RECOMMENDATION ITU-R BT.1563

Data encoding protocol using key-length-value

(Questions ITU-R 42/6 and ITU-R 20/6)

(2002)

The ITU Radiocommunication Assembly,

considering

a) that many countries are installing digital television production facilities based on the use of digital video components conforming to Recommendations ITU-R BT.601, ITU-R BT.656 and ITU-R BT.799;

b) that high-definition television (HDTV) production systems are being installed based on digital HDTV interfaces conforming to Recommendation ITU-R BT.1120;

c) that there exists the capacity within a signal conforming to Recommendation ITU-R BT.656 or ITU-R BT.799 for additional data signals to be multiplexed with the video data signal itself;

d) that there are operational and economic benefits to be achieved by the multiplexing of ancillary data signals with the video data signal;

e) that the operational benefits are increased if a minimum of different formats are used for ancillary data signals;

f) that formatting of ancillary data packets is specified in Recommendation ITU-R BT.1364;

g) that a generic formatting for a wide variety of data using ancillary data packets as one form of transmission will benefit broadcast transmission operations,

recommends

1 that the key-length-value (KLV) data formatting described in SMPTE Standard 336M 2001 – Data Encoding Protocol Using Key-Length-Value, be used as a method for a variety of data in the serial digital interface.

Summary of SMPTE Standard 336M-2001

This Recommendation defines an octet-level data encoding protocol for representing data items and data groups. This protocol defines a data structure that is independent of the application or transportation method used. An example of transportation would be ancillary data packets specified in Recommendation ITU-R BT.1364.

The Recommendation defines a KLV triplet as a data interchange protocol for data items where the key identifies the data, the length specifies the length of the data, and the value is the data itself. The KLV protocol provides a common interchange point for all compliant applications irrespective of the method of implementation or transport. The Recommendation also provides methods for combining associated KLV triplets in data sets where the set of KLV triplets is itself coded with

---

Note by the Secretariat: The SMPTE Standard 336M-2001 previously referred to as a web link site in electronic form, has been annexed to the text of this Recommendation.

KLV data coding protocol. Such sets can be coded in either full form (universal sets) or in one of four increasingly bit-efficient forms (global sets, local sets, variable-length packs, and fixed-length packs). The Recommendation provides a definition of each of these data constructs. The encoding octet range (length of the payload) specified in this Recommendation may generate unusually large volumes of data. Consequently, a specific application of KLV encoding may require only a limited operating data range and those details shall be defined in a relevant application document.

NOTE 1 – SMPTE Standard 336M-2001 is given in Annex 1. SMPTE Standard 336M-2001 and its summary refer to Version 2001 only, which is the version approved by Administrations of Member States of the ITU in application of Resolution ITU-R 1-3 on 30-04-02. By agreement between ITU and SMPTE, this Version was provided and authorized for use by SMPTE and accepted by ITU-R for inclusion in this Recommendation. Any subsequent version of SMPTE Standard 336M, which has not been accepted and approved by Radiocommunication Study Group 6 is not part of this Recommendation. For subsequent versions of SMPTE Documents, the reader should consult the SMPTE website: http://www.smpte.org/.

SMPTE 336M-2001

SMPTE STANDARD

for Television — Data Encoding Protocol using Key-Length-Value

1 Scope

This standard defines an octet-level data encoding protocol for representing data items and data groups. This protocol defines a data structure which is independent of the application or transportation method used.

The standard defines a key-length-value (KLV) triplet as a data interchange protocol for data items where the key identifies the data, the length specifies the length of the data, and the value is the data itself. The KLV protocol provides a common interchange for all compliant applications irrespective of the method of implementation or transport.

The standard also provides methods for combining associated KLV triplets in data sets where the set of KLV triplets is itself coded with KLV data coding protocol. Such sets can be coded in either full form (universal sets) or in one of four increasingly bit-efficient forms (global sets, local sets, variable-length packs, and fixed-length packs). The standard provides a definition of each of these data constructs.

The encoding octet range (length of the payload) specified in this standard may generate unusually large volumes of data. Consequently, a specific application of KLV encoding is capable of only a limited operating data range and those details shall be defined in a relevant application document.

2 Normative references

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

ANSI/SMPTE 298M-1997, Television — Universal Labels for Unique Identification of Digital Data

ISO/IEC 8825-1:1998 (ITU-T X.690), Information Technology — ASN.1 Encoding Rules — Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER), and Distinguished Encoding Rules (DER)

3 Key-length-value (KLV) protocol

Table 1 and figure 1 present an introductory view of the key-length-value (KLV) protocol for encoding data. The data encoded may be a single data item or a data group. The coding of data items is described in clause 4 while the coding of data groups is described in clause 5 of this standard.

The KLV coding protocol is composed of a universal label (UL) identification key (UL key), followed by a numeric length (value length), followed by the data value.

The composition of the UL key is described in 3.1 of this standard. The length of the full UL key shall be 16 octets. The length field is described in 3.2 of this standard. The value is described in 3.3 of this standard. The value is a sequence of octets of the data type as specified in a relevant standard and is not further specified by the KLV protocol. The length of the value field is variable and any limitations are defined in a relevant defining standard.

Approved
March 28, 2001

Table 1 – Key-length-value (KLV) fields for encoding of data

Field	Description	Length	Content/Format
UL key	Universal label for identification of the value	16 octets	Clause 3.1
Length	Length of the value field	Defined in a relevant dictionary, essence, application standard, but variable length	Clause 3.2
Value	Value associated with the UL key	Variable	Clause 3.3

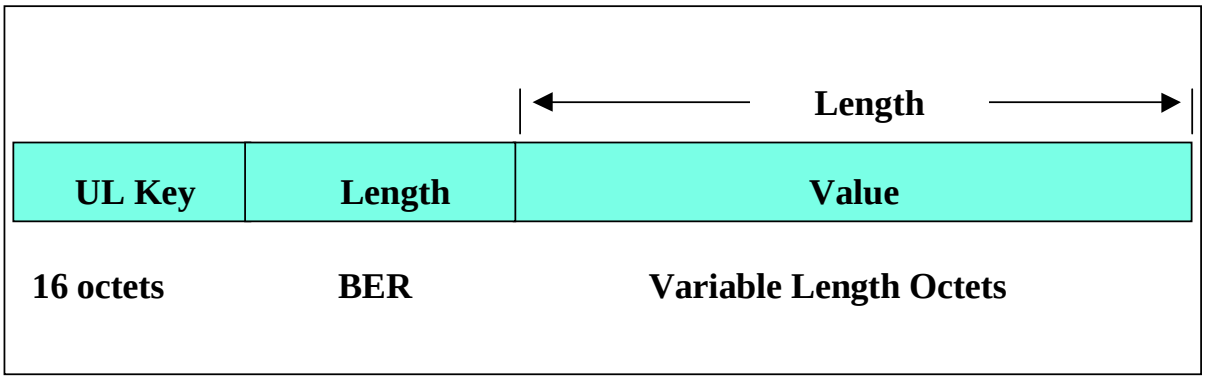

Figure 1 – Key-length-value (KLV) encoding

The bit-order (lsb or msb first) for KLV encoding shall be that of the transport used to carry the information.

3.1 ANSI/SMPTE 298M universal label key

KLV coding protocol shall use a 16-word universal label (UL) according to SMPTE 298M as the UL key to identify the data in the value field.

Each word in the SMPTE 298M UL is coded using the basic encoding rules (BER) for the encoding of an object identifier value specified in ISO/IEC 8825-1, paragraph 8.19. Each word of the UL key shall be limited to the range 0×00 to 0×7F and shall be represented by a single octet. The UL key shall have left-to-right significance with the first octet as the most significant. The leftmost octet of value 0×00 in the UL key shall define the termination of the label and all octets of lower significance shall also be set to 0×00. Octets of value 0×00 shall have no significance to the meaning of the UL key.

The full UL key consists of a 16-octet field including an object ID (OID) and the UL size (0×0E indicating a total UL key size of 16 octets) followed by a UL code and a series of subidentifiers which shall define the UL designators. The first two UL designators shall have reserved values for the KLV coding protocol according to this standard.

NOTE – The SMPTE UL itself adopts KLV coding with the object ID as the key, the UL size as the length, and the UL designators as the value.

