Unión Internacional de Telecomunicaciones

UIT-T

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

X.692

Enmienda 1

(08/2004)

SERIE X: REDES DE DATOS, COMUNICACIONES DE SISTEMAS ABIERTOS Y SEGURIDAD

Gestión de redes de interconexión de sistemas abiertos y aspectos de sistemas – Notación de sintaxis abstracta uno

---

Tecnología de la información – Reglas de codificación de notación de sintaxis abstracta uno: Especificación de la notación de control de codificación

Enmienda 1: Soporte de extensibilidad

Recomendación UIT-T X.692 (2002) – Enmienda 1

UIT-T

ITU Unión Internacional de Telecomunicaciones

RECOMENDACIONES UIT-T DE LA SERIE X

REDES DE DATOS Y COMUNICACIÓN ENTRE SISTEMAS ABIERTOS

REDES PÚBLICAS DE DATOS	
Servicios y facilidades	X.1–X.19
Interfaces	X.20–X.49
Transmisión, señalización y conmutación	X.50–X.89
Aspectos de redes	X.90–X.149
Mantenimiento	X.150–X.179
Disposiciones administrativas	X.180–X.199
INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Modelo y NOTAción	X.200–X.209
Definiciones de los servicios	X.210–X.219
Especificaciones de los protocolos en modo conexión	X.220–X.229
Especificaciones de los protocolos en modo sin conexión	X.230–X.239
Formularios para declaraciones de conformidad de implementación de protocolo	X.240–X.259
Identificación de protocolos	X.260–X.269
Protocolos de seguridad	X.270–X.279
Objetos gestionados de capa	X.280–X.289
Pruebas de conformidad	X.290–X.299
INTERFUNCIONAMIENTO ENTRE REDES	
Generalidades	X.300–X.349
Sistemas de transmisión de datos por satélite	X.350–X.369
Redes basadas en el protocolo Internet	X.370–X.399
SISTEMAS DE TRATAMIENTO DE MENSAJES	X.400–X.499
DIRECTORIO	X.500–X.599
GESTIÓN DE REDES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS Y ASPECTOS DE SISTEMAS	
Gestión de redes	X.600–X.629
Eficacia	X.630–X.639
Calidad de servicio	X.640–X.649
Denominación, direccionamiento y registro	X.650–X.679
Notación de sintaxis abstracta uno	**X.680–X.699**
GESTIÓN DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Marco y arquitectura de la gestión de sistemas	X.700–X.709
Servicio y protocolo de comunicación de gestión	X.710–X.719
Estructura de la información de gestión	X.720–X.729
Funciones de gestión y funciones de arquitectura de gestión distribuida abierta	X.730–X.799
SEGURIDAD	X.800–X.849
APLICACIONES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Compromiso, concurrencia y recuperación	X.850–X.859
Procesamiento de transacciones	X.860–X.879
Operaciones a distancia	X.880–X.899
PROCESAMIENTO DISTRIBUIDO ABIERTO	X.900–X.999
SEGURIDAD DE LAS TELECOMUNICACIONES	X.1000–

Para más información, véase la Lista de Recomendaciones del UIT-T.

NORMA INTERNACIONAL ISO/CEI 8825-3
RECOMENDACIÓN UIT-T X.692

Tecnología de la información – Reglas de codificación de notación de sintaxis abstracta uno: Especificación de la notación de control de codificación

Enmienda 1
Soporte de extensibilidad

Resumen

La presente enmienda añade la capacidad de utilizar la notación de control de codificación (ECN) necesaria para especificar la manera en que se codifican los tipos abiertos. También se mejora el mecanismo de "condiciones", que permite expresar condiciones más complejas.

Orígenes

La enmienda 1 a la Recomendación UIT-T X.692 (2002) fue aprobada el 29 de agosto de 2004 por la Comisión de Estudio 17 (2001-2004) del UIT-T por el procedimiento de la Recomendación UIT-T A.8. Se publica también un texto idéntico como Norma Internacional ISO/CEI 8825-3, Enmienda 1.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

La observancia de esta Recomendación es voluntaria. Ahora bien, la Recomendación puede contener ciertas disposiciones obligatorias (para asegurar, por ejemplo, la aplicabilidad o la interoperabilidad), por lo que la observancia se consigue con el cumplimiento exacto y puntual de todas las disposiciones obligatorias. La obligatoriedad de un elemento preceptivo o requisito se expresa mediante las frases "tener que, haber de, hay que + infinitivo" o el verbo principal en tiempo futuro simple de mandato, en modo afirmativo o negativo. El hecho de que se utilice esta formulación no entraña que la observancia se imponga a ninguna de las partes.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT no ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

ÍNDICE

NORMA INTERNACIONAL
RECOMENDACIÓN UIT-T

Tecnología de la información – Reglas de codificación de notación de sintaxis abstracta uno: Especificación de la notación de control de codificación

Enmienda 1
Soporte de extensibilidad

NOTA – Todo el texto nuevo o modificado en la presente enmienda está subrayado en las cláusulas pertinentes. En las nuevas cláusulas insertadas, sólo se resalta el encabezamiento subrayado. El texto eliminado figura tachado en esta enmienda. Al elaborar la Recomendación | Norma Internacional definitiva habrá de eliminarse el subrayado y el texto tachado.

1 Subcláusula 3.2.8

Sustitúyase la cláusula 3.2.8 por:

3.2.8 codificación condicional: Codificación que sólo se ha de aplicar si se cumple alguna condición ~~de límites o condición de gama de tamaños~~ especificada.

NOTA – La condición ha de ser una condición de límites o de gama de tamaños, u otro tipo de condición más compleja.

2 Nuevas subcláusulas 9.25 *bis* y 9.25 *ter*

Insértense las nuevas cláusulas 9.25 bis y 9.25 ter y actualícese el contenido:

9.25 *bis* Otras condiciones para la aplicación de codificaciones

9.25 *bis*.1 Hay una serie de condiciones, entre las que se cuentan el valor real y la gama de límites, que pueden probarse para seleccionar la codificación adecuada.

9.25 *bis*.2 También es posible exigir que se satisfagan todas las condiciones de una lista dada.

9.25 *bis*.3 Para la prueba de una condición se utiliza un único valor de enumeración (como "**`bounded-without-negatives`**") que contiene toda la prueba en la especificación de una o tres enumeraciones.

9.25 *bis*.4 Si se utiliza un trío de enumeraciones, la primera identifica el elemento que se prueba (por ejemplo "**`test-upper-bound`**"), la segunda, la naturaleza de la prueba (por ejemplo, "**`greater-than`**"), y la tercera el valor entero de la prueba.

