SMPTE STANDARD

Types Dictionary Structure

Table of Contents

Page

Foreword 2
Intellectual Property 2
1 Scope 3
2 Conformance Notation 3
3 Normative References 3
4 Types Dictionary Structure 4
4.1 Compatibility with Other Data Structures 5
4.2 Individual Classes of Types 5
4.3 Derivation of Types (Informative) 12
4.4 Dictionary Structure and Format 13
5 Types Dictionary Maintenance 19
5.1 Dictionary Version Information 19
5.2 Dictionary Management and Compatibility Requirements 19
5.3 Dictionary Availability 20
Annex A Glossary of Terms (Normative) 21
Annex B Registration Criteria (Normative) 23
B.1 Criteria for Modifications to Entries in Classes 1-7 23
B.2 Criteria for Modifications to Entries in Class 13 23
B.3 Criteria for Modifications to Entries in Class 14 24
Annex C Organization of References (Informative) 25
Annex D Bibliography (Informative) 26

Approved
January 13, 2012

Foreword

SMPTE (the Society of Motion Picture and Television Engineers) is an internationally-recognized standards developing organization. Headquartered and incorporated in the United States of America, SMPTE has members in over 80 countries on six continents. SMPTE's Engineering Documents, including Standards, Recommended Practices, and Engineering Guidelines, are prepared by SMPTE's Technology Committees. Participation in these Committees is open to all with a bona fide interest in their work. SMPTE cooperates closely with other standards-developing organizations, including ISO, IEC and ITU.

SMPTE Engineering Documents are drafted in accordance with the rules given in Part XIII of its Administrative Practices.

SMPTE ST 2003 was prepared by Technology Committee 30MR on Metadata and Registers.

Intellectual Property

At the time of publication no notice had been received by SMPTE claiming patent rights essential to the implementation of this Standard. However, attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. SMPTE shall not be held responsible for identifying any or all such patent rights.

1 Scope

This standard defines the structure of a dictionary of types for all registers defined by SMPTE ST 336. These types may be used in a range of applications such as production workflow applications, data exchange formats, and archival asset management systems.

The standard normatively defines universal identifiers, type names, definitions, and standardized symbols, as well as other normative and informative fields.

Applications of individual entries will vary, but, when used, shall conform to the definitions and formats in the types dictionary.

2 Conformance Notation

Normative text is text that describes elements of the design that are indispensable or contains the conformance language keywords: "shall", "should", or "may". Informative text is text that is potentially helpful to the user, but not indispensable, and can be removed, changed, or added editorially without affecting interoperability. Informative text does not contain any conformance keywords.

All text in this document is, by default, normative, except: the Introduction, any section explicitly labeled as "Informative" or individual paragraphs that start with "Note:"

The keywords "shall" and "shall not" indicate requirements strictly to be followed in order to conform to the document and from which no deviation is permitted.

The keywords, "should" and "should not" indicate that, among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others; or that a certain course of action is preferred but not necessarily required; or that (in the negative form) a certain possibility or course of action is deprecated but not prohibited.

The keywords "may" and "need not" indicate courses of action permissible within the limits of the document.

The keyword “reserved” indicates a provision that is not defined at this time, shall not be used, and may be defined in the future. The keyword “forbidden” indicates “reserved” and in addition indicates that the provision will never be defined in the future.

A conformant implementation according to this document is one that includes all mandatory provisions ("shall") and, if implemented, all recommended provisions ("should") as described. A conformant implementation need not implement optional provisions ("may") and need not implement them as described.

Unless otherwise specified, the order of precedence of the types of normative information in this document shall be as follows: Normative prose shall be the authoritative definition; Tables shall be next; followed by formal languages; then figures; and then any other language forms.

3 Normative References

Note: All references in this document to other SMPTE documents use the current numbering style (e.g. SMPTE ST 298:2009) although, during a transitional phase, the document as published (printed or PDF) may bear an older designation (such as SMPTE 298-2009). Documents with the same root number (e.g. 298) and publication year (e.g. 2009) are functionally identical.

The following standards contain provisions which, through reference in this text, constitute provisions of this recommended practice. At the time of publication, the editions indicated were valid. All standards are subject to revision, and parties to agreements based on this recommended practice are encouraged to investigate the possibility of applying the most recent edition of the standards indicated below.

SMPTE ST 298:2009, Universal Labels for Unique Identification of Digital Data

SMPTE ST 336:2007, Data Encoding Protocol Using Key-Length-Value

SMPTE ST 2029:2009, Uniform Resource Names for SMPTE Resources

ISO/IEC 646:1991, Information Technology — ISO 7-Bit Coded Character Set for Information Interchange

W3C Recommendation — Namespaces in XML 1.0 (Second Edition), World Wide Web Consortium, 16 August 2006. http://www.w3.org/TR/REC-xml-names/

4 Types Dictionary Structure

The types dictionary structure provides flexibility in capturing data and exchanging it among applications through a standardized hierarchy of universal labels (ULs) that uniquely identify the types, grouped to aid their management within a small but comprehensive number of classes. Type classes are collections of types with common characteristics or attributes. Additional classes are provided for user-defined public, private, and experimental types.

The types dictionary defined by this document provides two methods of referencing an individual item. The first is to use a unique, two-part, 16-byte universal label that is numerical (and hence language independent). The second method of referencing an item is to use its assigned symbol, which is a name that conforms to computer language syntax restrictions. Symbols are intended for use in computer languages such as the Extensible Markup Language (XML). The symbol shall be unique within a namespace that has been defined according to Section 4.4.7.

Note: The symbol, together with its namespace defined in Section 4.4.7, forms a unique identifier like the UL.

The KLV coding of data items and groups of data items is defined in SMPTE ST 336. The structure of the metadata element dictionary is defined in SMPTE ST 335, including the requirement for a type entry for each metadata element. This document defines the structure of a dictionary for type entries. The associated types registry includes all entries which have been approved according to the specific procedures defined in Annex B.

The exact format of the universal label shall be as defined in SMPTE ST 336. The first eight bytes of the universal label shall consist of the UL Header (2 bytes) and UL designator (6 bytes). The UL designator shall identify the item as belonging to a specific SMPTE register of a given category, structure, and version. The second eight bytes shall form the item designator as defined in SMPTE ST 336. The item designator shall be used to uniquely identify the meaning or definition of the item in the register.

The types dictionary shall be organized into nodes, leaves and children. The dictionary classes form class nodes and below these are further nodes at each subclass. To aid the management of the dictionary, these nodes and subnodes shall be assigned a universal label, so as to give clear breaks in the structure. Entries within a subclass form leaves, which are the type descriptions. Some classes of type require extra elements for a complete description, these elements are children and shall be regarded as part of the description of the parent type.

The universal labels used in the types dictionary defined by this document shall be constructed as shown in Table 1, which complies with SMPTE ST 336.

Table 1 – Construction of universal labels for the types dictionary

Byte Position	Description	Value	Meaning
	UL Header		
1	Object identifier	06h	Object identifier tag per SMPTE ST 298
2	UL length	0Eh	The byte length of the object identifier value is 14 bytes.
	UL designator		
3	UL code	2Bh	The administering organization is an ISO organization.
4	UL subcode	34h	The delegated organization is SMPTE.
5	Registry category designator	01h	The registry category is dictionaries.
6	Registry designator	04h	The specific register is a types dictionary.
7	Structure designator	01h	The dictionary structure conforms to this SMPTE standard.
8	Version number	01h to 7Fh	This indicates the version number of the register.
9-16	Item designator	Defined by the types dictionary	This identifies a specific type within the types dictionary.

