Registered PER Encoding Instructions (PER EIs)

(Updated December 2010)

Introduction

In accordance with ITU-T X.695 | ISO/IEC 8825-6 this Web page contains details of PER Encoding Instructions.

Each PER EI can be applied to one of several ASN.1 types, and modifies the specification in ITU-T X.691| ISO/IEC 8825-2 for the UNALIGNED variants of the BASIC-PER and of the CANONICAL-PER encodings**. PER EIs have no effect if the ALIGNED variants of BASIC-PER or if CANONICAL-PER encodings are specified.**

Without PER EIs, PER specifies encodings for all abstract values of all types that are permitted by any specified PER-visible constraints.

With PER EIs, there can be some abstract values that are permitted by PER-visible constraints (or lack of such constraints) that cannot be encoded for that type (see ITU-T X.695 | ISO/IEC 8825-6, clause 10). In general, users of PER EIs in specifications should try to avoid this situation by suitable use of PER-visible constraints, but it is not a specification error to allow this situation to arise. If there are permitted abstract values that cannot be encoded using the PER EI, there will be a failure when an encoding is attempted with those abstract values.

NOTE – This may occur in an application originating the transfer of messages, or in a relay application.

It is important to note that ITU-T X.695 | ISO/IEC 8825-6 prohibits the application of a PER EI to an ASN.1 type that is extensible for PER encodings.

Three categories of PER Encoding Instruction specification are detailed in ITU-T X.695 | ISO/IEC 8825-6 for recording on this Web-site:

a) NEEDED: The PER EI is under active consideration

b) READY: The PER EI has been approved by one or other of ITU-T SG17 or ISO/IEC JTC1/SC6, and the full specification has been published on this web-site. The dates at which either or both of these approvals occurred is stated for each PER EI, including the date at which the full specification has been posted on this web-site.

c) APPROVED: The PER EI has been approved by both ITU-T SG17 and ISO/IEC JTC1/SC6, and the full specification has been published on this web-site as READY for at least six months. The date of movement to APPROVED status is listed.

This page is currently maintained by the ASN.1 Rapporteur Group of ITU-T Study Group 17. Contact information can be obtained from the ITU-T TSB.

CONTENTS

1 Normative References

- Recommendation ITU-T X.691 (2008) | ISO/IEC 8825-2:2008, *Information technology – ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)*.
- Recommendation ITU-T X.695 (2008) | ISO/IEC 8825-6:2008, *Information technology – ASN.1 encoding rules: Registration and application of PER encoding instructions*.

2 Definition of terms

ITU-T X.691 | ISO/IEC 8825-2 and ITU-T X.695 | ISO/IEC 8825-6 define terms which are used in this document.

3 Abbreviations

PER Packed Encoding Instructions (ITU-T X.691 | ISO/IEC 8825-2)

4 Applicability of PER EIs in the READY or APPROVED categories

Table 1 lists all the types that can have PER EIs that are in the READY or APPROVED categories, with the PER EIs that can be applied to them.

Table 1 – Types which can have READY or APPROVED PER EIs applied to them

ASN.1 type	Available PER EIs	Part of encoding affected
INTEGER (see clause 6.1.4)	[SIZE *n*]	Number of bits in the encoding and the value to be encoded if the PER-visible lower-bound is non-zero
NULL (see clause 6.1.5)	[SIZE *n*]	Number of bits in the encoding
BOOLEAN (see clause 6.1.6)	[SIZE *n*]	Number of bits in the encoding
ENUMERATED (see clause 6.1.7)	[SIZE *n*]	Number of bits used for the enumeration index
CHOICE (see clause 6.1.8)	[SIZE *n*]	Number of bits used for the choice determinant
SEQUENCE (see clause 6.1.9)	[SIZE *n*]	Number of bits used for the optionality bit-map
SET (see clause 6.1.9)	[SIZE *n*]	Number of bits used for the optionality bit-map
Some Restricted Character String types (see clause 6.2)	[NULL]	Termination mechanism for the character string
All (see clause 6.3 for [LENGTH *n*] and clause 6.4 for [COUNT-BITS] and [COUNT-OCTETS])	[LENGTH *n*]	Inclusion and size of a preceding length field, if not already present in a PER encoding (also prevents use of indefinite length and subtraction of a lower bound)
	[COUNT-BITS]	Makes the length count a count of bits
	[COUNT-OCTETS]	Makes the length count a count of octets

5 Interactions between PER EIs applied to the same type

For the purpose of interactions, all that matters is that a type has a final encoding instruction of the specified sort applied. (See ITU-T X.695 | ISO/IEC 8825-6, clause 3.2 for the definition of final encoding instruction). Use of the phrase "shall not both be applied to" is to be interpreted as meaning that there shall not be a final encoding instruction of the two sorts.