9.25 *ter* Control de codificación para tipos abiertos

9.25 *ter*.1 Los tipos abiertos proporcionan con frecuencia un medio de extensibilidad, empleando un campo de identificación con nuevos valores para el campo identificación y nuevos tipos para el tipo abierto que se añaden en versiones sucesivas (y que generalmente se ponen a disposición en las extensiones específicas del vendedor).

9.25 *ter*.2 Estas dos características indican que puede pedirse al decodificador que decodifique un tipo abierto cuando esa implementación concreta no conoce el tipo que se ha codificado en ella.

9.25 *ter*.3 El soporte de codificación para los tipos abiertos es idéntico al de la mayoría de las otras clases en la categoría campo de bit, pero tiene la capacidad adicional de especificar que un conjunto de objetos de codificación distinto ha de aplicarse al tipo que ha de codificarse en el tipo abierto.

NOTA – La razón es que muchos protocolos prefieren utilizar un estilo distinto de codificación (generalmente basado en un enfoque tipo-longitud-valor) para el tipo contenido en un tipo abierto, manteniendo un estilo más compacto de codificación para los campos del mensaje que contiene el tipo abierto.

9.25 *ter*.4 El modelo utilizado para descodificar un tipo abierto reconoce que el decodificador no conocerá qué tipo satisface el tipo abierto (las tablas y constricciones de relación no son visibles para la PER o la ECN), aunque la aplicación puede determinar dicho tipo a partir de otro campo en el protocolo, o de un mensaje anterior o a partir de la dirección llamante (para las adiciones específicas del vendedor).

9.25 *ter*.5 El modelo es, por tanto, el siguiente: una vez tratado el relleno previo especificado y habiendo determinado el espacio de codificación y cualquier valor del relleno previo y posterior, el decodificador pedirá a la aplicación el tipo que se ha codificado. (En el caso de las herramientas, es bastante probable que la aplicación habrá preconfigurado la herramienta con una lista de tipos conocidos que puedan estar presentes y simplemente devolverá un puntero a uno de estos tipos.) La decodificación puede proseguir normalmente.

9.25 *ter*.6 No obstante, la aplicación puede devolver "unknown" (desconocido) (véase 9.25 *ter*.4) y el decodificador habrá de saber cómo determinar el final de la codificación desconocida. Esto se soluciona permitiendo que, en este caso, el especificador ECN proporcione una estructura de codificación y (opcionalmente) un conjunto de objetos de codificación para ella, que utilizarán los decodificadores para decodificar los tipos desconocidos de un tipo abierto. En la cláusula 23 se presenta la sintaxis para este caso.

NOTA – Un ejemplo de este tipo de estructura de codificación puede ser la que especifica la codificación generalmente conocida como "tipo, longitud, valor", cuyo final puede ser determinado sin conocer el tipo que se codifica.

3 Subcláusula 13.2.9

Sustitúyase la subcláusula 13.2.9 por lo siguiente:

13.2.9 En etapas posteriores de estos procedimientos, el punto de aplicación puede ser cualquiera de los elementos siguientes:

a) Un nombre de clase de codificación. Puede ser codificado por completo utilizando la especificación de un nombre de codificación de la misma clase (véase 17.1.7).

b) Un constructor de codificación (véase 16.2.12). Los procedimientos de construcción pueden ser determinados por la especificación contenida en un objeto de codificación de la clase constructor de codificación, pero ese objeto de codificación no determina la codificación de los componentes. La especificación del objeto de codificación que se aplica quizá requiera que uno o más componentes del constructor sean sustituidos por otras estructuras (parametrizadas) antes de que el punto de aplicación pase a los componentes.

c) Una clase en la categoría cadena de bits o cadena de octetos que tiene un tipo contenido como una propiedad asociada a los valores (véase 11.3.4.3 d). La codificación del tipo contenido depende de si está o no presente un **ENCODED BY** y de la especificación del objeto de codificación que se aplica (véase 22.11).

d) Una clase en la categoría tipo abierto. La codificación del componente del tipo abierto depende de si está o no presente **ENCODED WITH**, y de la especificación del objeto de codificación que se aplica (véase 23.9 *bis*.2).

~~d~~e) Un componente que es una clase de codificación (posiblemente precedida por una o más clases en la categoría rótulo), seguido por una clase de codificación en la categoría opcionalidad. Los procedimientos y las codificaciones para determinar la presencia o la ausencia vienen determinados por la especificación contenida en un objeto de codificación de la clase en la categoría opcionalidad. Este objeto de codificación puede requerir también la sustitución de la clase de codificación (junto con todas sus clases precedentes en la categoría rótulo) por una estructura de sustitución (parametrizada) antes de que la clase sea codificada. El punto de aplicación pasa entonces a la primera clase en la categoría rótulo (si la hay), o al componente, o a su reemplazante.

~~e~~f) Una clase de codificación precedida por una clase de codificación en la categoría rótulo. El número de rótulo asociado a la clase en la categoría rótulo se codifica utilizando la especificación de un objeto de codificación de la clase en la categoría rótulo, y el punto de aplicación pasa entonces a la clase rotulada.

~~f~~g) Cualquier otra clase de codificación incorporada. Puede ser codificada por completo utilizando la especificación contenida en un objeto de codificación de esa clase.

4 Subcláusula 13.2.10.5

Sustitúyase la nota en 13.2.10.5 por lo siguiente:

NOTA – Si el objeto de codificación que se aplica a una clase en la categoría tipo abierto contiene **ENCODED WITH**, esto determina el conjunto de objetos de codificación que se aplica al componente. En caso contrario, se aplica al componente el conjunto de objetos de codificación combinados que se aplica a esta clase (véase 23.9 *bis*.2).

5 Subcláusula 17.5.15

Sustitúyase 17.5.15 por lo siguiente:

17.5.15 Si se necesita una **REFERENCE** como parámetro real de cualquiera de los objetos de codificación o conjuntos de objetos de codificación utilizados en esta producción, puede ser suministrada como un parámetro de referencia del objeto de codificación que se define, o bien, como una "ComponentIdList" (en 15.3.1 puede encontrarse la sintaxis de "ComponentIdList", su significado en este contexto se especifica a continuación).

17.5.15 *bis* Si el gobernador no es un constructor en la categoría repetición, el primer (o único) "identifier" de la "ComponentIdList" será el "identifier" del "NamedType" presente textualmente (en algún nivel del anidado – véase 17.5.15 *ter*) de la construcción que se obtiene al desreferenciar el gobernador. Así se identifica toda la definición de dicho componente "NamedType", esté o no presente textualmente la definición.

