2005

MPEG-21 digital item declaration and Identification-principles and compression

I. S. Burnett  
University of Wollongong, ianb@uow.edu.au

S. J. Davis  
University of Wollongong, stdavis@uow.edu.au

G. M. Drury  
University of Wollongong, drury@uow.edu.au

Publication Details  
This article was originally published as: Burnett, IS, Davis, SJ & Drury, GM, MPEG-21 digital item declaration and Identification - principles and compression, IEEE Transactions on Multimedia, June 2005, 7(3), 400-407. Copyright IEEE 2005.
MPEG-21 digital item declaration and Identification-principles and compression

Abstract
At the core of the MPEG-21 Multimedia Framework is the concept of the Digital Item, a virtual container for a hierarchical structure of metadata and resources. This paper considers the Digital Item Declaration Language (DIDL), gives examples of its usage, and discusses how it is used to integrate other parts of MPEG-21. The paper then discusses how Digital Item Identification integrates with the DIDL to allow MPEG-21 to utilize standard identifiers from many application spaces. Finally, an alternative, compressed form of the XML Digital Item Declaration is described. This uses schema-based compression to significantly reduce the size of these XML documents.

Keywords
MPEG, multimedia communication, multimedia computing, multimedia databases, multimedia systems

Disciplines
Physical Sciences and Mathematics

Publication Details
This article was originally published as: Burnett, IS, Davis, SJ & Drury, GM, MPEG-21 digital item declaration and Identification - principles and compression, IEEE Transactions on Multimedia, June 2005, 7(3), 400-407. Copyright IEEE 2005.
Abstract—At the core of the MPEG-21 Multimedia Framework is the concept of the Digital Item, a virtual container for a hierarchical structure of metadata and resources. This paper considers the Digital Item Declaration Language (DIDL), gives examples of its usage, and discusses how it is used to integrate other parts of MPEG-21. The paper then discusses how Digital Item Identification integrates with the DIDL to allow MPEG-21 to utilize standard identifiers from many application spaces. Finally, an alternative, compressed form of the XML Digital Item Declaration is described. This uses schema-based compression to significantly reduce the size of these XML documents.

Index Terms—MPEG, multimedia communication, multimedia computing, multimedia databases, multimedia systems.

I. INTRODUCTION

MPEG-21, the Multimedia Framework, has been and continues to be an ambitious undertaking; the propensity of parts of the standard (currently 16) is an indication of the complexity of creating an overriding architecture for the creation and delivery of diverse multimedia content and applications. With MPEG-21, MPEG created a new interoperable unit for multimedia delivery and transaction—the Digital Item (DI), which is essentially a “virtual container” for metadata and content (called resources in MPEG-21 to avoid confusion). The base concept of a DI is similar, in part, to work undertaken in the Digital Library [1] and e-learning fields [2]. However, in MPEG-21, a complete and rich delivery framework based around a more versatile DI specification has been standardized. Thus, MPEG has shifted away from the previous bitstream syntax and semantics, decoder behavior (MPEG-1, MPEG-2, MPEG-4 [3], [4]) and multimedia description tools (MPEG-7 [5]) standardization activities to standardize the components of a higher level framework. The latter is intended to be a significant step toward achieving the mantra of “any content, any time, anywhere”. This paper concentrates on the mechanisms for declaring and identifying the distinctive core of MPEG-21, the DI.

DIs are the unit of transaction in MPEG-21 and, for that reason, the capitalization of “Digital Item” is deliberate. A DI, as defined specifically by MPEG in MPEG-21 [6] consists of resources, metadata, and structure. Given this definition it is helpful to contrast a DI with the familiar web page which could also easily be covered by such a description.

The primary distinction between a web page and a DI is the purpose of the underlying structure. For a DI, the structure is aimed purely at declaring the constituent parts, while for an HTML web page the structure is aimed at marking-up the text and resource content for presentation purposes. Hence, while the constituent parts of a DI may be present in a web page, the main aim of that web page is the presentation of material via a browser. In contrast, DIs are not required to contain information as to how the resources and metadata should be presented. Thus, DIs will either operate in application spaces where there are agreed “rules” for presentation or may contain presentation descriptions as resources. However, in both cases, the presentation to the user is clearly more flexible than that of a web page where the author enforces the presentation.

