorded on a certain physical medium; Materials 5 and 6 are stored elsewhere. There is a material manager (Material Mgr) that manages the AV materials according to their UMID values. The red arrows from the manager to the material denote that the manager recognizes the materials via their UMIDs and their locations (Materials 1, 2, 4, and 5); therefore, it can provide a URL for any of those materials in return for a given UMID.

Because the UMID-managed domain is composed only of materials managed by the material manager via their UMIDs, materials not recognized by the manager are outside the scope of the domain, regardless of their physical locations. For example, even though Material 3 is recorded on the same physical medium as those in the domain, it is not in the domain because it is not recognized by the manager; as a result, the material is not seen from an external application requesting the material manager for its UMID resolution.

Requirements of the Material Manager in the UMID-Managed Domain

The goal of the UMID-managed domain is to provide an embodiment of the UMID application principles. For file-based operations, the media files in the domain are properly managed via their respective UMIDs and made accessible via their URLs resolved from the given UMIDs.

As shown in **Fig. 4**, the material manager plays a key role to accomplish the UMID-managed domain. It is the material manager's responsibility to ensure the UMIDs in the domain are always valid. In addition, because of the UMID resolution to be realized, a database for a pair composed of the UMID and its corresponding URL for each media file (hereafter called the UMID-managed list) must be maintained in its latest state. As a result, the material manager needs to be involved in most kinds of operations of media files.

In the following sections, the expected behaviors of the material manager for a couple of typical media file operations are discussed.

Media File Import into the Domain

When a new media file is imported into the domain by copying or moving it from elsewhere, the basic behavior of the material manager is to attach a newly created UMID (if one is not yet attached) or replace the existing UMID with a new UMID value (if one is already attached). Therefore, the material manager must detect the UMID of an incoming media file, attach or replace it with a newly created value, and register the pair composed of the UMID and the resultant file's URL to the UMID-managed list.

Exceptional behavior may be applied when the original location of a media file is also known to be a UMID-managed domain. In this case, because the UMID attached to the incoming media file is valid, the UMID may be reused when the media file is imported into the domain exactly as is (as a clone). In this case, the material manager may just detect the UMID of an incoming media file and register the pair composed of the UMID and the resultant file's URL in the domain to the UMID-managed list after the completion of the copying or moving operation.

Media File Modification at Its Essence Within the Domain

When an existing media file is modified, for example, by insert editing, the material manager must find the UMID of the media file, replace it with a newly created value, and update the UMID value in the pair composed of the UMID and the file's URL existing in the UMID-managed list. This must be done even when only one media file exists for a given UMID in the domain, because a clone file might exist elsewhere in the world.