Note: As defined in SMPTE ST 298, a value of 00h at any position in a UL is treated as a terminator and all further values within that UL are required to be zero also.

4.1 Compatibility with Other Data Structures

The types dictionary structure is a framework that supports global interoperability by defining types in a way that enables the interchange of metadata from different sources, applications, and third-party organizations.

Note: Many different cataloging conventions are used by communities who focus on a specific domain or subject or who have specific needs for archive and retrieval of multimedia data including, for example, intellectual rights. The types dictionary is not intended to replace conventions already in use, for example in textual naming or keywords. Within the framework of the types dictionary structure, different content creation communities, media indexing professionals, or metadata extractors and users can develop metadata conventions that meet their specific requirements.

4.2 Individual Classes of Types

Within the types dictionary, types shall be organized into a hierarchical structure, where each is assigned to a type class as shown in the overview of Figure 1. The initial set of type classes in this standard consists of:

Class 1: Basic Types
Class 2: Enumerated Types
Class 3: Record Types
Class 4: Multiples
Class 5: Reference Types
Class 6: Choice Types

Class 8: Controlled Vocabularies
Class 13: Organizationally registered for public use
Class 14: Organizationally registered as private
Class 15: Experimental

These classes are further subdivided as described in the sections below.

The number of type classes can be extended in the future to a maximum of 127, and the class numbers that have not been assigned here shall be reserved for use by SMPTE.

The processes for registration of new types shall be as specified in normative Annex B.

Byte 9 of the UL identifies which of these type classes a metadata element belongs to. Subsequent bytes enable the hierarchical identification of subclasses.

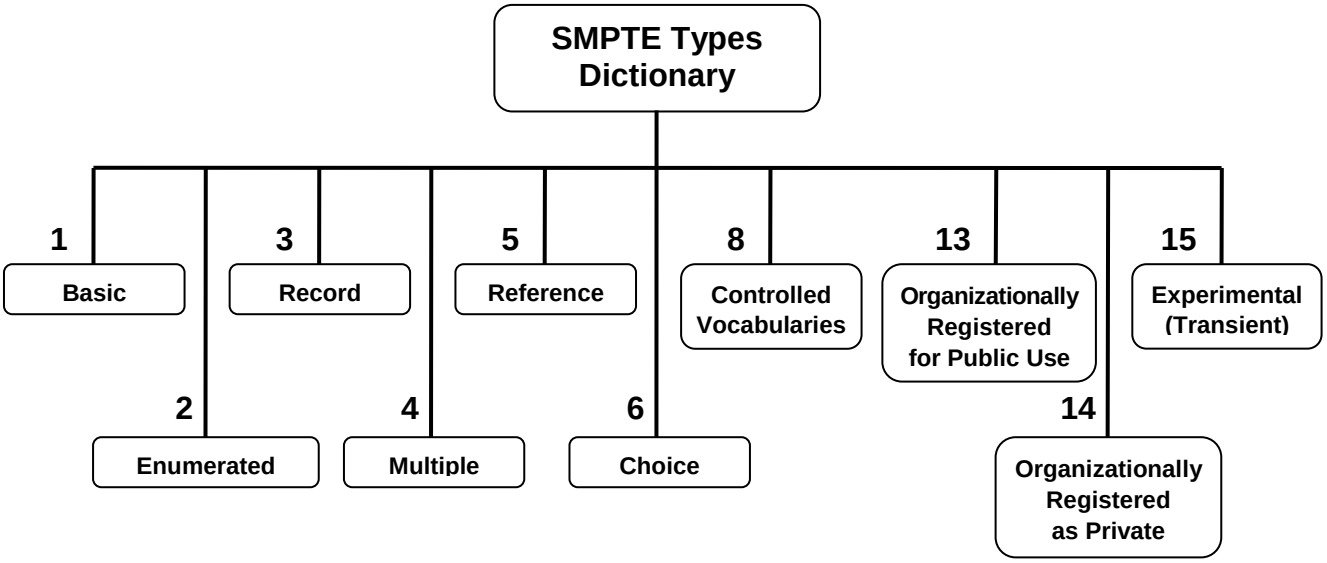

Figure 1 – Types class structure

4.2.1 Class 1: Basic Types

Types in this class shall be simple derivatives of bytestrings of various lengths, with or without significance attached to the most significant bit.

Note: When numeric data is stored in binary some types are formatted such that the most significant bit is used to store information such as the sign of the number. Basic types can be used to hold values formatted with this extra information, or without. For example a single-byte basic type could hold a signed integer between -128 and +127, or an unsigned integer between 0 and 255.

Sub-classes in this Class include:

- Integer Types
- Renamed Integer Types
- Number Types
- Inseparable Bytestrings
- Boolean Types

- Character Types
- Renamed Character Types

Note: Renamed types are used where a type is defined that is not fundamentally different in representation to another type, but the two are treated by other specifications as separate types. For example SMPTE ST 377-1 defines types called "Position" and "Length" which are 64-bit signed integers.

Other subclasses may be added through the process defined in Annex B.1.1.

4.2.2 Class 2: Enumerated Types

Types in this class shall consist of a base type, whose allowable values are controlled and have a symbolic identifier or “token” assigned to them. They are similar to `enums` in the C programming language, except that the base type may be any other type.

Enumerated Types where the base type is an integer shall be Enumerated Integer Types. Enumerated Types where the values are assigned using a method that can be guaranteed to yield universally unique numbers (such as the RFC 4122 UUID Algorithms) may be classed as Extendible Enumerations.

The definition of an Enumerated Integer Type shall include a list of all permitted values and the equivalent symbolic identifier for each.

The definition of an Extendible Enumeration may include a non-exclusive list of values with the equivalent symbolic identifier for each, or the definition may include a document that defines the register holding such values for that type. In this case the register shall be specified in the Defining Document field of the entry.

Sub-classes in this Class include:

- Enumerated Integer Types
    - Enumerated UInt8 Types
- Extendible Enumerations (see description above)

Other subclasses may be added through registration per Annex B.1.1.

4.2.3 Class 3: Record Types

Types in this class are all composed of fixed structures of other types. They are similar to `Records` in the Pascal programming language, `structs` in the C Programming Language, and `sequences` in XML.

The definition of a Record type shall include a list of all the component sub-types and the symbolic identifier used to refer to each, in the order in which they must appear in instances of the type.

Sub-classes in this Class include:

- Simple Record Types

Other subclasses may be added through registration per Annex B.1.1.

4.2.4 Class 4: Multiples

Types in this class are all composed of multiples of a single base type. The instances may be individually identified or anonymous, they may be ordered or unordered, and there may be an implicit or explicit count of instances, and instances may be of fixed or variable size.

The definition of a Multiple Type shall include the UL of the base type and the qualifiers necessary to describe the composition of the multiple.

Sub-classes in this Class include:

- Simple Arrays
- Record Arrays
- Simple Sets
- Bytestream Types

Other subclasses may be added through registration per Annex B.1.1.

