UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T X.680

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

(07/94)

REDES DE DATOS Y COMUNICACIÓN ENTRE SISTEMAS ABIERTOS

GESTIÓN DE REDES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS Y ASPECTOS DE SISTEMAS – NOTACIÓN DE SINTAXIS ABSTRACTA UNO

TECNOLOGÍA DE LA INFORMACIÓN – NOTACIÓN DE SINTAXIS ABSTRACTA UNO: ESPECIFICACIÓN DE LA NOTACIÓN BÁSICA

Recomendación UIT-T X.680

Reemplazada por una versión más reciente

(Anteriormente «Recomendación del CCITT»)

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. En el UIT-T, que es la entidad que establece normas mundiales (Recomendaciones) sobre las telecomunicaciones, participan unos 179 países miembros, 84 empresas de explotación de telecomunicaciones, 145 organizaciones científicas e industriales y 38 organizaciones internacionales.

Las Recomendaciones las aprueban los Miembros del UIT-T de acuerdo con el procedimiento establecido en la Resolución N.° 1 de la CMNT (Helsinki, 1993). Adicionalmente, la Conferencia Mundial de Normalización de las Telecomunicaciones (CMNT), que se celebra cada cuatro años, aprueba las Recomendaciones que para ello se le sometan y establece el programa de estudios para el periodo siguiente.

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. El texto de la Recomendación UIT-T X.680 se aprobó el 1 de julio de 1994. Su texto se publica también, en forma idéntica, como Norma Internacional ISO/CEI 8824-1.

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.

Reemplazada por una versión más reciente

RECOMENDACIONES UIT-T DE LA SERIE X

REDES DE DATOS Y COMUNICACIÓN ENTRE SISTEMAS ABIERTOS

(Febrero de 1994)

ORGANIZACIÓN DE LAS RECOMENDACIONES DE LA SERIE X

Dominio	Recomendaciones
REDES PÚBLICAS DE COMUNICACIÓN 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 PICS	X.240-X.259
Identificación de protocolos	X.260-X.269
Protocolos de seguridad	X.270-X.279
Objetos gestionados de red	X.280-X.289
Pruebas de conformidad	X.290-X.299
INTERFUNCIONAMIENTO ENTRE REDES	
Consideraciones generales	X.300-X.349
Sistemas móviles de transmisión de datos	X.350-X.369
Gestión	X.370-X.399
SISTEMAS DE TRATAMIENTO DE MENSAJES	X.400-X.499
DIRECTORIO	X.500-X.599
GESTIÓN DE REDES OSI Y ASPECTOS DE SISTEMAS	
Gestión de redes	X.600-X.649
Denominación, direccionamiento y registro	X.650-X.679
Notación de sintaxis abstracta uno (ASN.1)	X.680-X.699
GESTIÓN OSI	X.700-X.799
SEGURIDAD	X.800-X.849
APLICACIONES OSI	
Cometimiento, concurrencia y recuperación	X.850-X.859
Procesamiento de transacción	X.860-X.879
Operaciones a distancia	X.880-X.899
TRATAMIENTO ABIERTO DISTRIBUIDO	X.900-X.999

ÍNDICE

Página

Sumario

La presente Recomendación | Norma Internacional proporciona una notación llamada notación de sintaxis abstracta uno (ASN.1) para la definición de la sintaxis de datos de información. Se definen en ella varios tipos de datos sencillos y se especifica una notación para hacer referencia a esos tipos y especificar valores de los mismos.

La notación ASN.1 puede aplicarse cuando sea necesario definir la sintaxis abstracta de la información sin limitar de ningún modo la manera de codificar la información para la transmisión. Es aplicable especialmente, pero no exclusivamente, a los protocolos de capa de aplicación.

Introducción

Esta Recomendación | Norma Internacional presenta una notación normalizada para la definición de tipos y valores de datos. Un *tipo de dato* (o simplemente *tipo*) es una clase de información (por ejemplo, numérica, textual, de imagen fija o de vídeo). Un *valor de datos* (o simplemente *valor*) es un ejemplar de esa clase. En la presente Recomendación | Norma Internacional se definen varios tipos básicos y sus valores correspondientes, así como reglas para combinarlos en tipos y valores más complejos.

Aunque esta notación normalizada se define dentro del marco de la OSI, puede utilizarse para muchos otros objetivos. En las capas inferiores del modelo de referencia básico OSI (véase la Rec. UIT-T X.200 | ISO/CEI 7498) y en muchas otras arquitecturas de protocolos, cada mensaje se especifica como el valor binario de una secuencia de octetos. En la capa de presentación de OSI (véase la Rec. X.216 | ISO/CEI 8822), la naturaleza de los parámetros de datos de usuario cambia. Sin embargo, las normas relativas a la capa de aplicación requieren la definición de tipos de datos muy complejos para llevar sus mensajes, con independencia de su representación binaria. Para especificar los tipos de datos, precisan de una notación que no necesariamente determine la representación de cada valor. Esa notación ha de ser complementada mediante la especificación de uno o más algoritmos denominados **encoding rules** (reglas de codificación) que determinan el valor de los octetos de capa más baja que transportan los datos de aplicación (dicha especificación se denomina **transfer syntax** (sintaxis de transferencia). El protocolo de la capa de presentación de OSI (véase la Rec. UIT-T X.226 | ISO/CEI 8823) puede negociar las sintaxis de transferencia (**encodings**, codificaciones) que habrán de utilizarse.

Fuera del contexto de OSI hay un reconocimiento creciente de la noción de valor abstracto de una clase (por ejemplo, una determinada imagen en color 256) con independencia de los detalles de cualquier codificación particular en la que, para interpretar correctamente la representación del valor mediante un esquema de bits, es necesario saber (usualmente a partir del contexto) el tipo (la clase) del valor que se representa, así como el mecanismo de codificación que se emplea. La identificación de un tipo es, por ello, una parte importante de la presente Recomendación | Norma Internacional.

Una técnica muy general para definir un tipo complicado a nivel abstracto consiste en definir un pequeño número de tipos simples (**simple types**) definiendo todos los valores posibles de los tipos simples, y combinar entonces estos tipos simples de diversas maneras. A continuación se indican algunas maneras de definir nuevos tipos:

a) dada una lista (ordenada) de tipos existentes, se puede formar un valor como una secuencia (ordenada) de valores, formada por uno de cada uno de los tipos existentes; la colección de todos los valores posibles obtenidos de esta manera es un nuevo tipo (si todos los tipos existentes en la lista son distintos, este mecanismo puede ampliarse para permitir que algunos valores no sean incluidos en aquélla);

b) dado un conjunto no ordenado de tipos existentes (distintos), se puede formar un valor como un conjunto (no ordenado) de valores, formado por uno de cada uno de los tipos existentes; la colección de todos los posibles conjuntos de valores no ordenados obtenidos de esta manera es un nuevo tipo (también en este caso puede ampliarse el mecanismo para permitir la omisión de algunos valores);

c) dado un tipo existente único, puede formarse un valor como una lista (ordenada) o un conjunto (no ordenado) de cero, uno o más valores del tipo existente; la colección de todas las listas o conjuntos de valores posibles obtenidos de esta manera es un nuevo tipo;

d) dada una lista de tipos (distintos), se puede escoger un valor cualquiera de esos tipos distintos; el conjunto de todos los posibles valores obtenidos de esta manera es un nuevo tipo;

e) dado un tipo, se puede formar un nuevo tipo como un subconjunto de dicho tipo utilizando alguna relación de estructura o de orden entre los valores.

Un aspecto importante de los tipos que se combinan de esta manera es que las reglas de codificación deben reconocer las construcciones combinantes, proporcionando codificaciones inequívocas de la colección de valores de los tipos básicos. Así pues, a todo tipo básico definido utilizando la notación especificada en esta Recomendación | Norma Internacional se le asigna un **tag** (rótulo) para ayudar a la codificación inequívoca de los valores.

En la notación se especifican cuatro clases de rótulo.

La primera es la clase **universal**. Los rótulos de clase universal sólo se utilizan como se especifica en esta Recomendación | Norma Internacional, y cada rótulo:

a) se asigna a un solo tipo; o

b) se asigna a un mecanismo de construcción.

Los usuarios de esta notación no están autorizados a especificar explícitamente rótulos de clase universal en sus especificaciones ASN.1, ya que estos rótulos están incorporados (built-in) y sólo pueden ser especificados explícitamente en la presente Recomendación | Norma Internacional.

Las otras tres clases de rótulos se denominan rótulo de clase **application** (aplicación), rótulo de clase **private** (privada) y rótulo de clase **context-specific** (específica del contexto). No hay diferencia formal entre las utilizaciones de los rótulos de estas tres clases. Cuando se empleen rótulos de clase aplicación, podrán emplearse también, por lo general, rótulos de clase privada o específica al contexto, dependiendo de la elección y el estilo del usuario. La presencia de las tres clases se debe en buena medida a razones históricas, pero en F.2.1.2 se dan directrices respecto al modo de emplear normalmente las clases.

Los rótulos están destinados, sobre todo, a ser utilizados por la máquina, y no son esenciales para la notación destinada al ser humano, definida en esta Recomendación | Norma Internacional. No obstante, cuando es necesario exigir que ciertos tipos sean distintos, esto se expresa especificando que tengan rótulos distintos. Por esta razón, la atribución de rótulos es una parte importante de la utilización de esta notación.

NOTA – En esta Recomendación | Norma Internacional, los valores de rótulo se asignan a todos los tipos simples y mecanismos de construcción. Las restricciones impuestas a la utilización de la notación garantizan que los rótulos pueden utilizarse en la transferencia para identificar de manera inequívoca los valores.

Las cláusulas 8 a 29 (inclusive) definen los tipos simples soportados por ASN.1 y especifican la notación que ha de emplearse para referenciar tipos simples y para definir nuevos tipos utilizando éstos. Las cláusulas 8 a 29 especifican también la notación que ha de emplearse para especificar valores de tipos definidos mediante ASN.1.

Las cláusulas 30 y 31 (inclusive) definen los tipos soportados por la ASN.1 para llevar dentro de ellas la codificación completa de los tipos ASN.1.

Las cláusulas 32 a 37 (inclusive) definen los tipos cadena de caracteres.

Las cláusulas 38 a 41 (inclusive) definen ciertos tipos que se consideran de utilidad general, pero que no requieren reglas de codificación adicionales.

Las cláusulas 42 y 45 definen una notación que permite definir subtipos a partir de los valores de un tipo parent (progenitor).

El Anexo A forma parte integrante de la presente Recomendación | Norma Internacional y en él se dan directrices sobre cómo pueden los usuarios de esta Recomendación | Norma Internacional referirse a los tipos y valores de ASN.1 definidos utilizando la Recomendación X.208 del CCITT (1988) | ISO/CEI 8824:1990.

El Anexo B forma parte integrante de la presente Recomendación | Norma Internacional y en él se define el árbol de identificador de objeto para autoridades soportado por la ISO.

El Anexo C forma parte integrante de la presente Recomendación | Norma Internacional y en él se define el árbol de identificador de objeto para autoridades soportado por el UIT-T.

El Anexo D forma parte integrante de la presente Recomendación | Norma Internacional y en él se define el árbol de identificador de objeto para uso conjunto por la ISO y el UIT-T.

El Anexo E forma parte integrante de la presente Recomendación | Norma Internacional y contiene valores de identificador de objeto y descriptor de objeto asignados en esta Recomendación | Norma Internacional.

El Anexo F forma parte integrante de la presente Recomendación | Norma Internacional y proporciona ejemplos y sugerencias sobre la utilización de la notación ASN.1.

El Anexo G, que no forma parte integrante de la presente Recomendación | Norma Internacional, contiene un suplemento didáctico sobre las cadenas de caracteres ASN.1.

El Anexo H, que no forma parte integrante de la presente Recomendación | Norma Internacional, describe prestaciones de la versión anterior de ASN.1 que han sido reemplazadas.

El Anexo I, que no forma parte integrante de la presente Recomendación | Norma Internacional, da información detallada sobre la notación tipo ANY reemplazada.

El Anexo J, que no forma parte integrante de la presente Recomendación | Norma Internacional, da información detallada sobre la notación macro reemplazada.

El Anexo K, que no forma parte integrante de la presente Recomendación | Norma Internacional, proporciona un sumario de ASN.1 utilizando la notación de la cláusula 5.

NORMA INTERNACIONAL

RECOMENDACIÓN UIT-T

TECNOLOGÍA DE LA INFORMACIÓN – NOTACIÓN DE SINTAXIS ABSTRACTA UNO: ESPECIFICACIÓN DE LA NOTACIÓN BÁSICA

1 Alcance

Esta Recomendación | Norma Internacional proporciona una notación normalizada, llamada notación de sintaxis abstracta uno (ASN.1), que se utiliza para la definición de tipos de datos, valores y constricciones a los tipos de datos.

La presente Recomendación | Norma Internacional:

- define varios tipos simples con sus rótulos (tags), y especifica una notación para diferenciar estos tipos y para especificar los valores de estos tipos;
- define mecanismos para construir nuevos tipos a partir de tipos más básicos, y especifica una notación para definir tales tipos y asignarles rótulos y para especificar valores de estos tipos;
- define conjuntos (o juegos) de caracteres (por medio de referencias a otras Recomendaciones y/o Normas Internacionales) para uso dentro de la ASN.1;
- define varios tipos útiles (ASN.1), que pueden ser referenciados por usuarios de ASN.1.