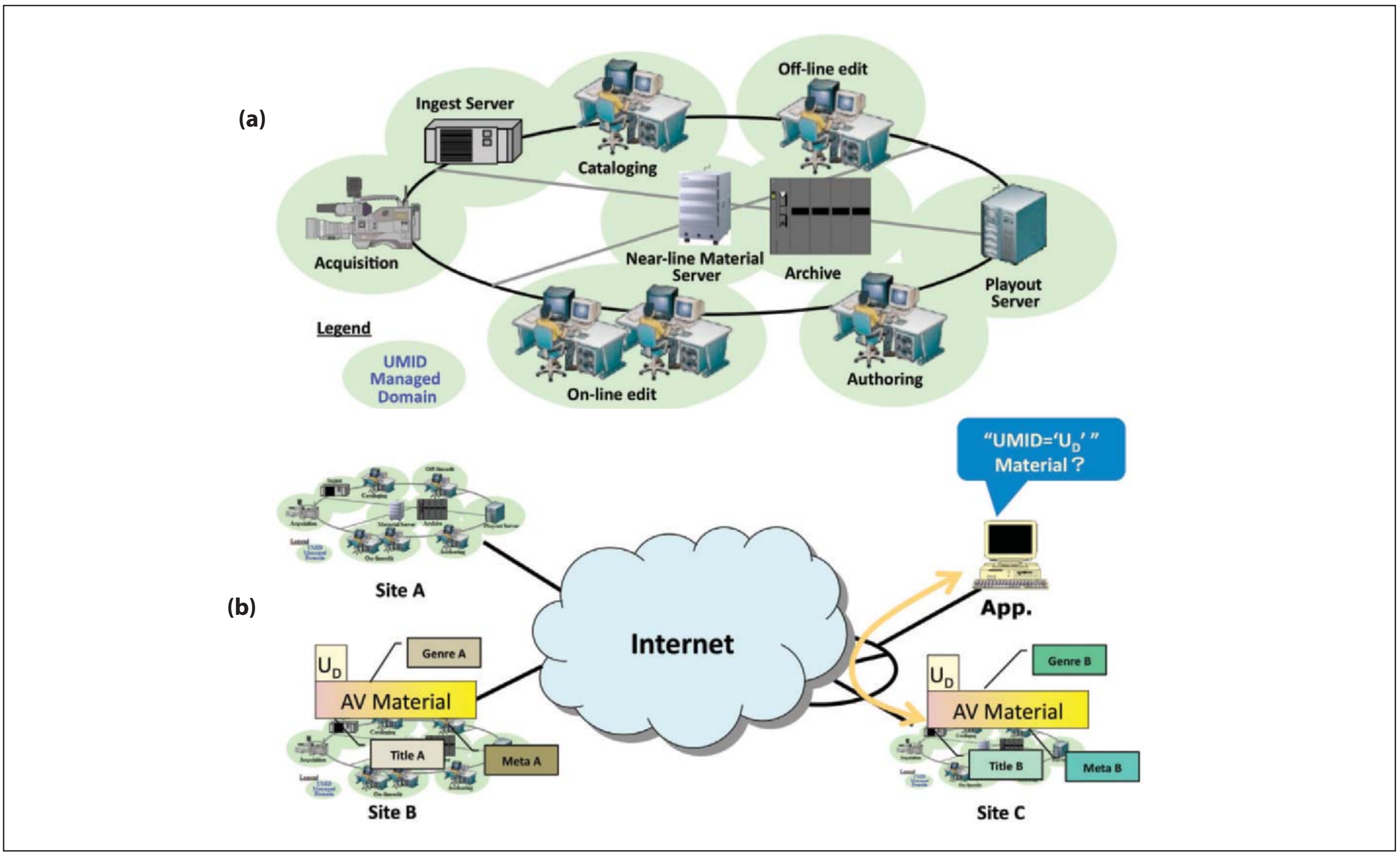

Figure 5. Extension of UMID-managed domain: (a) within a system; (b) among multiple sites over internet.

Extension of the UMID-Managed Domain

When a media file is imported into or exported from another UMID-managed domain, the original UMID attached to the source media file may be used as is for the destination media file. This is regarded as the equivalent of the copy operation of a media file within a single UMID-managed domain even though it is conducted over the boundary of the domains.

This situation implies that, with the help of the UMID resolution protocol language spoken among the material mangers for each domain, the plural UMID-managed domains can be merged to form a single UMID-managed domain in which the AV materials are uniformly managed in the same way as those in the original, individual domains by using the same UMIDs.

This leads to the realization of the so-called best-of-breed media system composed of products from multiple manufacturers. In **Fig. 5 (a)**, assuming all products participating in a media system support the UMID-managed domain appropriately, the UMID-managed domains are merged to cover the entire system; that is, a media system using the UMID as a common material identifier can be achieved.

Furthermore, because of the global uniqueness of the UMID, the UMID-managed domain can be extended globally. **Figure 5 (b)** shows one such case, in which AV materials are managed locally at each relevant broadcaster's site (Sites A, B, and C) but shared globally among them. Because the UMID does not depend on a specific implementation of the AV material management, such as for the media asset management (MAM) system, used in each site, AV materials to share the same UMID (U_D in **Fig. 5 (b)**) are guaranteed to bring the same essence representation at their playout. Therefore, an external application can selectively access the most convenient one among them, as shown in **Fig. 5 (b)**, where an application ac-

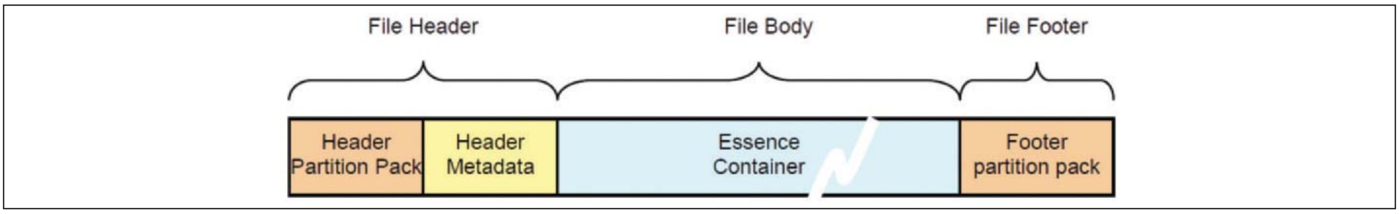

Figure 6. Simple MXF file.