The following terms should be used in the names of types having the qualifiers listed:

Table 2 – Terms for Multiple Types

Elements are	Elements are	Count of elements is	Size of each element is	Term
Anonymous	Ordered	Explicit	Fixed	***Array***
Identified	Unordered	Explicit	Fixed	***Set***
Identified	Ordered	Explicit	Fixed	***Vector***
Anonymous	Ordered	Implicit	Fixed	***Table***
Identified	Unordered	Implicit	Fixed	***Bag***
Identified	Ordered	Implicit	Fixed	***Series***
Anonymous	Ordered	Explicit	Variable	***Cluster***
Identified	Unordered	Explicit	Variable	***Bundle***
Identified	Ordered	Explicit	Variable	***Stack***
Anonymous	Ordered	Implicit	Variable	***StringTable***
Identified	Unordered	Implicit	Variable	***StringString***
Identified	Ordered	Implicit	Variable	***StringSeries***

Note: The term “Set” must not be confused with the group coding syntax defined by SMPTE ST 336.

An illustration of the derivation of Multiple types is given in Figure 2. The key of a KLV triplet identifies the metadata element, “Creatives”, in the metadata element dictionary. The metadata element dictionary, in turn, identifies the type, “Cluster of”, by means of a key that indexes the definition in the types dictionary. The definition of the Base Type, “ISO7”, is then indexed in the types dictionary using the Base Type Key.

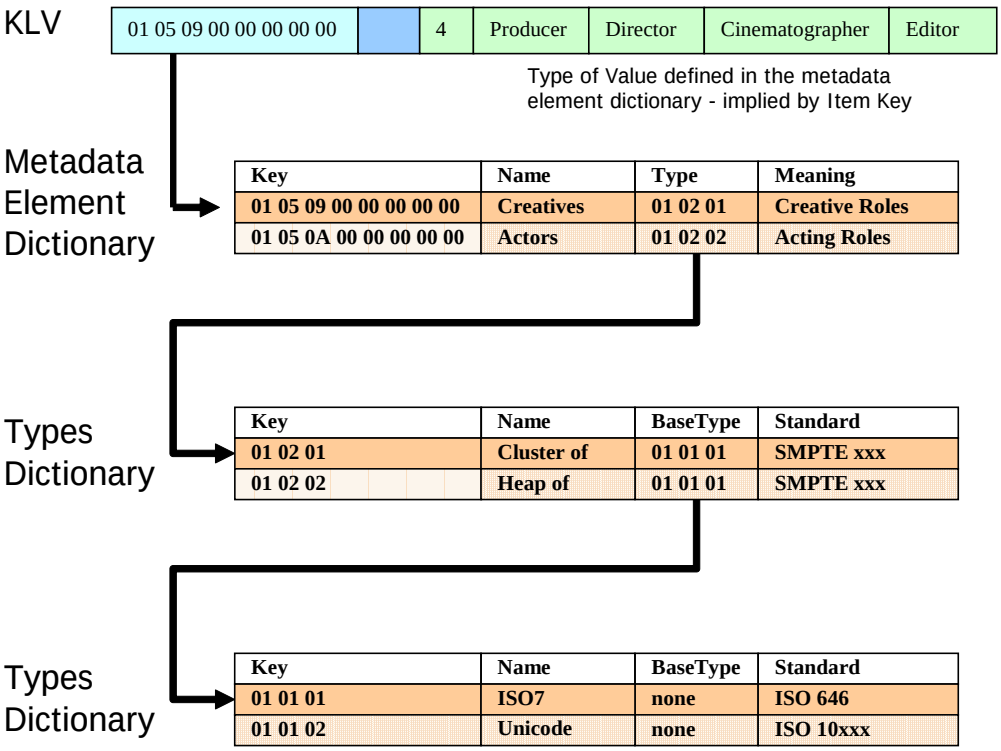

Figure 2 – Example of derivation of Multiple Types

An illustration of the representations of Multiple types is given in Figure 3 and Figure 4. This shows that Multiple types whose elements correspond to groups can be represented in two ways, either by reference to the groups or by directly embedding them.

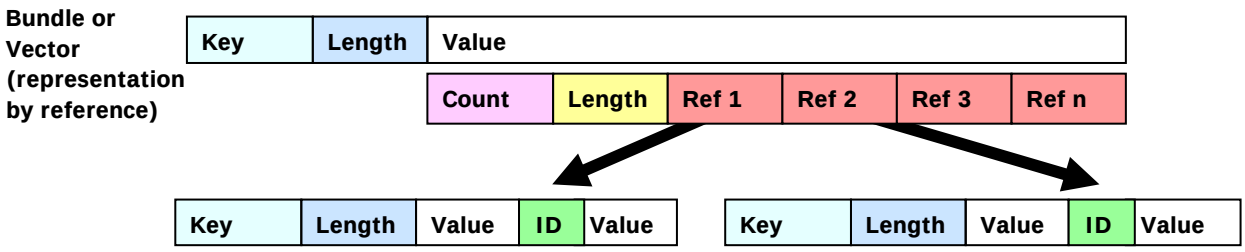

Figure 3 – Example of Multiple Types used for Aggregation by Reference

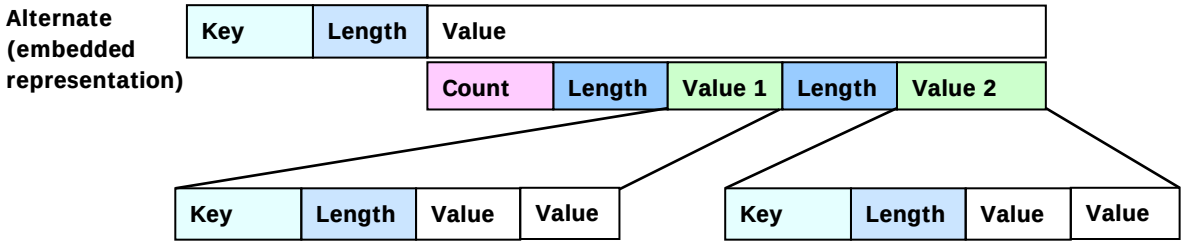

Figure 4 – Example of Multiple Types used for Aggregation by Embedding

Note: The Material Exchange Format (MXF) and the Advanced Authoring Format (AAF) do not use embedding.

4.2.5 Class 5: Reference Types

Types in this class are all composed of single and multiple instances of references to base types. The references may include strong and weak references, and may be references to any registered element, type or group.

The definition of a Reference Type shall include the UL of the type to which instances of the new type will refer.

Sub-classes in this Class include:

- Strong References
- Strong Reference Sets
- Strong Reference Vectors
- Weak References
- Weak Reference Sets
- Weak Reference Vectors
- Global Reference Sets

Other subclasses may be added through registration per Annex B.1.1.

An illustration of the representation of Reference Types is given in Figure 5. The owner of the target group has a strong reference to that group, but third parties may have weak references to the same group. In other words, a strong reference is a one-to-one relationship implying ownership, whereas a weak reference is a many-to-one relationship.