La notación ASN.1 puede aplicarse cuando sea necesaria para definir la sintaxis abstracta de información. Es aplicable en particular, aunque no exclusivamente, a protocolos de aplicación.

La notación ASN.1 también es referenciada por otras normas de la capa de aplicación que definen reglas de codificación para los tipos ASN.1.

2 Referencias normativas

Las siguientes Recomendaciones | Normas Internacionales contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación | Norma Internacional. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y Normas Internacionales son objeto de revisiones, con lo que se preconiza que los participantes en acuerdos basados en la presente Recomendación | Norma Internacional investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones | Normas Internacionales citadas a continuación. Los miembros de la CEI y de la ISO mantienen registros de las Normas Internacionales actualmente vigentes. La Oficina de Normalización de las Telecomunicaciones de la UIT mantiene una lista de las Recomendaciones UIT-T actualmente vigentes.

2.1 Recomendaciones | Normas Internacionales idénticas

- Recomendación UIT-T X.200 (1994) | ISO/CEI 7498-1:1994, *Tecnología de la información – Interconexión de sistemas abiertos – Modelo de referencia básico: El modelo básico*.
- Recomendación UIT-T X.216 (1994) | ISO/CEI 8822:1994, *Tecnología de la información – Interconexión de sistemas abiertos – Definición del servicio de presentación*.
- Recomendación UIT-T X.226 (1994) | ISO/CEI 8823-1:1994, *Tecnología de la información – Interconexión de sistemas abiertos – Protocolo de presentación con conexión: Especificación del protocolo*.
- Recomendación UIT-T X.681 (1994) | ISO/CEI 8824-2:1995, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación del objeto de información.*
- Recomendación UIT-T X.682 (1994) | ISO/CEI 8824-3:1995, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación de constricciones.*

- Recomendación UIT-T X.683 (1994) | ISO/CEI 8824-4:1995, *Tecnología de la información – Notación de sintaxis abstracta uno: Parametrización de las especificaciones de ASN.1.*
- Recomendación UIT-T X.690 (1994) | ISO/CEI 8825-1:1995, *Tecnología de la información – Reglas de codificación de notación de sintaxis abstracta uno: Especificación de las reglas de codificación básica, de las reglas de codificación canónica y de las reglas de codificación distinguida.*
- Recomendación UIT-T X.691 (1995) | ISO/CEI 8825-2:1995, *Tecnología de la información – Reglas de codificación de notación de sintaxis abstracta uno: Especificación de las reglas de codificación empaquetada.*

2.2 Referencias adicionales

- ISO *International Register of Coded Character Sets to be used with Escape Sequences.*
- ISO/CEI 646:1991, *Information technology – ISO 7-bit coded character set for information interchange.*
- ISO/CEI 2022:1994, *Information technology – Character code structure and extension techniques.*
- ISO 3166:1993, *Codes for the representation of names of countries.*
- ISO 6523:1984, *Data interchange – Structure for identification of organizations.*
- ISO 8601:1988, *Data elements and interchange formats – Information Interchange – Representation of dates and times.*
- ISO/CEI 10646-1:1993, *Information technology – Universal Multiple-Octet Coded Character Set (UCS): – Architecture and Basic Multilingual Plane.*
- Recomendación X.121 del CCITT (1988), *Plan de numeración internacional para redes públicas de datos.*
- Recomendación X.208 del CCITT (1988), *Tecnología de la Información – Interconexión de sistemas abiertos – Especificación de la notación de sintaxis abstracta uno (ASN.1).*
- ISO/CEI 8824:1990, *Information Technology – Open Systems Interconnection – Specification of Abstract Syntax Notation One (ASN.1).*

3 Definiciones

Para los fines de esta Recomendación | Norma Internacional se aplican las siguientes definiciones.

3.1 Especificación de objeto de información

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.681 | ISO/CEI 8824-2:

a) objeto de información (information object);

b) clase de objeto de información (information object class);

c) conjunto de objetos de información (information object set);

d) tipo ejemplar de (instance-of type);

e) tipo campo de clase de objeto (object class field type).

3.2 Especificación de constricción

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.682 | ISO/CEI 8824-3:

a) constricción de relación de componentes (component relation constraint);

b) constricción de tabla (table constraint).

3.3 Parametrización de la especificación de ASN.1

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.683 | ISO/CEI 8824-4:

a) tipo parametrizado (parameterized type);

b) valor parametrizado (parameterized value).

3.4 Definición del servicio de presentación

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Rec. UIT-T X.216 | ISO/CEI 8822:

a) (una) sintaxis abstracta [(an) abstract syntax];

b) nombre de sintaxis abstracta (abstract syntax name);

c) conjunto de contextos definidos (defined context set);

d) valor de datos de presentación (presentation data value);

e) (una) sintaxis de transferencia [(a) transfer syntax];

f) nombre de sintaxis de transferencia (transfer syntax name).

3.5 Especificación del protocolo de presentación

Esta Recomendación | Norma Internacional utiliza el siguiente término definido en la Rec. UIT-T X.226 | ISO/CEI 8823:

– identificador de contexto de presentación (presentation context identifier).

3.6 Estructura para la identificación de organizaciones

Esta Recomendación | Norma Internacional utiliza también los siguientes términos definidos en la Norma ISO 6523:

a) organización emisora (issuing organization);

b) código de organización (organization code);

c) designador de indicativo internacional (international code designator).

3.7 Conjunto de caracteres codificados multiocteto universal (UCS)

Esta Recomendación | Norma Internacional utiliza los siguientes términos definidos en la Norma ISO/CEI 10646-1:

a) plano multiidioma básico (BMP, *basic multilingual plane*);

b) célula (cell);

c) carácter combinante (combining character);

d) símbolo gráfico (graphic symbol)

e) grupo (group);

f) subconjunto limitado (limited subset);

g) plano (plane);

h) fila (row);

i) subconjunto seleccionado (selected subset).

3.8 Definiciones adicionales

3.8.1 carácter abstracto: El conjunto de información asociada con una célula en una tabla que define un repertorio de caracteres.

NOTA – La información incluirá normalmente algunos o la totalidad de los siguientes ítems:

a) una forma de carácter;

b) un nombre de carácter; o

c) la definición de funciones asociadas con el carácter cuando se utiliza en entornos particulares.

3.8.2 valor abstracto: Un valor cuya definición se basa en el tipo, con independencia de cómo se representa en cualquier regla de codificación.

NOTA – El empleo de la expresión "valor abstracto" es con frecuencia una declaración de lo que se dice, probablemente varíe en base a las reglas de codificación utilizadas.

3.8.3 conjunto (o juego) de caracteres ASN.1: El juego de caracteres, especificado en la cláusula 8, que se utiliza en la notación ASN.1.

3.8.4 especificación ASN.1: Colección de uno o más módulos ASN.1.

3.8.5 tipo asociado: Un tipo que se utiliza solamente para definir la notación de valor y subtipo de un tipo.

NOTA – En la presente Recomendación | Norma Internacional se definen tipos asociados cuando es necesario establecer claramente que puede haber una diferencia significativa entre el modo de definir el tipo en ASN.1 y el modo de codificarlo. Los tipos asociados no aparecen en las especificaciones de usuario.

3.8.6 tipo cadena de bits: Un tipo simple cuyos valores distinguidos son una secuencia ordenada de cero, uno o más bits.

NOTA – Cuando es necesario llevar codificaciones de un valor abstracto, la utilización del tipo embedded-pdv (pdv incrustado) proporcionará, por lo general, un mecanismo más flexible para el anuncio o el acuerdo sobre la naturaleza de la codificación que el tipo cadena de bits.

3.8.7 tipo booleano: Un tipo simple con dos valores distinguidos.

3.8.8 carácter: Un miembro de un conjunto de elementos utilizados para la organización, control o representación de datos.

NOTA – Esto significa, por ejemplo, que un carácter que combina acento y letra «e» son dos caracteres en la versión francesa de ISO 646 y no un solo carácter é.

3.8.9 sintaxis abstracta de caracteres: Cualquier sintaxis abstracta cuyos valores se especifican como el conjunto de cadenas de caracteres de cero, uno o más caracteres tomados de una colección de caracteres especificada.

3.8.10 repertorio de caracteres: Caracteres de un conjunto de caracteres sin indicación alguna de cómo se codifican.

3.8.11 tipos cadenas de caracteres: Tipos simples cuyos valores son cadenas de caracteres tomados de algún juego de caracteres definido.

3.8.12 sintaxis de transferencia de caracteres: Cualquier sintaxis de transferencia para una sintaxis abstracta de caracteres.

NOTA – La ASN.1 no admite sintaxis de transferencias de caracteres que no codifiquen todas las cadenas de caracteres como un múltiplo entero de 8 bits.

3.8.13 tipos elección: Tipos definidos por referencia a una lista de tipos diferentes; cada valor del tipo elección es un valor de los tipos componentes.

3.8.14 tipo componente: Uno de los tipos referenciados cuando se define un CHOICE (elección), SET (conjunto), SEQUENCE (secuencia), SET OF (conjunto de) o SEQUENCE OF (secuencia de).

3.8.15 constricción: Una notación que puede utilizarse en asociación con la notación para un tipo, para definir un subtipo de ese tipo.

3.8.16 caracteres de control: Caracteres que aparecen en algunos repertorios de caracteres que han recibido un nombre (y quizá una función definida en relación con determinados entornos) pero a los que no se ha asignado una forma de carácter, y que no son caracteres con avance de espacio.

NOTA – NEWLINE y TAB son ejemplos de caracteres de control a los que se les ha asignado una función de formateo en un entorno de impresión. DLE es un ejemplo de carácter de control al que se ha asignado una función en un entorno de comunicación.

3.8.17 Tiempo Universal Coordinado (u hora universal coordinada) (UTC, *coordinated universal time*)**:** La escala de tiempo (u horaria) mantenida por el Bureau Internationale de l'Heure (Oficina Internacional de la Hora) que constituye la base de una diseminación coordinada de frecuencias patrón y señales horarias.

NOTAS

1 La fuente de esta definición es la Recomendación 460-2 del Comité Consultivo Internacional de Radiocomunicaciones (CCIR). El CCIR también ha definido UTC como el acrónimo para Tiempo Universal Coordinado.

2 UTC y tiempo medio (u hora media) de Greenwich son dos patrones horarios alternativos que, para la mayoría de los fines prácticos, determinan la misma hora.

3.8.18 elemento: Un miembro de una clase elemento que puede distinguirse de todos los elementos de la misma clase.

3.8.19 clase elemento: Un tipo (cuyos elementos son sus valores) o clase de objeto de información (cuyos elementos son todos los objetos posibles de esa clase).

3.8.20 conjunto de elementos: Uno o más elementos de la misma clase elemento.

3.8.21 tipo pdv incrustado: Un tipo cuyo conjunto de valores es la unión de los conjuntos de valores en todas las sintaxis abstractas posibles. Este tipo es una parte de una especificación ASN.1 que lleva un valor cuyo tipo puede ser definido externamente a esa especificación ASN.1. También lleva una identificación del tipo del valor que se transporta así como una identificación de la regla de codificación utilizada para codificar el valor.

3.8.22 codificación: El esquema de bits resultante de la aplicación de un conjunto de reglas de codificación a un valor de una sintaxis abstracta específica.

3.8.23 reglas de codificación (ASN.1): Reglas que especifican la representación, durante una transferencia, de los valores de tipos ASN.1. Permiten también recuperar los valores a partir de la representación, si se conoce el tipo.

NOTA – A los fines de la especificación de reglas de codificación, las diversas notaciones referenciadas de tipo (y de valor) que pueden proporcionar notaciones alternativas para tipos (y valores) incorporados no son relevantes.

3.8.24 tipos enumerados: Tipos simples a cuyos valores se les da identificadores distintos como parte de la notación de tipo.

3.8.25 tipo externo: Un tipo que es una parte de una especificación ASN.1 y que lleva un valor cuyo tipo puede ser definido externamente a esa especificación ASN.1. También lleva una identificación del tipo del valor que se transporta.

3.8.26 referencia externa: Una referencia de tipo, referencia de valor, objeto, etc., que se define en un módulo distinto de aquel al que se hace referencia y que es referido prefijando el nombre del módulo al ítem referido.

EJEMPLO – ModuleName, TypeReference.

3.8.27 falso: Uno de los valores distinguidos de tipo booleano (véase «verdadero»).

3.8.28 gobernante, gobernador: En relación con algún objeto, conjunto de objetos, conjunto de valores, valor o subtipo, la clase de objeto de información o tipo que controla su interpretación restringiendo el ítem o los ítems que entran en juego para ser notación de valor de esa clase o tipo, respectivamente.

3.8.29 tipo entero: Un tipo simple cuyos valores distinguidos son los números enteros positivos y negativos, incluido el cero (como un valor único).

NOTA – La gama de un entero puede verse limitada por reglas de codificación particulares, pero tales limitaciones se establecen de tal modo que no afecten a ningún usuario de ASN.1.

3.8.30 ítems: Secuencias nombradas de caracteres tomadas en los juegos de caracteres ASN.1, especificadas en la cláusula 9, que se utilizan para formar la notación ASN.1.

3.8.31 módulo: Uno o más ejemplares de la utilización de la notación ASN.1 para definición de tipo, valor, etc., encapsuladas utilizando la notación de módulo ASN.1 (véase la cláusula 10).

3.8.32 tipo nulo: Un tipo simple que consta de un solo valor, que también se llama nulo (null).

3.8.33 objeto: Un elemento bien definido de información, definición o especificación, que requiere un nombre para identificar su utilización en un ejemplar de comunicación.