Another important feature of a DI is the ability to configure the DI, taking into account such factors as the usage environment, terminal capabilities, and network conditions. This assists in enabling transparent and augmented use of DIs across a wide range of networks and devices used by different communities.

The paper is divided into three main sections describing the DI Declaration, DI Identification, and then a more exploratory section on the compression of DI Declarations.

II. MPEG-21 Part 2—Digital Item Declaration

This section considers DIs and their structure in detail. The structure of the DI is provided by a DI Declaration (DID) [6]. We first detail the model used for the declaration, and its representation in XML, before considering an example DI, validation and a brief summary of updates proposed for a second edition of [6].

A. DIs and Declarations

The DID formally expresses and identifies the resources (e.g., MP3 files) and the metadata (e.g., Dublin Core [7] descriptions) which are considered by the author to be the constituents of the DI. Further, the DID binds together individual and groups of resources and metadata. This is further extended by the capability to allow metadata to be anchored to certain fragments in a media resource.

Part 2 of MPEG-21 takes care to divide the representation of DIs into three distinct parts [6].

• DID Model: A set of abstract terms and concepts to form a useful model for defining DIs. Within this model, a DI is the digital representation of “a work”, and as such, it is the thing that is acted upon (managed, described, exchanged, collected, etc.) within the model.
• **Representation:** The description of the syntax and semantics of each of the DID elements, as represented in XML.

• **Schema:** An XML Schema [8], [9] comprising the entire grammar of the DID representation in XML.

It should be noted that the DID Model is an abstract model of building blocks useful for declaring DIs. This abstract model could be represented in different ways. MPEG-21 Part 2 defines a normative XML representation of the DID Model. This XML-based representation is the Digital Item Declaration Language (DIDL).

In this section, we summarize the DIDL, but for a detailed description of the DID Model, the representation, and the XML Schema for DIDL the reader is referred to [6]. In general, an XML element of the DIDL corresponds to a building block of the DID Model. Therefore the syntax, semantics, and relationships of the DIDL elements reflect the semantics and relationships of the corresponding building blocks of the DID Model. In some cases, parts of the DID Model are represented by attributes of elements in DIDL. DIDL also includes some additional elements not part of the DID Model, but useful when declaring DIs using DIDL. In addition to the XML Schema, DIDL includes validation rules not expressed in the XML Schema. These additional validation rules are an integral part of the semantics of the DIDL elements and we consider these in Section II-C.

Section IV of this paper considers an alternative representation of the DID Model in a binary rather than textual form (Part 16 of MPEG-21 [10]). However, the approach taken in that section is strongly related to the DIDL schema and the accurate compression of the XML representation of the DID.

### B. DIDL

The grammar of the DIDL is presented as an XML Schema. Fig. 1 shows a partial graphical representation of the hierarchy.

In this section we consider the fifteen elements that are available to declare a DI. In this paper, the names of DIDL elements and attributes appear in **bold**. When reference is made to the building blocks of the DID Model, *italics* are used.

The root element of any DIDL document is the **DIDL** element. It exists only in the representation and schema. The **DIDL** element may contain either a single **Item** or **Container** as a child. It may also contain a **Declarations** element before the **Item** or **Container**.

An **Item** comprises sub **Item** elements and/or **Component** elements. It may also contain **Descriptor** elements containing additional information bound to the **item** represented.

A **Container** element provides for the grouping of **Item** elements or other **Container** elements. An example usage would be to provide “a shelf” on which a collection of “books” represented by **Item** elements might be placed.

A **Declarations** element allows declaration of other DIDL elements without instantiating them. The declared elements can then be used later in the DID by using an internal reference (see the **Reference** element below). The **Declarations** element exists only in the representation and schema.

The building blocks of an **Item** element are the **Component** elements. By using sub **Item** elements the **DID** can also define the structure of a DI.

A **Component** element binds a set of **Descriptor** elements to a **Resource** element.