Applications of both `[LENGTH n]` (with or without `[COUNT-BITS]` and `[COUNT-OCTETS]`) and `[SIZE n]` are independent, and have no interaction.

Applications of `[COUNT-BITS]` and `[COUNT-OCTETS]` without `[LENGTH n]` shall not be used.

`[COUNT-BITS]` and `[COUNT-OCTETS]` shall not both be applied to the same type.

Use of `[NULL]` prohibits use of `[LENGTH n]`

There are no other interactions between encoding instructions in the APPROVED or READY categories.

6 Specification of PER EIs in the APPROVED category

NOTE – All the following PER EIs were approved by ITU-T SG 17 on 16th April 2010 and were approved by ISO/IEC SC 6 on 1st Oct 2010 and the full specification was posted to this web-site in June 2010. These PER EIs were therefore moved to the approved category in December 2010.

6.1 Size of a field used to encode part or all of an abstract value (in bits)

6.1.1 Identifying Keyword and EIDetail productions

NOTE – The notation used in the following productions is specified in ITU-T X.695 | ISO/IEC 8825-6, clause 5.

The productions are:

IdentifyingKeyword ::= `SIZE`

EIDetail ::= number

where number shall be positive, non-zero, and shall not be greater than 8192 (for pragmatic reasons).

6.1.2 Syntax

```
[SIZE n]
```

EXAMPLES:

```
SIZE-EXAMPLE1 ::= [SIZE 8] INTEGER (0..256)

SIZE-EXAMPLE2 ::= [SIZE 8] INTEGER (-128..127)

SIZE-EXAMPLE3 ::= [SIZE 16] INTEGER

SIZE-EXAMPLE4 ::= [SIZE 8] NULL

SIZE-EXAMPLE5 ::= [SIZE 4] BOOLEAN

SIZE-EXAMPLE6 ::= [SIZE 2] ENUMERATED {enum1, enum2, enum3}

SIZE-EXAMPLE7 ::= [SIZE 16] CHOICE {c1 Type1, c2 Type2, c3 Type3}

SIZE-EXAMPLE8 ::= [SIZE 3] SEQUENCE {
      s1 Type1 OPTIONAL,
      s2 Type2,
      s3 Type3 OPTIONAL }

SIZE-EXAMPLE9 ::= [SIZE 3] SET {
      s1 Type1 OPTIONAL,
      s2 Type2,
      s3 Type3 OPTIONAL }
```

6.1.3 Simple description

6.1.3.1 This encoding instruction can be applied to the following ASN.1 types (and no others): INTEGER, NULL, BOOLEAN, ENUMERATED, CHOICE, SEQUENCE and SET (see Table 1).

6.1.3.2 This specifies the number of bits to be used for encoding an integer, a null, a Boolean, an enumeration determinant, a choice index or the optionality bit-map in a sequence or set.

6.1.3.3 This also eliminates the subtraction of the lower bound in a PER-visible constraint where applicable.

6.1.4 Effect on INTEGER encodings

6.1.4.1 The presence of a `[SIZE n]` as a final encoding instruction specifies the number of bits to be used for encoding the integer.

6.1.4.2 The procedures in the following subclauses shall replace the procedures of ITU-T X.691 | ISO/IEC 8825-2, clause 13.

6.1.4.3 The value shall be encoded without subtraction of a PER-visible lower bound. If there are no negative values permitted by PER-visible constraints, the encoding shall be the non-negative-binary-integer encoding as specified in ITU-T X.691 | ISO/IEC 8825-2, 11.3, otherwise the encoding shall be the 2's-complement-binary-integer encoding as specified in ITU-T X.691 | ISO/IEC 8825-2, 11.4. In both cases, the number of bits used for the encoding shall be the number specified in the PER EI.

6.1.4.4 The encoding shall always be added to the field list, even if there is only a single value permitted by PER-visible constraints.