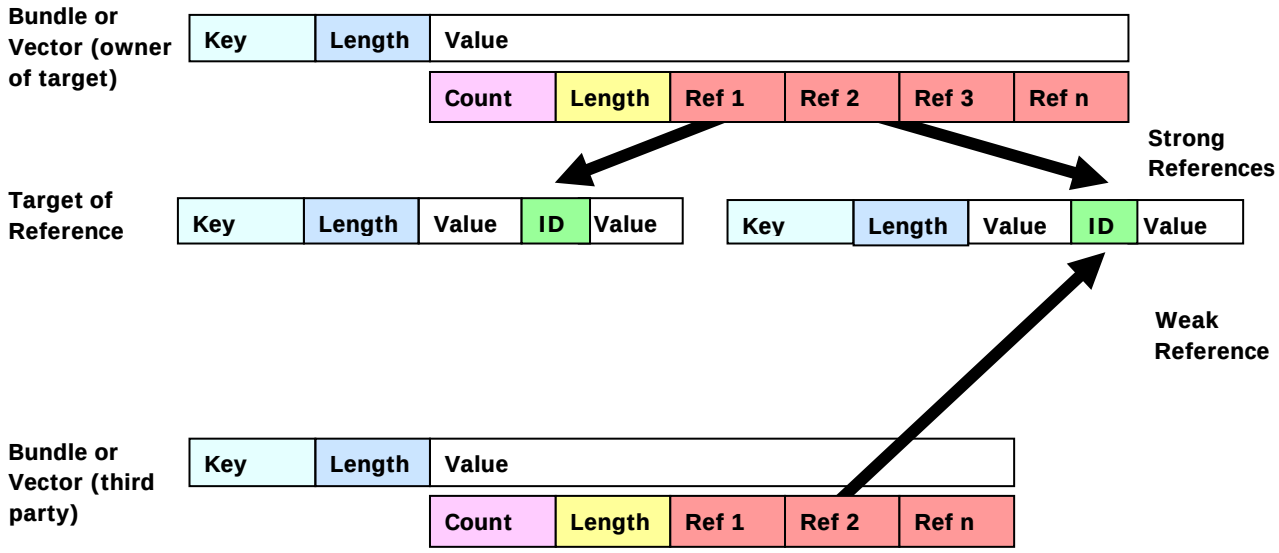

Figure 5 – Example of representation of Reference Types

4.2.6 Class 6: Choice Types

Types in this class are all composed of one of a number of defined alternative base types. They are similar to `unions` in the C programming language and to XML `choice`.

The definition of a Choice Type shall include a list of one or more alternate base types. Values of the Choice Type may have any legal value of one of the alternates.

Subclasses may be added through registration per Annex B.1.1.

An illustration of the derivation of Choice Types is given in Figure 6. The key of a KLV triplet identifies the value as a “UID” in the metadata element dictionary. The type key for this element indexes the type definition in the types dictionary, which indicates that the type is a “Choice of” one of two base types, each identified by a key. In this example, the definitions of the identified base types, “UMID” and “ISAN”, are contained in the metadata element dictionary. These metadata elements in turn have a type, specified by a type key.

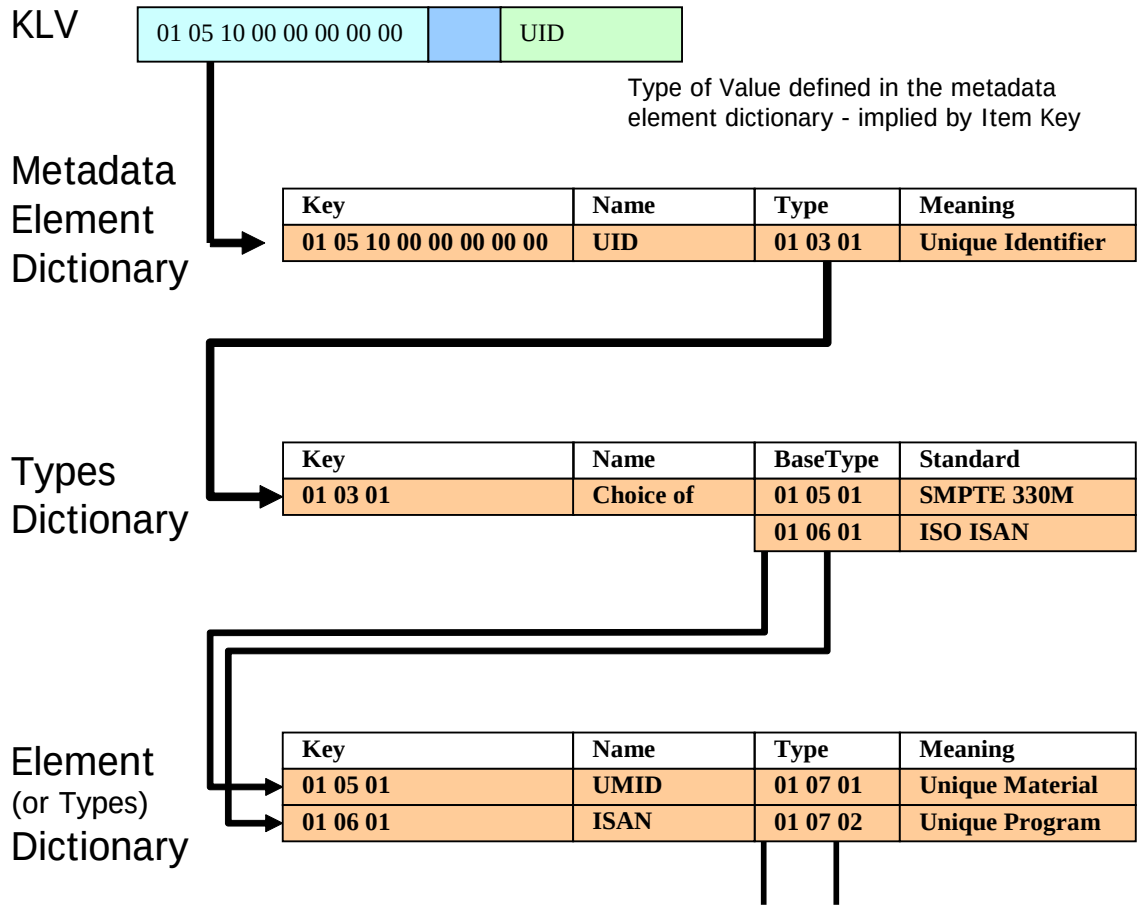

Figure 6 – Example of derivation of Choice Types

4.2.7 Class 8: Controlled Vocabularies

Types in this class shall consist of elements that have defined values held in a register which also adds further semantic definitions. The definition of a Controlled Vocabulary Type include the document that defines the register holding permitted values for that type. The register shall be specified in the Defining Document field of the entry.

4.2.8 Class 13: Organizationally registered for public use

Types in this class shall consist of elements that have been registered by a specific organization and are therefore reserved and managed separately from the other classes (1-7) of types. Subclasses and types used by the class 13 registrant shall be published in the types dictionary. Class 13 types shall be managed by the SMPTE Registration Authority and its approval shall be consistent with Annex B.2.

4.2.9 Class 14: Organizationally registered as private

Types in this class shall consist of individual elements whose definitions are held by a specific organization and are therefore reserved and managed separately from the other classes (1 -7) of types.

Each allocated top-level node shall be publicly identified in the types dictionary and universal labels under that shall be reserved for use by the registered organization. Types elements used by the class 14 registrant shall not be published in the types dictionary. Allocation of top-level nodes in class 14 shall be managed by the SMPTE Registration Authority and its approval shall be consistent with Annex B.3.

Note: If a device or application is required to parse metadata instances with types defined in class 14, it will require a definition that is supplied by the owning organization rather than attempting to use any version of the SMPTE register defined by this document.

4.2.10 Class 15: Experimental

Class 15 metadata shall only be used in multimedia research or other limited access, experimental environments where experimentation with new types and applications does not depend on strict conformance to approved standards and which remain within a test or laboratory environment.

4.3 Derivation of Types (Informative)

The definitions of types can stand alone, or they could be constructed from the definitions of other types. This is shown pictorially below in the following two figures.

In Figure 7, which gives an example of a stand-alone type definition, the type of the “Version Title” metadata element is identified as “ISO7” via its type key. The definition of the ISO7 type is defined by the ISO 646 standard and does not reference any other types.