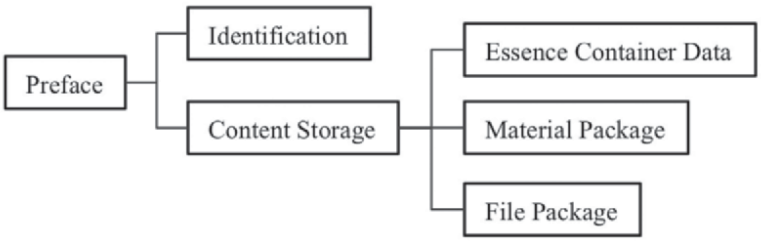

Figure 7. Logical structure of MXF header metadata.

cesses the AV material of UMID U_D managed in Site C because of its closer location in the network.

UMID APPLICATIONS IN MXF

MXF Overview

What Is the MXF?

The MXF is a container format standardized in SMPTE ST377-16 and its accompanying document suite. MXF is a container in the sense that MXF contains any kind and any form of AV essence. MXF also carries metadata, which are to be associated with the whole or particular points of the AV essence it contains.

An MXF file, in its physical representation, is composed of a sequence of SMPTE key-length-value (KLV)–coded chunks of data (hereafter called the KLV packet),[7] which logically form a file header, a file footer, and the file body placed between them, as shown in **Fig. 6**.[6]

In a typical MXF file, the essence data are individually KLV-wrapped to form the essence container and are stored in the file body. The granularity of the essence to be KLV-wrapped varies, depending on the type of MXF file (i.e., from per frame to an entire essence).

Metadata associated with the whole or parts of the file are called the header metadata and are contained in the file header—and in the file footer if desired. Two specific kinds of header metadata are defined in MXF: structural metadata and descriptive metadata.

Structural metadata describe different essence types and their relationships along a timeline. The structural metadata are mandatory because they specify how to playout the essence, including its picture size and frame rate. Because the essence contained in the file body is typically pure video and/or audio digital data without temporal information, there is no way for the essence to be played out without the structural metadata. Consequently, modification of structure metadata may strongly affect how the essence is to be played out.

Descriptive metadata provide information mainly for human use, such as titles and synopsis. While it is important for a user to search a desired MXF file efficiently, we don't go into more detail about this in this paper because it does not affect the playout of the essence.

The header metadata are also a sequence of KLV packets called sets, but they logically form a tree structure by connecting them via strong references (which link sets together). **Figure 7** shows a simplified logical structure of MXF header metadata (the structural metadata).

In **Fig. 7**, the preface set, as a root of the header metadata, signals the kind of MXF file, as well as the types of essences container and the kinds of essence contained in the file.

Below the preface set come the identification and content storage sets. The identification set specifies the MXF file version information, date of creation, and so on, and the content storage set contains three fundamental metadata items: the essence container data, the material package, and the file package. The essence container

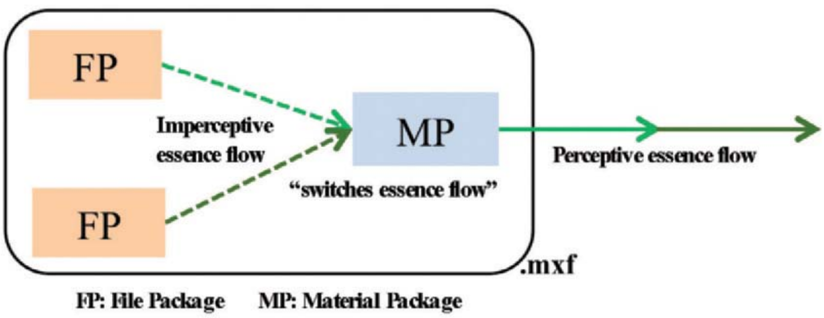

Figure 8. MXF internal model at playout.