17.5.15 *ter* Si hay más de un identificador coincidente de este tipo, el identificador coincidente elegido estará determinado por la primera correspondencia en un barrido (en orden textual) de los identificadores de nivel externo, seguido de un barrido (en orden textual) de los identificadores de segundo nivel, y por un barrido (en orden textual) de los identificadores de tercer nivel, etc.

17.5.15 *quat* Todos los "identifier" subsiguientes de la "ComponentIdList" (de haberla) serán "identifier" en un "NamedType" de la estructura identificada por la parte anterior de la "ComponentIdList", e identificarán toda la definición de dicho componente "NamedType", estén o no textualmente presenten en la definición de la estructura identificada por la parte anterior de la "ComponentIdList".

17.5.15 *quin* Si el gobernador es constructor en la categoría repetición, el parámetro real para **REFERENCE** será una "ComponentIdList" cuyo primer "identifier" identifica un componente textualmente presente en la "EncodingStructure" en la "RepetitionStructure" obtenido desreferenciando la repetición (véase 17.5.15 *ter*). Se aplicarán a continuación las subcláusulas 17.5.15 *ter* y 17.5.15 *quat*.

17.5.15 *sex* Si se requiere la **REFERENCE** para identificar un contenedor, puede ser suministrada también como:

a) **STRUCTURE** (siempre que el constructor de la estructura que se codifica no sea una categoría alternativa), cuando se refiere a esa estructura;

b) **OUTER**, cuando se refiere al contenedor de la codificación completa.

NOTA – La "EncodeStructure" es la única producción en la que se pueden suministrar las **REFERENCE**s, salvo si se utilizan parámetros ficticios o se utiliza **OUTER**, o cuando las referencias soportan **flag-to-be-used** o **flag-to-be-set** en la definición de un objeto de codificación de una clase en la categoría repetición que utiliza sustitución.

6 Subcláusula 18.2.6

Sustitúyase la nota de 18.2.6 por lo siguiente:

NOTA – El conjunto de objetos de codificación combinados que aplican estos objetos de codificación al tipo elegido para utilizarlo con la clase **#OPEN-TYPE** es siempre el mismo que el conjunto de objetos de codificación combinados aplicado a la clase **#OPEN-TYPE**, ya que estos objetos de codificación no contienen un **ENCODED WITH** (véanse 13.2.10.5 y 13.2.9 d).

7 Subcláusula 21.11.1

Sustitúyase 21.11.1 por lo siguiente:

21.11.1 El tipo "**RangeCondition**" es:

```
RangeCondition ::= ENUMERATED
              {unbounded-or-no-lower-bound,
               semi-bounded-with-negatives,
               bounded-with-negatives,
               semi-bounded-without-negatives,
               bounded-without-negatives,
               test-lower-bound,
               test-upper-bound,
               test-range}
```

8 Subcláusula 21.11.4

Sustitúyase 21.11.4 por lo siguiente:

21.11.4 El predicado se satisface para cada uno de los cinco primeros valores enumeración de 21.11.1 si, y solamente si, los límites impuestos a la clase codificación en la categoría entero cumplen las condiciones siguientes:

a) **unbounded-or-no-lower-bound**: o bien no hay límites o bien hay un sólo límite superior pero no hay límite inferior;

b) **semi-bounded-with-negatives**: hay un límite inferior que es menor que cero, pero no hay límite superior;

c) **bounded-with-negatives**: hay un límite inferior que es menor que cero y un límite superior;

d) **semi-bounded-without-negatives**: hay un límite inferior que es mayor o igual a cero pero no hay límite superior;

e) **bounded-without-negatives**: hay un límite inferior que es mayor o igual a cero y un límite superior.

NOTA – Para cualquier conjunto de límites dado, se satisfará exactamente un predicado.

9 Nueva subcláusula 21.11.5

Añádase la nueva subcláusula 21.11.5:

21.11.5 Si se utilizan los tres últimos valores enumeración de 21.11.1, se proporcionará un valor del tipo "**Comparison**" (véase 21.11 *bis*) junto con un valor **comparator** entero. Si se utilizan otros valores enumeración, no se proporcionarán estos valores.

10 Nueva subcláusula 21.11 *bis*

Añádase la nueva cláusula 21.11bis después de 21.11, e inclúyase en el índice:

21.11 *bis* Tipo Comparison

21.11 *bis*.1 El tipo "**Comparison**" es:

```
Comparison ::= ENUMERATED
               {equal-to,
                not-equal-to,
                greater-than,
                less-than,
                greater-than-or-equal-to,
                less-than-or-equal-to}
```

21.11 *bis*.2 No hay valor por defecto para la propiedad de codificación de este tipo.

21.11 *bis*.3 Se utiliza la propiedad de codificación del tipo "**Comparison**" para probar la propiedad identificada de una clase con respecto al valor entero (el **comparator**).

21.11 *bis*.4 El predicado utilizando una "**Comparison**" se satisface para cada valor enumeración si, y solamente si, la propiedad identificada satisface las siguientes condiciones:

a) **equal-to**: su valor es igual al del valor **comparator** entero especificado;

b) **not-equal-to**: su valor es distinto al del valor **comparator** entero especificado;

c) **greater-than**: su valor es superior al del valor **comparator** entero especificado;

d) **less-than**: su valor es inferior al del valor **comparator** entero especificado;

e) **greater-than-or-equal-to**: su valor es superior o igual al del valor **comparator** entero especificado;

f) **less-than-or-equal-to**: su valor es inferior o igual al del valor **comparator** entero especificado.

11 Subcláusula 21.12.1

Sustitúyase 21.12.1 por lo siguiente:

21.12.1 El tipo "**SizeRangeCondition**" es:

```
SizeRangeCondition ::= ENUMERATED
              {no-ub-with-zero-lb,
               ub-with-zero-lb,
               no-ub-with-non-zero-lb,
               ub-with-non-zero-lb,
               fixed-size,
               test-lower-bound,
               test-upper-bound,
               test-range}
```

12 Subcláusula 21.12.4

Sustitúyase 21.12.4 por lo siguiente:

21.12.4 El predicado se satisface para cada uno de los cinco primeros valores enumeración de 21.12.1 si, y solamente si, la constricción de tamaño efectivo cumple las condiciones siguientes:

a) **no-ub-with-zero-lb**: no hay un límite superior impuesto al tamaño y el límite inferior es cero;

b) **ub-with-zero-lb**: hay un límite superior impuesto al tamaño y el límite inferior es cero;

c) **no-ub-with-non-zero-lb**: no hay un límite superior impuesto al tamaño y el límite inferior es distinto de cero;

d) **ub-with-non-zero-lb**: hay un límite superior impuesto al tamaño y el límite inferior es distinto de cero;

f) **fixed-size**: el límite superior y el límite inferior impuestos al tamaño tienen el mismo valor.