3.8.34 tipo descriptor de objeto: Un tipo cuyos valores distinguidos son texto legible por el ser humano y que proporcionan una breve descripción de un objeto.

NOTA – Un valor de descriptor de objeto está usualmente asociado con un solo objeto. Unicamente un valor de identificador de objeto identifica inequívocamente un objeto.

3.8.35 identificador de objeto: Un valor (distinguible de todos los demás) que está asociado con un objeto.

3.8.36 tipo identificador de objeto: Un valor simple cuyos valores distinguidos son el conjunto de todos los identificadores de objeto atribuidos de conformidad con las reglas de esta Recomendación | Norma Internacional.

NOTA – Las reglas de esta Recomendación | Norma Internacional permiten a una amplia gama de autoridades asociar independientemente identificadores de objetos con objetos.

3.8.37 tipo cadena de octetos: Un tipo simple cuyos valores distinguidos son una secuencia ordenada de cero, uno o más octetos, estando formado cada octeto por una secuencia ordenada de 8 bits.

3.8.38 notación de tipo abierto: Una notación ASN.1 utilizada para denotar un conjunto de valores procedentes de más de un tipo ASN.1.

NOTAS

1 En el cuerpo de esta Recomendación | Norma Internacional se utiliza el término «tipo abierto» con el mismo significado que "notación de tipo abierto".

2 Todas las reglas de codificación ASN.1 proporcionan codificaciones inequívocas de los valores de un solo tipo ASN.1 único. No necesariamente son inequívocas, las codificaciones que proporcionan para «notación de tipo abierto» que lleva valores de tipos ASN.1 que normalmente no se determinan en el momento de la especificación. Es preciso conocer el tipo del valor que se codifica en la «notación de tipo abierto» para que se pueda determinar inequívocamente el valor abstracto de ese campo.

3 En esta Recomendación | Norma Internacional, la única notación que es una notación de tipo abierto es el "ObjectClassFieldType" especificado en la Rec. UIT-T X.681 | ISO/CEI 8824-2, donde el "FieldName" denota un campo de tipo o un campo de valor de tipo variable. La notación "ANY" definida en la Rec. X.208 del CCITT (1988) | ISO/CEI 8824:1990, y que se describe en el Anexo I, es una notación de tipo abierto.

3.8.39 tipo progenitor (de un subtipo): El tipo que se constriñe cuando se define un subtipo.

NOTA – El tipo progenitor puede ser, a su vez, un subtipo de algún otro tipo.

3.8.40 producción: Una parte de la notación formal utilizada para especificar ASN.1.

3.8.41 tipo real: Un tipo simple cuyos valores distinguidos (especificados en la cláusula 18) son miembros del conjunto de números reales.

3.8.42 definiciones recursivas: Un conjunto de definiciones ASN.1 que no pueden ser reordenadas de un modo tal que todos los tipos utilizados en una construcción se definan antes de la definición de la construcción.

NOTA – En ASN.1 se permiten definiciones recursivas: el usuario de la notación asume la responsabilidad de asegurar que los valores (de los tipos resultantes) que se utilizan tienen una representación finita.

3.8.43 tipo cadena de caracteres restringida: Un tipo cadena de caracteres cuyos caracteres se toman de un repertorio de caracteres fijo, identificado en la especificación de tipo.

3.8.44 tipos selección: Tipos definidos por referencia a un tipo componente de un tipo choice (elección) y cuyos valores son precisamente los valores de ese tipo componente.

3.8.45 tipos secuencia: Tipos definidos por referencia a una lista ordenada de tipos (algunos de los cuales pueden ser declarados como opcionales); cada valor del tipo secuencia es una lista ordenada de valores, uno de cada tipo componente.

NOTA – Cuando un tipo componente esté declarado como opcional, no es necesario que un valor del tipo secuencia contenga un valor de ese tipo componente.

3.8.46 tipos secuencia de: Tipos definidos por referencia a un solo tipo componente; cada valor del tipo secuencia de es una lista ordenada de cero, uno o más valores del tipo componente.

3.8.47 tipos conjunto: Tipos definidos por referencia a una lista fija, no ordenada, de tipos distintos (algunos de los cuales pueden ser declarados como opcionales); cada valor del tipo conjunto es una lista no ordenada de valores, uno de cada tipo componente.

NOTA – Cuando un tipo componente esté declarado como opcional, no es necesario que el nuevo tipo contenga un valor de ese tipo componente.

3.8.48 tipos conjunto de: Tipos definidos por referencia a un solo tipo componente; cada valor del tipo conjunto de es una lista no ordenada de cero, uno o más valores del tipo componente.

3.8.49 tipos simples: Tipos definidos especificando directamente el conjunto de sus valores.

3.8.50 carácter con avance de espacio: Un carácter de un repertorio de caracteres que se incluye con caracteres gráficos en la impresión de una cadena de caracteres, pero que está representado en la reproducción física por un espacio vacío; normalmente no se le considera como un carácter de control (véase 3.8.16).

NOTA – En un repertorio de caracteres puede haber un solo carácter con avance de espacio o múltiples caracteres con avance de espacio con anchuras variables.

3.8.51 subtipo (de un tipo progenitor): Un tipo cuyos valores son un subconjunto (o el conjunto completo) de los valores de algún otro tipo (el tipo progenitor).

3.8.52 rótulo: Una denotación de tipo que va asociada con todo tipo ASN.1.

3.8.53 tipos rotulados: Tipos definidos por referencia a un solo tipo existente y a un rótulo; el nuevo tipo es isomórfico con el tipo existente, pero es diferente de éste.

3.8.54 rotulación: Sustitución del rótulo existente (que pudiera ser el rótulo por defecto) de un tipo por un rótulo especificado.

3.8.55 verdadero: Uno de los valores distinguidos del tipo booleano (véase "falso").

3.8.56 tipo: Un conjunto de valores que tiene un nombre.

3.8.57 nombre de referencia de tipo: Un nombre asociado de modo único con un tipo dentro de algún contexto.

NOTA – Se asignan nombres de referencia a los tipos definidos en esta Recomendación | Norma Internacional; estos nombres están universalmente disponibles en ASN.1. Otros nombres de referencia están definidos en otras Recomendaciones | Normas Internacionales y son aplicables solamente en los respectivos contextos de esas Recomendaciones | Normas Internacionales.

3.8.58 tipo cadena de caracteres no restringida: Un tipo cuyos valores son valores de una sintaxis abstracta de caracteres identificada separadamente para cada ejemplar de uso de ese tipo.

3.8.59 usuario (de ASN.1): El individuo u organización que define la sintaxis abstracta de un elemento particular de información utilizando ASN.1.

3.8.60 valor: Un miembro distinguido de un conjunto de valores.

3.8.61 nombre de referencia de valor: Un nombre asociado de manera única con un valor dentro de algún contexto.

3.8.62 conjunto de valores: Una colección de valores de un tipo. Semánticamente equivalente a un subtipo.

3.8.63 espacio en blanco: Cualquier acción de formatación que produce un espacio en blanco en una página impresa; por ejemplo, el carácter SPACE o el carácter TAB, o usos múltiples de esos caracteres.

4 Abreviaturas

ASN.1	Notación de sintaxis abstracta uno (*abstract syntax notation one*)
BER	Reglas de codificación básicas de ASN.1 (*basic encoding rules of ASN.1*)
DCC	Indicativo de país de datos (*data country code*)
DNIC	Código de identificación de red de datos (*data network identification code*)
ICD	Designador de código internacional (*international code designator*)
IEC	Comisión Electrónica Internacional
ISO	Organización Internacional de Normalización (*International Standards Organization*)
UIT-T	Sector de Normalización de las Telecomunicaciones de la Unión Internacional de Telecomunicaciones
PDV	Valor de datos de presentación (*presentation data value*)
PER	Reglas de codificación empaquetada de ASN.1 (*packed enconding rules of ASN.1*)
EER	Empresa de explotación reconocida
UCS	Conjunto de caracteres codificados multiocteto universal (*universal multiple-octet coded character set*).

5 Notación

La notación ASN.1 consiste en una secuencia de caracteres del conjunto (o juego) de caracteres ASN.1 especificados en la cláusula 8.

Cada vez que se utiliza la notación ASN.1, caracteres pertenecientes al conjunto de caracteres ASN.1 son agrupados para formar ítems. La cláusula 9 especifica todas las secuencias de caracteres que forman ítems ASN.1 y el nombre de cada ítem.

La notación ASN.1 se especifica en la cláusula 10 (y siguientes), para lo cual especifica la colección de secuencias de ítems que forman ejemplares válidos de la notación ASN.1, y se especifica la semántica de cada secuencia.

Para especificar esas colecciones, esta Recomendación | Norma Internacional utiliza una notación formal definida en las siguientes subcláusulas.

5.1 Producciones

Una colección nueva (más compleja) de secuencias ASN.1 se define mediante una producción. Esta utiliza los nombres de colecciones de secuencias de producción definidas en la presente Recomendación | Norma Internacional y forma una nueva colección de secuencias especificando o bien que:

a) la nueva colección de secuencias de producción habrá de consistir en cualquier secuencia contenida en cualquiera de las colecciones originales; o

b) que la nueva colección habrá de consistir en cualquier secuencia de producción que pueda ser generada tomando exactamente un carácter de cada colección, y yuxtaponiéndolos en un orden especificado.

Cada producción consta de las siguientes partes, que ocupan una o varias líneas, y siguen este orden:

a) un nombre para la nueva colección de secuencias de producción;

b) los caracteres

::=

c) una o más colecciones alternativas de secuencias, definidas en 5.2, separadas por el carácter

|

Una secuencia de producción está presente en la nueva colección si lo está en una o más de las colecciones alternativas. La nueva colección es referenciada en esta Recomendación | Norma Internacional por el nombre a que se refiere el anterior inciso a).

NOTA – Si la misma secuencia de producción aparece en más de una alternativa, cualquier ambigüedad semántica en la notación resultante es resuelta por otras partes de la secuencia de producción ASN.1 completa.

5.2 Las colecciones alternativas

Cada una de las colecciones alternativas de secuencias de producción en "una o más colecciones alternativas de" [véase 5.1 c)] se especifica por una lista de nombres. Cada nombre es, o bien el nombre de un ítem, o el nombre de una colección de secuencias de producción definida por una producción en esta Recomendación | Norma Internacional.

La colección de secuencias de producción definida por la alternativa consiste en todas las secuencias obtenidas tomando una cualquiera de las secuencias de producción (o el ítem) asociada con el primer nombre, en combinación con (y seguida por) cualquiera de las secuencias de producción (o ítem) asociada con el segundo nombre, en combinación con (y seguida por) cualquiera de las secuencias de producción (o ítem) asociada con el tercer nombre, y así sucesivamente hasta el último nombre (o ítem), inclusive, de la alternativa.

5.3 Ejemplo de una producción

```
BitStringValue   ::=
                     bstring    |
                     hstring    |
                     "{" IdentifierList "}"
```

es una producción que asocia al nombre BitStringValue las secuencias de producción siguientes:

a) cualquier bstring (un ítem); y

b) cualquier hstring (un ítem); y

c) cualquier secuencia asociada con IdentifierList, precedida por un "{" y seguida por un "}".

NOTA – "{" y "}" son los nombres de ítems que contienen los caracteres únicos { y } (véase 9.15).

En este ejemplo, IdentifierList se definiría por una ulterior producción, que iría antes o después de la producción que define BitStringValue.

5.4 Disposición (layout)

Cada producción utilizada en esta Recomendación | Norma Internacional va precedida y seguida por una línea vacía. Las producciones no contienen líneas vacías. La producción puede ocupar una sola línea, o varias. La disposición (layout) no es significativa.

5.5 Recursión

Las producciones especificadas en esta Recomendación | Norma Internacional son frecuentemente recursivas. En tal caso, las producciones deberán ser reaplicadas continuamente hasta que no se genere más ninguna nueva secuencia.

NOTA – En muchos casos, tal reaplicación da lugar a una colección no acotada de secuencias permitidas, algunas de las cuales, o todas ellas, pueden ser a su vez inacotadas. Esto no es un error.

5.6 Referencias a una colección de secuencias

Esta Recomendación | Norma Internacional hace referencia a una colección de secuencias (parte de la notación ASN.1) referenciando el primer nombre (antes de ::=) de una producción; el nombre va encerrado entre un par de " (comillas) para distinguirlo de texto en lenguaje natural, a menos que aparezca como parte de una producción.

5.7 Referencias a un ítem

Esta Recomendación | Norma Internacional hace referencia a un ítem referenciando el nombre del ítem; el nombre va encerrado entre un par de " (comillas) para distinguirlo de texto en lenguaje natural, a menos que aparezca como parte de una producción y no sea un ítem de un solo carácter, "::=", "..", o "...".

5.8 Notaciones abreviadas

Para obtener producciones más concisas y legibles se utilizan las siguientes notaciones abreviadas en la definición de las colecciones de secuencias de producción ASN.1 de la Rec. UIT-T X.681 | ISO/CEI 8824-2, la Rec. UIT-T X.682 | ISO/CEI 8824-3 y la Rec. UIT-T X.683 | ISO/CEI 8824-4 (no se utiliza en ningún otro lugar de la presente Recomendación | Norma Internacional):

a) un asterisco (*) que sigue a dos nombres, "A" y "B", denota el ítem empty (vacío) (véase 9.7) o una secuencia de producción asociada con "A", o una serie alternante de secuencias de producción asociadas con "A" y "B" que empiezan y terminan con una secuencia de producción asociada con "A". Así:

   C ::= A B *

   es equivalente a

   C ::= D | empty
   D ::= A | A B D

   siendo D un nombre auxiliar que no aparece en otra parte de las producciones.