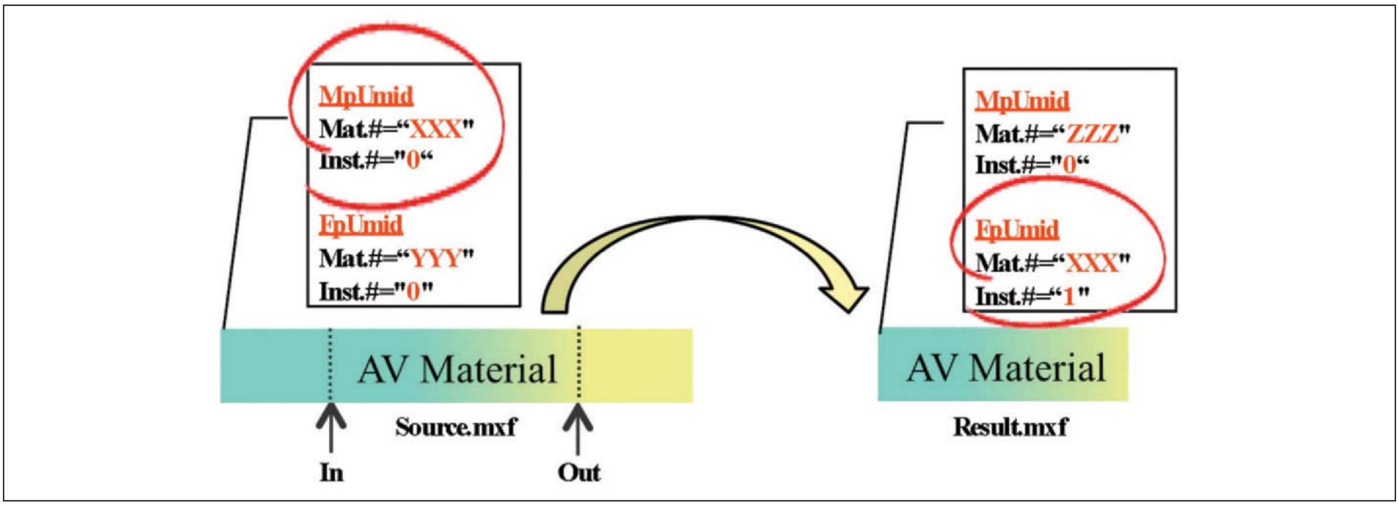

Figure 9. UMID application in partial retrieval of an MXF file.

data describe the individual essence container in the file body. The material and file packages describe an output timeline of the MXF file and the essence stored in an essence container, respectively, which constitute one of the most characteristic parts of MXF and are vital for UMID applications in MXF.

MXF with Dual Structure

Among other valuable characteristics of MXF, its dual structure is one of its most unique characteristics. In short, what is to be played out is not always the same as what is contained in the file.

Because of its dual structure, an MXF file has two kinds of metadata describing the essence data: the file package and the material package (**Fig. 7**). Specifically, the file package is used to describe an essence container (and thus the essence contained in it); therefore, the same number of instances of the file package and of the essence containers exist in the file. The material package is used to describe the essence to be played out. In other words, the material package specifies what we observe when an MXF file is played out.

Because of such roles of the material and file packages, the following features are observed:

- The file package represents the temporal information of an essence container (input timeline) and its technical properties, such as codec and frame rate, by using the file descriptor.
- The material package represents the temporal information of an essence to be played out (output timeline) but does not include the technical properties.
- The material package specifies how to play out the one or more essences described by the file package or packages by referencing the file packages with their begin and end time codes.

These may be schematically modeled as shown in **Fig. 8**. Based on this figure, the internal behavior at the playout of an MXF file is described as follows:

1. Each file package provides an imperceptive baseband essence flow to the material package.
2. The material package controls the essence flow along the output timeline and makes the essences perceptive, resulting in the essence representation we observe at playout.

The reason the technical properties of any essence are omitted for the material package may be understood as follows: the standard baseband essence flows are generated by each file package and supplied to the material package; the material package just receives and switches them at playout.

UMID Applications in MXF

UMIDs in MXF

According to SMPTE ST377-1,[6] the basic UMID is used as a unique identifier (UID) of a package that describes the essence on a timeline. In the following discussion, let MpUmid and FpUmid denote the package UIDs of material and file packages, respectively.

Applications of the Material Package UID (MpUmid)

While an originally expected role of the MpUmid would be as a unique identifier of an instance of the material package, it is also considered to uniquely identify the essence on an output timeline; that is, it uniquely identifies the essence representation at its playout. Hence, based on the UMID application principles, the MpUmid is regarded as a globally unique identifier of an MXF file.