NOTA – Sólo el caso "**fixed-size**" se solapa con otros predicados.

13 Nueva subcláusula 21.12.5

Añádase la nueva cláusula 21.12.5 después de 21.12.4:

21.12.5 Si se utilizan los tres últimos valores enumeración de 21.12.1, se proporcionará un valor de tipo "**Comparison**" (véase 21.11 *bis*) junto con un valor **comparator** entero. Si se utilizan los otros valores enumeración, no se proporcionarán dichos valores.

14 Nueva subcláusula 21.16

Añádase una nueva cláusula 21.16:

21.16 Tipo IntegerMapping

21.16.1 El tipo "**IntegerMapping**" es:

```
IntegerMapping ::= SET OF SEQUENCE {
               source  SET OF INTEGER,
               result INTEGER} (CONSTRAINED BY {/* the intersection of the source
                                                   components shall be empty */})
```

21.16.2 El tipo "**IntegerMapping**" se utiliza para especificar explícitamente una transformada ints-to ints (entero a entero).

15 Subcláusula 23.2.3.8

Sustitúyase 23.2.3.8 por lo siguiente:

23.2.3.8 Si un objeto de codificación de la lista ordenada "**REPETITION-ENCODINGS**" se define utilizando "**IF**" o "**IF-ALL**", todos los objetos de codificación precedentes de esa lista se definirán utilizando "**IF**" o "**IF-ALL**".

16 Subcláusula 23.4.3.8

Sustitúyase 23.4.3.8 por lo siguiente:

23.4.3.8 Si un objeto de codificación de la lista ordenada "**REPETITION-ENCODINGS**" se define utilizando "**IF**" o "**IF-ALL**", todos los objetos de codificación precedentes de esa lista se definirán utilizando "**IF**" o "**IF-ALL**".

17 Subcláusula 23.6.2.3

Sustitúyase 23.6.2.3 por lo siguiente:

23.6.2.3 Si un objeto de codificación de la lista ordenada "**ENCODINGS**" se define utilizando "**IF**" o "**IF-ALL**", todos los objetos de codificación precedentes de esta lista se definirán utilizando "**IF**" o "**IF-ALL**".

18 Subcláusula 23.7.1

Sustitúyase 23.7.1 por lo siguiente:

23.7.1 Sintaxis definida

La sintaxis de la definición de objetos de codificación de la clase **#CONDITIONAL-INT** se define como sigue:

```
#CONDITIONAL-INT ::= ENCODING-CLASS {

        -- Condition (see 21.11)
        &range-condition                RangeCondition  OPTIONAL,
        &comparison                     Comparison OPTIONAL,
        &comparator                     INTEGER OPTIONAL,
        &Range-conditions               RangeCondition ORDERED OPTIONAL,
        &Comparisons                    Comparison ORDERED OPTIONAL,
        &Comparators                    INTEGER ORDERED OPTIONAL,

        -- Structure-only replacement specification (see 22.1)
        &#Replacement-structure         OPTIONAL,
        &replacement-structure-encoding-object  &#Replacement-structure OPTIONAL,

        -- Pre-alignment and padding specification (see 22.2)
        &encoding-space-pre-alignment-unit Unit (ALL EXCEPT repetitions)
                                        DEFAULT bit,
        &encoding-space-pre-padding     Padding DEFAULT zero,
        &encoding-space-pre-pattern     Non-Null-Pattern (ALL EXCEPT
                                               different:any) DEFAULT bits:'0'B,

        -- Start pointer specification (see 22.3)
        &start-pointer                  REFERENCE    OPTIONAL,
        &start-pointer-unit             Unit (ALL EXCEPT repetitions)
                                        DEFAULT bit,
        &Start-pointer-encoder-transforms #TRANSFORM ORDERED OPTIONAL,

        -- Encoding space specification (see 22.4)
        &encoding-space-size            EncodingSpaceSize
                                        DEFAULT self-delimiting-values,
        &encoding-space-unit            Unit (ALL EXCEPT repetitions)
                                        DEFAULT bit,
        &encoding-space-determination   EncodingSpaceDetermination
                                        DEFAULT field-to-be-set,
        &encoding-space-reference       REFERENCE OPTIONAL,
        &Encoder-transforms             #TRANSFORM ORDERED OPTIONAL,
        &Decoder-transforms             #TRANSFORM ORDERED OPTIONAL,

        -- Value encoding
        &Transform                      #TRANSFORM ORDERED OPTIONAL,
        &encoding                       ENUMERATED
                                        {positive-int, twos-complement,
                                         reverse-positive-int,
                                         reverse-twos-complement}
                                        DEFAULT twos-complement,
```

```
-- Value padding and justification (see 22.8)
&value-justification              Justification DEFAULT right:0,
&value-pre-padding                Padding DEFAULT zero,
&value-pre-pattern                Non-Null-Pattern DEFAULT bits:'0'B,
&value-post-padding               Padding DEFAULT zero,
&value-post-pattern               Non-Null-Pattern DEFAULT bits:'0'B,
&unused-bits-determination        UnusedBitsDetermination
                                  DEFAULT field-to-be-set,
&unused-bits-reference            REFERENCE OPTIONAL,
&Unused-bits-encoder-transforms   #TRANSFORM ORDERED OPTIONAL,
&Unused-bits-decoder-transforms   #TRANSFORM ORDERED OPTIONAL,

-- Identification handle specification (see 22.9)
&exhibited-handle                 PrintableString OPTIONAL,
&Handle-positions                 INTEGER (0..MAX) OPTIONAL,
&Handle-value                     HandleValue DEFAULT tag:any,

-- Bit reversal specification (see 22.12)
&bit-reversal                     ReversalSpecification
                                  DEFAULT no-reversal
}
WITH SYNTAX {
[IF &range-condition [&comparison  &comparator]]
[IF-ALL &Range-conditions [&Comparisons &Comparators]]
[ELSE]
[REPLACE
      [STRUCTURE]
      WITH &#Replacement-structure
            [ENCODED BY &replacement-structure-encoding-object]]
[ALIGNED TO
      [NEXT]
      [ANY]
            &encoding-space-pre-alignment-unit
            [PADDING &encoding-space-pre-padding
            [PATTERN &encoding-space-pre-pattern]]]
[START-POINTER   &start-pointer
      [MULTIPLE OF     &start-pointer-unit]
      [ENCODER-TRANSFORMS    &Start-pointer-encoder-transforms]]
ENCODING-SPACE
      [SIZE &encoding-space-size
            [MULTIPLE OF &encoding-space-unit]]
      [DETERMINED BY &encoding-space-determination]
      [USING &encoding-space-reference
            [ENCODER-TRANSFORMS &Encoder-transforms]
            [DECODER-TRANSFORMS &Decoder-transforms]]
[TRANSFORMS &Transforms]
[ENCODING &encoding]
[VALUE-PADDING
      [JUSTIFIED &value-justification]
      [PRE-PADDING &value-pre-padding
            [PATTERN &value-pre-pattern]]
      [POST-PADDING &value-post-padding
            [PATTERN &value-post-pattern]]
      [UNUSED BITS
            [DETERMINED BY &unused-bits-determination]
            [USING &unused-bits-reference
                  [ENCODER-TRANSFORMS &Unused-bits-encoder-transforms]
                  [DECODER-TRANSFORMS &Unused-bits-decoder-transforms]]]]
[EXHIBITS HANDLE &exhibited-handle AT &Handle-positions
      [AS &handle-value]]
[BIT-REVERSAL &bit-reversal]
}
```