In Figure 8, the type “Cluster of” is a constructed type. This is because “Cluster of” is constructed from the base type “ISO7”.

The metadata fields required to define the derivation of types by construction are defined in Section 4.4.

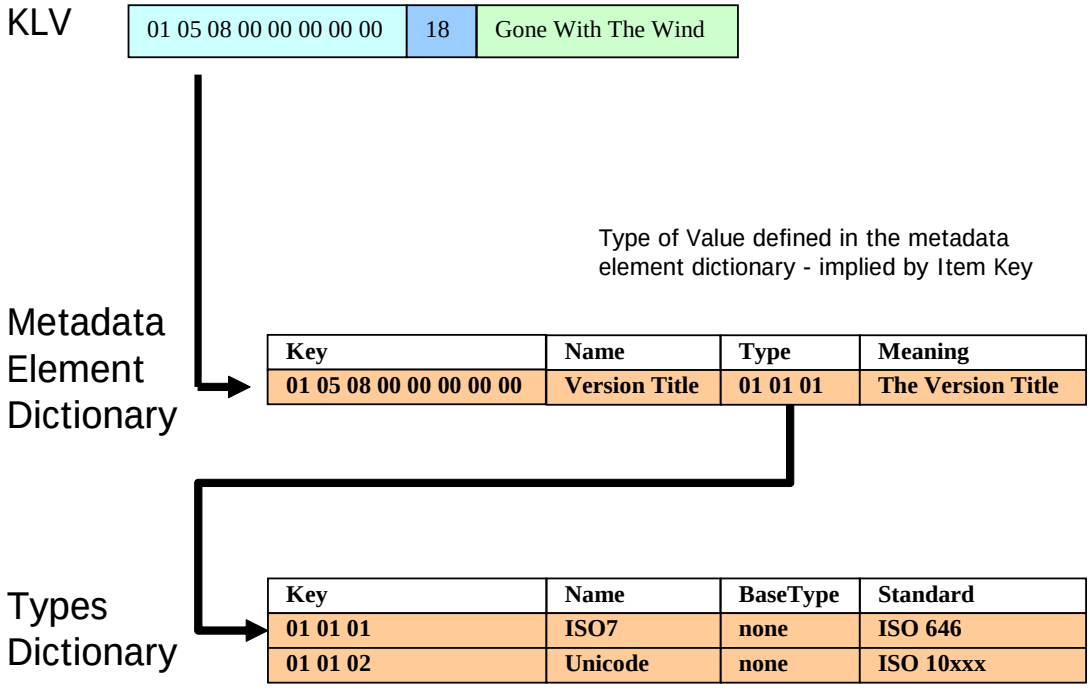

Figure 7 – Example of standalone types

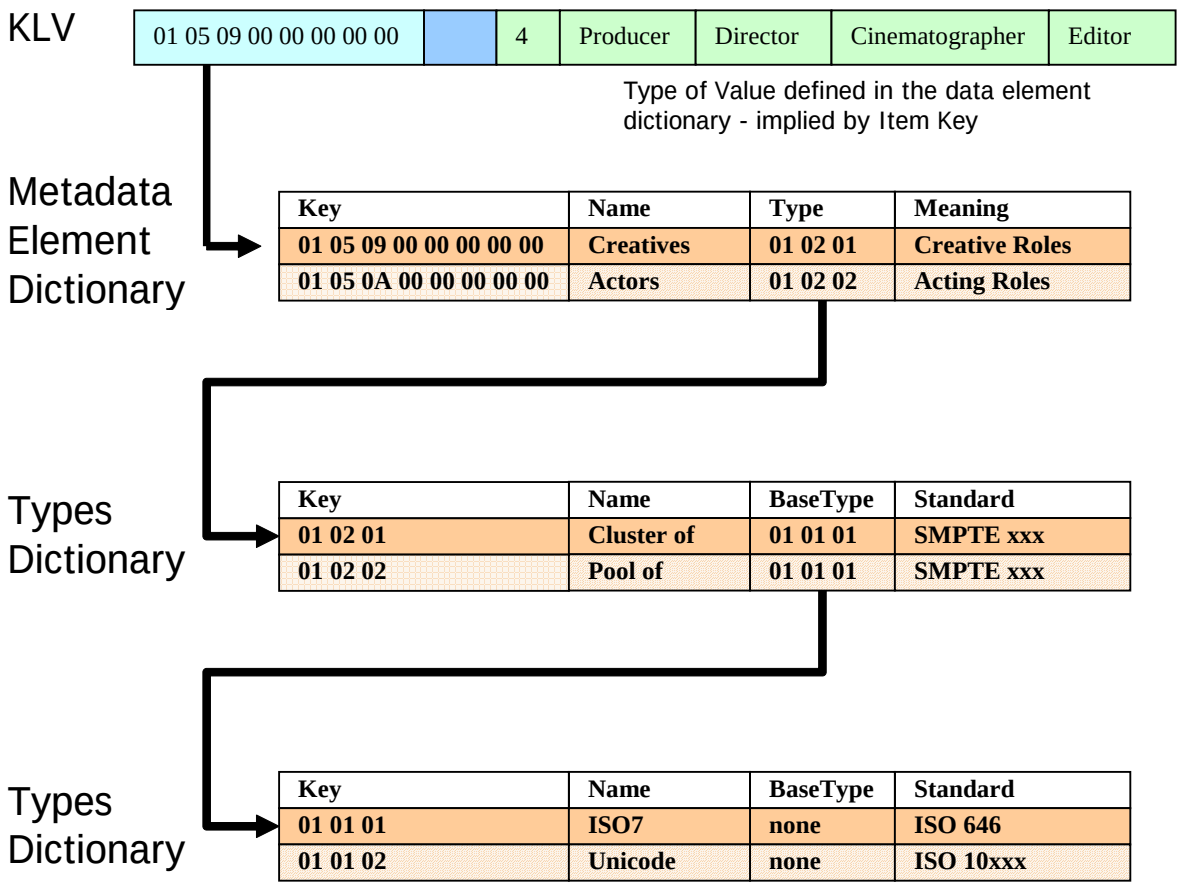

Figure 8 – Example of types constructed from other types

4.4 Dictionary Structure and Format

Each types element or types class/subclass shall be described by a number of fields, which are classified as normative, informative, queried, stated, or calculated.

The fields that apply to leaves or nodes are indicated in the following subsections. In the types dictionary, nodes describe classes/subclasses and leaves describe types elements.

Normative fields of the types dictionary are those that are required for the description of a types element or class/subclass. They may be omitted only if they are not applicable.

Informative fields provide additional information that is intended to help users of the types dictionary. In the case of a conflict between an informative field and a normative field, the normative field shall take precedence.

Queried fields are derived by using an identifier (universal label) to look up information contained in the normative fields of another element in the types dictionary.

Stated fields contain values that were stated by the originator of the item and were used to derive the normative fields. Derivation may be manual (i.e., by the document editor) or calculated by formula. In some cases, derived normative fields belong to a different register, such as the Types Register.

Calculated fields are non-normative fields that are derived from other fields. Note that this definition excludes derived normative fields. In the case of conflict between a calculated field and a normative field, the normative field shall take precedence.

The following sections list the specific normative and informative fields. Descriptions of the stated, calculated, and queried fields are also given. The order in which the fields appear in the register need not reflect the order in which they are presented below.