   EJEMPLO: "C ::= A B *" es la notación abreviada de las siguientes alternativas de C:

   empty
   A
   A B A
   A B A B A
   A B A B A B A
   ...

b) un signo más (+) produce los mismos efectos que el asterisco en a), salvo el de excluir el ítem empty (vacío). Así:

   E ::= A B +

   es equivalente a

   E ::= A | A B E

   EJEMPLO: "E ::= A B +" es la notación abreviada de las siguientes alternativas de E:

   A
   A B A
   A B A B A
   A B A B A B A
   ...

c) un signo de interrogación (?) que sigue a un nombre denota, o bien el ítem empty (véase 9.7) o una secuencia de producción asociada con "A". Así:

F ::= A ?

es equivalente a

F ::= empty | A

6 Rótulos

6.1 Un rótulo se especifica dando su clase y el número dentro de la clase. La clase es una de las siguientes

- universal;
- application (aplicación);
- private (privada);
- context-specific (específica al contexto).

6.2 El número es un entero no negativo, especificado en notación decimal.

Las restricciones a los rótulos asignados por el usuario de ASN.1 se especifican en la cláusula 28.

El Cuadro 1 recapitula la asignación de los rótulos de la clase universal que se especifican en esta Recomendación | Norma Internacional.

Cuadro 1 – Asignaciones de rótulos de clase universal

UNIVERSAL 0	Reservado para uso en las reglas de codificación
UNIVERSAL 1	Tipo booleano
UNIVERSAL 2	Tipo entero
UNIVERSAL 3	Tipo cadena de sets
UNIVERSAL 4	Tipo cadena de octetos
UNIVERSAL 5	Tipo nulo
UNIVERSAL 6	Tipo identificador de objeto
UNIVERSAL 7	Tipo descriptor de objeto
UNIVERSAL 8	Tipos externo y ejemplar de
UNIVERSAL 9	Tipo real
UNIVERSAL 10	Tipo enumerado
UNIVERSAL 11	Tipo PDV inscrustado
UNIVERSAL 12-15	Reservado para futuras ediciones de esta Recomendación | Norma Internacional
UNIVERSAL 16	Tipos secuencia y secuencia de
UNIVERSAL 17	Tipos conjunto y conjunto de
UNIVERSAL 18-22, 25-30	Tipos cadenas de caracteres
UNIVERSAL 23-24	Tipos tiempo
UNIVERSAL 31-...	Reservado para adiciones a esta Recomendación | Norma Internacional

6.3 Algunas reglas de codificación exigen un orden canónico de los rótulos. En 6.4 se define dicho orden, a efectos de uniformidad.

NOTA – Este ordenamiento no se utiliza en ningún otro punto de la presente Recomendación | Norma Internacional, pero es referenciado por otras Recomendaciones | Normas Internacionales.

6.4 El orden canónico de los rótulos se define de la siguiente manera:

a) los elementos o alternativas con rótulos de la clase universal aparecerán en primer lugar, seguidos por aquellos que tengan rótulos de clase aplicación, seguidos a su vez por aquellos que tengan rótulos específicos del contexto, seguidos por último por los de rótulo de clase privada;

b) dentro de cada clase de rótulos, los elementos o alternativas aparecerán en el orden ascendente de sus números de rótulo.

7 Utilización de la notación ASN.1

7.1 La notación ASN.1 para una definición de tipo será "Type" (véase 14.1).

7.2 La notación ASN.1 para un valor de un tipo será "Value" (véase 14.7).

NOTA – Por lo general, no es posible interpretar la notación de valor sin conocer el tipo.

7.3 La notación ASN.1 para asignar un tipo a un nombre de referencia de tipo será "TypeAssignment" (véase 13.1), "ValueSetTypeAssignment" (véase 13.4), "ParameterizedTypeAssignment" (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4, subcláusula 8.2) o "ParameterizedValueSetTypeAssignment" (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4, subcláusula 8.2).

7.4 La notación ASN.1 para asignar un valor a un nombre de referencia de valor será "ValueAssignment" (véase 13.2) o "ParameterizedValueAssignment" (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4, subcláusula 8.2).

7.5 Las alternativas de producción de la notación "Assignment" sólo se utilizarán dentro de la notación "ModuleDefinition" (excepto lo especificado en la Nota 2 de 10.1).

8 El conjunto (o juego) de caracteres ASN.1

8.1 Un ítem ASN.1 consistirá en una secuencia de los caracteres indicados en el Cuadro 2, excepto lo especificado en 8.2 y 8.3.

Cuadro 2 – Caracteres ASN.1

A a Z
a a z
0 a 9
: = , { } < . @ ( ) [ ] – ' " | & ^ * ; !

NOTA – Cuando los organismos nacionales de normalización hayan elaborado normas de derivación equivalentes, podrán aparecer caracteres adicionales en los siguientes ítems:

typereference	(véase 9.2)
identifier	(véase 9.3)
valuereference	(véase 9.4)
modulereference	(véase 9.5)

Cuando se introduzcan caracteres adicionales para acomodar un lenguaje en el que la distinción entre letras mayúsculas y minúsculas no es significativa, la distinción sintáctica conseguida imponiendo que el primer carácter de alguno de los ítems mencionados anteriormente sea una letra mayúscula o minúscula tendrá que conseguirse de alguna otra forma. Con esto se pretende facilitar la escritura de especificaciones ASN.1 válidas en diversos idiomas.

8.2 Cuando se utilice la notación para especificar el valor de un tipo cadena de caracteres, todos los símbolos gráficos del juego de caracteres definidos podrán aparecer en la notación ASN.1, entre comillas (véase 9.11).

8.3 En el ítem "comment" pueden aparecer formas de caracteres adicionales alternativas (véase 9.6).

8.4 No se dará significado al estilo tipográfico, tamaño, color, intensidad y otras características de visualización.

8.5 Las letras mayúsculas y minúsculas deberán considerarse distintas.

9 Ítems ASN.1

9.1 Reglas generales

9.1.1 Las siguientes subcláusulas especifican los caracteres de ítems ASN.1. En cada caso se da el nombre del ítem, junto con la definición de las secuencias de caracteres que forman el ítem.

9.1.2 Cada ítem especificado en las siguientes subcláusulas (excepto "bstring", "hstring" y "cstring") aparecerá en una sola línea y, salvo para los ítems "comment", "bstring", "hstring" y "cstring", no contendrá espacios en blanco (véanse 9.9, 9.10 y 9.11).

9.1.3 La longitud de la línea no está limitada.

9.1.4 Los ítems de las secuencias especificadas por esta Recomendación | Norma Internacional (la notación ASN.1) pueden aparecer en una línea o en varias y pueden estar separados por espacio en blanco, por líneas vacías o por comentarios.

9.1.5 Un ítem estará separado del siguiente por un espacio en blanco, una línea nueva o un comentario, si el carácter (o caracteres) inicial del ítem siguiente es un carácter (o caracteres) autorizado para su inclusión al final de los caracteres del ítem anterior.

9.2 Referencias de tipo

Nombre de ítem – typereference

9.2.1 Una "typereference" estará constituida por un número arbitrario de letras (una o más), dígitos y guiones. El carácter inicial será una letra mayúscula. Un guión no será el último carácter. Un guión no irá seguido inmediatamente de otro guión.

NOTA – Las reglas referentes a los guiones están pensadas para evitar ambigüedades con comentarios (que posiblemente sigan).

9.2.2 Una "typereference" no será una de las secuencias de caracteres reservadas indicadas en 9.16.

9.3 Identificadores

Nombre de ítem – identifier

Un "identifier" estará constituido por un número arbitrario de (una o más) letras, dígitos y guiones. El carácter inicial será una letra minúscula. Un guión no será el último carácter. Un guión no irá seguido inmediatamente de otro guión.

NOTA – Las reglas referentes a los guiones están pensadas para evitar ambigüedades con comentarios (que posiblemente sigan).

9.4 Referencia de valor

Nombre de ítem – valuereference

Una "valuereference" estará constituida por la secuencia de caracteres especificada para un "identifier" en 9.3. Al analizar un ejemplar de utilización de esta notación, una "valuereference" se distingue de un "identifier" por el contexto en el que aparece.

9.5 Referencia de módulo

Nombre de ítem – moduloreference

Una "moduloreference" estará constituida por la secuencia de caracteres especificada para una "typereference" en 9.2. Al analizar un ejemplar de utilización de esta notación, una "moduloreference" se distingue de una "typereference" por el contexto en el que aparece.

9.6 Comentario

Nombre de ítem – comment

9.6.1 Un "comment" no está referenciado en la definición de la notación ASN.1. Puede, sin embargo, aparecer en cualquier momento entre otros ítems y no tiene significado sintáctico.

NOTA – No obstante, en el contexto de una Recomendación | Norma Internacional que utiliza ASN.1, un comentario ASN.1 puede contener texto normativo relacionado con la semántica de la aplicación o constricciones a la sintaxis.

9.6.2 Un "comment" comenzará con un par de guiones adyacentes y terminará con el próximo par de guiones adyacentes o al final de la línea, lo que suceda primero. Un "comment" no contendrá más pares de guiones adyacentes que el par que lo abre y el par (de haberlo), que lo cierra. Puede incluir símbolos gráficos que no están en el juego de caracteres especificados en 8.1 (véase 8.3).

9.7 Ítem vacío

Nombre de ítem – empty

El ítem "empty" no contiene caracteres. Se usa en la notación de la cláusula 5 cuando se especifican conjuntos alternativos de secuencias de producción, para indicar que es posible la ausencia de todas las alternativas.

9.8 Ítem número

Nombre de ítem – number

Un "number" estará constituido por uno o más dígitos. El primer dígito no será cero a menos que el "number" tenga un solo dígito.

NOTA – El ítem "number" se hace corresponder (se mapea) siempre con (a) un valor entero interpretándolo como notación decimal.

9.9 Ítem cadena binaria

Nombre de ítem – bstring

Una "bstring" estará constituida por un número arbitrario (que puede ser cero) de ceros, unos, espacios en blanco y caracteres de nuevo renglón, precedidos por un solo apóstrofo (’) y seguidos por el par de caracteres:

’B

Los espacios en blanco y los caracteres de nuevo renglón que aparecen dentro de un ítem cadena binaria no tienen significado alguno.

EJEMPLO: ’01101100’B

9.10 Ítem cadena hexadecimal

Nombre de ítem – hstring

9.10.1 Una "hstring" estará constituida por un número arbitrario (que puede ser cero) de los caracteres

A B C D E F 0 1 2 3 4 5 6 7 8 9

o espacios en blanco o caracteres de nuevo renglón, precedidos por un solo apóstrofo (’) y seguidos por el par de caracteres:

’H

Los espacios en blanco y los caracteres de nuevo renglón que aparecen dentro de un ítem cadena hexadecimal no tienen significado alguno.

EJEMPLO: ’AB0196’H

9.10.2 Cada carácter se utiliza para denotar el valor de un semiocteto utilizando una representación hexadecimal.

9.11 Ítem cadena de caracteres

Nombre de ítem – cstring

9.11.1 Una "cstring" estará constituida por un número arbitrario (que puede ser cero) de símbolos gráficos y caracteres con avance de espacio tomados del conjunto de caracteres referenciado por el tipo cadena de caracteres, precedido y seguido por " (comillas). Si el conjunto de caracteres incluye el carácter comillas ("), este carácter (si está presente en la cadena de caracteres representada por la "cstring") debe estar representado en la "cstring" por un par de caracteres " (comillas) en la misma línea, sin espacio separador. La "cstring" puede abarcar más de una línea de texto, en cuyo caso la cadena de caracteres representada no incluirá caracteres con avance de espacio en la posición que precede o sigue al final de línea de la "cstring". El espacio en blanco que aparece inmediatamente antes o después del final de la línea de la "cstring" no tiene significado alguno.

NOTAS

1 La "cstring" sólo puede utilizarse para representar cadenas de caracteres a cada uno de cuyos caracteres se le ha asignado un símbolo gráfico o bien se trata de un carácter con avance de espacio. Se dispone de sintaxis ASN.1 alternativa para cuando sea preciso denotar una cadena de caracteres que contiene caracteres de control (véase la cláusula 32).

2 La cadena de caracteres representada por una "cstring" está constituida por los caracteres asociados con los símbolos gráficos impresos y los caracteres con avance de espacio. Los caracteres con avance de espacio que preceden o siguen inmediatamente cualquier fin de línea de la "cstring" no son parte de la cadena de caracteres que se representa (son ignorados). Cuando se incluyen caracteres con avance de espacio en la "cstring" o cuando símbolos gráficos del repertorio de caracteres no son inequívocos, la cadena de caracteres denotada por "cstring" puede resultar imprecisa.

EJEMPLO 1 – "屎 屍 市 弒"

EJEMPLO 2 – The "cstring"

```
"ABCDE   FGH
      IJK""XYZ"
```

puede utilizarse para representar un valor de cadena de caracteres de tipo IA5String. El valor representado consta de los caracteres

```
ABCDE   FGHIJK"XYZ
```

en donde el número exacto de espacios que se pretende que haya entre E y F puede resultar poco preciso si en la especificación se utiliza un tipo de carácter de espaciamiento proporcional (como el empleado más arriba).

