

SMPTE STANDARD

Types Dictionary Structure



Page 1 of 26 pages

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

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
1	UL Header		
	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.
3	UL designator		
	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.

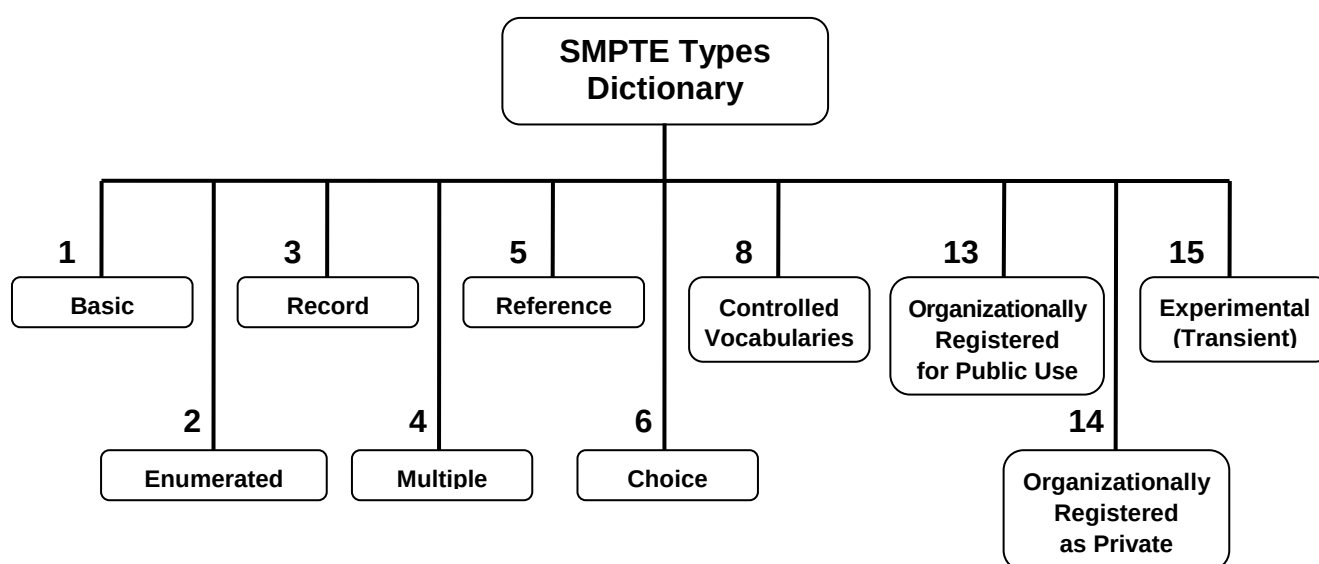


Figure 1 – Types class structure

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
 - o 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	<i>Array</i>
Identified	Unordered	Explicit	Fixed	<i>Set</i>
Identified	Ordered	Explicit	Fixed	<i>Vector</i>
Anonymous	Ordered	Implicit	Fixed	<i>Table</i>
Identified	Unordered	Implicit	Fixed	<i>Bag</i>
Identified	Ordered	Implicit	Fixed	<i>Series</i>
Anonymous	Ordered	Explicit	Variable	<i>Cluster</i>
Identified	Unordered	Explicit	Variable	<i>Bundle</i>
Identified	Ordered	Explicit	Variable	<i>Stack</i>
Anonymous	Ordered	Implicit	Variable	<i>StringTable</i>
Identified	Unordered	Implicit	Variable	<i>StringString</i>
Identified	Ordered	Implicit	Variable	<i>StringSeries</i>

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.