When applying the SMPTE UL to the coding of data according to this standard, these fields shall be categorized as the UL header, UL designators, and item designator as shown in table 2.

Table 2 – Field descriptions for the universal label key for the KLV encoding of data

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenated subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE subidentifier	1 octet	Always 0×34
5	Registry category designator	Registry category designator identifying the category of registry described (e.g., dictionaries)	1 octet	See table 3
6	Registry designator	Registry designator identifying the specific registry in a category (e.g., metadata dictionary)	1 octet	See table 3
7	Structure designator	Designator of the structure variant within the given registry	1 octet	Clause 3.1.2
8	Version number	Version of the given registry which first defines the item specified by the item designator	1 octet	Incrementing number
9-16	Item designator	Unique identification of the particular item within the context of the UL designator	8 octets	See relevant standard and version

Annex C shows an informative example of a metadata UL key from the SMPTE metadata dictionary in tabular and figure formats.

Note that decoders which recognize the UL key but do not want to, or cannot, decode the associated value may ignore the item and shall continue the decoding process of subsequent items using the length value to skip the value of the undecoded item. If decoders only store or forward the item, they shall forward the item unaltered.

3.1.1 UL designators

Table 3 defines octet values for the designators to be used in octets 5 through 7 of the UL designators. SMPTE standards and recommended practices which define a UL key with the value of octet 5 (registry category designator) in the range 0×01 to 0×04 shall register the full UL key or keys used with the SMPTE Registration Authority in the registry identified by octets 6 and 7 (registry designator and structure designator).

3.1.1.1 Dictionaries

SMPTE standards and recommended practices which define the value of word 5 of the UL key as 0×01 are dictionary standards and shall be used to define single data items with the KLV data construct.

The coding of individual items is defined in clause 4.

3.1.1.2 Sets and packs

SMPTE standards and recommended practices which define the value of word 5 of the UL key as 0×02 are set and pack standards and shall be used to define groups of KLV coded data items.

The coding of data groups is defined in clause 5.

3.1.1.3 Wrappers and containers

SMPTE standards and recommended practices which define the value of word 5 of the UL key as 0×03 are wrapper and container standards and use the UL key to identify the wrapper or container and its contents. Wrappers and containers differ from sets and packs in that they do not necessarily employ an overall KLV data construct for the entire contents of the wrapper or container. It is recommended that individual parts of a wrapper or container encode data using the KLV coding protocol, but these parts may be bound together by other techniques. In some cases, a wrapper or container may employ an overall KLV

Table 3 – UL designators for octets 5 through 7

Registry category Octet 5	Registry Octet 6	Structure Octet 7	Described in
01 – Dictionaries			Clause 4
	01 – Metadata dictionaries	0×01-0×7F– Structure designator	Other standard/practice
	02 – Essence dictionaries	0×01-0×7F – Structure designator	Other standard/practice
	03 – Control dictionaries	0×01-0×7F – Structure designator	Other standard/practice
	04 – Types dictionaries	0×01-0×7F – Structure designator	Other standard/practice
02 – Groups (sets and packs)			Clause 5
	01 – Universal sets	0×01-0×7F – Structure designator	Clause 5.1
	02 – Global sets (default)	0×01-0×7F – Structure designator	Clause 5.2
	03 – Local sets (default)	0×01-0×7F – Structure designator	Clause 5.3
	04 – Variable-length packs (default)	0×01-0×7F – Structure designator	Clause 5.4
	05 – Fixed-length packs	0×01-0×7F – Structure designator	Clause 5.5
03 – Wrappers and containers			
	01 – Simple wrappers	0×01-0×7F – Structure designator	Other standard/practice
	02 – Complex wrappers	0×01-0×7F – Structure designator	Other standard/practice
04 – Labels			Clause 6
	01 – Labels dictionary	0×01-0×7F – Structure designator	Other standard/practice

construct in certain applications (such as a streaming interface) but employ another technique in other applications (such as a storage container). In these cases, the wrapper or container is not redefined as a set or a pack but retains the definition as a container or wrapper for consistency of identification.

Simple wrappers and containers are defined as embedding all the data into a single framework with no external references.

Complex containers and wrappers are defined by frameworks where individual data items may be included in a file by reference rather than embedding. Complex containers and wrappers can be more efficient and are suited to local environments where references can be easily resolved.

The definition of wrappers and containers is outside the scope of this standard and can be found in other documents.

3.1.2 Structure designator

Octet 7 shall contain the structure designator for the given registry.

Structure designators are allocated to distinguish between incompatible versions of the same registry. They may be thought of as a major version number.

NOTE – Different structure designators may be assigned when a registry becomes so cluttered that it has outlived its usefulness, or when changes to the contents, structure, or class relationships of the registry are necessary which prevent backward compatibility.

Different structure designators may also be assigned to distinguish between different syntax rules for construction and interpretation of values.

Different structure designators may also be assigned when more than one standard or practice is used to define the contents of a specific registry.

The precise discipline for allocation of structure designators is described for each specific registry in the clauses below.

3.1.3 Version number

Octet 8 shall contain the version number of the given registry which first defines the item specified by the item designator.

New items may be added to registries after initial approval of the controlling standard or practice. Each time a set of item definitions is added, the current version number of the particular registry is incremented. Each entry in a registry includes the version number in which the item was first defined. It is this number which is carried in octet 8.

Parsers may use the version number as an additional guide and consistency check in the process of parsing a UL key.

3.1.4 Item designator

Octets 9 through 16 of the UL key comprise the item designator.

The item designator field is fixed 8 octets in length. Item designators are from 1 to 8 bytes long, and are padded on the right with zero octets to fill the 8-byte field. ASN.1 object identifier coding is used to provide for allocation of varying quantities of subidentifiers in a hierarchical manner.

The precise meaning and construction of the item designator depends upon the specific registry and structure variant, and is described further in the clause below.

3.2 Encoding of the KLV length field

In the KLV coding protocol, the value of the length field shall be encoded using the basic encoding rules (BER) for either the short form or long form encoding of length octets specified in ISO/IEC 8825-1, paragraph 8.1.3. This method of value length encoding is self-contained and allows for efficient parsing of KLV encoded data. When the KLV coding protocol is applied to groups of KLV coded units, the length field for individual units may adopt a different method as defined by the standard for the coding of that group (see clause 5).

NOTE 1 – Bits are numbered bit 8 to bit 1 with bit 8 as the MSB.

NOTE 2 – Receivers/decoders must decode the length field correctly whether the long or short form is used.

3.2.1 BER short-form length encoding

The following normative section (including example) is quoted from ISO/IEC 8825-1:

> 8.1.3.4 In the short form, the length octets shall consist of a single octet in which bit 8 is zero and bits 7 to 1 encode the number of octets in the contents [value] octets (which may be zero), as an unsigned binary integer with bit 7 as the most significant bit.

Example: L = 38 can be encoded as 00100110_2.

The short form for length encoding shall be used whenever the data value length is less than 128 octets.

3.2.2 BER long-form length encoding

The following normative section (including example) is quoted from ISO/IEC 8825-1:

> 8.1.3.5 In the long form, the length octets shall consist of an initial octet and one or more subsequent octets. The initial octet shall be encoded as follows:
>
> a) bit 8 shall be one;
>
> b) bits 7 to 1 shall encode the number of subsequent octets in the length octets, as an unsigned binary integer with bit 7 as the most significant bit;
>
> c) the value 11111111_2 shall not be used.

NOTE – This restriction is introduced for possible future extensions.

Bits 8 to 1 of the first subsequent octet, followed by bits 8 to 1 of the second subsequent octet, followed in turn by bits 8 to 1 of each further octet up to and including the last subsequent octet, shall be the encoding of an unsigned binary integer equal to the number of octets in the value field, with bit 8 of the first subsequent octet as the most significant bit.

NOTE – This is sometimes known as big-endian byte order.

Example:

L = 201 can be encoded as:

Octet 1 = 10000001_2 Octet 2 = 11001001_2 [b8 ... b1]

NOTE – While there are no restrictions in this standard on the maximum number of octets in the data value length field, the presence of large data value lengths can be determined from the first octet in the BER long-form length encoding.

Where appropriate, individual application standards and recommended practices may define the maximum octet length of the length field or may place limitations on the value range of the length field in order to simplify decoder requirements.