A **Resource** element references an individually identifiable resource such as a video or audio clip, or an image. In the DID Model, a **component** contains only one **resource**. However, in DIDL, a **Component** element may contain one or more **Resource** elements. This allows multiple references (for example at different locations) for the represented **resource**. Each

![Fig. 1. Partial graphical representation of the DID Schema.](image-url)
Resource element in a Component element must reference a bit equivalent resource.

A Descriptor element allows information to be associated with the enclosing element. The information can be represented either as a Component or a Statement. The latter contains textual information that can be bound to other elements.

The elements described so far provide the basic building blocks for declaring a DI using DIDL. The next four elements are the key elements in enabling a DI to be configured. They allow parts of a DI to be made available based on conditions or user choice.

A Choice element groups a related set of selections represented by Selection child elements of the Choice element. Each Selection element represents a decision about an associated predicate that will affect one or more Condition elements in an Item. If a Selection is chosen, its predicate becomes true; if it is rejected, its predicate becomes false; if it is left unresolved, its predicate is left undecided.

A Condition element allows the optional inclusion of the enclosing element based on a defined set of predicates associated with Selection elements. For the Condition to be satisfied, predicates listed in the except attribute of the Condition must be false, and predicates listed in the require attribute must be true. Predicate tests within a single Condition element are combined as a conjunction (an AND relationship). Multiple Condition elements within a given parent element are combined as a disjunction (an OR relationship).

The final element relevant to conditional availability of parts of the DID is the Assertion element. This element allows the (partial or full) configuration state of a Choice element to be defined by asserting as true, false, or undecided a number of predicates associated with the Selection children of the Choice.

Two of the final three elements are related to the description of resources and other elements: An Anchor element allows Descriptor elements to be bound to a fragment (a specific location or part) of a resource (represented by a Resource element). An Annotation element allows information to be associated with an identified element without altering or adding to the element.

The final element of DIDL is the Reference element. This allows the contents of an internally or externally referenced DIDL element to be included in the content of the parent element of the Reference. The Reference element exists only in the representation and schema.

C. DIDL Example

One of the simplest DIs that we could envisage would be a Music album. An abbreviated example is given in Fig. 2. Space in this paper does not allow us to consider a full album with all its metadata and this example should be understood to be illustrative only. The DID opens with the XML namespace declarations familiar to users of XML and the root DIDL element. Within this DIDL element we have a single Item which has an id attribute, allowing external or internal referencing of the Item. At the top level of this root Item, the album has been identified with a DII Identifier element included in a Descriptor/Statement combination. This will be discussed fully in Section III of this paper, however the key point worth noting is that the root Item is the “subject” of the DID and is thus identified with the base identifier of the DI. Other identifiers could be spread throughout the DI to identify content but are not included here for brevity. A human readable text Descriptor/Statement combination is also included at this level giving the title and artist. This could equally have used XML metadata from e.g., the MPEG-7 standard [11]. In a real world application it is likely that such metadata standards will be used rather than plain text in order to provide interoperability. The metadata standards chosen will be those agreed within a given application space.

The Choice is identified for reference purposes as “BR” and represents a choice in the DID between content at a bit rate of 128 kb/s or 192 kb/s. One, and only one, of these bit rates must be selected (the configuration state of the Choice) and thus both minSelections and maxSelections are set to 1. Each of the Selection elements has an appropriate select_id that can be used to resolve the state of Condition elements elsewhere in the DID.

The album tracks are included in a sub Item of the root Item. The author of the DI has chosen to attach a Descriptor to the tracks which contains a Component/Resource combination which references a JPEG image of the cover art of this CD: “debutcd.jpg”. Each track, only one of which is shown in Fig. 2, is included as a further sub Item of the album tracks Item. Note that DIs can naturally build a structured hierarchy of content and most complex multimedia content can be thought of in this manner. The first track Item includes a Descriptor/Statement combination with the track name “Billy in the Lowground”, and then two Component elements. Only one Component will be available depending on the state of the Condition elements which, in turn, depend on state of Selection elements in the bit rate Choice. This track Item format is repeated for all thirteen tracks in the album.

Following the album track Item, a further sub Item of the root Item is included for supplementary information (“TONY_INFO”). This includes a Component/Resource combination which references a web site. The Component binds a Descriptor with the Resource. The Descriptor contains descriptive information about the resource. In this case, a plain text Statement describing the purpose of the Resource, i.e., where the CD of this album can be purchased. This simple DID then closes with the obligatory closing XML elements.