6.1.4.5 It is possible that the encoding using the PER EI will be incapable of encoding some or all of the abstract values of the type that are permitted by PER-visible constraints. If there are no abstract values that can be encoded, it is a PER EI specification error. If there are one or more abstract values that cannot be encoded, then an attempt to encode any of those abstract values will result in an encoding failure.

NOTE – In the latter case, the encoding failure could result from an attempt to convert from an encoding that is not restricted by PER EIs to the encoding specified here.

6.1.5 Effect on NULL encodings

PER normally encodes a **`NULL`** with zero bits (adding nothing to the field list). The presence of a final encoding instruction of `[SIZE n]` requires the addition of ***n*** bits to the field list, with all bits set to zero by a conforming encoder and ignored by a conforming decoder.

NOTE – This is one of several ways of obtaining a reserved field (or padding bits) in the encoding.

6.1.6 Effect on BOOLEAN encodings

The presence of a `[SIZE n]` final encoding instruction specifies the number of bits to be used for encoding the Boolean. The value of ***n*** shall not be zero. The least significant bit of the ***n***-bit field shall be set to 1 for **`TRUE`** and 0 for **`FALSE`**. The other bits shall all be set to zero by an encoder and ignored by a decoder.

NOTE - To use all zeros as **`TRUE`** and any non-zero for the ***n***-bits as **`FALSE`**, or to reverse the encodings of **`TRUE`** and **`FALSE`**, requires ECN, and is not currently supported by PER EIs.

6.1.7 Effect on ENUMERATED encodings

6.1.7.1 The presence of a `[SIZE n]` final encoding instruction specifies the number of bits to be used for encoding the enumeration index.

6.1.7.2 In a normal PER encoding, the enumeration index is zero for the first index, one for the second, etc, and the number of bits used is the minimum necessary to encode the largest integer value that appears (that of the last enumeration). With this PER EI, a field of ***n*** bits (as specified in `[SIZE n]`) is used for the encoding (where ***n*** may be greater than necessary.

6.1.7.3 It is a PER EI specification error to assign a `[SIZE n]` PER EI if ***n*** is insufficient to allow a non-negative-binary-integer encoding (as specified in ITU-T X.691 | ISO/IEC 8825-2, 11.3) of the last enumeration using only **`n`** bits.

6.1.8 Effect on CHOICE encodings

6.1.8.1 The presence of a `[SIZE n]` final encoding instruction specifies the number of bits to be used for encoding the choice determinant.

6.1.8.2 In a normal PER encoding, the choice determinant is zero for the first choice alternative, one for the second, etc, and the number of bits used is the minimum necessary to encode the largest integer value that appears (that of the last choice determinant).

6.1.8.3 It is a PER EI specification error to assign a `[SIZE n]` PER EI if ***n*** is insufficient to allow a positive integer encoding of the last choice determinant. If ***n*** is greater than this, then earlier bits shall be all set to zero by a conforming encoder and shall be ignored by a conforming decoder.

6.1.9 Effect on SEQUENCE and SET encodings

6.1.9.1 The presence of a `[SIZE n]` final encoding instruction specifies the number of bits to be used for encoding the optionality bit-map for a sequence or set that has optional elements. If there are no optional elements, then there is a padding field of n bits set to zero and ignored by the recipient.

NOTE – This was done to allow for possible future additions of optional elements.

6.1.9.2 In a normal PER encoding, there is one bit for each of the optional elements, indicating their presence or absence. The effect of a `[SIZE n]` final encoding instruction is to insert additional zero bits at the end of the normal PER optionality bit-map if ***n*** is greater than the number of optional elements. The additional bits shall all be set to zero by a conforming encoder and shall be ignored by a conforming decoder.

NOTE - The left-justification of the optionality bit-map is to allow for the possible addition later of further optional elements at the end of the set or sequence. This provides a crude extensibility mechanism.

6.1.9.3 The `[SIZE n]` final encoding instruction is a specification error if ***n*** is less than the number of optional elements.

6.2 Null-terminated character strings

6.2.1 Restrictions

This can only be applied to **`IA5String`**, **`VisibleString`**, **`BMPString`**, **`UniversalString`**, **`UTF8String`**, **`NumericString`** and **`PrintableString`**.

6.2.2 Identifying Keyword and EIDetail productions

NOTE – The notation used in the following productions is specified in ITU-T X.695 | ISO/IEC 8825-6, clause 5.