Implementations shall make every effort to apply a valid value to the length field. However, in certain operations, it may prove impractical to establish the length of the value field. Such a case is an incoming data stream which is assigned a key and a length field at the start point. In this case, the value of the length cannot be established until the termination of the stream and at that point, it may prove impossible to return to the length field to enter the value. In such cases, the length field shall be set to [0×80] which shall indicate a nondeterministic length of the value field. Any application document which allows the length of the value field to be undefined must define an alternative method of locating the end of the value field.

NOTE – The length value [0×80] is used because it is normally meaningless as a BER long-form value as it indicates zero subsequent octets.

3.3 Encoding of data values

Data values may be either individual data items or data groups. In either case, the data is an octet string whose length is specified by the length value. The last octet of the value shall be the terminating octet of the data sequence.

3.4 Empty metadata items

Specifications for contiguity of KLV packets including any gaps between KLV packets are outside the scope of this standard, and are addressed in the appropriate transport layer documents.

However, should applications so require it, breaks in the data sequence can be inserted by the use of a specific empty metadata item. Use of empty metadata items is not mandatory.

The empty metadata item is a KLV coded packet which shall define a length value followed by an empty value field. No attempt should be made to interpret the data in the value field.

The empty metadata item shall be defined in the metadata dictionary.

Empty metadata items may be coded as individual items or within sets, when allowed by the specific set definition.

Applications may delete or skip any or all empty metadata items upon receipt. Applications may insert empty metadata items, but shall not require other applications to preserve such items.

4 KLV coding of individual data items

The KLV coding of individual data items is a simple application of the key, length, and value as defined in clause 3.

The UL key of individual data items is defined in a dictionary together with the ranges of length and the specification of the value itself. For individual data items, the value of word 5 of the UL key shall be 0×01.

4.1 Identification of value data representations

The value of many dictionary items can be represented in more than one way. For example, a start time in the metadata dictionary can be represented as a character string of the time code or as an efficient bit-packed form. The first offers direct mapping to a display whereas the second offers high-transmission efficiency for use in narrow-band data channels. There are many such dictionary items which have multiple data representations for the same descriptor.

Where a dictionary item has more than one data representation for the value, one representation shall be designated as the default representation and shall be assigned a key with at least one trailing zero octet. Alternate representations shall be assigned keys by replacing the leftmost trailing zero octet with nonzero values, which shall be assigned sequentially. Each representation shall be documented in the dictionary.

Example:

01.02.03.04.00.00.00.00 is name (default data representation in UTF 16 unicode characters);

01.02.03.04.01.00.00.00 is name (different data representation in ISO 7-bit characters);

01.02.03.04.02.00.00.00 is name (another data representation in UTF 8 unicode characters).

The parser treats all representations as the same item; i.e., it recognizes 01.02.03.04.00, then looks for xx in place of the 00 to identify different encodings. Since the default representation is defined, the extra non-zero term in the fifth position is known to be a new data representation of the default dictionary item and not a new dictionary entry.

Many dictionary values share a common set of definitions for multiple data representations. To simplify the dictionary definitions, a types dictionary shall be used to define these data representations. The types dictionary shall be used as a shared resource for all other dictionaries.

Annex D shows an example of KLV coding for a single metadata item.

5 KLV group coding

Group coding of data elements can be used to reduce the overhead of repeating redundant information that appears in the key of each unit. Group coding also allows logical groups of individual data elements, or groups of elements, to be encoded together and provides options for increased coding efficiency. In steps of increasing code efficiency, the KLV coding protocol can be used to support universal sets, global sets, local sets, variable-length packs, and fixed-length packs described as follows:

– *Universal sets* shall be used to construct a logical grouping of data elements and other KLV encoded items. Universal sets use the full KLV coding construct throughout.

– *Global sets* are defined as per universal sets, but offer coding efficiency by sharing a common key header. This coding gain is lossless and every UL key can be fully recovered from the data in the global set alone.

– *Local sets* are defined as per universal sets, but offer coding efficiency through the use of short local tags whose meaning is defined only within the context of the local set. Local sets retain the KLV data construct, but require a separate standard or recommended practice to define the meaning of the local tags and to provide a map from the local tag value to the UL key value.

– *Variable-length packs* are defined as a further grouping of data elements that eliminates the use of UL keys and local tags for all individual elements within the group. Variable-length packs, therefore, rely on a standard or recommended practice which defines the order of data elements within the pack.

– *Fixed-length packs* are the most efficient (and least flexible) grouping of data elements that eliminates the use of both UL keys and local tags and removes the length for all individual elements within the group. Thus, fixed-length packs rely on a standard or recommended practice which defines both the order of data elements and the length of each data element within the pack.

Sets and packs shall consist of a number of individual data elements which are coded as a group by the KLV set or pack data construct. The set or pack shall be defined by a full UL key whose value shall be registered with the SMPTE Registration Authority. A set may encode data elements which are themselves sets or packs as well as individual dictionary items. This is called KLV recursive coding and this standard provides no limit on the number of levels of recursion which may be used by any particular application. A pack shall only encode a group of individual dictionary items; i.e., packs shall not use recursive coding.

The presence of sets or packs shall be indicated by 0×02 in the registry category designator field (octet 5) of the set or pack UL key. The registry designator field (octet 6) shall be used to identify the type of set or pack. The set or pack standard or practice and registry shall be identified by the structure designator field (octet 7) and the version of the registry shall be identified by the version number field (octet 8).

The length of a set or pack shall be encoded as either BER short-form or long-form coding. Application standards or recommended practices may provide an upper bound to the value of the BER coded length to ease decoder requirements.

Application standards or recommended practices may also provide a fixed-length coding scheme specific to the application for sets or packs embedded within sets or packs. However, this is discouraged.

Where the length fields of the KLV items in a global set, a local set, and a variable-length pack are identified as fixed length, the most significant byte shall be encoded first (big-endien).

The set or pack value shall be comprised of a number of individual data elements with coding as defined by the set or pack type.

The following clauses define how the data elements are encoded for universal sets, global sets, local sets, variable-length packs, and fixed-length packs.

5.1 Universal sets

The UL key of a universal set shall be defined by an accompanying standard or recommended practice including a structure designator and an accompanying universal set registry including a version number.

The UL key of a universal set shall be 16 octets in length.

The length of a universal set shall be coded as per ASN.1 notation; BER long or short form as required.

The value of a universal set shall be a sequence of KLV-encoded elements whose total length is given by the length field. Each and every data element in a universal set shall apply KLV data coding protocol including the full UL key value.

Relevant application standards or practices may specify constraints upon the value of a universal set such as the number and size of items, the allowed sequence of items, and whether any items are mandatory or optional.

The UL key for universal sets is described in table 4. Figure 2 illustrates the data structure for the encoding of universal sets.

5.2 Global sets

The UL key of a global set shall be defined by an accompanying standard or recommended practice including a structure designator and an accompanying global set registry including a version number.

The UL key of a global set shall be 16 octets in length.

The length of a global set shall be coded as per ASN.1 notation; BER long or short form as required.

The value of a global set shall be a sequence of KLV-encoded elements whose total length is given by the length field. Each and every data element in a global set shall apply KLV data coding protocol, but with a shortened global tag value replacing the UL key as described next.

The global set UL shall be defined in two parts:

– The first group of 8 octets (UL header and UL designator) shall be registered with the SMPTE Registration Authority and shall be used to identify the global set standard or recommended practice including the structure designator. Each entry in the global set registry shall record the version number in which it was first defined.

– The second group of 8 octets is called the global set designator and shall be used to define the common UL header and UL designator for all the UL keys within the global set. This second group of 8 octets shall include the UL header fields together with as much of the UL designator as is common to all items in the global set. The global set designator may be terminated by a zero-value octet to indicate termination of the common UL designator root. The significant length of the second group to the zero value terminator shall be 2 to 8 octets. If the length of the second group is 8 octets, then the zero-value terminator octet is not required.

Each global tag is from 2 to 12 words in length. Global tags of length less than 12 words shall be terminated by a single zero value thus removing redundant UL data.

The full 16-octet UL key of each item in the global set can be losslessly recreated by concatenating the nonzero octets of the global set designator and the global tag of the item. If the resulting concatenation is less than 16 octets in length, the remaining octets in the 16-octet space shall be zero filled.