D. DIDL Validation

One of the key aspects of DIDL is that it is more than an XML Schema [6]. This is true on two levels: 1) firstly, DIDs create a DI hierarchy which gives powerful structural semantics that take DIDL beyond a simple descriptive schema and 2) the result of 1) is that DIDL requires extra validation rules beyond simply validating the XML against the schema. If an application is purely reading DID documents (i.e., receiving and not creating DIs), the extra validation rules are only significant in that the application can rely on certain combinations and possibilities valid in the DIDL schema not being present. However, an application creating DIDs must be cogniscent of these extra rules. It is not
possible in this paper to consider all of the extra validation rules, but we will consider those of three key elements.

1) Item Validation Rules: The Item element has a single validation rule which is intended to prevent the effective deletion of that Item due to the state of a Condition as a result of a Selection included somewhere in the hierarchy of that Item. The exact wording of the validation rule in 21-000-2 is:

   “An Item element cannot be conditional on any of its descendant Selection elements. In other words, an Item cannot contain a Condition element specifying a select.id value that identifies any descendant Selection element within the Item” [6].

2) Choice Validation Rules: The Choice element is one of the most complex and powerful elements in the DIDL. It allows DI authors to configure and tailor a DI to meet a user’s or terminal’s needs. The first two rules are designed to guarantee that a valid set of selected Selection elements of Choice can actually be made:

   “The value of the maxSelections attribute must be no less than the value of the minSelections attribute” [6].

   “The value of the minSelections attribute must be no larger than the number of Selection children” [6].

Clearly, these validation rules are commonsense but necessary to ensure that an application does not create a DID in which one must, e.g., select three Selection elements in a Choice when there are only two available. The third validation rule for Choice elements continues in a similar vein ensuring that the default of the Choice (if present) is actually one of the child Selection elements and that the number of default values specified for the Choice do not contravene the minimum and maximum number of selections available in that Choice.

3) Condition Validation Rules: Condition elements are intrinsically linked to Selection elements and thus when a DID parser finds a Condition it will necessarily need to search for any associated Selection. To ensure that parsers are not needlessly consumed with an impossible task a simple validation rule has been added to the Condition element:

   “Each ID value specified in the require and except attributes must match a select.id attribute value defined in a Selection element located somewhere within an Item element that is an ancestor of the Condition” [6].

Further, there is an ambiguity if neither an except or require attribute is included in a Condition. Hence, a validation rule is included to prevent empty Condition elements.

Thus to decide if a DIDL document is valid to [6] it is necessary to first validate the XML against the DIDL schema and then to perform checks on the DIDL to ensure that within the document no validation rules have been breached.

E. DIDL Second Edition

The first edition of MPEG-21 Part 2 [6] was published as an international standard in 2003. A second edition is currently under development which will extend the functionality of DIDL. Many of these extensions have resulted from the application of DIDL in practical areas such as digital libraries and archives. Three of the key extensions are considered here.

1) Embedded XML in Resource: In the first edition of MPEG-21 Part 2, an individually identifiable resource is referenced by a Resource element by specifying its Uniform Resource Identifier (URI) in a ref attribute of the Resource. Alternatively, the resource data itself can be embedded as character data content of the Resource. If the data is anything other than a text-based format, the data must be encoded as base64. MPEG-21 Part 2 second edition extends the embedding functionality of the Resource element by also allowing the direct embedding of well-formed XML content in the Resource.
and DIA also standardizes some attribute of the and attribute is not present, the data element by also allowing the elements to be referenced by specifying a URI identifying the statement in a ref attribute of the statement element.

ISO/IEC 21000-2 second edition contains base64 encoded data, the encoding mechanism also provides, e.g., for a movie (as a DI Resource) to be transferred mid-session. At the simplest level, this allows the state of a terminal to adapt the content or make decisions on content retrieval. For example, in Fig. 4, the DIA metadata regarding the terminal status of resource -->

Fig. 3. Inclusion of a REL License in a Descriptor/Statement combination.