19 Subcláusula 23.7.2.2

Sustitúyase 23.7.2.2 por lo siguiente:

23.7.2.2 La sintaxis permite la especificación de una sola condición impuesta a los límites del entero para que se aplique esta codificación (utilización de "**IF**"). Permite además la especificación de todo un conjunto de condiciones

que han de satisfacerse (utilización de "**IF-ALL**"). También permite la especificación de la no existencia de condición alguna. La utilización de "**ELSE**", o la omisión de "**IF**", "**IF-ALL**" y "**ELSE**" especifica que no hay ninguna condición. "**IF-ALL**" se utilizará con tres listas, si una o más de las condiciones de gama de tamaño requiere una comparación, y se utilizará con una lista en cualquier otro caso. Cuando se utilicen tres listas, las condiciones de gama de tamaño que no requieran una comparación o un comparador (de haberlo) irán detrás de las que sí lo requieren, y no habrá anotaciones correspondientes en la segunda y tercera listas. Cuando se utilice "**IF-ALL**" con tres listas, éstas se interpretarán como una lista de predicados utilizando los valores en las posiciones correspondientes en las tres listas.

NOTA – Se recomienda que las tres listas se formateen para presentar una condición en cada columna.

Ejemplo:

```
IF-ALL {test-lower-bound, test-range      , bounded-with-negatives }
              {greater-than    , less-than-or-equal-to }
              {-10             , 20                    }
```

20 Subcláusula 23.7.2.4

Sustitúyase 23.7.2.4 por lo siguiente:

23.7.2.4 De las condiciones "**IF**", "**IF-ALL**" y "**ELSE**", como máximo estará presente una.

21 Subcláusulas 23.7.2.6, 23.7.2.7 y 23.7.2.8

Sustitúyanse 23.7.2.6, 23.7.2.7 y 23.7.2.8 por lo siguiente:

23.7.2.6 Si cualquier transformada de las "**TRANSFORMS**" no es reversible para el valor abstracto al que se aplica, se trata de un error de la especificación ECN o de la aplicación. La primera transformada de las "**TRANSFORMS**", si las hay, tendrá un origen que será un entero y la última transformada tendrá un resultado que será un entero.

NOTA – La prueba de las condiciones "**IF**" e "**IF-ALL**" tiene lugar en los límites del valor original, y no se ve afectada por estas transformadas.

23.7.2.7 La transformada "**INT-TO-INT**" con el valor "**subtract:lower-bound**" se incluirá si, y solamente si, la condición "**IF**" o "**IF-ALL**" restringe la aplicación de esta codificación a clases en la categoría entero con un límite más bajo, y (si está presente) será la primera transformada de la lista.

23.7.2.8 "**ENCODING-SPACE SIZE**" no se fijará en "**fixed-to-max**" a menos que la condición "**IF**" o "**IF-ALL**" restrinja la codificación a una clase con un límite más alto y un límite más bajo.

22 Subcláusula 23.9.3.8

Sustitúyase 23.9.3.8 por lo siguiente:

23.9.3.8 Si un objeto de codificación de la lista ordenada "**REPETITION-ENCODINGS**" se define utilizando "**IF**" o "**IF-ALL**", todos los objetos de codificación precedentes de esa lista se definirán utilizando "**IF**" o "**IF-ALL**".

23 Nueva subcláusula 23.9 *bis*

Insértese la nueva subcláusula 23.9 bis detrás de la 23.9 y actualícense los contenidos:

23.9 *bis* Definición de objetos de codificación de clases en la categoría tipo abierto