Relevant application standards or recommended practices may specify constraints upon the value of a global set such as the number and size of items, the allowed sequence of items, and whether any items are mandatory or optional.

Table 4 – Field descriptions for the UL key for the KLV encoding of universal sets

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenation of subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE subidentifier	1 octet	Always 0×34
5	Registry category designator	Sets and packs	1 octet	Always 0×02
6	Registry designator	Universal sets	1 octet	Always 0×01
7	Structure designator	Designator of the structure variant within the universal set registry	1 octet	Incrementing number
8	Version number	Version of the given registry which first defines the item specified by the item designator	1 octet	Incrementing number
9-16	Universal set designator	Unique identification of the particular universal set	8 octets	See universal set registry

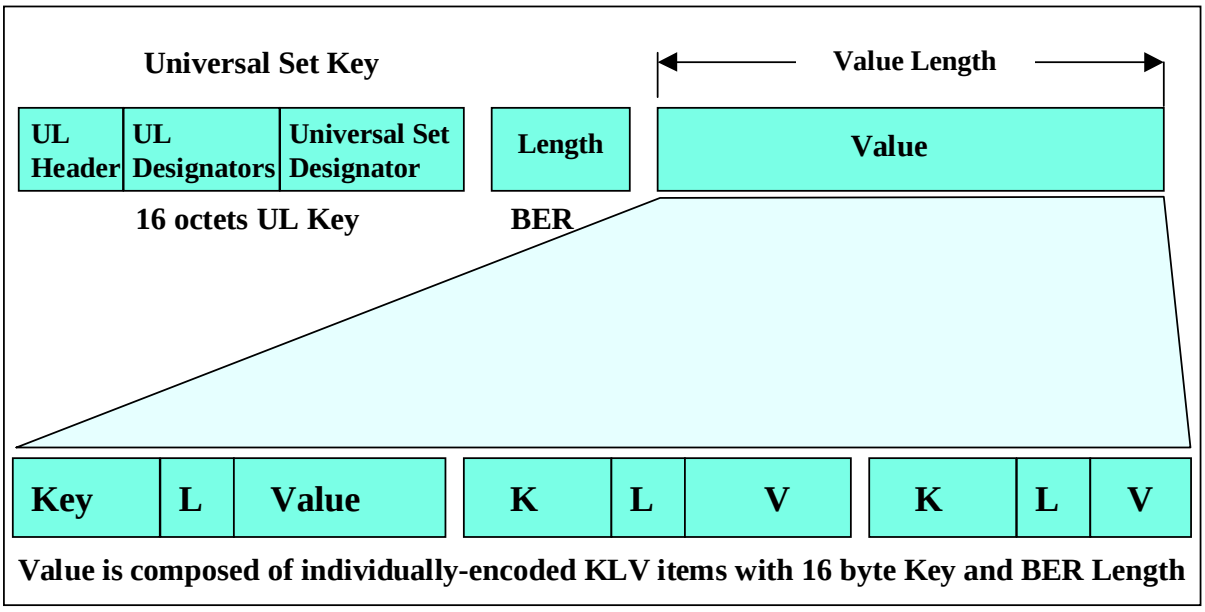

Figure 2 – KLV coded universal set data structure

The UL key for global sets is described in table 5.

Figure 3 illustrates the structure for the encoding of global data sets.

The 16-octet global set UL key is followed by the global set length (encoded using BER short- or long-form encoding) which is followed by a number of elements which shall each be triplets of global tag, length, and value.

The preferred specification of the length fields of individual elements is BER short- or long-form encoding. The full range of allowed length field lengths is defined by the registry designator, according to table 6. All length fields in the global set shall follow the same syntax.

Global sets can accommodate recursion, so that the UL key linked to a global tag may identify either a single data element from a dictionary or a data set or pack from a set or pack standard or recommended practice and the corresponding registry.

An informative example of the operation of global sets is given in annex F.

5.3 Local sets

A local set is defined as a number of data elements that are grouped to reduce the length of the keys for each element within the set. Elements may be in any order within the local set and may be present or absent.

The UL key of a local set shall be defined by an accompanying standard or recommended practice including a structure designator and an accompanying local set registry including a version number.

The UL key of a local set shall be 16 octets in length.

The length of a local set shall be coded by default as per BER length notation; long or short form as required.

The value of a local set shall be a sequence of KLV-encoded elements whose total length is given by the length field.

The UL key for local sets is described by table 7. The local set designator is defined within the last 8 octets in the local set UL key. The 16-octet local set key shall be defined in an associated standard or recommended practice and the local set UL key value shall be registered with the SMPTE Registration Authority to guarantee a unique local set UL key value.

The data structure for the encoding of local sets is illustrated in figure 4.

The 16-octet local set UL key is followed by the set length, which is followed by a number of elements, which shall each be triplets of local tag, length, and value.

The preferred size of the local tag fields is 1 octet, and the preferred specification of the length fields is BER short- or long-form encoding. The full range of allowed combinations of tag and length field lengths is defined by the registry designator, according to table 8.

A relevant local set standard or recommended practice shall define the link between the local tag of each element and the corresponding UL key value. This link shall be defined in a relevant local set standard or recommended practice that provides for each local tag the UL key of the defining element. This linking definition is a mechanism that gives users of this standard the flexibility to define their own aliases for highly efficient coding. Developers of local sets shall provide the mapping between each tag in a local set and the defining UL key. Unlike universal sets and global sets, where the UL key of each element in the set can be losslessly recreated, the UL key of each local set tag cannot be reconstructed without recourse to the defining standard or recommended practice and the corresponding registry.

Local sets can accommodate recursion so that the UL key linked to a local tag may identify either a single data element from a dictionary or a data set or pack from a set or pack standard or recommended practice and the corresponding registry.

Figure 5 is an informative illustration of the linking between a local tag and a full UL key.

5.4 Variable-length packs

A variable-length pack is similar to a local set, but does not have local tags. Thus, each element of a variable-length pack comprises only a length field and a value field. Elements in a variable-length pack must appear in the defined order.

The UL key of a variable-length pack shall be defined by an accompanying standard or recommended practice including a structure designator and an accompanying variable-length pack registry including a version number.

Table 5 – Field descriptions for the UL key for global set encoding

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenation of subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE designator	1 octet	Always 0×34
5	Registry category designator	Sets and packs	1 octet	Always 0×02
6	Registry designator	Global sets	1 octet	See table 6
7	Structure designator	Designator of the structure variant within the global set registry	1 octet	See the accompanying standard or practice
8	Version number	Version of the global set registry which first defines the item specified by the global set designator	1 octet	Incrementing number
	Global set designator:			
9-16	Global set designator	The common portion of the UL key shared by all global tags	8 octets	Active number defines the octets needed to establish the common root for all global tags (2-8 octets)