Table 3 – Classification of fields used in the types dictionary

Field(s)	Description	Classification	Scope	Required?	Format
Register	4.4.1	normative	nodes, leaves, children	Required	Text (enumerated)
Dictionary version at introduction	4.4.2	calculated	nodes, leaves	Derived	Integer
Node/Leaf/Child	4.4.3	normative	nodes, leaves, children	Required	Text
Level	4.4.4	calculated	nodes, leaves, children	Derived	Integer
SMPTE designator and item designator	4.4.5	stated	nodes, leaves, some children	Required for derivation of normative UL	16 hexadecimal bytes (as individual fields)
URN representation of the universal label	4.4.6	normative	nodes, leaves, some children	Required	URN representing the UL
Namespace Name	4.4.7	normative	nodes	Required for top-level class 13/14 node	URI
Symbol	4.4.8	normative	nodes, leaves	Required	Text restricted to character pattern defined in 4.4.8
Name	4.4.9	normative	nodes, leaves	Required	Text
Definition	4.4.10	normative	nodes, leaves, children	Required	Text
Type kind	4.4.11	normative	leaves	Required	Text (enumerated)
Type size	4.4.12	normative	leaves	Required for some leaf kinds	Unsigned Integer
Base type	4.4.13	normative	leaves	Required for some leaf kinds	URN representing the UL
Base type symbol	4.4.14	queried	leaves	Required for some leaf kinds	Text restricted to character pattern defined in [fill-in]
Type qualifiers	4.4.15	normative	leaves	Optional	Batch of Enumeration
Facets	4.4.16	normative	leaves	Required for some leaf kinds	Array of Records
Facet symbol	4.4.16	normative	children	Required for children that are facets	Symbol
Facet name	4.4.16	normative	children	Required for children that are facets	Text (enumerated)
Facet type	4.4.16	normative	children	Required for children that are facets	URN representing the UL
Facet value	4.4.16	normative	children	Required for children that are facets	Text

Facet parent	4.4.16	normative	children	Required for children that are facets	URN representing the UL
Defining document	4.4.17	normative	nodes, leaves	Optional	Text
Context scope	4.4.18	informative	leaves	Required	Text (enumerated)
Applications	4.4.19	informative	nodes, leaves, children	Optional	Text
Notes	4.4.20	informative	nodes, leaves, children	Optional	Text
isDeprecated	4.4.21	normative	nodes, leaves, children	Required	Boolean

When any version of the register is made available for ballot, or published in any form, all fields classified in Table 3 as normative or informative shall be included. Other fields may be included.

4.4.1 Register

This normative field shall identify the register to which an item belongs, for example, “Elements”, “Groups”, “Types”, or “Labels”. For the dictionary defined by this standard, this field shall have the value “Types”.

4.4.2 Dictionary version at introduction

This calculated field shall record the version number of the dictionary which first recorded the allocation of a type or class/subclass description against its UL.

For all entries in classes 1-7, and for any nodes registered in class 14, the version number shall be the version of the dictionary which first contained the entry. Class 13 entries may use the same method for selecting version numbers, or another method at the discretion of the organization responsible for those entries.

Byte 8 of the UL shall also hold the version number.

4.4.3 Node/Leaf/Child

This normative field shall define whether the item is a node, leaf, or a child as defined in Section 4.

4.4.4 Level

This calculated field shall indicate the level of an item in the class hierarchy of the types dictionary. It shall be calculated from the position of the last active byte of the item designator.

4.4.5 SMPTE designator and item designator

These stated fields shall contain the 16 individual bytes of the universal label in hexadecimal notation.

The SMPTE designator (first 8 bytes) shall be consistent with the provisions of Table 1.

The item designator (last 8 bytes) uniquely identifies the specific item in the dictionary in a hierarchical fashion. Classes are designated with the first byte in the item designator and subsequent bytes enable the hierarchical identification of subclasses and/or individual types. The item designator shall reflect the class/subclass that is most appropriate for that item and shall not be identical to the item designator assigned to another item in any draft or published version of the dictionary.

4.4.6 URN representation of the universal label

This normative field shall be derived from the value of the universal label as defined by Section 4.4.5 and represented in a normative text format that has been approved for use in SMPTE registers. This provides a consistent textual format for representing the underlying UL.

The normative text format of the universal label is the internationally recognized Uniform Resource Name (URN) notation. SMPTE ST 2029 defines the urn:smpte:ul representation of a SMPTE ST 298 UL, which shall be used for this value.

4.4.7 Namespace Name

This normative field shall define the scope over which symbols, which are defined in Section 4.4.8, are unique. For classes 1-7 the Namespace Name shall be identified by the Uniform Resource Identifier (URI) http://www.smpte-ra.org/reg/2003/<revision>, where <revision> shall be a string denoting the year of publication of this structure standard as 4 decimal digits, and may be appended with 2-month decimal digits in the range 01-12. For classes 13 and 14, a Namespace Name shall be specified by the registrant for the top-level node and may be specified for any sub-node.

Organizations that have defined types in classes 13 and 14 may choose any valid Namespace Name (URI) for the symbols identifying these types in accordance with the XML-Namespace recommendation; this is subject to the restriction that this Namespace Name shall not correspond to the XML namespace used for class 1-7 symbols. The Namespace Name (URI) shall be specified as a normative field of the class 13 or 14 node to which this namespace applies. All sub-nodes shall belong to this namespace, unless another namespace has been specified. Therefore, if no Namespace Name (URI) for a node is given, it shall be inherited from the ancestor node.

4.4.8 Symbol

This normative field shall define the symbol that identifies a class/subclass or type. A symbol is a name that conforms to computer language syntax restrictions, and it is intended for use in computer languages such as the Extensible Markup Language (XML). To enable the use of symbols in a wide range of computer languages, a symbol shall be a string composed only of the characters A-Z, a-z, 0-9, and _, and it shall begin with an alpha character (A-Z, a-z) or an underscore (_).

Symbols shall be defined for both nodes and leaves and shall be unique within the XML namespace identified by Section 4.4.7.

Note 1: This implies that class 1-7 symbols are unique within the types dictionary.

Note 2: The combination of Namespace Name and Symbol produces a unique identification for a type in the same way that namespace name and local name are used in W3C REC-xml-names to form an expanded name.

4.4.9 Name

This normative field shall be the name for the type or class/subclass identified by the universal label or symbol. It shall be written in U.S. English. The name should use title case capitalization and should follow SMPTE guidelines for element or node naming to ensure the name is descriptive and unambiguous.

4.4.10 Definition

This normative field shall be the detailed and unambiguous definition of the item or class/subclass. It shall be written in U.S. English.

4.4.11 Type Kind

This normative field shall specify whether the type is a basic type or is composed out of another type or types. This entry selects which type qualifiers are meaningful for this type, and which if any facets must be specified.

4.4.12 Type Size

This normative field shall give the size of the data contained in the type, in bytes. The number does not include any overhead needed to construct the type (such as number of elements for a vector, or a terminating null for a string). For multiples, this entry shall give the number of instances (if it is fixed) or zero (if the count is not fixed).

4.4.13 Base Type

For non-basic types or renamed types, this normative field shall specify the UL of the underlying type from which the type is built. For example, a new type called ActivationMode might be an Enumeration of base type UInt8 (it would have Base Type set to the UL of the entry named "UInt8"). An array of these values may be defined as a Multiple Type and its Base Type would be the UL of ActivationMode.. The format of this field shall be as specified in Section 4.4.6.

Note: Despite the similarity in names, the value of base type is not limited to only basic types.

4.4.14 Base Type Symbol

This queried field shall provide the symbol of the base type.

4.4.15 Type Qualifiers

This normative field shall contain an unordered list of attributes of the type (it may be empty if none apply).