9.11.2 Cuando un carácter sea carácter combinante, se denotará en la "cstring" como un carácter individual. No se sobreimprimirá con los caracteres con los que se combina. (Así se asegura que el orden de los caracteres combinantes en el valor cadena queda definido de manera inequívoca.)

EJEMPLO – El carácter combinante acento y la letra "e" son dos caracteres en la versión francesa de ISO 646 y, por eso, en una "cstring" correspondiente se escriben como dos caracteres y no como un solo carácter é.

9.11.3 La "cstring" no se utilizará para representar valores de cadena de caracteres que contengan caracteres de control. Sólo se permiten caracteres gráficos y caracteres con avance de espacio.

9.12 Ítem asignación

Nombre de ítem – "::="

Este ítem estará constituido por la secuencia de caracteres

```
::=
```

NOTA – Esta secuencia no contiene ningún carácter espacio en blanco (véase 9.1.2).

9.13 Separador de gama

Nombre de ítem – ".."

Este ítem estará constituido por la secuencia de caracteres

```
..
```

NOTA – Esta secuencia no contiene ningún carácter espacio en blanco (véase 9.1.2).

9.14 Elipsis

Nombre de ítem – "..."

Este ítem estará constituido por la secuencia de caracteres

...

NOTA – Esta secuencia no contiene ningún carácter espacio en blanco (véase 9.1.2).

9.15 Ítems carácter único (o carácter individual)

Nombres de ítems –

"{"
"}"
"<"
","
"."
"("
")"
"["
"]"
"-" (guión)
":"
";"
"@"
"|"
"!"
"^"

Un ítem con cualquiera de estos nombres estará constituido por el carácter único sin las comillas.

9.16 Palabras reservadas

Nombres de palabras reservadas –

ABSENT	EMBEDDED	INSTANCE	REAL
ABSTRACT-SYNTAX	END	INTEGER	SEQUENCE
ALL	ENUMERATED	INTERSECTION	SET
APPLICATION	EXCEPT	ISO646String	SIZE
AUTOMATIC	EXPLICIT	MAX	STRING
BEGIN	EXPORTS	MIN	SYNTAX
BIT	EXTERNAL	MINUS-INFINITY	T61String
BMPString	FALSE	NULL	TAGS
BOOLEAN	FROM	NumericString	TeletexString
BY	GeneralizedTime	OBJECT	TRUE
CHARACTER	GeneralString	ObjectDescriptor	UNION
CHOICE	GraphicString	OCTET	UNIQUE
CLASS	IA5String	OF	UNIVERSAL
COMPONENT	TYPE-IDENTIFIER	OPTIONAL	UniversalString
COMPONENTS	IDENTIFIER	PDV	UTCTime
CONSTRAINED	IMPLICIT	PLUS-INFINITY	VideotexString
DEFAULT	IMPORTS	PRESENT	VisibleString
DEFINITIONS	INCLUDES	PrintableString	WITH
		PRIVATE	

Un ítem con cualquiera de estos nombres estará constituido por la secuencia de caracteres que forman el nombre. Son secuencias de caracteres reservadas.

NOTAS

1 Estas secuencias no contienen espacios en blanco.

2 Las palabras clave CLASS, CONSTRAINED, INSTANCE, SYNTAX y UNIQUE no se utilizan en esta Recomendación | Norma Internacional; se utilizan en la Rec. UIT-T X.681 | ISO/CEI 8824-2, en la Rec. UIT-T X.682 | ISO/CEI 8824-3 y en la Rec. UIT-T X.683 | ISO/CEI 8824-4.

10 Definición de módulo

10.1 Una "ModuleDefinition" es especificada por las siguientes producciones:

```
ModuleDefinition ::=
        ModuleIdentifier
        DEFINITIONS
        TagDefault
        "::="
        BEGIN
        ModuleBody
        END

ModuleIdentifier ::=
        modulereference
        DefinitiveIdentifier

DefinitiveIdentifier ::=
        "{" DefinitiveObjIdComponentList "}" | empty

DefinitiveObjIdComponentList  ::=
        DefinitiveObjIdComponent     |
        DefinitiveObjIdComponent DefinitiveObjIdComponentList

DefinitiveObjIdComponent  ::=
        NameForm        |
        DefinitiveNumberForm    |
        DefinitiveNameAndNumberForm

DefinitiveNumberForm     ::= number

DefinitiveNameAndNumberForm ::= identifier "(" DefinitiveNumberForm ")"

TagDefault ::=
        EXPLICIT TAGS           |
        IMPLICIT TAGS           |
        AUTOMATIC TAGS      |
        empty

ModuleBody ::=
        Exports Imports AssignmentList        |
        empty

Exports ::=
        EXPORTS SymbolsExported ";"       |
        empty

SymbolsExported ::=
        SymbolList |
        empty

Imports ::=
        IMPORTS SymbolsImported ";"       |
        empty

SymbolsImported ::=
        SymbolsFromModuleList |
        empty

SymbolsFromModuleList ::=
        SymbolsFromModule                 |
        SymbolsFromModuleList  SymbolsFromModule

SymbolsFromModule ::=
        SymbolList FROM GlobalModuleReference

GlobalModuleReference ::=
        modulereference AssignedIdentifier

AssignedIdentifier ::=
        ObjectIdentifierValue       |
        DefinedValue                |
        empty
```

```
SymbolList ::=
        Symbol              |
        SymbolList "," Symbol

Symbol ::=
        Reference           |
        ParameterizedReference

Reference ::=
        typereference       |
        valuereference      |
        objectclassreference |
        objectreference     |
        objectsetreference

AssignmentList ::=
        Assignment          |
        AssignmentList Assignment

Assignment ::=
        TypeAssignment      |
        ValueAssignment     |
        ValueSetTypeAssignment |
        ObjectClassAssignment |
        ObjectAssignment    |
        ObjectSetAssignment |
        ParameterizedAssignment
```

NOTAS

1 La utilización de una ParameterizedReference en las listas EXPORTS e IMPORTS se especifica en la Rec. UIT-T X.683 | ISO/CEI 8824-4

2 En los ejemplos (y en la definición de esta Recomendación | Norma Internacional de tipos con rótulos de la clase universal), el "ModuleBody" puede utilizarse fuera de una "ModuleDefinition".

3 Las producciones "Typeassignment" y "Valueassignment" se especifican en la cláusula 13.

4 La agrupación de tipos de datos ASN.1 para constituir módulos no entraña necesariamente que valores de datos de presentación formen sintaxis abstractas denominadas, destinadas a la definición de contextos de presentación.

5 El valor de "TagDefault" para la definición de módulo sólo afecta a los tipos definidos explícitamente en el módulo. No afecta a la interpretación de tipos importados.

6 El carácter punto y coma (;) no aparece en la especificación de la lista de asignaciones ni en ninguna de sus producciones subordinadas, y está reservado para uso por diseñadores de herramientas ASN.1.

10.2 El "TagDefault" se toma como "EXPLICIT TAGS" si es "empty" (vacío).

NOTA – La cláusula 28 da el significado de "EXPLICIT TAGS", "IMPLICIT TAGS" y "AUTOMATIC TAGS".

10.3 Cuando se selecciona la alternativa "AUTOMATIC TAGS" de "TagDefault", se dice que se ha seleccionado la rotulación automática del módulo; de no ser así, se dice que no se ha seleccionado. La rotulación automática es una transformación sintáctica que se aplica (con condiciones adicionales) a las producciones "ComponentTypeList" y "AlternativeTypeList" que aparecen en la definición del módulo. Esta transformación está especificada formalmente en 22.5, 24.3 y 26.2 a propósito de las notaciones de los tipos secuencia, los tipos conjunto y los tipos elección, respectivamente.

10.4 A la "modulereference" que aparece en la producción "ModuleDefinition" se le llama nombre del módulo.

NOTA – La utilización de la posibilidad de definir un módulo ASN.1 único haciendo aparecer varios "ModuleBody" con la misma "modulereference" asignada fue (discutiblemente) permitida en especificaciones anteriores. No se permite en la presente Recomendación | Norma Internacional.

10.5 Los nombres de módulo deberán utilizarse una vez solamente (salvo lo especificado en 10.8) dentro de la esfera de interés de la definición del módulo.

10.6 Si el "DefinitiveIdentifier" (identificador definitivo) no es "empty" (vacío), el valor de identificador de objeto denotado identifica inequívocamente y de forma única el módulo que se está definiendo. En la definición del valor de identificador de objeto no podrá utilizarse un valor definido.

NOTA – En esta Recomendación | Norma Internacional no se trata la cuestión de los cambios en un módulo exigidos por un nuevo "DefinitiveIdentifier".

10.7 Si el "AssignedIdentifier" (idenficador asignado) no es "empty" (vacío), las alternativas "ObjectIdentifierValue" (valor de identificador de objeto) y "DefinedValue" (valor definido) identifican inequívocamente y de forma única el módulo del cual se están importando ítems. Cuando se utilice la alternativa "DefinedValue" de "AssignedIdentifier", deberá ser un valor de identificador de objeto de tipo. Cada "valuereference" que aparezca textualmente dentro de un "AssignedIdentifier" deberá satisfacer una de las reglas siguientes:

a) Está definida en la "AssignmentList" (lista de asignaciones) del módulo que se está definiendo, y todas las "valuereferences" (referencias de valor) que aparecen textualmente en el lado derecho del enunciado de asignación también satisfacen esta regla (regla "a") o la regla siguiente (regla "b").

b) Aparece como un "Symbol" (símbolo) en un "SymbolsFromModule" (símbolos procedentes de módulo) cuyo "AssignedIdentifier" no contiene textualmente ninguna "Valuereference".

NOTA – Se recomienda asignar un identificador de objeto de tal modo que otros puedan hacer referencia inequívocamente al módulo.

10.8 La "GlobalModuleReference" de un "SymbolsFromModule" deberá aparecer en la "ModuleDefinition" de otro módulo, con la excepción de que, si incluye un "DefinitiveIdentifier" no vacío, la "modulereference" (referencia de módulo) puede no ser la misma en los dos casos.

NOTA – Sólo deberá utilizarse una "modulereference" diferente a la utilizada en el otro módulo cuando deban importarse símbolos de dos módulos que tienen el mismo nombre (por haber sido denominados los módulos sin tener en cuenta 10.5). Cuando se utilizan otros nombres distintos, dichos nombres pueden emplearse en el cuerpo del módulo (véase 10.14).

10.9 Cuando en la referencia de un módulo se utiliza una "modulereference" y un "AssignedIdentifier" no vacío, éste deberá considerarse definitivo.

10.10 Cuando el módulo referenciado tenga un "DefinitiveIdentifier" no vacío, la "GlobalModuleReference" que referencia ese módulo no podrá tener un "AssignedIdentifier" vacío.

10.11 Cuando se seleccione la alternativa "SymbolsExported" de "Exports":

a) cada "Symbol" de "SymbolExported" deberá satisfacer una, y solamente una, de las condiciones siguientes:

   i) está definido únicamente en el módulo que se está construyendo, o

   ii) aparece exclusivamente una vez en la alternativa "SymbolsImported" de "Imports";

b) todo "Symbol" al que se haga referencia desde el exterior del módulo de manera apropiada deberá incluirse en el "SymbolsExported" y sólo estos símbolos ("Symbol"s) podrán ser referenciados desde el exterior del módulo; y

c) si no existen tales símbolos, deberá seleccionarse entonces la alternativa "empty" de "SymbolsExported" (no de "Exports").

10.12 Cuando se selecciona la alternativa "empty" (vacío) de "Exports", todo "Symbol" definido en el módulo puede ser referenciado desde otros módulos.

NOTA – La alternativa "empty" de "Exports" se incluye para asegurar la retrocompatibilidad (backward compatibility).

10.13 Los identificadores que aparecen en una "NamedNumberList", "Enumeration" o "NamedBitList" son exportados implícitamente si la referencia de tipo que los define es exportada o aparece como un componente (o un subcomponente) dentro de un tipo exportado.

10.14 Cuando se seleccione la alternativa "SymbolsImported" de "Imports":

a) Cada "Symbol" de "SymbolsFromModule" deberá estar definido en el cuerpo del módulo, o presente en la cláusula IMPORTS, del módulo denotado por la "GlobalModuleReference" en "SymbolsFromModule". La importación de un "Symbol" presente en la cláusula IMPORTS del módulo referenciado se permite únicamente si el "Symbol" aparece solamente una vez en esa cláusula y no está definido en el módulo referenciado.

   NOTA 1 – Esto no impide que el mismo nombre de símbolo definido en dos módulos diferentes se importe en otro módulo. Sin embargo, si el mismo nombre de "Symbol" aparece más de una vez en la cláusula IMPORTS del módulo A, ese nombre de "Symbol" no puede ser exportado de A para importarlo en otro módulo B.

b) Si la alternativa "SymbolExported" de "Exports" está seleccionada en la definición del módulo denotado por la "GlobalModuleReference" en "SymbolsFromModule", el "Symbol" deberá aparecer en su "SymbolsExported".

c) Sólo los símbolos ("Symbol"s) que aparezcan en la "SymbolsList" de un "SymbolsFromModule" podrán aparecer como el símbolo en cualquier "External<X>reference" que tenga la "modulereference" denotada por "GlobalModuleReference" de ese "SymbolsFromModule" (donde <X> es "value", "type", "object", "objectclass", u "objectset").

d) Si no existen tales símbolos, deberá seleccionarse entonces la alternativa "empty" de "SymbolsImported".