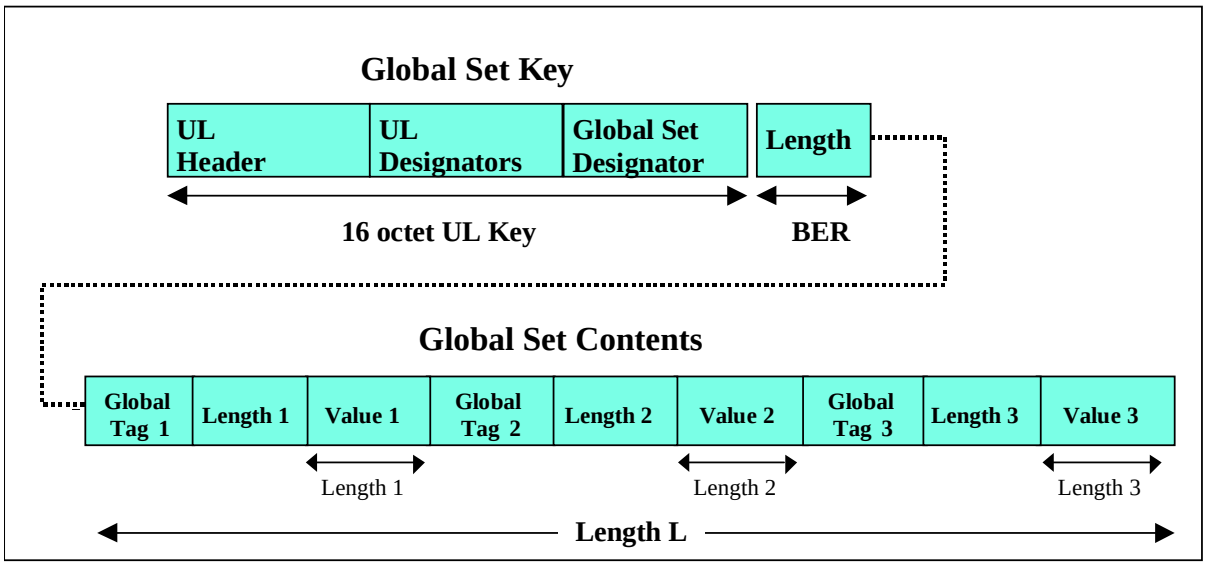

Figure 3 – KLV coded global set data structure

Table 6 – Coding of registry designator (octet 6) for global set syntax

Octet 6 value	Length fields	Description
0×02	BER short or long	Any length
0×22	1 octet	Length up to 255
0×42	2 octets	Length up to 65535
0×62	4 octets	Length up to 2^{32}-1

Table 7 – Field descriptions for the UL key for local set encoding

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenation of subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE designator	1 octet	Always 0×34
5	Registry category designator	Sets and packs	1 octet	Always 0×02
6	Registry designator	Local sets	1 octet	See table 8
7	Structure designator	Designator of the structure variant within the local set registry	1 octet	Defined by the local set registry and standard or practice
8	Version number	Version of the local set registry which first defines the item specified by the local set designator	1 octet	Incrementing number
	Local set designator:			
9-16	Local set designator	Defines the local set placement in a hierarchical structure	8 octets	Defined by the local set registry and standard or practice

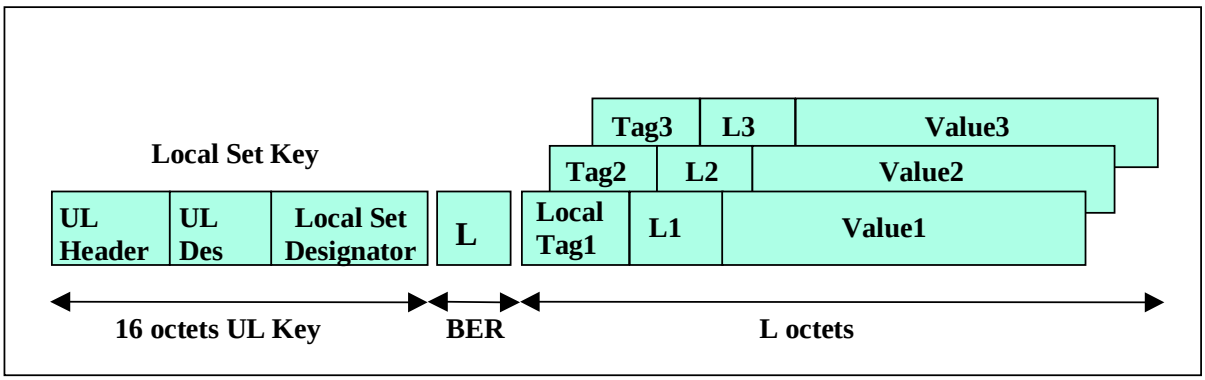

Figure 4 – KLV coded local set structure

Table 8 – Coding of registry designator (octet 6) for local set syntax

Octet 6 value	Length fields	Local tag fields length	Description
0×03	BER short or long	1 octet	Any length
0×13	BER short or long	2 octets	
0×1B	BER short or long	4 octets	
0×23	1 octet	1 octet	Length up to 255
0×33	1 octet	2 octets	
0×3B	1 octet	4 octets	
0×43	2 octets	1 octet	Length up to 65535
0×53	2 octets	2 octets	
0×5B	2 octets	4 octets	
0×63	4 octets	1 octet	Length up to 2^{32}-1
0×73	4 octets	2 octets	
0×7B	4 octets	4 octets	

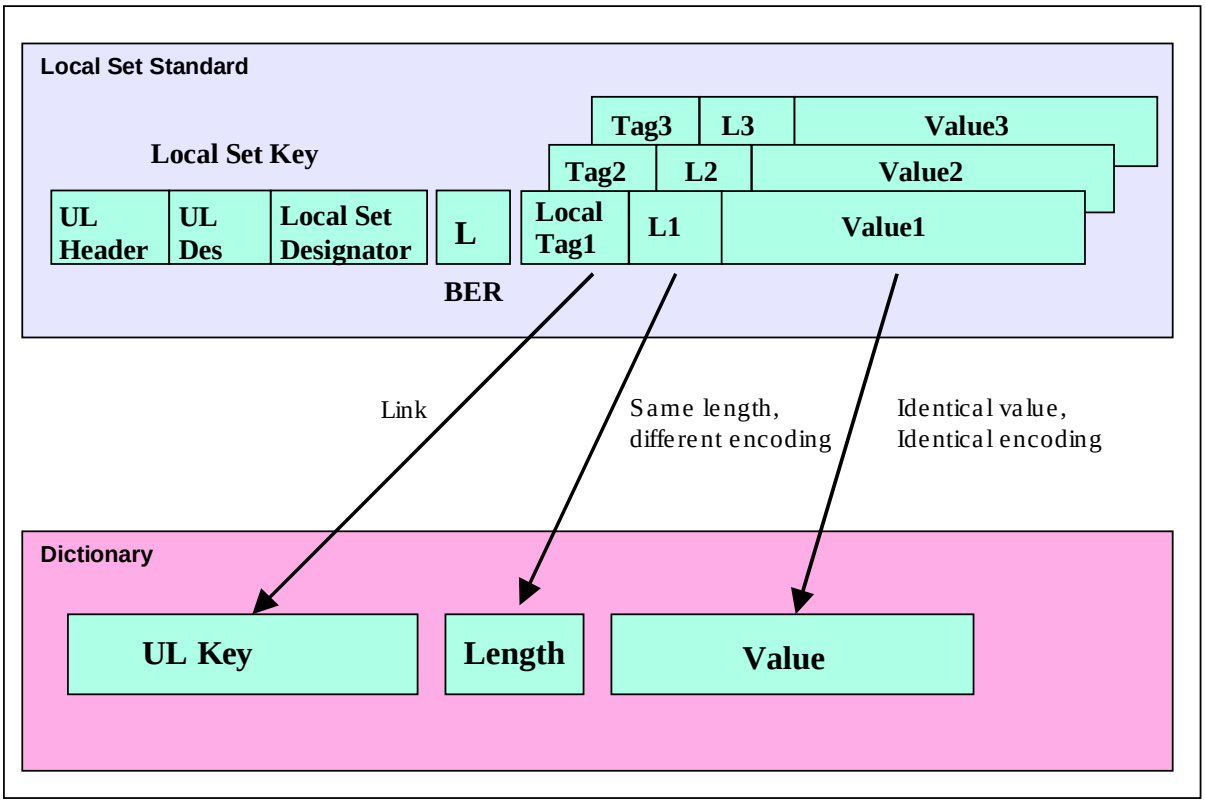

Figure 5 – Informative illustration of local set label to global key linking

The UL key of a variable-length pack shall be 16 octets in length.

The length of a variable-length pack shall be coded by default as per BER notation; long or short form as required.

The value of a variable-length pack shall be a sequence of KLV-encoded elements whose total length is given by the length field.

The UL key for variable-length pack is described in table 9. The variable-length pack designator is defined within the last 8 octets in the local set UL key. The variable-length pack key shall be defined in an associated standard or recommended practice and the variable-length pack UL key value shall be registered with the SMPTE Registration Authority to guarantee a unique variable-length pack UL key value.

The data structure for the encoding of variable-length packs is illustrated in figure 6.

The 16-octet variable-length pack UL key is followed by the variable-length pack length (encoded using BER short- or long-form encoding) which is followed by a number of elements which shall each be doublets of length and value.

The default specification of the length fields of individual elements is BER short- or long-form encoding. The full range of allowed length field lengths is defined by the registry designator, according to table 10. All length fields in the variable-length pack shall follow the same syntax.