23.9 *bis*.1 Sintaxis definida

La sintaxis de la definición de objetos de codificación de clases en la categoría tipo abierto se define como sigue:

```
#OPEN-TYPE ::= ENCODING-CLASS {

              -- Structure-only replacement specification (see 22.1)
              &#Replacement-structure           OPTIONAL,
              &replacement-structure-encoding-object
                                                &#Replacement-structure OPTIONAL,

              -- Pre-alignment and padding specification (see 22.2)
```

```
&encoding-space-pre-alignment-unit
                                  Unit (ALL EXCEPT repetitions)
                                  DEFAULT bit,
&encoding-space-pre-padding       Padding DEFAULT zero,
&encoding-space-pre-pattern       Non-Null-Pattern (ALL EXCEPT different:any)
                                  DEFAULT bits:'0'B,

-- Start pointer specification (see 22.3)
&start-pointer                    REFERENCE OPTIONAL,
&start-pointer-unit               Unit (ALL EXCEPT repetitions) DEFAULT bit,
&Start-pointer-encoder-transforms #TRANSFORM ORDERED OPTIONAL,

-- Encoding space specification (see 22.4)
&encoding-space-size              EncodingSpaceSize
                                  DEFAULT self-delimiting-values,
&encoding-space-unit              Unit (ALL EXCEPT repetitions)
                                  DEFAULT bit,
&encoding-space-determination     EncodingSpaceDetermination
                                  DEFAULT field-to-be-set,
&encoding-space-reference         REFERENCE OPTIONAL,
&Encoder-transforms               #TRANSFORM ORDERED OPTIONAL,
&Decoder-transforms               #TRANSFORM ORDERED OPTIONAL,

-- Open-type encoding
&Known-structure-encodings        #ENCODINGS OPTIONAL,
&Unknown-structure                OPTIONAL,
&Unknown-structure-encodings      #ENCODINGS OPTIONAL,

-- Value padding and justification (see 22.8)
&value-justification              Justification DEFAULT right:0,
&value-pre-padding                Padding DEFAULT zero,
&value-pre-pattern                Non-Null-Pattern DEFAULT bits:'0'B,
&value-post-padding               Padding DEFAULT zero,
&value-post-pattern               Non-Null-Pattern DEFAULT bits:'0'B,
&unused-bits-determination        UnusedBitsDetermination
                                  DEFAULT field-to-be-set,
&unused-bits-reference            REFERENCE OPTIONAL,
&Unused-bits-encoder-transforms   #TRANSFORM ORDERED OPTIONAL,
&Unused-bits-decoder-transforms   #TRANSFORM ORDERED OPTIONAL,

-- Bit reversal specification (see 22.12)
&bit-reversal                     ReversalSpecification
                                  DEFAULT no-reversal
}
WITH SYNTAX {
[REPLACE
      [STRUCTURE]
      WITH &#Replacement-structure
            [ENCODED BY &replacement-structure-encoding-object]]
[ALIGNED TO
      [NEXT]
      [ANY]
            &encoding-space-pre-alignment-unit
            [PADDING &encoding-space-pre-padding
            [PATTERN &encoding-space-pre-pattern]]]
[START-POINTER &start-pointer
      [MULTIPLE OF     &start-pointer-unit]
      [ENCODER-TRANSFORMS    &Start-pointer-encoder-transforms]]
ENCODING-SPACE
      [SIZE &encoding-space-size
            [MULTIPLE OF &encoding-space-unit]]
      [DETERMINED BY &encoding-space-determination]
      [USING &encoding-space-reference
            [ENCODER-TRANSFORMS &Encoder-transforms]
            [DECODER-TRANSFORMS &Decoder-transforms]]
[ENCODED WITH &Known-structure-encodings]
[UNKNOWN IS &Unknown-structure
      [ENCODED WITH &Unknown-structure-encodings]]
[VALUE-PADDING
      [JUSTIFIED &value-justification]
      [PRE-PADDING &value-pre-padding
```

```
                [PATTERN &value-pre-pattern]]
          [POST-PADDING &value-post-padding
                [PATTERN &value-post-pattern]]
          [UNUSED BITS
                [DETERMINED BY &unused-bits-determination]
                [USING &unused-bits-reference
                      [ENCODER-TRANSFORMS &Unused-bits-encoder-transforms]
                      [DECODER-TRANSFORMS &Unused-bits-decoder-transforms]]]
    [EXHIBITS HANDLE &exhibited-handle AT &Handle-positions
          [AS &handle-value]]
    [BIT-REVERSAL &bit-reversal]
    }
```

23.9 *bis*.2 Modelo para la codificación de clases en la categoría tipo abierto

23.9 *bis*.2.1 El modelo de codificación de tipo abierto es el siguiente:

a) La clase en la categoría tipo abierto puede sustituirse por otra estructura para delimitar la longitud, de ser necesario.

b) El objeto de codificación definido para esta categoría aplica el conjunto de objetos de codificación "**ENCODED WITH**" al tipo cuyo valor va a codificarse para el tipo abierto. Si no existe "**ENCODED WITH**", se utiliza el conjunto de objetos de codificación combinados actual.

c) El decodificador pedirá a la aplicación que identifique el tipo codificado en el tipo abierto. La aplicación devolverá la identificación del tipo, que se decodifica a continuación, o indicará que el tipo codificado en el tipo abierto no puede determinarse (respuesta "unknown") (desconocido).

d) Si la respuesta es " unknown " y está presente "**UNKNOWN IS**", el decodificador utilizará la estructura "**UNKNOWN IS**" y el "**ENCODED WITH**" dentro del "**UNKNOWN IS**" (de haberlo) para determinar el final del espacio de codificación.

e) Si la respuesta es " unknown " y no hay "**UNKNOWN IS**", el tamaño del espacio de codificación puede determinarse mediante "**ENCODING-SPACE**" (véase 23.9 *bis.3.3*), y el decodificador devolverá a la aplicación todos los bits contenidos en el espacio de codificación definido excepto el valor previo y posterior al relleno.

23.9 *bis*.2.2 En el caso de una decodificación desconocida, el decodificador remitirá a la aplicación los bits que forman la codificación desconocida como valor de tipo abierto.

23.9 *bis*.3 Objetivo y restricciones

23.9 *bis*.3.1 Esta sintaxis se utiliza para definir cómo se codifica un tipo abierto, así como los medios que utiliza el descodificador para determinar el final de la codificación de un tipo desconocido en un tipo abierto.

23.9 *bis*.3.2 Si "**REPLACE STRUCTURE**" está configurado, no se configurarán otros parámetros.

23.9 *bis*.3.3 Si "**ENCODING-SPACE SIZE**" es "**self-delimiting**", se configurará "**UNKNOWN IS**".

23.9 *bis*.4 Acciones del codificador

23.9 *bis*.4.1 Para cualquier grupo de propiedades de codificación que se fije, el codificador efectuará las acciones de codificador especificadas en la cláusula 22, en el orden siguiente y de acuerdo con la definición del objeto de codificación:

a) sustitución;

b) prealineación y relleno;

c) puntero de comienzo;

d) espacio de codificación (véase 23.9 *bis*.4.3);

e) codificación de tipo abierto (véase 23.9 *bis*.4.2);

f) relleno y justificación de valor (véase 23.9 *bis*.4.5);

g) inversión de bits.

23.9 *bis*.4.2 El codificador codificará el valor del tipo proporcionado por la aplicación utilizando el conjunto de objetos de codificación "**ENCODED WITH**", de estar presente, o utilizará, en cualquier otro caso, el conjunto de objetos de codificación combinados actual.

23.9 *bis*.4.3 Si "**ENCODING-SPACE SIZE**" es "**variable-with-determinant**" o "**encoder-option-with-determinant**", será el número mínimo de unidades de "**MULTIPLE OF**" necesarias para contener el esquema ("s", say), de acuerdo con 23.9 *bis*.4.5.

23.9 *bis*.4.4 El codificador (como opción de codificador) puede incrementar "s" (como se indica en 23.9 *bis*.4.3) en unidades de "**MULTIPLE OF**" (sujeto a las restricciones que suponen la gama de valores de cualquier "**added-field**" o "**asn1-field**") si "**ENCODING-SPACE SIZE**"se pone a "**encoder-option-with-determinant**".

23.9 *bis*.4.5 Si el número de bits no utilizados es distinto de cero, se aplicará "**VALUE-JUSTIFICATION**" utilizando cualquiera de los valores fijados o de los valores por defecto.