Consequently, the UMID-managed domain composed of MXF files with their MpUmids needs to be established. The material manager in the domain is expected to behave in the way previously discussed. For example, when an MXF file is imported from somewhere other than the UMID-managed domain, the material manager must detect the MpUmid of the incoming MXF file, replace it with a newly created value, and then register the pair composed of the MpUmid and the resultant file's URL to the UMID-managed list.

As for MXF file modification at its essence, the material manager must detect the MpUmid of the modified MXF file, replace it with a newly created value, and update the pair composed of the MpUmid and the file's URL in the UMID-managed list. Because the material package controls the essence to be played out, modification of the material package often requires its MpUmid to be updated—except that it does not affect the essence to be played out.

Applications of the File Package UID (FpUmid)

The FpUmid is a unique identifier of an instance of the file package and thus can be considered to uniquely identify the essence flow from its file package to the material package for its playout. Because this is a conceptual essence flow effective only within an MXF file, there is theoretically no way for an external application to access it—even when a given UMID is resolved to an MXF file containing the file package with the UMID value as FpUmid. Hence, the usefulness of managing the FpUmid is quite limited in general.

However, the uniqueness of the FpUmid is crucial within relevant MXF file or files because the file package is uniquely referenced by the material package within the file or files.

Based on these considerations, though default treatment of the FpUmid is to assign a newly created value for it, there is another application for the FpUmid, which can be made more useful in practical situations if the scope of the FpUmid uniqueness to be within MXF file or files is accepted.

An example of such an FpUmid use is given by partial retrieval of an MXF file. Suppose a resulting MXF file, Result.mxf, is obtained by partially retrieving a source MXF file, Source.mxf, with its in/out points as shown in **Fig. 9**.

Because Result.mxf is a newly created MXF file, its MpUmid must be the newly created value with its zero instance number. But if its FpUmid is permitted not to be globally unique, another use of UMID as a linking tool can apply; that is, because the essence described by the file package is created from the output of Source.mxf, it is reasonable for the FpUmid to also be created from the MpUmid of Source.mxf. Specifically, the material number of the FpUmid for Result.mxf is inherited from that of the MpUmid for Source.mxf, while its instance number is a nonzero value to indicate its use as a linking tool.

Because an application executing such an operation can fully manage the creation of all UMIDs to be contained in the resulting MXF file or files, it is easy for the application to assign UMIDs that are unique within the MXF file or files just by controlling the nonzero instance number.

This kind of FpUmid treatment brings additional usefulness of the FpUmid. While it is obvious in **Fig. 9**, the resolution of the FpUmid of resulting Result.mxf with its instance number masked to zero leads to the source MXF file from which the resulting MXF file is created.

UMID RESOLUTION PROTOCOL

Requirements for the UMID Resolution Protocol

Although AV materials can be fully managed by using their respective UMID values, it is useless if these materials cannot be accessed from external applications. Therefore, the method to access material via the UMID must be standardized for an application to consistently access any materials managed by any products.

While the detailed specifications of the UMID resolution protocol are still under study, we suggest its requirements and plausible solutions in this subsection. The requirements identified so far include the following:

- The UMID must be resolved to the URL of its corresponding media file.
- Some technical metadata that hint for the requester to judge their playability should be provided, because the UMID value alone contains no information as to whether it is playable.

Domain Naming System–like Protocol

One possible solution would be to borrow the idea of the domain naming system, in which each 4-byte internet protocol address is resolved to its associated URL, and vice versa. In this approach, the following items need to be considered:

- The dedicated port number to be registered to the Internet Assigned Numbers Authority
- Payload as a binary format for request and response
- RP224 (SMPTE label)–based technical metadata[8]

Standard Operation in a Web Service Interface

With the service-oriented architecture–based technology becoming popular in the media and entertainment industry,[9] it is convenient to specify the web service version of the UMID resolution protocol. In this approach, the following items need to be considered:

- Independency of the underlying protocol
- Extensible Markup Language schema for messages to be exchanged and its message exchange pattern
- Both the simple object access protocol and representational state transfer

CONCLUSION

The reason the originally intended use of UMID as a standard globally unique identifier has been seldom seen in practice is revealed by a lack of fundamental rules for its application, including the UMID resolution protocol. Although the UMID application has been described in SMPTE RP205,[2] we cannot say its content is sufficient when we observe UMID uses in reality. We believe this paper should be a contribution for the recommended practice document to be updated to improve its scope and quality.