Because the elements within a pack do not have a local tag, the order of the elements shall be specified by the defining standard or recommended practice.

A relevant variable-length pack standard or recommended practice shall define the link between each element and the corresponding UL key value by providing the UL key of the defining element. This linking definition is a mechanism that gives users of this standard the flexibility to define their own aliases for highly efficient coding. Developers of variable-length packs shall register the mapping between each tag in a variable-length pack and the defining UL key. Unlike universal sets and global sets, where the UL key of each element in the set can be losslessly recreated, the UL key of each element in a variable-length pack cannot be reconstructed without recourse to the defining standard or recommended practice and the corresponding registry.

Variable-length packs can accommodate recursion so that the UL key linked to an element may identify either a single data element from a dictionary or a data group from a set or pack standard or recommended practice and the corresponding registry.

5.5 Fixed-length packs

A fixed-length pack is defined as per the local set section, but does not have local tags. Thus, each element of a fixed-length pack is comprised of only a value field.

A fixed-length pack is similar to a variable length pack, but does not have length fields. Thus, each element of a fixed-length pack comprises only a value field. Elements in a fixed-length pack must appear in the defined order.

The UL key of a fixed-length pack shall be defined by an accompanying standard or recommended practice including a structure designator and an accompanying fixed-length pack registry including a version number.

The UL key of a fixed-length pack shall be 16 octets in length.

The length of a fixed-length pack shall be coded as per BER notation; long or short form as required.

The value of a fixed-length pack shall be a sequence of elements whose total length is given by the length field.

The UL key for a fixed-length pack is described in table 11. The fixed-length pack designator is defined within the last 8 octets in the local set UL key. The fixed-length pack key shall be defined in an associated standard or recommended practice and the fixed-length pack UL key value shall be registered with the SMPTE Registration Authority to guarantee a unique fixed-length pack UL key value.

The data structure for the encoding of fixed-length packs is illustrated in figure 7.

Because the elements within a fixed-length pack do not have a local tag, the order of the elements shall be specified by the defining standard or practice.

Table 9 – Field descriptions for the UL key for variable-length pack encoding

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenation of subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE designator	1 octet	Always 0×34
5	Registry category designator	Sets and packs	1 octet	Always 0×02
6	Registry designator	Variable-length packs	1 octet	See table 10
7	Structure designator	Designator of the structure variant within the variable-length pack registry	1 octet	Defined by the variable-length pack registry and standard or practice
8	Version number	Version of the variable-length pack registry which first defines the item specified by the variable-length pack designator	1 octet	Incrementing number
	Variable-length pack designator:			
9-16	Variable-length pack designator	Defines the variable-length pack placement in a hierarchical structure	8 octets	Defined by the variable-length pack registry and standard or practice

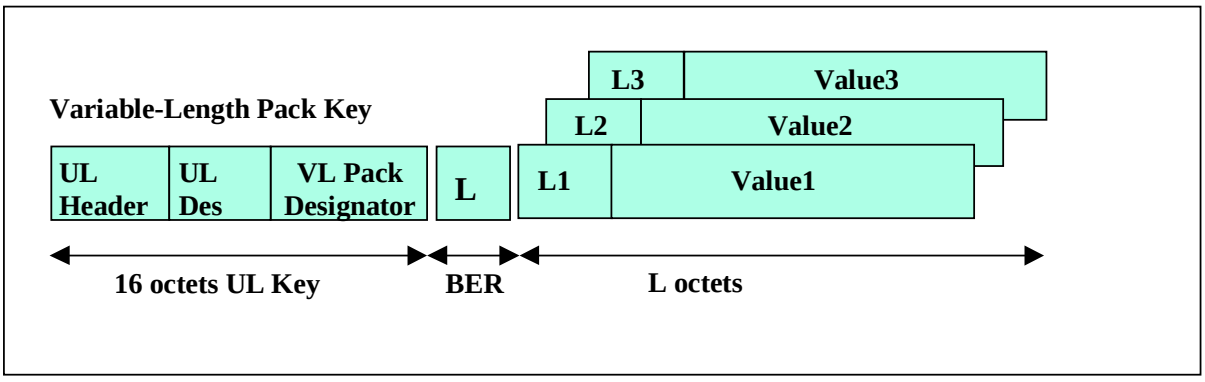

Figure 6 – KLV coded variable-length pack structure

Table 10 – Coding of registry designator (octet 6) for variable-length pack syntax

Octet 6 value	Length fields	Description
0×04	BER short or long	Default
0×24	1 octet	Length up to 255
0×44	2 octets	Length up to 65535
0×64	4 octets	Length up to 2^{32}-1

Table 11 – Field descriptions for the UL key for fixed-length pack encoding

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenation of subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE designator	1 octet	Always 0×34
5	Registry category designator	Sets and packs	1 octet	Always 0×02
6	Registry designator	Fixed-length packs	1 octet	Always 0×05
7	Structure designator	Designator of the structure variant within the fixed-length pack registry	1 octet	Incrementing number
8	Version number	Version of the fixed-length pack registry which first defines the item specified by the fixed-length pack designator	1 octet	Incrementing number
	Fixed-length pack designator:			
9-16	Fixed-length pack designator	Defines the fixed-length pack placement in a hierarchical structure	8 octets	Defined by the fixed-length pack registry and standard or practice

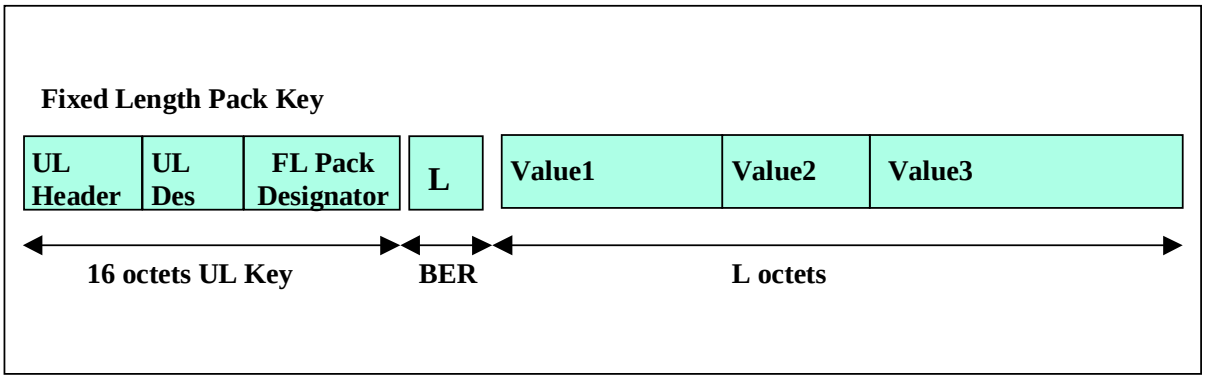

Figure 7 – KLV coded fixed-length pack structure

A relevant fixed-length pack standard or recommended practice shall define the link between each element and the corresponding UL key value by providing the UL key of the defining element. This linking definition is a mechanism that gives users of this standard the flexibility to define their own aliases for highly efficient coding. Developers of fixed-length pack packs shall register the mapping between each tag in a fixed-length pack pack and the defining UL key. Unlike universal sets and global sets, where the UL key of each element in the set can be losslessly recreated, the UL key of each element in a fixed-length pack pack cannot be reconstructed without recourse to the defining standard or recommended practice and the corresponding registry.

Fixed-length packs can accommodate recursion so that the UL key linked to an element may identify either a single data element from a dictionary or a data group from a set or pack standard or recommended practice and the corresponding registry.

6 Labels

Labels shall be used to identify any object whose meaning is entirely conveyed by the label itself, and there is no requirement for an independent value. Thus, a label does not need either a length field or a value field. Labels shall be defined in the labels dictionary. A label is illustrated in figure 8.

Within wrappers and containers, and sometimes even in sets, there is sometimes the need to identify aspects of the data contents which are not identified by the set, wrapper, or container UL key. Such an aspect can be identified by including a label in the set, wrapper, or container as a data item. It is necessary to define the presence of a label at a high level in the UL key so that decoders are aware that the item is a label only and not a KLV coded item.

NOTE – The label can also be the value in KLV encoding.