2) Referencing of Statement Content: In the first edition of MPEG-21 Part 2 [6], a statement is included as the content of a Statement element. ISO/IEC 21000-2 second edition extends the functionality of the Statement element by also allowing the statement to be referenced by specifying a URI identifying the statement in a ref attribute of the Statement element.

3) Encoding Format of Resource and Statement: MPEG-21 Part 2 extends the Resource and Statement elements by adding an encoding attribute to these elements. If the Resource or Statement contains base64 encoded data, the encoding attribute must be present and its value must be set to base64. If the encoding attribute is not present, the data must be unencoded. This removes an ambiguity in the first edition of MPEG-21 Part 2 when the content is a text-based format that is base64 encoded.

F. Integration Role of the DID: REL and DIA

One of the key roles of the DID in the DI is that it is the core of the structure that makes a DI a functional virtual object rather than just a random collection of metadata and resources. Thus, at this point it is useful to consider how other parts of MPEG-21 fit into the DID structure that we have considered. For this purpose we will concentrate on the integration of Rights Expressions [12] and Digital Item Adaptation (DIA) [13] into the DID. These sections of the standard are discussed in detail in other papers in this issue [14], [15]. The key elements for both DIA and the Rights Expression Language (REL) are the Descriptor/Statement combinations; these are designed to hold metadata relevant to a resource and can thus be used to hold Rights Expressions or DIA metadata.

A simple example of the incorporation of an REL license would be as shown in Fig. 3. This is just one example of incorporating REL expressions into the DID and even with this mechanism it may be advantageous to include the Rights Expression by reference to facilitate flexibility in licenses.

For DIA, the Descriptor/Statement combination is a useful way of carrying metadata which might be used by a terminal to adapt the content or make decisions on content retrieval. For example, in Fig. 4, the DIA metadata regarding the network characteristics are relayed; these could be attached to, e.g., a Component or an Item. DIA also standardizes some specific tools for DIDs. One in particular, allows devices to shift the DI “Session” of a User from one device to another or to store that session. At the simplest level, this allows the state of predicates embodied by Selection elements contained by Choice elements in a DI to be maintained (synchronized) but the mechanism also provides, e.g., for a movie (as a DI Resource) to be transferred mid-flow from one device to another. The Session Mobility mechanism uses DIDs to transfer the state of a user’s session. An example of a “Session Transfer DID” which is termed a context DI is given in Fig. 5. Further details of Session Mobility and its usage are given in [14] in this Special Issue.

As can be seen from the foregoing, DIDs give a useful structural framework in which to transmit and store a variety of “menu” information for digital content. Through the combination of XML schemas for descriptive frameworks (such as DIA, REL and MPEG-7), metadata and resources can be interrelated efficiently and simply. However, it is important to note that the DID can be used effectively without the addition of such schemas as was demonstrated in the example of Fig. 2. In the following section, we consider Digital Item Identification (DII) metadata in some detail.
III. MPEG-21 Part 3—Digital Item Identification

One of the key aspects of the digital world is the ability to identify digital objects and items. While this was also an important part of the “analogue” world [example identifier schemes are the International Standard Recording Code (ISRC) and the International Standard Book Number (ISBN)] where physical objects were identified by number, with the digital domain the propensity of copies and the ability to easily move and change material makes identification vital. MPEG-21, however does not provide a new identification scheme. There are already many identification schemes available and the key to MPEG-21 usage will be that it can integrate with the existing identifiers used in a given application space. Hence, the DII part of MPEG-21 [16] concentrates on how to integrate existing identification schemes into the MPEG-21 framework. In this section names of DII elements also appear in bold.

A. DII Elements

The two XML elements introduced in DII are very simple: Identifier and RelatedIdentifier. Both of these elements are intended to contain URIs. The difference between the two elements is subtle but important as it results in clear differences in usage and implementation. The Identifier is intended to contain a URI that identifies a DI, container, component, or fragment. In contrast, the RelatedIdentifier carries identifiers that are related to the DI (or parts thereof). One example is the identification of an abstraction of the work (e.g., a composition as an abstraction of a sound recording). The distinction between, and usage of, these elements is clarified in the example shown in Fig. 6 taken from the DII standard [16].