It is fortunate that the MXF has adopted the UMID as its mandatory item. Based on the fundamental rules established in this paper, an MXF file is appropriately managed globally by using its material package unique identifier (UID). Furthermore, because of the characteristics of MXF, a link to the source material, one of the most frequently requested behaviors, can be successfully implemented.

Clearly, such fundamental rules about the UMID application should be widely disseminated in the industry to maximize its usability. Therefore, standardization activity for the UMID application is to be proposed shortly for the UMID, together with the MXF, to constitute part of the informational infrastructure for file-based media systems.

NOTE

Based on this paper, the standardization activity for the UMID application has been proposed to the SMPTE standard committee. The proposal includes, as its first step, the study of existing UMID applications, the establishment of the UMID application principles, the SMPTE RP205 update, and the identification of future relevant standardization activities including the UMID resolution protocols.

REFERENCES

1. SMPTE ST330-2011, "Television—Unique Material Identifier (UMID)," www.smpte.org.
2. SMPTE RP205-2009, "Application of Unique Material Identifiers in Production and Broadcast Environments," www.smpte.org.
3. Matthias De Geyter, Nico Oorts, and Luk Overmeire, "Integration Demands on MAM Systems: A Proof of Concept Solution," *SMPTE Mot. Imag. J.*, 117(8):38-46, Nov./Dec. 2008.
4. Yoshiaki Shibata and Jim Wilkinson, "White Paper on UMID and Its Applications," the technology brochure distributed at the Sony booth at in NAB2002 (April 2002), http://metafrontier.jp/drupal/sites/default/files/papers/umid4Nab2002TechBrochure.pdf.
5. SMPTE ST298-2009, "Universal Labels for Unique Identification of Digital Data," www.smpte.org.
6. SMPTE ST377-1-2011, "Material Exchange Format (MXF)—File Format Specification," www.smpte.org.
7. SMPTE ST336-2007, "Data Encoding Protocol Using Key-Length-Value," www.smpte.org.
8. SMPTE RP224v11-2011, "SMPTE Labels Register," www.smpte.org.
9. European Broadcasting Union–Advanced Media Workflow Association Framework for Interoperable Media Service Task Force, http://wiki.amwa.tv/ebu/index.php/Main_Page.

Presented at the SMPTE 2011 Annual Technical Conference & Exhibition, 25-27 October 2011.

The Authors

Yoshiaki Shibata has started his career as a semiconductor researcher at Sony Corp. in 1991. He then moved into the IT/Multimedia field in 1996, studying video over internet and video compression technology.

In 1998, Shibata joined the MPEG-7 standardization activity; he took a leadership in the MPEG-7 schema design as a working group chair and a project editor for ISO/IEC 15938-5.

Since 2001, he has been working in the media and entertainment industry, where initially he made a significant contribution to update the UMID specification (SMPTE ST330/RP205) as well as invent the EssenceMark for professional AV products. In 2003, he took part in the XDCAM development project, where he played a crucial role in the metadata part of the product development and applied for more than 40 patents in metadata related technology.

In 2011, Shibata left Sony Corp., making a trail to provide Japan's first independent consulting service for the media and metadata technology in the media and entertainment industry. Then, following a one-year preparation period, he founded metaFrontier.jp, LLC in Jan. 2012.

Jim Wilkinson first worked in the area of broadcasting, joining the Independent Broadcasting Authority's (IBA) Engineering Headquarters in1974. During his time with the IBA, he worked on DICE, video compression using differential pulse-code modulation (DPCM) at 34 Mbits/sec and other video projects.

In 1979, Wilkinson joined the newly created Advanced Development Laboratories of Sony Broadcast as one of the founding members. His many activities and projects during his 28 years with Sony included digital video recording, image compression and processing, digital audio, metadata and file/stream formats.

Wilkinson has been awarded more than 80 patents; both U.K. and international. He has participated in many standards activities in the Audio Engineering Society, SMPTE, and the European Broadcasting Union, with particular emphasis on metadata, compression and file/stream formats.

In 1995, he was awarded the Alexander M. Poniatoff Gold Medal for Technical Excellence by SMPTE. He is a Chartered Engineer and a fellow of the Institution of Engineering and Technology and SMPTE.

In 2006, Wilkinson retired from Sony and now works as an engineering consultant in the areas of video and sound with an ongoing interest in compression and file/stream formats.