23.9 *bis*.5 Acciones del decodificador

23.9 *bis*.5.1 Para cualquier grupo de propiedades de codificación que se fije, el descodificador efectuará las acciones del decodificador especificadas en la cláusula 22, en el orden siguiente y de acuerdo con la definición del objeto de codificación:

a) prealineación y relleno;

b) puntero de comienzo;

c) espacio de codificación;

d) inversión de bits;

e) relleno y justificación de valor;

f) descodificación de tipo abierto (véase 23.9 *bis*.5.2).

23.9 *bis*.5.2 Para la decodificación de tipo abierto, el decodificador pedirá a la aplicación el tipo que se ha codificado y decodificará el valor de dicho tipo o de la estructura "**UNKNOWN IS**", de acuerdo con las especificaciones de "**ENCODED WITH**" en "**UNKNOWN IS**".

23.9 *bis*.5.3 Si la decodificación es de tipo desconocido, los bits que forman la codificación desconocida (sin bits de relleno previo y sin valor de bits de relleno previo y posterior, de haberlos) se transferirán a la aplicación como valor del tipo abierto.

24 Subcláusula 23.12.2.3

Sustitúyase 23.12.2.3 por lo siguiente:

23.12.2.3 Si un objeto de codificación de la lista ordenada "**REPETITION-ENCODING**" se define utilizando "**IF**" o "**IF-ALL**", todos los objetos de codificación precedentes de esa lista se definirán utilizando "**IF**" o "**IF-ALL**".

25 Subcláusula 23.13.1

Sustitúyase 23.13.1 por lo siguiente:

23.13.1 Sintaxis definida

La sintaxis de la definición de objetos de codificación de la clase **#CONDITIONAL-REPETITION** se define como sigue:

```
#CONDITIONAL-REPETITION ::= ENCODING-CLASS {

        -- Condition (see 21.12)
        &size-range-condition            SizeRangeCondition  OPTIONAL,
        &comparison                      Comparison OPTIONAL,
        &comparator                      INTEGER OPTIONAL,
        &Size-range-conditions           SizeRangeCondition ORDERED OPTIONAL,
        &Comparisons                     Comparison ORDERED OPTIONAL,
        &Comparators                     INTEGER ORDERED OPTIONAL,

        -- Structure or component replacement specification (see 22.1)
        &#Replacement-structure          OPTIONAL,
        &replacement-structure-encoding-object  &#Replacement-structure OPTIONAL,
        &#Head-end-structure             OPTIONAL,

        -- Pre-alignment and padding specification (see 22.2)
        &encoding-space-pre-alignment-unitUnit (ALL EXCEPT repetitions) DEFAULT bit,
        &encoding-space-pre-padding      Padding DEFAULT zero,
```

```
&encoding-space-pre-pattern       Non-Null-Pattern (ALL EXCEPT different:any)
                                  DEFAULT bits:'0'B,

-- Start pointer specification (see 22.3)
&start-pointer                    REFERENCE     OPTIONAL,
&start-pointer-unit               Unit (ALL EXCEPT repetitions) DEFAULT bit,
&Start-pointer-encoder-transforms #TRANSFORM ORDERED OPTIONAL,

-- Repetition space specification (see 22.7)
&repetition-space-size            EncodingSpaceSize
                                  DEFAULT self-delimiting-values,
&repetition-space-unit            Unit DEFAULT bit,
&repetition-space-determination   RepetitionSpaceDetermination
                                  DEFAULT field-to-be-set,
&main-reference                   REFERENCE OPTIONAL,
&Encoder-transforms               #TRANSFORM ORDERED OPTIONAL,
&Decoder-transforms               #TRANSFORM ORDERED OPTIONAL,
&handle-id                        PrintableString
                                  DEFAULT "default-handle",
&termination-pattern              Non-Null-Pattern (ALL EXCEPT
                                  different:any) DEFAULT '0'B,

-- Repetition alignment
&repetition-alignment             ENUMERATED {none, aligned}
                                  DEFAULT none,

-- Value padding and justification (see 22.8)
&value-justification              Justification DEFAULT right:0,
&value-pre-padding                Padding DEFAULT zero,
&value-pre-pattern                Non-Null-Pattern DEFAULT bits:'0'B,
&value-post-padding               Padding DEFAULT zero,
&value-post-pattern               Non-Null-Pattern DEFAULT bits:'0'B,
&unused-bits-determination        UnusedBitsDetermination
                                  DEFAULT field-to-be-set,
&unused-bits-reference            REFERENCE OPTIONAL,
&Unused-bits-encoder-transforms   #TRANSFORM ORDERED OPTIONAL,
&unused-bits-decoder-transforms   #TRANSFORM ORDERED OPTIONAL,

-- Identification handle specification (see 22.9)
&exhibited-handle                 PrintableString OPTIONAL,
&Handle-positions                 INTEGER (0..MAX) OPTIONAL,
&Handle-value                     HandleValue DEFAULT tag: any,

-- Bit reversal specification (see 22.12)
&bit-reversal                     ReversalSpecification
                                  DEFAULT no-reversal
}
WITH SYNTAX {
[IF &size-range-condition [&comparison  &comparator]]
[IF-ALL &Size-range-conditions [&Comparisons &Comparators]]
[ELSE]
[REPLACE
      [STRUCTURE]
      [COMPONENT]
      [ALL COMPONENTS]
      WITH &Replacement-structure
      [ENCODED BY &replacement-structure-encoding-object
            [INSERT AT HEAD &#Head-end-structure]]]
[ALIGNED TO
      [NEXT]
      [ANY]
            &encoding-space-pre-alignment-unit
            [PADDING &encoding-space-pre-padding
            [PATTERN &encoding-space-pre-pattern]]]
[START-POINTER   &start-pointer
      [MULTIPLE OF     &start-pointer-unit]
      [ENCODER-TRANSFORMS    &Start-pointer-encoder-transforms]]
REPETITION-SPACE
      [SIZE &repetition-space-size
            [MULTIPLE OF &repetition-space-unit]]
      [DETERMINED BY &repetition-space-determination
```

```
            [HANDLE &handle-id]]
      [USING &main-reference
            [ENCODER-TRANSFORMS &Encoder-transforms]
            [DECODER-TRANSFORMS &Decoder-transforms]]
      [PATTERN &termination-pattern]
[ALIGNMENT &repetition-alignment]
[VALUE-PADDING
      [JUSTIFIED &value-justification]
      [PRE-PADDING &value-pre-padding
            [PATTERN &value-pre-pattern]]
      [POST-PADDING &value-post-padding
            [PATTERN &value-post-pattern]]
      [UNUSED BITS
            [DETERMINED BY &unused-bits-determination]
            [USING &unused-bits-reference
                  [ENCODER-TRANSFORMS &Unused-bits-encoder-transforms]
                  [DECODER-TRANSFORMS &Unused-bits-decoder-transforms]]]]
[EXHIBITS HANDLE &exhibited-handle AT &Handle-positions
[AS &handle-value]]
[BIT-REVERSAL &bit-reversal]
}
```