NOTA 2 – Una consecuencia de c) y d) es que "IMPORTS" implica que el módulo no puede contener una "External<X>Reference".

e) todos los "SymbolsFromModule" de la "SymbolsFromModuleList" deberán incluir ocurrencias de "GlobalModuleReference" de tal modo que:

i) cada "modulereference" en ellos sea diferente de cada una de las demás, y también de la "modulereference" asociada con el módulo referenciante; y

ii) el "AssignedIdentifier", cuando no esté vacío, denota valores de identificador de objeto cada uno de los cuales será diferente de cada uno de los demás, y también del valor de identificador de objeto (si existe) asociado con el módulo referenciante.

10.15 Cuando se selecciona la alternativa "empty" de "Imports", el módulo puede, aun así, referenciar "Symbol"s definidos en otros módulos mediante una "External<X>Reference".

NOTA – La alternativa "empty" de "Imports" se incluye por razones de retrocompatibilidad.

10.16 Los identificadores que aparecen en una NamedNumberList, Enumeration o NamedBitList son importados implícitamente si la referencia de tipo que los define ha sido importada o aparece como un componente (o subcomponente) dentro de un tipo exportado.

10.17 Un "Symbol" de un "SymbolsFromModule" puede aparecer en "ModuleBody" como una "Reference". El significado asociado con el "Symbol" es el que tiene en el módulo denotado por la correspondiente "GlobalModuleReference".

10.18 Cuando el "Symbol" aparezca también en una "AssigmentList" (lo cual se desaconseja), o en otro u otros varios ejemplares de "SymbolsFromModule", sólo deberá utilizarse en una "External<X>Reference". Cuando no aparezca de esta forma, se utilizará directamente como una "Reference".

10.19 Las diversas alternativas de "Assignment" se definen en la siguientes subcláusulas de esta Recomendación | Norma Internacional, excepto cuando se indica otra cosa:

Alternativa de "Assignment"	*Subcláusula definidora*
"TypeAssignment"	13.1
"ValueAssignment"	13.2
"ValueSetTypeAssignment"	13.4
"ObjectClassAssignment"	9.1 de la Rec. UIT-T X.681 | ISO/CEI 8824-2
"ObjectAssignment"	11.1 de la Rec. UIT-T X.681 | ISO/CEI 8824-2
"ObjectSetAssignment"	12.1 de la Rec. UIT-T X.681 | ISO/CEI 8824-2
"ParameterizedAssignment"	8.1 de la Rec. UIT-T X.683 | ISO/CEI 8824-4

El primer símbolo de cada "Assignment" es una de las alternativas de "Reference", que denota el nombre de referencia que se está definiendo. En una "AssignmentList" no podrá haber dos asignaciones que tengan los mismos nombres de referencia.

11 Referenciación de definiciones de tipo y valor

11.1 Las producciones tipo definido y valor definido

DefinedType ::=
Externaltypereference |
typereference |
ParameterizedType |
ParameterizedValueSetType

DefinedValue ::=
Externalvaluereference |
valuereference |
ParameterizedValue

especifican las secuencias que deberán utilizarse para referenciar definiciones de tipo y valor. El tipo identificado por un "ParameterizedType" (tipo parametrizado) y por un ParameterizedValueSetType (tipo de conjunto de valores parametrizados) y el tipo y valor identificados por un "ParameterizedValue" (valor parametrizado) se especifican en la Rec. UIT-T X.683 | ISO/CEI 8824-4.

11.2 Salvo lo especificado en 10.17, las alternativas "typereference", "valuereference", "ParameterizedType", "ParameterizedValueSetType" o "ParameterizedValue" no deberán utilizarse a menos que la referencia esté dentro del "ModuleBody" en que se ha asignado un tipo o valor (véanse 13.1 y 13.2) a la typereference o la valuereference:

11.3 La "Externaltypereference" y la "Externalvaluereference" no deberán utilizarse a menos que a la correspondiente "typereference" o "valuereference":

a) se le haya asignado un tipo o un valor, respectivamente (véanse 13.1 y 13.2), o

b) están presentes en la cláusula IMPORTS.

dentro del "ModuleBody" utilizado para definir el "modulereference" correspondiente, sólo se permitirá la referenciación de un elemento en la cláusula IMPORTS de otro módulo si el "Symbol" no aparece más que una vez en dicha cláusula.

NOTA – Esto no impide que el mismo "Symbol" definido en dos módulos diferentes sea importado en otro módulo. Sin embargo, si el mismo "Symbol" aparece más de una vez en la cláusula IMPORTS de un módulo A, ese "Symbol" no puede ser referenciado utilizando el módulo A en una referencia externa.

11.4 Una referencia externa solamente deberá utilizarse en un módulo para referenciar un ítem que está definido en un módulo diferente, y se especifica por las producciones siguientes:

Externaltypereference ::=
modulereference
"."
typereference

Externalvaluereference ::=
modulereference
"."
valuereference

NOTA – En la Rec.UIT-T X.681 | ISO/CEI 8824-2 se especifican producciones de referencia externa adicionales (ExternalClassReference, ExternalObjectReference y ExternalObjectSetReference).

11.5 Cuando el módulo referenciante se haya definido utilizando la alternativa "SymbolsImported" de "IMPORTS", la "modulereference" de la referencia externa deberá aparecer en la "GlobalModuleReference" de exactamente uno de los "SymbolsFromModule" de los "SymbolsImported". Cuando el módulo referenciante se haya definido utilizando la alternativa "empty" de "IMPORTS", la "modulereference" de la referencia externa deberá aparecer en la "ModuleDefinition" del módulo (distinto del módulo referenciante) en que está definida la "Reference".

12 Notación para soportar referencias a componentes ASN.1

12.1 También se requiere hacer referencia formal a componentes de tipos, valores, etc. ASN.1 para muchos otros fines. Uno de estos casos se da cuando es necesario, al escribir un texto, identificar un tipo específico dentro de algún módulo ASN.1. Esta subcláusula define una notación que puede utilizarse para proporcionar tales referencias.

12.2 La notación permite identificar cualquier componente de un tipo conjunto o secuencia (cuya presencia en el tipo puede ser obligatoria u opcional).

NOTA – La Rec. UIT-T X.682 | ISO/CEI 8824-3 especifica el significado de la notación en cada ejemplar de utilización. En particular, el significado de una referencia a un campo que está ausente en algunos valores del tipo circundante (enclosing type) no está dentro del ámbito de esta Recomendación | Norma Internacional.

12.3 Cualquier parte de cualquier definición de tipo ASN.1 puede ser referenciada mediante la utilización de la construcción sintáctica "AbsoluteReference":

```
AbsoluteReference ::= "@" GlobalModuleReference
                      "."
                      ItemSpec

ItemSpec ::=
                      typereference |
                      ItemId "." ComponentId

ItemId ::= ItemSpec
ComponentId ::=
                      identifier | number | "*"
```

NOTA – La producción AbsoluteReference no se utiliza en ningún otro punto de esta Recomendación | Norma Internacional. Se proporciona a los fines indicados en 12.1.

12.4 La "GlobalModuleReference" identifica un módulo ASN.1 (véase 10.1).

12.5 La "typereference" hace referencia a cualquier tipo ASN.1 definido en el módulo identificado por "GlobalModuleReference".

12.6 El "ComponentId" de cada "ItemSpec" identifica un componente del tipo que ha sido identificado por "ItemId". Será el último "ComponentId" si el componente que identifica no es un tipo conjunto, secuencia, conjunto de, secuencia de o elección.

12.7 La forma "identifier" (identificador) de "ComponentId" puede utilizarse si el "ItemId" progenitor es un tipo conjunto o secuencia y es preciso que sea uno de los identificadores ("identifier"s) del "NamedType" de la "ComponentTypeList" de ese conjunto o de esa secuencia. También puede utilizarse si el "ItemId" identifica un tipo elección, y se requiere entonces que sea uno de los identificadores de un "NamedType" de la "AlternativeTypeList" de ese tipo elección. No puede utilizarse en ninguna otra circunstancia.

12.8 La forma "number" (número) de "ComponentId" puede utilizarse únicamente si el "ItemId" es un tipo secuencia de o conjunto de. El valor del número identifica el ejemplar del tipo en la secuencia de o en el conjunto de, identificando el valor "1" el primer ejemplar del tipo. El valor cero identifica un componente tipo entero conceptual [no explícitamente presente en transferencia y denominado **iteration count** (cuenta de iteraciones)] que contiene una cuenta del número de ejemplares del tipo en la secuencia de o en el conjunto de que están presentes en el valor del tipo circundante.

12.9 La forma "*" de "ComponentId" puede utilizarse únicamente si el "ItemId" es una secuencia de o un conjunto de. Toda semántica asociada con la utilización de la forma "*" de "ComponentId" se aplica a todos los componentes de la secuencia de y del conjunto de.

NOTA – En el siguiente ejemplo:

```
M DEFINITIONS ::= BEGIN
    T ::= SEQUENCE {
        a    BOOLEAN,
        b    SET OF INTEGER
    }
END
```

los componentes de "T" podrían ser referenciados por texto fuera de un módulo ASN.1 (o en un comentario), tal como:

```
si (@M.T.b.0 es impar) entonces
    (@M.T.b.*será un entero impar)
```

que se utiliza para indicar que si el número de componentes en "b" es impar, todos los componentes de "b" deben ser impares.

13 Asignación de tipos y valores

13.1 Para asignar un tipo a una "typereference" se utiliza la notación especificada por la producción "Typeassignment":

```
TypeAssignment ::=
                      typereference
                      "::="
                      Type
```

La "typereference" no será una palabra reservada en ASN.1 (véase 9.16).

13.2 Para asignar un valor a una "valuereference" se utiliza la notación especificada por la producción "Valueassignment":

ValueAssignment ::=
valuereference
Type
"::="
Value

El "Value" que se asigne a la "valuereference" deberá ser una notación válida para el valor del tipo definido por "Type" (especificado en 13.3).

13.3 "Value" es una notación para un valor de un tipo si, o bien:

a) "Value" es una notación "BuiltinValue" para el tipo (véase 14.8); o

b) "Value" es una notación "DefinedValue" para un valor de tipo.

13.4 A una "typereference" le puede ser asignado un conjunto de valores por la notación especificada por la producción "ValueSetTypeAssignment":

ValueSetTypeAssignment ::= typereference
Type
"::="
ValueSet

Esta notación asigna a la "typereference" el tipo definido como un subtipo del tipo denotado por "Type" y que contiene exactamente los valores que están especificados en, o permitidos por, "ValueSet". La "typereference" no será una palabra reservada ASN.1 (véase 9.16) y podrá ser referenciada como un tipo. "ValueSet" se define en 13.5.

13.5 Un conjunto de valores gobernado por algún tipo será especificado por la notación "ValueSet":

ValueSet ::= "{" ElementSetSpec "}"

El conjunto de valores comprende todos los valores, de los que habrá por lo menos uno, especificado por "ElementSetSpec" (véase la cláusula 44).

14 Definición de tipos y valores

14.1 Un tipo deberá especificarse por la notación "Type":

Type ::= BuiltinType | ReferencedType | ConstrainedType

14.2 Los tipos incorporados (built-in) de ASN.1 se especifican por la notación "BuiltinType", que se define como sigue:

BuiltinType ::=
BitStringType |
BooleanType |
CharacterStringType |
ChoiceType |
EmbeddedPDVType |
EnumeratedType |
ExternalType |
InstanceOfType |
IntegerType |
NullType |
ObjectClassFieldType |
ObjectIdentifierType |
OctetStringType |
RealType |
SequenceType |
SequenceOfType |
SetType |
SetOfType |
TaggedType

Las diversas notaciones "BuiltinType" se definen en las siguientes cláusulas (de esta Recomendación | Norma Internacional, a menos que se indique otra cosa):

BitStringType	19
BooleanType	15
CharacterStringType	33
ChoiceType	26
EmbeddedPDVType	30
EnumeratedType	17
ExternalType	31
InstanceOfType	Rec. UIT-T X.681 | ISO/CEI 8824-2, Anexo C
IntegerType	16
NullType	21
ObjectClassFieldType	Rec. UIT-T X.681 | ISO/CEI 8824-2, subcláusula 14.1
ObjectIdentifierType	29
OctetStringType	20
RealType	18
SequenceType	22
SequenceOfType	23
SetType	24
SetOfType	25
TaggedType	28

14.3 Los tipos referenciados de ASN.1 se especifican por la notación "ReferencedType":

```
ReferencedType ::=
    DefinedType          |
    UsefulType           |
    SelectionType        |
    TypeFromObject       |
    ValueSetFromObjects
```

La notación "ReferencedType" proporciona otro posible medio de hacer referencia a algún otro tipo (y, en último término, a un tipo incorporado). Las diversas notaciones "ReferencedType" y la forma en que se determina el tipo a que ellas se refieren, se especifican en los siguientes lugares de esta Recomendación | Norma Internacional (a menos que se indique otra cosa):

DefinedType	11.1
UsefulType	38.1
SelectionType	27
TypeFromObject	Rec. UIT-T X.681 | ISO/IEC 8824-2, cláusula 15
ValueSetFromObjects	Rec. UIT-T X.681 | ISO/IEC 8824-2, cláusula 15

14.4 El "ConstrainedType" se define en la cláusula 42.

14.5 Esta Recomendación | Norma Internacional exige la utilización de la notación "NamedType" al especificar los componentes del tipo conjunto, del tipo secuencia y de los tipos elección. La notación para "NamedType" es:

```
NamedType ::= identifier Type
```

14.6 El "identifier" (identificador) (en la primera alternativa de "NamedType" o el "SelectionType") se utiliza para referirse de manera inequívoca a los componentes de un tipo conjunto, un tipo secuencia o un tipo elección en la notación de valor y en las constricciones de la relación de componentes (véase la Rec. UIT-T X.682 | ISO/CEI 8824-3). No forma parte del tipo y no influye en él.