The structure of the UL key for labels is defined by table 12.

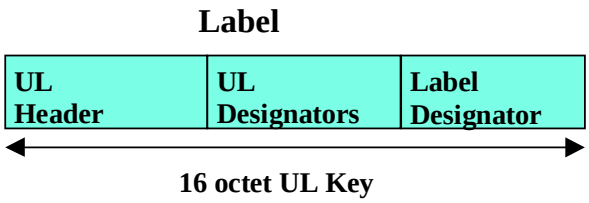

Figure 8 – UL key for labels

Table 12 – Field descriptions for the UL key for labels

No.	Field	Description	Length	Content/Format
	UL header:			
1	OID	Object identifier	1 octet	Always 0×06
2	UL size	16-octet size of the UL	1 octet	Always 0×0E
	UL designators:			
3	UL code	Concatenation of subidentifiers ISO, ORG	1 octet	Always 0×2B
4	SMPTE designator	SMPTE designator	1 octet	Always 0×34
5	Registry category designator	Labels	1 octet	Always 0×04
6	Registry designator	Specific labels registry	1 octet	Incrementing number
7	Structure designator	Designator of the structure variant within the labels registry	1 octet	Incrementing number
8	Version number	Version of the labels registry which first defines the item specified by the label designator	1 octet	Incrementing number
	Label designator:			
9-16	Label designator	Defines the label's placement in a hierarchical structure	8 octets	Defined by the labels registry and standard or practice

Annex A (normative)
Glossary of terms

A.1 basic encoding rules (BER): An ISO standard encoding for various constructs in ASN.1. Includes the encoding of object identifiers and also of length fields. The length octets of the KLV packet shall conform to the basic encoding rules (BER) for either the short-form or long-form encoding specified in ISO/IEC 8825-1, pars. 8.1.3.4 and 8.1.3.5.

A.2 container: A generic name for a data object which provides a framework to contain different kinds of information. The term is commonly applied to multimedia where audio, video, data essence, and metadata are formed into a single data object.

A.3 essence: Identified by the EBU/SMPTE Task Force for Harmonized Standards for the Exchange of Program Material as Bitstreams as video, audio, and/or data information. Essence could also be graphics, telemetry, photographs, or other information.

A.4 metadata: Generally referred to as data about data or data describing other data. More specifically, information that is considered ancillary to or otherwise directly complementary to the essence. Any information that a content provider considers useful or of value when associated with the essence being provided.

A.5 metadata dictionary: The standard database of approved, registered data element tags, their definitions, and their allowed formats.

A.6 metadata item: A broad term for a unit of metadata.

A.7 object identifier (OID): The first octet in the UL that identifies it as a UL — abbreviated OID. Always 06 in hexadecimal (hex) notation (0×06).

A.8 octet: An 8-bit word; directly equivalent to the commonly used byte.

A.9 primitive encoding: In ASN.1 notation, a definite-length encoding method that applies to simple encoding types and types derived from simple types by implicit tagging. It requires that the length of subidentifiers be known in advance.

A.10 SMPTE Registration Authority: A registration organization which keeps a record of the use of ANSI/SMPTE 298M UL keys and other reference data.

A.11 type or data type: Information about the representation of the data value.

A.12 wrapper: Identified by the EBU/SMPTE Task Force for Harmonized Standards for the Exchange of Program Material as Bitstreams as a means of wrapping video, audio, data essence, and metadata information into a common framework. In this definition, it is identical to the definition of a container, but wrappers may further be used to wrap further metadata around an already defined container. In this sense, a container is a multipurpose box which has audio-visual information and a wrapper is the packaging around the box including labeling and other supporting metadata.

Annex B (informative)
Organization of references

The organization of SMPTE standards and recommended practices addressing the coding of individual data items such as essence and metadata is illustrated in figure B.1. No single standard can contain all of the information needed to describe and encode all data. This encoding protocol standard and the metadata dictionary standards form the SMPTE normative standards for defining metadata and its coding. Informative SMPTE documents supplement the standards for encoding with examples and administrative instructions on managing the data standardization and registration process.

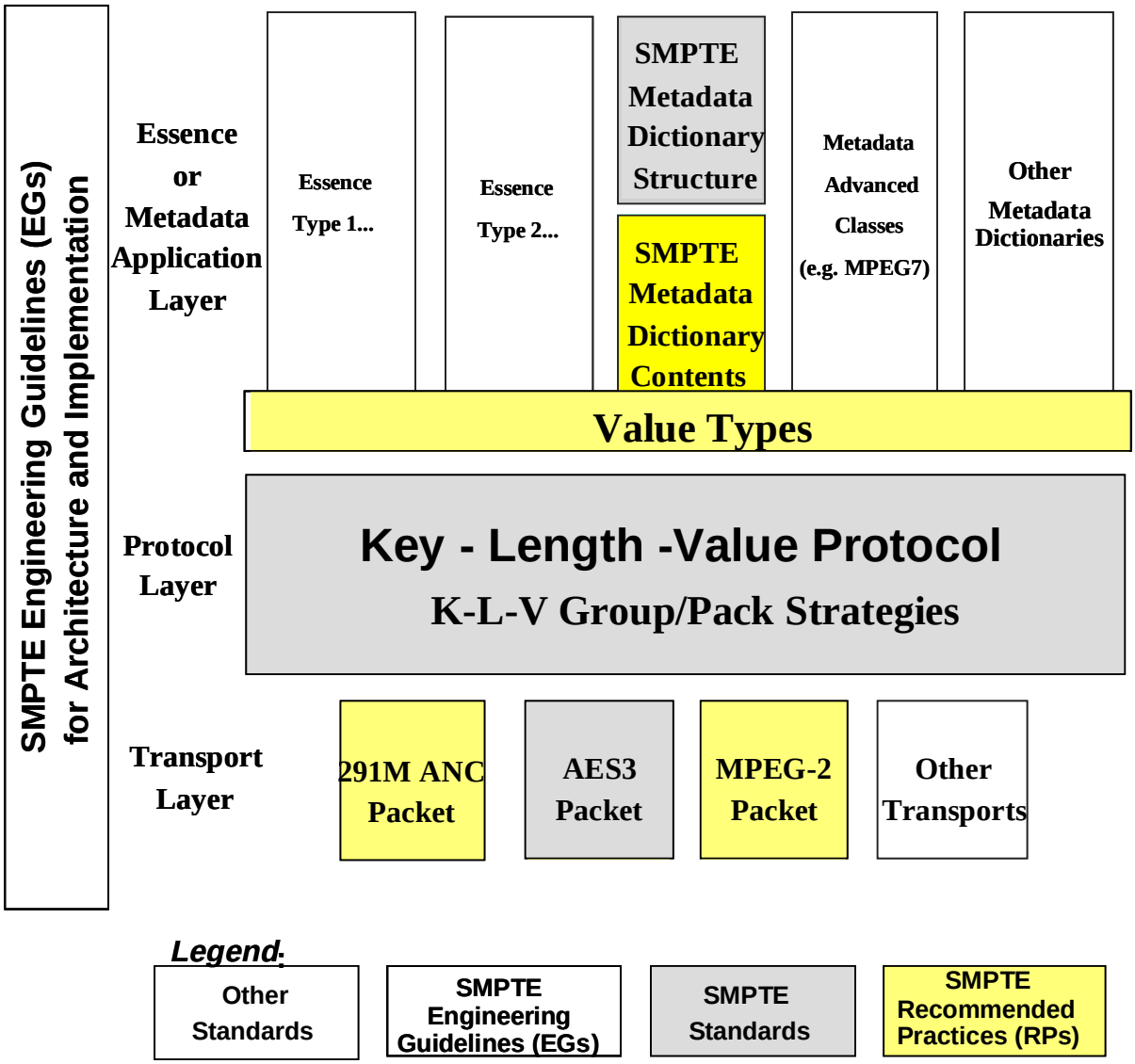

Figure B.1 – Organization of references

Annex C (informative)
Example usage of SMPTE UL

An expanded example of SMPTE universal label fields for metadata encoding is shown in table C.1. An example of SMPTE universal label fields for metadata encoding is shown in figure C.1

Table C.1 – Expanded example of SMPTE universal label fields for metadata encoding

Octet No.	Value (hex)	Example explanation	Reference
1	0×06	UL object identifier	ANSI/SMPTE 298M
2	0×0E	UL size	ANSI/SMPTE 298M
3	0×2B	Concatentation of designators ISO, ORG	ANSI/SMPTE 298M
4	0×34	SMPTE designator	ANSI/SMPTE 298M
5	0×01	SMPTE dictionaries designator	SMPTE Registration Authority
6	0×01	Metadata dictionary designator	SMPTE Registration Authority
7	0×01	Structure standard reference number	SMPTE Registration Authority
8	0×01	Standard version number	SMPTE Registration Authority
9	0×01	Metadata class: Identifiers and locators	Metadata dictionary
10	0×01	Unique identifier subclass	Metadata dictionary
11	0×11	ISO identifiers	Metadata dictionary
12	0×01	ISO audio-visual number (ISAN)	Metadata dictionary
13	0×00	Unused	Metadata dictionary
14	0×00	Unused	Metadata dictionary
15	0×00	Unused	Metadata dictionary
16	0×00	Unused	Metadata dictionary

06 0E 2B 34 01 01 01 01 01 01 11 01 00 00 00 00

NOTE – Hex-encoded octets; 0x removed and separated for readability.