26 Subcláusulas 23.13.2.1 y 23.13.2.2

Sustitúyanse 23.13.2.1 y 23.13.2.2 por lo siguiente:

23.13.2.1 Esta sintaxis se utiliza para definir la codificación de una clase en la categoría repetición a reserva del cumplimiento de una condición basada en los límites de la repetición (utilización de "**IF**"). Permite además la especificación del cumplimiento de todo un conjunto de condiciones (utilización de "**IF-ALL**"). También permite la especificación de la no existencia de condición alguna. La utilización de "**ELSE**", o la omisión de "**IF**", "**IF-ALL**" y "**ELSE**" especifica que no hay ninguna condición. "**IF-ALL**" se utilizará con tres listas, si una o más de las condiciones de gama de tamaño requiere una comparación, y se utilizará con una lista en cualquier otro caso. Cuando se utilicen tres listas, las condiciones de gama de tamaño que no requieran una comparación o un comparador (de haberlo) irán detrás de las que sí la requieren, y no habrá anotaciones correspondientes en la segunda y tercera lista. Al utilizar "**IF-ALL**" con tres listas, éstas se interpretarán como una lista de predicados utilizando los valores de las posiciones correspondientes en las tres listas.

NOTA – Se recomienda que las tres listas se formateen para presentar una condición en cada columna (véase el ejemplo en 23.7.2.2).

23.13.2.2 De las condiciones "**IF**", "**IF-ALL**" y "**ELSE**", como máximo estará presente una.

27 Subcláusula 23.15

Sustitúyase 23.15 por lo siguiente:

23.15 Definición de objetos de codificación para clases en las demás categorías

En esta versión de la presente Recomendación | Norma Internacional no hay sintaxis definida para clases en las categorías siguientes:

```
objectidentifier
open-type
real
```

28 Subcláusula 24.3.1

Sustitúyase 24.3.1 por lo siguiente:

24.3.1 La transformada int-to-int (entero a entero) utiliza la propiedad de codificación siguiente:

```
&int-to-int             CHOICE
                        {increment        INTEGER (1..MAX),
                         decrement        INTEGER (1..MAX),
                         multiply         INTEGER (2..MAX),
                         divide           INTEGER (2..MAX),
```

```
                        negate          ENUMERATED{value},
                        modulo          INTEGER (2..MAX),
                        subtract        ENUMERATED{lower-bound},
                        mapping         IntegerMapping
                      } OPTIONAL
```

29 Nueva subcláusula 24.3.2 *bis*

Añádase la nueva subcláusula 24.3.2 bis después de 24.3.2:

24.3.2 *bis* La definición del tipo utilizado en la transformada int-to-int es:

```
IntegerMapping ::= SET OF SEQUENCE {
               source    SET OF INTEGER,
               result    INTEGER} (CONSTRAINED BY {/* the intersection of the source
                                                     components shall be empty
                                                     (see 21.16) */})
```

30 Nueva subcláusula 24.3.8 *bis*

Añádase la nueva subcláusula 24.3.8 bis después de 24.3.8:

24.3.8 *bis* La transformada del valor "**mapping:integerMapping**" se define de la siguiente manera. El valor entero original se sustituye por el valor asociado al conjunto de valores al que pertenece. Si la intersección de los conjuntos de valores no está vacía, se trata de un error de la especificación ECN. Si el entero original no pertenece a uno de los conjuntos de valore, se trata de un error de la aplicación.

31 Cuadro 6

Añádase la siguiente fila al final del cuadro 6:

mapping:integerMapping	*Conjuntos de valores de origen, que contiene cada uno de ellos un sólo valor, y los valores resultantes son distintos.*

32 Subcláusula C.1

Sustitúyase la producción "Governor" en C.1 por lo siguiente:

Governor ::=
- **EncodingClassFieldType**
- | **REFERENCE**
- | **DefinedOrBuiltinEncodingClass**
- | **#ENCODINGS**
- | **Type**

33 Subcláusula C.4

Sustitúyase C.4 por lo siguiente:

C.4 Lista de parámetros reales

La cláusula 9.5 de la Rec. UIT-T X.683 | ISO/CEI 8824-4 se modifica como sigue:

9.5 La "ActualParameterList" es:

ActualParameterList ::=
"{<" ActualParameter "," + ">}"

ActualParameter ::=
- **Value**
- | **ValueSet**
- | **OrderedValueList**

| **DefinedOrBuiltinEncodingClass**
| **EncodingObject**
| **EncodingObjectSet**
| **OrderedEncodingObjectList**
| ~~**identifier**~~
| **ComponentIdList**
| **STRUCTURE**
| **OUTER**

Si el parámetro ficticio correspondiente es:

a) un valor: se seleccionará la alternativa "Value";

b) un conjunto de valores: se seleccionará la alternativa "ValueSet";

c) una lista de valores ordenada de tipo fijo: se seleccionará la alternativa "OrderedValueList";

d) una clase de codificación: se seleccionará la alternativa "DefinedOrBuiltinEncodingClass";

e) un objeto de codificación: se seleccionará la alternativa "EncodingObject";

f) un conjunto de objetos de codificación: se seleccionará la alternativa "EncodingObjectSet";

g) una lista ordenada de objetos de codificación: se seleccionará la alternativa "OrderedEncodingObjectList";

h) una referencia: se seleccionará la alternativa "ComponentIdList", STRUCTURE o OUTER.

STRUCTURE sólo se seleccionará cuando el parámetro real se utilice como se especifica en 17.5.15. **OUTER** puede utilizarse siempre que se requiera una referencia para identificar un contenedor, e identifica el contenedor de la codificación completa.

34 Subcláusula G.2.4

Sustitúyase la producción "Governor" en G.2.4 por lo siguiente:

```
Governor ::=
        EncodingClassFieldType
    |   REFERENCE
    |   DefinedOrBuiltinEncodingClass
    |   #ENCODINGS
    |   Type
```

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedios
Serie I	Red digital de servicios integrados
Serie J	Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedios
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	Gestión de las telecomunicaciones, incluida la RGT y el mantenimiento de redes
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	**Redes de datos, comunicaciones de sistemas abiertos y seguridad**
Serie Y	Infraestructura mundial de la información, aspectos del protocolo Internet y Redes de la próxima generación
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

27072

Impreso en Suiza

Ginebra, 2005