14.7 Un valor de algún tipo será especificado por la notación "Value":

```
Value ::= BuiltinValue | ReferencedValue | ObjectClassFieldValue
```

NOTA – ObjectClassFieldValue se define en la Rec. X.681 | ISO/CEI 8824-2, subcláusula 14.6.

14.8 Los valores de tipo incorporados de ASN.1 pueden especificarse por la notación "BuiltinValue", definida como sigue:

```
BuiltinValue ::=
        BitStringValue          |
        BooleanValue            |
        CharacterStringValue    |
        ChoiceValue             |
        EmbeddedPDVValue        |
```

```
    EnumeratedValue          |
    ExternalValue            |
    InstanceOfValue          |
    IntegerValue             |
    NullValue                |
    ObjectIdentifierValue    |
    OctetStringValue         |
    RealValue                |
    SequenceValue            |
    SequenceOfValue          |
    SetValue                 |
    SetOfValue               |
    TaggedValue
```

Cada una de las diversas notaciones "BuiltinValue" se define en la misma cláusula que la correspondiente notación "BuiltinType", como se ha indicado en 14.2.

14.9 Los valores referenciados de ASN.1 se especifican por la notación "ReferencedValue":

```
ReferencedValue ::=
    DefinedValue |
    ValueFromObject
```

La notación "ReferencedValue" proporciona otro posible medio de referir a algún otro valor (y, en último término, a un valor incorporado). Las diversas notaciones "ReferencedValue", y la forma en que se determina el valor a que ellas se refieren, se especifican en los siguientes lugares de esta Recomendación | Norma Internacional (a menos que se indique otra cosa):

DefinedValue	11.1
ValueFromObject	Rec. UIT-T X.681 | ISO/IEC 8824-2, cláusula 15

14.10 Con independencia de si un tipo es un "BuiltinType", "ReferenceType" o "ConstrainedType", sus valores pueden ser especificados por un "BuiltinValue" o un "ReferenceValue" de ese tipo.

14.11 El valor de un tipo referenciado mediante la notación "NamedType" será definido por la notación "NamedValue":

```
NamedValue ::= identifier Value
```

donde el "identifier" (identificador) (en la primera alternativa de "NamedType" o "SelectionType" es el mismo que se utiliza en la notación "NamedType".

NOTA – El "identifier" forma parte de la notación, no del valor propiamente dicho. Se utiliza para referirse inequívocamente a los componentes de un tipo conjunto, un tipo secuencia o un tipo elección.

15 Notación para el tipo booleano

15.1 El tipo booleano (boolean) (véase 3.8.7) será referenciado por la notación "BooleanType":

```
BooleanType ::= BOOLEAN
```

15.2 El rótulo para tipos definidos por esta notación es de clase universal, número 1.

15.3 El valor de un tipo booleano (véase 3.8.55 y 3.8.27) será definido por la notación "BooleanValue":

```
BooleanValue ::= TRUE | FALSE
```

16 Notación para el tipo entero

16.1 El tipo entero (integer) (véase 3.8.29) será referenciado por la notación "IntegerType":

```
IntegerType ::=
            INTEGER                 |
            INTEGER "{" NamedNumberList "}"

NamedNumberList ::=
            NamedNumber             |
            NamedNumberList "," NamedNumber
```

```
NamedNumber ::=
        identifier "(" SignedNumber ")"        |
        identifier "(" DefinedValue ")"

SignedNumber ::= number | "-" number
```

16.2 La segunda alternativa de "SignedNumber" no se utilizará si el "number" (número) es cero.

16.3 La "NamedNumberList" no es significativa en la definición de un tipo. Se utiliza solamente en la notación de valor especificada en 16.9.

16.4 La "valuereference" en "DefinedValue" deberá ser de tipo entero (integer).

NOTA – Puesto que un "identifier" no puede utilizarse para especificar el valor asociado con "NamedNumber", el "DefinedValue" nunca puede interpretarse erróneamente como un "IntegerValue". Por eso, en el caso siguiente:

```
a INTEGER ::= 1
T1 ::= INTEGER { a(2) }
T2 ::= INTEGER { a(3), b(a) }
c T2 ::= b
d T2 ::= a
```

"c" denota el valor 1, ya que no puede ser una referencia a la segunda ni a la tercera ocurrencia de "a", y "d" denota el valor 3.

16.5 El valor de cada "SignedNumber" o "DefinedValue" que aparece en la "NamedNumberList" deberá ser diferente, y representa un valor distinguido del tipo entero.

16.6 Cada "identifier" (identificador) que aparece en la "NamedNumberList" deberá ser diferente.

16.7 El orden de las secuencias "NamedNumber" en "NamedNumberList" no es significativo.

16.8 El rótulo para tipos definidos por esta notación es de clase universal, número 2.

16.9 El valor de un tipo entero será definido por la notación "IntegerValue":

```
IntegerValue ::=
        SignedNumber |
        identifier
```

16.10 El "identifier" del "IntegerValue" será uno de los "identifier"s (identificadores) del "IntegerType" con el que está asociado el valor, y representará el número correspondiente.

NOTA – Cuando se referencia un valor entero para el cual se ha definido un "identifier", deberá preferirse la utilización de la forma "identifier" de "IntegerValue".

17 Notación para el tipo enumerado

17.1 El tipo enumerado (enumerated) (véase 3.8.24) será referenciado por la notación "EnumeratedType":

```
EnumeratedType ::=
        ENUMERATED "{" Enumeration "}"

Enumeration ::=
        EnumerationItem | EnumerationItem "," Enumeration

EnumerationItem ::=
        identifier | NamedNumber
```

NOTAS

1 Cada valor de un "EnumeratedType" tiene un identificador que está asociado con un entero distinto. Sin embargo, no se espera que los valores propiamente dichos tengan una semántica de entero. La especificación de la alternativa "NamedNumber" de "EnumerationItem" proporciona el control de la representación del valor, a fin de facilitar las extensiones compatibles.

2 Los valores numéricos dentro de los "NamedNumber" de la "Enumeration" no tienen que estar ordenados ni ser contiguos.

17.2 Para cada "NamedNumber", el "identifier" y el "SignedNumber" serán distintos de todos los demás "identifier"s y de los "SignedNumber"s de la "Enumeration". Las subcláusulas 16.2 y 16.4 se aplican también a cada "NamedNumber".

17.3 A cada "EnumerationItem" (de un "EnumeratedType") que sea un "identifier" se le asigna un número entero no negativo distinto; estos enteros serán sucesivos. Con este fin se asignan los enteros sucesivos comenzando por 0, pero se excluyen los que han sido empleados en "EnumerationItem"s y son "NamedNumber"s.

NOTA – Un valor entero está asociado con un "EnumerationItem" para ayudar en la definición de las reglas de codificación. De no ser así, no se utiliza en la especificación ASN.1.

17.4 El tipo enumerado tiene un rótulo de clase universal, número 10.

17.5 El valor de un tipo enumerado será definido por la notación "EnumeratedValue":

EnumeratedValue ::= identifier

17.6 El "identifier" de "EnumeratedValue" será igual al de un "identifier" de la secuencia "EnumeratedType" con la que está asociado el valor.

18 Notación para el tipo real

18.1 El tipo real (véase 3.8.41) será referenciado por la notación "RealType":

RealType ::= REAL

18.2 El tipo real tiene un rótulo de clase universal número 9.

18.3 Los valores del tipo real son los valores PLUS-INFINITY y MINUS-INFINITY junto con los números reales que pueden ser especificados por la siguiente fórmula que consta de tres enteros, M, B y E:

$$M \times B^E$$

donde M se denomina la mantisa, B la base y E el exponente.

18.4 El tipo real tiene un tipo asociado que se utiliza para dar precisión a la definición de los valores abstractos del tipo real y que se utiliza también para sustentar las notaciones de valor y subtipo del tipo real.

NOTA – Las reglas de codificación pueden definir un tipo diferente que se utiliza para especificar codificaciones, o puede especificar codificaciones sin referencia a los tipos asociados. En particular, la codificación en BER y PER proporciona una codificación decimal codificado en binario (BCD, *binary-coded decimal*) si la base es 10 y una codificación que permite la transformación eficiente hacia y desde las representaciones de coma flotante de los soportes físicos si la base es 2.

18.5 El tipo asociado para la definición de valores y a efectos de subtipificación es (con comentarios normativos):

```
SEQUENCE {
    mantissa  INTEGER,
    base INTEGER (2|10),
    exponent  INTEGER
        -- El número real asociado es la "mantisa"
        -- multiplicado por la "base" elevada a la potencia del "exponente"
}
```

NOTAS

1 Los valores representados por base 2 y por base 10 se consideran valores abstractos distintos incluso si equivalen al mismo valor en números reales y pueden llevar semánticas de aplicación diferentes.

2 La notación "REAL (WHIT COMPONENTS { ... , base (10)})" puede utilizarse para restringir el conjunto de valores a valores abstractos en base 10 (y de manera similar para valores abstractos en base 2).

3 Este tipo es capaz de llevar una representación finita exacta de cualquier número que pueda ser almacenado en soportes físicos típicos de coma flotante y de cualquier número que tenga una representación finita decimal en caracteres.

18.6 El valor de un tipo real se definirá por la notación "RealValue":

```
RealValue ::=
        NumericRealValue | SpecialRealValue

NumericRealValue ::=  0         |
        SequenceValue   -- Value of the associated sequence type

SpecialRealValue ::=
        PLUS-INFINITY | MINUS-INFINITY
```

La forma "0" deberá utilizarse para valores cero, y la forma alternativa para "NumericRealValue" no deberá utilizarse para valores cero.

19 Notación para el tipo cadena de bits

19.1 El tipo cadena de bits (bitstring) (véase 3.8.6) será referenciado por la notación "BitStringType":

```
BitStringType ::=
        BIT STRING
        BIT STRING "{" NamedBitList "}"

NamedBitList ::=
        NamedBit        |
        NamedBitList "," NamedBit

NamedBit ::=
        identifier "(" number ")" |
        identifier "(" DefinedValue ")"
```

19.2 El primer bit de una cadena de bits se denomina **bit cero** (**bit zero**). El último bit de una cadena de bits se denomina bit final o **bit de cola** (**trailing bit**).

NOTA – Esta terminología se utiliza en la especificación de la notación de valor y en las reglas de codificación.

19.3 El "DefinedValue" deberá ser una referencia a un valor no negativo de tipo entero.

19.4 El valor de cada "number" (número) o "DefinedValue" que aparezca en la "NamedBitList" deberá ser diferente y será el número (de la posición) de un bit distinguido en un valor bistring (cadena de bits).

19.5 Cada "identifier" (identificador) que aparece en la "NamedBitList" deberá ser diferente.

NOTAS

1 El orden de las secuencias de producción "NamedBit" de la "NamedBitList" no es significativo.

2 Puesto que un "identifier" que aparezca dentro de la "NamedBitList" no puede ser utilizado para especificar el valor asociado con un "NamedBit", el "DefinedValue" nunca podrá interpretarse erróneamente como un "IntegerValue". Por eso, en el caso siguiente:

```
a  INTEGER ::= 1

T1 ::= INTEGER { a(2) }

T2 ::= BIT STRING { a(3), b(a) }
```

la última ocurrencia de "a" denota el valor 1, ya que no puede ser una referencia a la segunda ni a la tercera ocurrencia de "a".

19.6 La presencia de una "NamedBitList" no tiene efecto alguno en el conjunto de valores de este tipo. Están permitidos valores que tengan bits 1 distintos de los bits nombrados.

19.7 Cuando se utiliza una "NamedBitList" en la definición de un tipo cadena de bits (bitstring), las reglas de codificación ASN.1 pueden añadir (o eliminar) libremente, de manera arbitraria, muchos bits 0 de cola a (o de) los valores que están siendo codificados o decodificados. Los diseñadores de aplicaciones deben asegurarse, por consiguiente, de que no se asocian semánticas diferentes con unos valores que sólo difieren en el número de bits 0 de cola.

19.8 Este tipo tiene un rótulo de clase universal, número 3.

19.9 El valor de un tipo cadena de bits (bitstring) se definirá por la notación "BitStringValue":

```
BitStringValue ::=
        bstring                         |
        hstring                         |
        "{" IdentifierList "}"          |
        "{" "}"

IdentifierList ::=
        identifier          |
        IdentifierList "," identifier
```

19.10 Cada "identifier" de "BitStringValue" deberá ser el mismo que un "identifier" de la secuencia "BitStringType" con la que está asociado el valor.

19.11 La notación "BitStringValue" denota un valor cadena de bits (bitstring) con unos en las posiciones de bits especificadas por los números correspondientes a los "identifier" y con ceros en todas las demás posiciones de bits.

NOTA – La secuencia de producción "{" "}" se utiliza para denotar la cadena de bits que no contiene ningún bit.

19.12 Al especificar las reglas de codificación de una cadena de bits, los bits serán referenciados por los términos **primer bit** y **bit de cola**, en donde el primer bit tiene el número cero (véase 19.2).

19.13 Cuando se utiliza la notación "bstring", el **primer bit** está a la izquierda y el **bit de cola** está a la derecha.