The productions are:

IdentifyingKeyword ::= `NULL`

EIDetail ::= empty

6.2.3 Syntax

```
[NULL]
```

EXAMPLES:

```
NULL-EXAMPLE1 ::= [NULL] IA5String

NULL-EXAMPLE2 ::= [NULL] VisibleString

NULL-EXAMPLE2 ::= [NULL] BMPString

NULL-EXAMPLE3 ::= [NULL] UniversalString

NULL-EXAMPLE4 ::= [NULL] UTF8String

NULL-EXAMPLE5 ::= [NULL] NumericString

NULL-EXAMPLE6 ::= [NULL] PrintableString
```

6.2.4 Simple description

The concept of null-termination of an ASCII string is well-understood, and is often used in in-core data structures. This encoding instruction extends that option to the PER encoding.

6.2.5 Effect on encodings

6.2.5.1 Use of this PER EI disables fragmentation of the string for long strings.

6.2.5.2 Use of this PER EI removes any length field encoding.

6.2.5.3 Abstract characters that would encode as if they were the null-terminator cannot be encoded if this EI is applied.

NOTE – Allowing such characters in the specification is not an error, but applications should normally apply a suitable constraint to eliminate such abstract characters.

6.2.5.4 Use of this EI makes all alphabet constraints not PER-visible.

NOTE – This means that all characters encode directly as ASCII, 16-bit BMP characters, 32-bit UCS characters, or UTF8 characters.

6.2.5.5 Any size constraint on the abstract string applies only to the number of abstract characters encoded. The null-terminator is an addition to the encoding of the abstract characters.

6.2.5.6 The null-terminator encoding shall be as follows:

– For **`IA5String`**: 8 zero bits.

– For **`VisibleString`** (**`ISO646String`**): 8 zero bits.

– For **`BMPString`**: 16 zero bits.

– For **`UniversalString`**: 32 zero bits.

– For **`UTF8String`**: 8 zero bits.

– For **`NumericString`**: 8 zero bits.

– For **`PrintableString`**: 8 zero bits.

6.3 Presence and size of a length field.

6.3.1 Identifying Keyword and EIDetail productions

NOTE – The notation used in the following productions is specified in ITU-T X.695 | ISO/IEC 8825-6, clause 5.

The productions are:

IdentifyingKeyword ::= `LENGTH`

EIDetail ::= number

where number shall be positive, non-zero, and shall not be greater than 512 (for pragmatic reasons).

6.3.2 Syntax

```
[LENGTH n]
```

EXAMPLES:

```
LENGTH-EXAMPLE1 ::= [LENGTH 4] INTEGER (0..2048)

/* The above would provide a 4-bit length field before the integer encoding, which
would accommodate values up to a 15 bit encoding, but the abstract syntax
constraint still applies. */

LENGTH-EXAMPLE2 ::= [LENGTH 8] INTEGER (-128..127)

LENGTH-EXAMPLE3 ::= [LENGTH 16] INTEGER

LENGTH-EXAMPLE4 ::= [LENGTH 32] SEQUENCE {
      s1 Type1 OPTIONAL,
      s2 Type2,
      s3 Type3 OPTIONAL }

LENGTH-EXAMPLE5 ::= [LENGTH 64] SEQUENCE OF SomeOtherType
```

6.3.3 Simple description

This can be applied to any ASN.1 type, and specifies that the type shall **always** be encoded with a preceding length field of a specified length and with specified units (bits, octets, components, or characters) for the length field.

NOTE 1 – The units can be changed by applying additionally a `[COUNT-BITS]` or `[COUNT-OCTETS]` PER EI.

NOTE 2 – If the PER encoding already specifies a preceding length field, then this PER EI modifies the preceding length field encoding as specified in 6.3.4.3 – it does not add a further preceding length field.

NOTE 3 – This PER EI also disables fragmentation and the subtraction of a lower bound from a length encoding due to a PER-visible constraint.

TUTORIAL: PER has relatively complex rules on when a preceding length field is to be encoded, which can be summarised by saying that if PER-visible constraints (for example on the size of an integer or on the number of iterations in a sequence-of or set-of) can be used to determine the content of the length field, it is not encoded. A sequence type never has a length field. PER also has a fragmentation mechanism to permit the encoding of arbitrarily large integer values or arbitrarily long sequence-of iterations. Finally, the units of a length field (bits, octets, components, or characters) vary according to the type being encode, but can be changed by use of a `[COUNT-BITS]` or `[COUNT-OCTETS]` PER EI.