Allowable qualifiers for basic types are:

isNumeric –	shall be present if instances of the type may sensibly be treated as numbers
isSigned –	shall be present if instances of the type may sensibly be treated as signed numbers

Note: When isSigned is used isNumeric will also need to be used.

Allowable qualifiers for multiples, as defined by Section 4.2.4, are:

isIdentified –	shall be present if every instance of the base type includes a unique identifier (to distinguish it from other instances)
isOrdered –	shall be present if the instances of the base type are in a significant order
isCountImplicit –	shall be present if the number of instances of the base type must be inferred from the context (i.e., is not explicitly stated)
isSizeImplicit –	shall be present if the size of each instances of the base type must be inferred from the context (i.e., is not explicitly stated)

4.4.16 Facets

Some types require a list of related items to complete their definition. These items are called facets. Facets shall be used as follows:

- Records shall include an ordered list of facets defining the members of the record;
- Enumerations shall include an unordered list of facets defining the permitted values and their meanings (these are called tokens);
- Choices shall include an unordered list of facets defining the valid alternates.

Each facet shall have the following fields:

– **Facet Symbol**
This normative field shall be the symbolic identifier of the member, token or alternate. It shall follow the same format as defined in Section 4.4.8

– **Facet Name**
This normative field shall be the name of the facet. It shall be written in U.S. English. The name should use title case capitalization and should follow SMPTE guidelines for element or node naming to ensure the name is descriptive and unambiguous.

– **Facet Type**
This normative field shall be the UL of the Type of the member or alternate, formatted as per Section 4.4.6. The facet type shall be omitted for token.

– **Facet Value**
This normative field shall be the value of the token.

– **Parent**
This normative field shall give the UL of the type to which the facet belongs, formatted as per Section 4.4.6.

4.4.17 Defining document

This normative field shall reference the primary standard or authoritative document that provides further information about an item, if such a reference is available. For example, a floating point type could reference the document that defines the binary representation of the value.

4.4.18 Context scope

This informative field shall indicate if a type has been defined within a specific application or context.

- Defined context: A type that has a defined context has been defined within a specific application or context. An example is a type that is defined for use by a specific element defined in the metadata elements register. Other applications may use this type provided that due regard is given to the data relationships defined by the original application or context.

- Abstract context: A type that is defined as abstract may be used anywhere that the semantics of the type description apply. Abstract types can be used to map and re-use data across diverse applications.

- Unknown: A type with an unknown context scope has a context scope that has not been determined. Note: This means that the type description is incomplete. It is a warning that the interpretation of a data value may differ between applications.

The values used to identify the context scope shall be “DefinedContext”, “AbstractContext”, and “UnknownContext”, respectively.

4.4.19 Applications

This informative field shall be an informative listing of some known applications that use a particular type.

4.4.20 Notes

This informative field may be used to provide additional information that may assist in the interpretation and correct application of the type or a class/subclass of the types dictionary. This information cannot be deduced from the other normative and informative fields.

4.4.21 isDeprecated

This normative field is an indication to system designers that the type should no longer be used.

The field shall contain a boolean value that is true for types that have been classified as deprecated according to the processes described in Annex B.1.3 or Annex B.2.3. All other entries shall carry the value false. Where a node is flagged as deprecated no new nodes or leaves shall be allocated under that node. Leaves under a deprecated node may be flagged as deprecated or left usable.

Where a leaf is flagged as deprecated, all children under that leaf shall also be flagged as deprecated. Children under a non-deprecated leaf may be flagged as deprecated to indicate that values of an enumeration or choice type should no longer be used. Children of record types shall not be flagged as deprecated unless the entire type is deprecated.

Note: Deprecation can be used in situations where it has been determined that the entry is erroneous or could cause compatibility problems, so great care is required to avoid them. However some situations can require use of a type after it has been deprecated, such as reading from a large archive of material that contains elements of the deprecated type. In these situations developers will need to exercise great caution.

5 Types Dictionary Maintenance

The principles for maintenance and administration of the types dictionary are defined in the following clauses:

5.1 Dictionary Version Information

The following information shall be provided by the SMPTE Registration Authority with each update to the types dictionary:

Standard name: Types Dictionary

Structure designator: One-byte unsigned integer that indicates that the types dictionary is defined by this structure document. The structure designator shall have a value of 01h.

Version number: One-byte unsigned integer in the range of 1 to 127

Effective date: Date of publication of any updates to the register on the web site of the SMPTE Registration Authority (www.smpte-ra.org) as provided by the SMPTE Administrative Practices.

Register Administrator: SMPTE Registration Authority

Contact information: Text provided on the SMPTE Registration Authority's website, www.smpte-ra.org

Users of the types dictionary should check the SMPTE-RA web site regularly for updates to the register.

5.2 Dictionary Management and Compatibility Requirements

To ensure reliable and correct interpretation of legacy material in the future, changes to the types dictionary shall be carried out in accordance with the registration procedures defined in Annex B. Annex B specifies the provisions and corresponding requirements for additions, deletions, deprecations, and changes to items in classes 1-7 (Annex B.1), class 13 (Annex B.2), and class 14 (Annex B.3).

The addition process shall be carried out and documented in accordance with Annex B by the SMPTE Registration Authority. It shall occur on request from the appropriate SMPTE technology committee and shall be administered in accordance with Annex B. The version number of the dictionary shall be incremented each and every time an addition (or group of additions) is approved since this is critical to ensuring the operational compatibility of metadata decoders. The incrementing of the version number shall not prevent use of unaffected universal labels, structure, or contents by a decoder that conforms to the prior version.

Note: It is inevitable, given the above addition process, that eventually the dictionary will become cluttered with legacy entries to the point where the responsible SMPTE technology committee determines it has reached the limit of its usefulness. At this stage, or when other changes to the dictionary contents, to an existing approved dictionary structure, or to relationships between types that prevent backward compatibility are necessary, a new structure standard and the associated register will be created. These will be made readily accessible online by the SMPTE Registration Authority to allow upgrades to decoders. The superseded standard will then undergo no further revision unless essential under the SMPTE five-year rule.

5.3 Dictionary Availability

The latest version of the types dictionary shall be made available on the SMPTE Registration Authority website, www.smpte-ra.org, in a defined electronic publishing format with an accompanying document specification. A minimum of the two immediate previous versions should also be available in a clearly indicated archive.

Annex A Glossary of Terms (Normative)

A.1 Attribute: A characteristic of a type.

A.2 Batch of: Multiple instances of the items specified, in no particular order.

A.3 Class: The broad category of data that forms the first level of hierarchy for all registered data types.

A.4 Child: An entry in the register under a leaf that gives added information about the type defined in the leaf.

A.5 Context: The circumstance, purpose, and perspective under which something is defined or used.

A.6 Designator: A sub-identifier within a universal label.

A.7 Facet (Record): A child entry defining a member of a record.

A.8 Facet (Enumeration): A child entry defining a token belonging to an enumeration.

A.9 Facet (Choice): A child entry defining an alternate belonging to a choice.

A.10 Identifier: A sequence of numbers or characters, capable of uniquely identifying that with which it is associated, within a specified context.

A.11 Inseparable Bytestring: A sequence of bytes that, although possibly consisting of defined sections, only maintains its intended meaning when complete. For example a UUID is an inseparable bytestring (even though it may contain an IEEE 802 address) but a date field composed of day, month and year is not.

A.12 Item (register): An object in a register that instantiates a defined set of attributes. An item in the types dictionary is a description of a type, a class/subclass or a facet of a type.