B. DII Example

In Fig. 6, a sound recording in the form of an MPEG Layer III (MP3) file is identified using the Identifier element and an ISRC while the RelatedIdentifier is included in the form of a URI version of the International Standard Musical Work Code (ISWC), which identifies the underlying musical work recorded in the MP3. It can be seen that the DII elements are included in Descriptor/Statement DIDL combinations which are bound to the Resource using a Component. It is important to note, however, that an application will need to be DII aware to find and act upon these identifiers in an intelligent and meaningful manner. The DID is valid regardless of whether a given application understands the DII part of the MPEG-21 standard [16].

C. Registering Identifiers for Usage in DII

We have seen how identifiers can easily be incorporated into the DID, however it is also important to consider the identifiers in some extra detail. In Section III-A it was explained that the DII elements must contain valid URIs. While many internationally used identifiers are now available in this form and possess their own urn: namespace, it is feasible that users of MPEG-21 may wish to use other identifiers not available in this form. To allow this to happen with relative ease, MPEG has created a Registration Authority mechanism that allows identifiers to be used in the MPEG-21 framework to be recognizable. This provides the ability for identifiers to be expressed in the form urn:mpegra:mpeg21:diis:sss:nnn. where the string sss denotes the identifier for an Identification System and nnn denotes a unique identifier within that Identification System.

IV. Binarization of DIDs

The term “Binarization” simply refers to the process of converting XML from the Textual domain into the Binary domain [17]. XML provides a document which ensures human readability in a structured form, however, this has the consequence that it can bloat the raw data. Transmitting or storing these bloated documents requires additional resources which could otherwise be better utilized. This is especially true in mobile environments where bandwidth is often shared and/or limited. Binarization attempts to reduce the overall file size, through compression, whilst retaining enough important information to ensure exact semantic reconstruction of the XML.

MPEG-21 has realized the need for Binarization and a new part of MPEG-21 has been created for the Binary Format (part 16). A number of contributions comparing different XML and data compression techniques were submitted to MPEG during the 68th MPEG meeting in March 2004. The consensus of these MPEG input documents was that BiM [17] should be considered as the starting point as the Binarization tool in MPEG-21. Currently, experiments are underway to determine whether BiM needs changes or additional functionality for use in MPEG-21. Since DID Binarization is still in its infancy, we will only consider some key issues in applying BiM to DIDs.

A. Binarizing DIDs Using BiM

MPEG-7 BiM is the tool used for Binarization of MPEG-7 descriptions. It utilizes a priori knowledge of an XML Schema.
to reduce the amount of transmitted information. This method uses a tree-based compression approach, but furthermore takes advantage of the additional information (i.e., restrictions) present in the schema. To ensure full reconstruction of the binary stream as XML, the receiver must have the same version of the Schema used to generate the binary stream.

To better illustrate the MPEG-7 Binarization process, we shall consider a simple example to demonstrate the BiM concepts throughout this section. To simplify the example, we use the Schema for the DID as shown in Fig. 1 in which only the nodes used in this example have been expanded.

Before the Binarization process can begin, a syntax tree from the included schema(s) is generated. Fig. 7 illustrates an example of part of a syntax tree. This shows the syntax tree for the node \texttt{DID} and only branches to its immediate child nodes. Syntax tree generation creates a tree structure based on schema information.

The Binarization process can be thought of as stepping through the XML and the syntax tree together. For each node in the XML document, decisions about number of bits, mandatory elements etc., are decided from the syntax tree. The binary output is a result of recursively traversing the XML and at each node writing the appropriate bits.

As an example we consider the simple DID shown in Fig. 8. After applying BiM to this DID, the Binarized DID (resulting bitstream) is shown in Fig. 8, as well as the decomposed bitstream, which demonstrates the general process of creating a bitstream for the DID. The ones and zeros in bold are the resulting bits for each node and the reference line numbers indicate the order of the compressed DID bits. As illustrated in the decomposed view, the structure of the binary output retains structure information which corresponds directly to the syntax tree structure. This has the major advantage that applications need not decompress the entire bitstream, but, instead, navigate directly to the sought data. It also allows data to be added and removed without decompressing the entire bitstream. For example, if we decided to add a sub \texttt{Item} to the \texttt{Item} in the DID, we simply change the 0 on line 16 to a 1 and insert the appropriate binary at this point.