6.3.4 Effect on encodings

6.3.4.1 The ASN.1 type, with all PER-visible constraints may or may not have a preceding length field in the PER encoding. The effects are different in the two cases.

6.3.4.2 If the PER encoding does not have a preceding length field, then a preceding length field shall be added, counting in bits, unless there is also a `[COUNT-OCTETS]` PER EI (see also 6.4.3.2).

6.3.4.3 If the PER encoding has a preceding length field, then an additional preceding length field shall not be added, and the `[LENGTH n]` PER EI changes the size of the length field, but does not affect the units unless there is also a `[COUNT-BITS]` or `[COUNT-OCTETS]` PER EI.

6.3.4.4 This encoding instruction disables fragmentation as specified in ITU-T X.691 | ISO/IEC 8825-2, 11.9.

NOTE – This implies that a single length field is included using the original units (unless overridden by a `[COUNT-BITS]` or `[COUNT-OCTETS]` PER EI).

6.3.4.5 The preceding length field shall be *n* bits, with a PER-specified count of the units (unless the units are overridden by a `[COUNT-BITS]` or `[COUNT-OCTETS]` PER EI). The lower bound from an ASN.1 size constraint shall not be subtracted before encoding the length. The encoding of the length shall be the non-negative-binary-integer encoding as specified in ITU-T X.691 | ISO/IEC 8825-2, 11.3.

6.3.4.6 It is possible that the encoding using the PER EI will be incapable of encoding some or all of the abstract values of the type that are permitted by PER-visible constraints. If there are no abstract values that can be encoded, it is a PER EI specification error. If there are one or more abstract values that cannot be encoded, then an attempt to encode any of those abstract values will result in an encoding failure.

NOTE – In the latter case, the encoding failure could result from an attempt to convert from an encoding that is not restricted by PER EIs to the encoding specified here.

6.4 Units used for the count in a length field

6.4.1 Identifying Keyword and EIDetail productions

NOTE – The notation used in the following productions is specified in ITU-T X.695 | ISO/IEC 8825-6, clause 5.

The productions are:

IdentifyingKeyword ::= `COUNT-BITS | COUNT-OCTETS`

EIDetail ::= empty

6.4.2 Syntax

```
[COUNT-BITS]
[COUNT-OCTETS]
```

NOTE – `[COUNT-COMPONENTS]` is not provided, as this is the default for all types to which such an EI could be applied.

EXAMPLES:

```
COUNT-EXAMPLE1 ::= [COUNT-BITS] [LENGTH 128] SEQUENCE {......}

COUNT-EXAMPLE2 ::= [COUNT-OCTETS] [LENGTH 128] OCTET STRING

COUNT-EXAMPLE3 ::= [COUNT-OCTETS] BIT STRING /* This is a specification error
because [LENGTH n] is not present, but even if it was present, [COUNT-OCTETS] could
not be used on an unconstrained bit string as some abstract values will not produce
an encoding that is a multiple of 8 bits (see 6.4.3.2) */
```

6.4.3 Restrictions

6.4.3.1 This can only be applied if there is a `[LENGTH n]` EI applied (see 6.3).

6.4.3.2 It is a specification error if `[COUNT-OCTETS]` is used and some abstract values do not produce an encoding that is a multiple of 8 bits.

NOTE - This can sometimes be hard to determine statically. Tools are likely to be silent if it is clear that there is not a problem, and may give an error if they can determine that the specification is in error, and a warning otherwise.

6.4.4 Simple description

This specifies the units to be used for the length determinant required by the `[LENGTH` ***n*** `]` PER EI.

6.4.5 Effect on encodings

This PER EI qualifies the `[LENGTH` ***n*** `]` PER EI. It determines that the units used in the preceding length field shall be either bits (`[COUNT-BITS]`) or octets (`[COUNT-OCTETS]`) – see also 6.3.

7 Specification of PER EIs in the READY category

None.

8 Specification of PER EIs in the NEEDED category

There are no PER EIs in the needed category, but there are some in the POSSIBLE category that are recorded by the ASN.1 Group as a Standing Document, as required by ITU-T X. 695 | ISO/IEC 8825-6. These will be transferred to the NEEDED category and progressed when a need to use these is identified in user specifications.