A.13 Item designator (SMPTE ST 336): The last 8 bytes of the universal label, which uniquely identify a particular item within the context of the UL designator.

A.14 Leaf: An entry in the register that defines a type.

A.15 Level number: The last non-zero value of a UL of a node or leaf. Which value in a UL is the level number depends on the level of the entry, i.e. how many entries are above it in the hierarchy.

A.16 Metadata element: A data element defined by the metadata element dictionary.

A.17 Node: An entry in the registry that is used to provide a hierarchical structure for leaves. A node may have any number of nodes and leaves under it that are logically grouped by the node. All entries under a node share the non-zero bytes of the node's UL.

A.18 Parent: The leaf of which a specific child forms part of the definition. Each child has a field containing the UL of its parent.

A.19 Registry: An information system for registering metadata (e.g. metadata elements, types, groups).

A.20 Register: The information store or database maintained by a registry.

A.21 Registration authority: An organization responsible for maintaining a register.

A.22 Token: A defined value of an enumeration.

A.23 Top-level node: A node in class 13 or 14 under which an individual or organization other than SMPTE controls the entries.

A.24 Type: Information about the representation of the data or data value.

A.25 Types dictionary: The register, as defined by this standard, of approved type identifiers and the attributes of the types that they identify.

A.26 Universal label: Specifically a SMPTE-administered universal label, which is 16 bytes. The syntax of the universal labels used in the data element dictionary is defined in SMPTE ST 336, which describes the mechanism by which the UL is used as a key that explicitly identifies a predefined value or group of values.

A.27 UL designator (SMPTE ST 336): A sequence of sub-identifiers (bytes 3-8 of a 16-byte universal label) designating the ISO/ITU organization, registry category, registry, registry structure, and version number.

A.28 Value: An instance of information.

Annex B Registration Criteria (Normative)

This annex defines the specific registration criteria for entries in each of the defined types classes.

B.1 Criteria for Modifications to Entries in Classes 1-7

Classes 1 to 7 of the register shall be administered by the Metadata and Registers Committee, 30MR or another body as appointed by the Standards Committee from time-to-time. All changes shall only take place after successful completion of a ballot of the administering body.

Changes may be instigated by the administering body, or by any organization or individual that is a member of SMPTE upon providing the following information:

1) Contact information for the organization, individual, or committee requesting the change;

2) Details of the requested change of the registered type along with a justification for the change;

3) Details of any supporting document that may require the change of the registered type;

B.1.1 Additions to Entries in Classes 1-7

Additions to the register in classes 1 to 7 shall be subject to review for adequacy of information, including technical description, non-conflict with existing engineering documents, and compliance with the requirements in this section. Class 1 to 7 additions shall not require a supporting SMPTE engineering document.

B.1.2 Changes to Entries in Classes 1-7

During the ballot process for a change of an entry whose class is in the range 1 to 7, negative votes based upon procedural issues, including adequacy of technical description, shall be accepted.

B.1.3 Deprecation of Entries in Classes 1-7

Any request for flagging of a registered type as deprecated shall result in the posting of an appropriate public notice describing the proposed deprecation.

B.1.4 Deletion of Entries in Classes 1-7

Entries in classes 1 to 7 shall not be deleted.

B.2 Criteria for Modifications to Entries in Class 13

Class 13 of the register shall be administered by the Metadata and Registers Committee 30MR, or another body as appointed by the Standards Committee from time-to-time.

Nodes in class 13 may be allocated to a specified organization which becomes responsible for instigating changes under that node. A node shall be allocated after successful completion of a consensus vote of the administering body. The allocated node shall be regarded as the top-level node for that organization within the class 13 register.

All changes in class 13 entries are at the discretion of the appropriate organization. All changes shall be reviewed for adequacy of information and compliance with the provisions of this document.

Changes may be instigated upon submitting the following information to the administering body:

1) Contact information for the organization, individual, or committee requesting the change;

2) Details of the requested change of the registered type along with a justification for the change;

3) Details of any supporting document that may require the change of the registered type.

The administering body shall check that all required information has been supplied and include the modifications in the next published version of the register.

B.2.1 Additions to Entries in Class 13

Additions to the register in class 13 shall be subject to review for adequacy of information; this shall be limited only to compliance with the requirements in this section. Class 13 additions shall not require a supporting SMPTE engineering document.

B.2.2 Changes to Entries in Class 13

Changes to class 13 entries shall be subject to review for adequacy of information this shall be limited only to compliance with the requirements in this section.

B.2.3 Deprecation of Entries in Class 13

Any request for flagging of a registered type as deprecated should result in the posting of an appropriate public notice describing the proposed deprecation.

B.2.4 Deletion of Entries in Classes 13

Entries in class 13 shall not be deleted.

B.3 Criteria for Modifications to Entries in Class 14

Class 14 of the register shall be administered by the Metadata and Registers Committee 30MR, or another body as appointed by the Standards Committee from time-to-time.

Nodes in class 14 may be allocated to a specified organization which becomes responsible for all entries under that node. A node shall be allocated after successful completion of an administrative vote of the administering body. The allocated node shall be regarded as the top-level node for that organization within the class 14 register.

A request for a class 14 node shall include the following information:

1) Contact information for the organization, individual, or committee requesting the node;

2) Statement of intention to apply the registered type node, and intended date of first use;

Note: A fee might apply for the registration of a class 14 node.

Changes to the register in class 14 shall not be subject to review.

Annex C Organization of References (Informative)

No single standard can contain all of the information needed to describe and encode metadata. Hence, a layered approach is used to convey the information so the user can select the applicable standard(s) for the level of implementation needed. The SMPTE normative standards for metadata include:

- The standard for the groups register structure (SMPTE ST 395) and the groups register itself;
- The standard for the metadata element dictionary structure (SMPTE ST 335) and the metadata element dictionary itself;
- This standard for the types register structure (SMPTE ST 2003) and the types register itself;
- The standard for the SMPTE labels structure (SMPTE ST 400) and the labels register itself;
- The standard for key-length-value (KLV) data encoding (SMPTE ST 336).

Annex D Bibliography (Informative)

Note: All references in this document to other SMPTE documents use the current numbering style (e.g. SMPTE ST 395:2003) although, during a transitional phase, the document as published (printed or PDF) may bear an older designation (such as SMPTE 395M-2003). Documents with the same root number (e.g. 395) and publication year (e.g. 2003) are functionally identical.

SMPTE ST 335:2012, Metadata Element Dictionary Structure

SMPTE ST 395:2003, Metadata Groups Registry Structure

SMPTE ST 400:2004, Television — SMPTE Label Structure

ISO/IEC 646:1991, InformationTtechnology — ISO 7-Bit Coded Character Set for Information Interchange

ISO 7185:1990 Pascal

ISO/IEC 9899:1999, Programming Languages — C

ISO/IEC 11179-1, Information Technology — Metadata Registries (MDR) — Part 1: Framework

The Unicode Consortium. The Unicode Standard, Version 5.1.0, defined by: *The Unicode Standard, Version 5.0* (Boston, MA, Addison-Wesley, 2007. ISBN 0-321-48091-0), as amended by *Unicode 5.1.0* (http://www.unicode.org/versions/Unicode5.1.0).

Extensible Markup Language (XML) 1.0 (Fifth Edition), W3C Recommendation, 26th November 2008, http://www.w3.org/TR/2008/REC-xml-20081126/

Merriam Webster's Collegiate Dictionary, 11th Edition, July 2003, Merriam-Webster