BiM’s effectiveness for even this simple DID is demonstrated clearly; the original XML in our example is 345 bytes and after Binarization the size is just 22 bytes, a reduction of 94%. This simple example demonstrates the application of the BiM approach to DID compression but clearly many concepts have not been discussed. Interested readers are referred to [17] for full details on BiM and the future part 16 of MPEG-21[10].

Using this Binarization process of making decisions from the syntax tree does, however, introduce a weakness when applied to DIDs. Since a DID may have any type of embedded content in \texttt{Statement} and \texttt{Resource} nodes, BiM resorts to the use of a default codec (string) to encode the data since it cannot determine its type from the syntax tree. If this data was e.g., XML valid to a given schema, rather than using a default codec, BiM would be applied, and as already demonstrated, even for simple XML this could result in significant savings.

B. Advantages of Binarizing DIDL Second Edition DIDs

Although BiM performs well for MPEG-21 DIDs, utilizing some of the features of DIDL second edition, as well as a DIDL extension to BiM, will increase the efficiency of the compression. Using DIDL second edition the encoder/decoder need not only rely on schema information, but may also utilize information authors provide in the DID. In particular, DIDL second edition proposes changes to and the addition of attributes on the
Statement and Resource elements: namely, mimeType and encoding. The Statement and Resource elements allow any type of data to be embedded into the DID and a traditional schema based compression method would resort to using a generic coder (e.g., gzip). Utilizing these attributes, however, gives the encoder/decoder extra knowledge of the embedded resource, thus making codec selections based on type feasible. Two simple examples of embedded data utilizing this information are as follows.

- Embedded XML, which is valid to a schema—without the use of mimeType to indicate XML, a simple string codec would be used. However, with the mimeType attribute, BiM can be used on this XML.

- Embedded base64 data—conversion of binary to base64 comes with a penalty such that the resulting file is larger than the original file. Utilizing the encoding attribute, direct embedding of the original binary content in the compressed DID is likely the most efficient.

V. CONCLUSIONS

This paper has examined the concept of the MPEG-21 DI, its declaration using the XML Schema based DIDL and has given examples of the usage. The integration role of the DIDL has also been illustrated using examples from the MPEG-21 DIA and the MPEG-21 Rights Expression Language. The linkage provided by DII to identification schemes was considered and the compression of simple DIDs discussed. Through several examples we have sought to show the ability of the DI mechanisms to deliver broad ranging content with related and linked metadata. Further papers in this special issue build on these concepts in specific areas and describe the specifics of metadata schemas standardized in the parts of MPEG-21.

ACKNOWLEDGMENT

The authors would like to thank F. Pereira who was the Associate Editor responsible for this paper—as always, his contribution was beyond that expected. We would also like to thank Dr T. Eyers, a fellow member of Faculty at Wollongong, but, most importantly here, a renowned harmonica player who provided a sample DI music album.

REFERENCES


Ian S. Burnett (M’87–SM’02) received the B.Sc., M.Eng., and Ph.D. degrees in electrical and electronic engineering from the University of Bath, Bath, U.K.

He is an Associate Professor at the University of Wollongong, NSW, Australia. He has been an active participant in MPEG and MPEG-21 in recent years, most notably as Australian Head of Delegation. He is also the CTO of enikos pty ltd. (www.enikos.com), a company specialising in MPEG-21 based software. His current research interests are multimedia processing and delivery, speech and audio coding, three-dimensional audio, and audio separation.

Stephen J. Davis received the B.E. degree in computer engineering in 2001 from the University of Wollongong, NSW, Australia, where he is currently pursuing the Ph.D. degree, funded by the Smart Internet Technology CRC.

His research interests involve the compression and delivery of XML and binarization within the MPEG-21 framework.

Gerrard M. Drury received the B.Math. (computer science) degree from the University of Wollongong, NSW, Australia, in 1990.

He has extensive experience as a Software Engineer, and since 2002 has been working for the Telecommunications and Information Technology Research Institute, University of Wollongong, developing MPEG-21 related software, as well as participating in the MPEG-21 standardization process.