Figure C.1 – Example SMPTE universal label fields for metadata encoding

Annex D (informative)
Example of the KLV encoding of a single metadata item

Table D.1 shows an example of the fields that comprise the KLV protocol used for an individual value encoding of a human-assigned main title for a video segment. For clarity, each octet of the UL key is separated by spaces.

Table D.1 – Informative example of K-L-V individual value encoding of metadata

Key (hex encoded)	06 0E 2B 34 01 01 01 01 01 05 01 02 $00\ 00\ 00\ 00_h$
Description	Main title (ISO 7-bit char)
Length (binary) [hex]	0×10
Value (ASCII)	Yesterdays World

Annex E (informative)
Example of a universal set

In the example universal metadata set in table E.1, the three elements of main title, ISAN number, and supply organization can be in any order and each entry is self-contained with its own individual universal key-length-value. Similarly, if one or more of the elements of the defined universal set is missing, the remaining valid parts of the universal set can still be recovered because each has its own key-length-value combination.

Table E.1 – Informative example of K-L-V universal set encoding of metadata (octets separated by spaces for readability)

Universal set UL	06 0E 2B 34 02 01 01 01 01 01 01 01 00 00 00 00
Universal set length	0×59
Universal key 1	06 0E 2B 34 01 01 01 01 01 05 01 02 00 00 00 00
Description	Main title (ISO 7-bit char)
Length 1	0×10
Value 1	Yesterdays World
Universal key 2	06 0E 2B 34 01 01 01 01 01 01 11 00 $00\ 00\ 00\ 00_h$
Description	ISAN number
Length 2	0×10
Value 2	01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
Universal key 3	06 0E 2B 34 01 01 01 01 02 01 01 00 $00\ 00\ 00\ 00_h$
Description	Supply organization (ISO 7-bit char)
Length 3	0×06
Value 3	WXYZ15

Annex F (informative)
Example of a global set

In the example global metadata set in table F.1, the three elements of main title, ISAN number, and supply organization can be in any order and each entry is self-contained with its own individual global tag-length-value. Similarly, if one or more of the elements of the defined global set is missing, the remaining valid parts of the global set can still be recovered because each has its own tag-length-value combination.

Table F.1 – Informative example of K-L-V global set encoding of metadata (octets separated by spaces for readability)

Global set UL	06 0E 2B 34 02 02 01 01 06 0E 2B 34 01 01 01 01
Global set length	0×36
Global tag 1	01 05 01 02 00
Original UL key	06 0E 2B 34 01 01 01 01 01 05 01 02 00 00 00 00
Description	Main title (ISO 7-bit char)
Length 1	0×10
Value 1	Yesterdays World
Global tag 2	01 01 11 00
Original UL key	06 0E 2B 34 01 01 01 01 01 01 11 00 00 00 00 00
Description	ISAN number
Length 2	0×10
Value 2	01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
Global tag 3	02 01 01 00
Original UL key	06 0E 2B 34 01 01 01 01 02 01 01 00 00 00 00 00
Description	Supply organization (ISO 7-bit char)
Length 3	0×06
Value 3	WXYZ15

Annex G (informative)
Example of a local set

In the example local metadata set in table G.1, the three elements of main title, ISAN number, and supply organization can be in any order and each entry is self-contained with its own individual local tag-length-value. Similarly, if one or more of the elements of the defined local set is missing, the remaining valid parts of the local set can still be recovered because each has its own local tag-length-value combination.

Table G.1 – Informative example of K-L-V local set encoding of metadata (octets separated by spaces for readability)

Local set UL	06 0E 2B 34 02 03 01 01 06 0E 2B 34 01 01 01 01
Local set length	0×2c
Local tag 1	0×01
Original UL key	06 0E 2B 34 01 01 01 01 01 05 01 02 00 00 00 00
Description	Main title (ISO 7-bit char)
Length 1	0×10
Value 1	Yesterdays World
Local tag 2	0×02
Original UL key	06 0E 2B 34 01 01 01 01 01 01 11 00 00 00 00 00
Description	ISAN number
Length 2	0×10
Value 2	01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
Local tag 3	0×03
Original UL key	06 0E 2B 34 01 01 01 01 02 01 01 00 00 00 00 00
Description	Supply organization (ISO 7-bit char)
Length 3	0×06
Value 3	WXYZ15

Annex H (informative)
Example of a variable-length pack

In the example variable-length pack in table H.1, the three elements of main title, ISAN number, and supply organization must be in the order specified and must all be present. If one or more of the elements of the defined variable-length pack is missing, the remaining valid parts of the variable-length pack cannot be recovered.

Table H.1 – Informative example of K-L-V variable-length pack encoding of metadata (octets separated by spaces for readability)

VL pack UL	06 0E 2B 34 02 04 01 01 06 0E 2B 34 01 01 01 01
VL pack length	0×29
Original UL key	06 0E 2B 34 01 01 01 01 01 05 01 02 00 00 00 00
Description	Main title (ISO 7-bit char)
Length 1	0×10
Value 1	Yesterdays World
Original UL key	06 0E 2B 34 01 01 01 01 01 01 11 00 00 00 00 00
Description	ISAN number
Length 2	0×10
Value 2	01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
Original UL key	06 0E 2B 34 01 01 01 01 02 01 01 00 00 00 00 00
Description	Supply organization (ISO 7-bit char)
Length 3	0×06
Value 3	WXYZ15

Annex I (informative)
Example of a fixed-length pack

In the example fixed-length pack in table I.1, the three elements of main title, ISAN number, and supply organization must be in the order and of the length specified. If one or more of the elements of the defined fixed-length pack is missing, the remaining valid parts of the fixed-length pack cannot be recovered.

Table I.1 – Informative example of K-L-V fixed-length pack encoding of metadata (octets separated for readability)

FL pack UL	06 0E 2B 34 02 01 01 01 06 0E 2B 34 01 01 01 01
FL pack length	0×26
Original UL key	06 0E 2B 34 01 01 01 01 01 05 01 02 00 00 00 00
Description	Main title (ISO 7-bit char)
Value 1	Yesterdays World
Original UL key	06 0E 2B 34 01 01 01 01 01 01 11 00 00 00 00 00
Description	ISAN number
Value 2	01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16
Original UL key	06 0E 2B 34 01 01 01 01 02 01 01 00 00 00 00 00
Description	Supply organization (ISO 7-bit char)
Value 3	WXYZ15_ _ _ _ _ _ _ _ _ _

Annex J (informative)
Example of a label

An example of a label is given in table J.1.

Table J.1 – Example of a label

Label UL	06 0E 2B 34 04 01 01 01 11 22 33 44 55 00 00 00
Description	1/2-in type J cassette
NOTE – Octets separated by spaces for readability.	

Annex K (informative)
Bibliography

ISO/IEC 8824-1:1998 (ITU-T X.680), Information Technology — Abstract Syntax Notation One (ASN.1) — Specification of Basic Notation

Final Report: Analyses and Results, EBU/SMPTE Task Force for Harmonized Standards for the Exchange of Program and Material as Bitstreams, July 1998. SMPTE J.107:605-815