19.14 Cuando se utiliza la notación "hstring", el bit más significativo de cada dígito hexadecimal corresponde al bit situado más a la izquierda en la cadena de bits.

NOTA – Esta notación no constriñe en forma alguna la manera en que las reglas de codificación colocan una cadena de bits en octetos para la transferencia.

19.15 La notación "hstring" no debe utilizarse a menos que el valor cadena de bits (bitstring) esté constituido por un múltiplo de cuatro bits.

EJEMPLO

'A8A'H

y

'1010100110001010'B

son dos notaciones posibles para el mismo valor cadena de bits (bitstring). Si el tipo se definió utilizando una "NamedBitList", el cero final (único) no forma parte del valor que, de esta forma, tiene una longitud de 15 bits. Si el tipo se definió sin una "NamedBitList", el cero final sí forma parte del valor, que tiene entonces una longitud de 16 bits.

20 Notación para el tipo cadena de octetos

20.1 El tipo cadena de octetos (octetstring) (véase 3.8.37) será referenciado por la notación "OctetStringType":

OctetStringType ::= OCTET STRING

20.2 Este tipo tiene un rótulo que es de clase universal, número 4.

20.3 El valor de un tipo cadena de octetos se definirá por la notación: "OctetStringValue":

OctetStringValue ::=
bstring |
hstring

20.4 Al especificar las reglas de codificación para una cadena de octetos, los octetos son referenciados por los términos **first octet** (primer octeto) y **trailing octet** (octeto final o de cola), y los bits en un octeto son referenciados por los términos **most significant bit** (bit más significativo) y **least significant bit** (bit menos significativo).

20.5 Cuando se use la notación "bstring", el bit situado más a la izquierda será el bit más significativo del primer octeto. Si la "bstring" no es múltiplo de ocho bits, se interpretará como si contuviera bits postreros cero adicionales para hacer el próximo múltiplo de ocho.

20.6 Cuando se use la notación "hstring", el dígito hexadecimal más a la izquierda será el semiocteto más significativo del primer octeto.

20.7 Si la "hstring" no tiene un número par de dígitos hexadecimales, se interpretará como si contuviera un solo dígito hexadecimal cero postrero adicional.

21 Notación para el tipo nulo

21.1 El tipo nulo (null) (véase 3.8.32) será referenciado por la notación "NullType":

NullType ::= NULL

21.2 El rótulo para este tipo es de clase universal, número 5.

21.3 El valor del tipo nulo se referenciará por la notación "NullValue":

NullValue ::= NULL

22 Notación para tipos secuencia

22.1 La notación para definir un tipo secuencia (sequence) (véase 3.8.45) a partir de otros tipos será "SequenceType":

```
SequenceType ::=
        SEQUENCE "{" ComponentTypeList "}"   |
        SEQUENCE "{"  "}"

ComponentTypeList ::=
        ComponentType                        |
        ComponentTypeList  ","  ComponentType

ComponentType ::=
        NamedType                            |
        NamedType  OPTIONAL                  |
        NamedType  DEFAULT  Value            |
        COMPONENTS  OF  Type
```

22.2 Cuando aparece una producción "ComponentTypeList" dentro de la definición de un módulo para el que se ha seleccionado rotulación automática (véase 10.3), y ninguna de las ocurrencias de "NamedType" en cualquiera de las tres primeras alternativas para "ComponentType" contiene un "TaggedType", se selecciona la transformación de rotulación automática para la totalidad de la "ComponentTypeList"; de no ser así, no se selecciona.

NOTAS

1 La utilización de la notación "TaggedType" dentro de la definición de la lista de componentes de un tipo secuencia da el control de los rótulos al especificador, al contrario de lo que ocurre con la asignación automática por el mecanismo de rotulación automática. Por ello, en el siguiente caso:

T ::= SEQUENCE { a INTEGER, b [1] BOOLEAN, c OCTET STRING }

no se aplica rotulación automática a la lista de componentes a, b, c, incluso si esta definición de tipo secuencia T se produce dentro de un módulo para el que ha sido seleccionada la rotulación automática.

2 Sólo las ocurrencias de la producción "ComponentTypeList" dentro de un módulo en el que se ha seleccionado la rotulación automática son candidatas a la transformación por rotulación automática.

22.3 La decisión de aplicar la transformación de rotulación automática se toma individualmente para cada ocurrencia de la "ComponentTypeList" y *con anterioridad* a la transformación COMPONENTS OF especificada por 22.4. Sin embargo, tal como se especifica en 22.6 la transformación de rotulación automática, si se aplica, lo es *después* de la transformación COMPONENTS OF.

NOTA – Lo anterior tiene como efecto que la aplicación de rótulos automáticos queda suprimida por los rótulos presentes de manera explícita en la "ComponentTypeList", pero no por los rótulos presentes en el "Type" que sigue a "COMPONENTS OF".

22.4 El "Type" de la cuarta alternativa del "ComponentType" deberá ser un tipo secuencia. La notación "COMPONENTS OF Type" deberá utilizarse para definir la inclusión, en este punto de la lista de componentes, de todos los tipos de componentes del tipo referenciado.

NOTA – Esta transformación se completa lógicamente antes de la cumplimentación de los requisitos de las siguientes subcláusulas.

22.5 Para cada serie de una o más ocurrencias consecutivas de "ComponentType" marcadas como OPTIONAL o DEFAULT, los rótulos de esos "ComponentType" y de cualquier "ComponentType" que siga inmediatamente, deberán ser distintos (véase la cláusula 28). Si se seleccionó rotulación automática, la exigencia de que los rótulos sean distintos sólo es aplicable después de que se haya efectuado la rotulación automática y se satisfacerá siempre, si se ha aplicado la rotulación automática.

22.6 La transformación de rotulación automática de una ocurrencia de "ComponentTypeList" se efectúa lógicamente *después de* la transformación especificada por la subcláusula 22.4, pero solamente si la subcláusula 22.2 determina que debe aplicarse a esa ocurrencia de "ComponentTypeList". La transformación de rotulación automática repercute en cada "ComponentType" de la "ComponentTypeList" al sustituir el "Type" que figura originalmente en la producción "NamedType" por una ocurrencia del "TaggedType" de sustitución especificado en 22.7.

22.7 El "TaggedType" de sustitución se especifica de la siguiente manera:

a) la notación de "TaggedType" de sustitución utiliza la alternativa "Tag Type";

b) la "Class" del "TaggedType" de sustitución es vacío (es decir, la rotulación es específica del contexto);

c) el "ClassNumber" del "TaggedType" de sustitución es rótulo de valor cero para el primer ComponentType, uno para el segundo, y así sucesivamente, procediendo con números de rótulo crecientes;

d) el "Type" del "TaggedType" de sustitución es el "Type" original que se sustituye.

NOTAS

1 Las reglas que rigen la especificación de la rotulación implícita o de la rotulación explícita de los "TaggedTypes" de sustitución se indican en 28.6. La rotulación automática es siempre rotulación implícita, a menos que el "Type" sea un tipo elección o una notación de tipo abierto o una "DummyReference" (referencia ficticia) (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4, subcláusula 8.3), en cuyo caso se trata de rotulación explícita.

2 Una vez satisfecha 22.6, los rótulos de los componentes quedan determinados por completo y no son modificados incluso cuando el tipo secuencia es referenciado en la definición de un componente dentro de otra "ComponentTypeList" para el que es aplicable la transformación de rotulación automática. Así pues, en el caso siguiente:

T ::= SEQUENCE { a Ta, b Tb, c Tc }

E ::= SEQUENCE { f1 E1, f2 T, f3 E3 }

los rótulos atribuidos a *a*, *b* y *c* no resultan afectados por la posible rotulación automática aplicada a los componentes de E.

3 Cuando aparece un tipo secuencia como el "Type" en "COMPONENTS OF Type", cada ocurrencia de "ComponentType" en él es reproducida por la aplicación de la subcláusula 22.4 antes de la posible aplicación de la rotulación automática al tipo secuencia referenciador. Así pues, en el caso siguiente:

T ::= SEQUENCE { a Ta, b SEQUENCE { b1 T1, b2 T2, b3 T3}, c Tc }

W ::= SEQUENCE { x Wx, COMPONENTS OF T, y Wy }

los rótulos de a, b y c dentro de T no es preciso que sean los mismos que los rótulos de a, b y c dentro de W si W se ha definido en un entorno de rotulación automática, pero los rótulos de b1, b2 y b3 son los mismos tanto en T como en W. En otras palabras, la transformación de rotulación automática sólo se aplica una vez a una "ComponentTypeList" dada.

4 La subtipificación no influye en la rotulación automática.

5 En presencia de rotulación automática, la inserción de componentes nuevos puede dar lugar a cambios en otros componentes, debido al efecto lateral de la modificación de los rótulos.

22.8 Si están presentes "OPTIONAL" o "DEFAULT", puede omitirse el valor correspondiente en un valor del tipo nuevo.

22.9 Si ocurre "DEFAULT", la omisión de un valor para ese tipo deberá equivaler exactamente a la inserción del valor definido por "Value", que será una notación de valor para un valor del tipo definido por "Type" en la secuencia de producción "NamedType".

22.10 Los "identifier"s (identificadores) de todas las secuencias de producción "NamedType" de la "ComponentTypeList" (junto con los obtenidos por expansión de COMPONENTS OF) serán distintos.

22.11 Todos los tipos secuencia (sequence) tienen un rótulo que es de clase universal, número 16.

NOTA – Los tipos secuencia de (sequence of) tienen el mismo rótulo que los tipos secuencia (véase 23.2).

22.12 La notación para la definición de un valor de un tipo secuencia será "SequenceValue":

SequenceValue ::=
"{" ComponentValueList "}" |
"{" "}"

ComponentValueList ::=

NamedValue |
ComponentValueList "," NamedValue

22.13 La notación "{" "}" sólo se utilizará si:

a) todas las secuencias "ComponentType" del "SequenceType" están marcadas "DEFAULT" u "OPTIONAL", y todos los valores han sido omitidos; o

b) la notación de tipo era "SEQUENCE{}".

22.14 Deberá haber un "NamedValue" para cada "NamedType" que no esté marcado "OPTIONAL" o "DEFAULT", y los valores deberán estar en el mismo orden que las secuencias "NamedType" correspondientes.

23 Notación para tipos secuencia de

23.1 La notación para definir un tipo secuencia de (sequence-of) (véase 3.8.46) a partir de otro tipo será "SequenceOfType".

SequenceOfType ::= SEQUENCE OF Type

23.2 Todos los tipos secuencia de tienen un rótulo que es de clase universal, número 16.

NOTA – Los tipos secuencia (sequence) tienen el mismo rótulo que los tipos sequence-of (véase 22.11).

23.3 La notación para definir un valor de un tipo secuencia de será "SequenceOfValue":

SequenceOfValue ::= "{" ValueList "}" | "{" "}"

ValueList ::=
Value |
ValueList "," Value

La notación "{" "}" se utiliza cuando el SequenceOfValue es una lista vacía.

23.4 Cada "Value" de la "ValueList" deberá ser del tipo especificado en el "SequenceOfType".

NOTA – Puede darse un significado semántico al orden de estos valores.

24 Notación para tipos conjunto

24.1 La notación para definir un tipo conjunto (set) (véase 3.8.47) a partir de otros tipos será "SetType":

SetType ::=
SET "{" ComponentTypeList "}" |
SET "{" "}"

"ComponentTypeList" se especifica en 22.1.

24.2 El tipo de la cuarta alternativa del "ComponentType" (véase 22.1) deberá ser un tipo conjunto. La notación "COMPONENTS OF Type" deberá utilizarse, en este punto de la "ComponentTypeList", para definir la inclusión de todas las secuencias "ComponentType" que aparecen en el tipo referenciado.

NOTA – Esta notación se realiza lógicamente antes de la cumplimentación de los requisitos de las siguientes cláusulas.

24.3 Todos los tipos "ComponentType" de un tipo conjunto deberán tener rótulos diferentes. (Véase la cláusula 28.)

NOTA – Cuando el "TagDefault" del módulo en que aparece esta notación es "AUTOMATIC TAGS", esto se consigue, cualesquiera que sean los "ComponentType", como resultado de la aplicación de 22.6.

24.4 Las subcláusulas 22.2 y 22.6 a 22.10 se aplican también a los tipos set.

24.5 Todos los tipos conjunto tienen un rótulo que es de clase universal, número 17.

NOTA – Los tipos conjunto de tienen el mismo rótulo que los tipos set (véase 25.2).

24.6 El orden de los valores en un tipo conjunto no debe tener asociado ningún significado.

24.7 La notación para definir un valor de un tipo conjunto será "SetValue":

SetValue ::= "{" ComponentValueList "}" | "{" "}"

"ComponentValueList" se especifica en 22.12.

24.8 El "SetValue" será solamente "{" "}" si:

a) todas las secuencias "ComponentType" en el "SetType" están marcadas "DEFAULT" u "OPTIONAL", y todos los valores han sido omitidos; o

b) la notación de tipo era "SET{}".

24.9 Deberá haber un "NamedValue" para cada "NamedType" que no esté marcado "OPTIONAL" o "DEFAULT".

NOTA – Estos "NamedValues" pueden aparecer en cualquier orden.

25 Notación para tipos conjunto de

25.1 La notación para definir un tipo conjunto de (set of) (véase 3.8.48) a partir de otro tipo será "SetOfType".

SetOfType ::=
SET OF Type

25.2 Todos los tipos conjunto de tienen un rótulo que es de clase universal, número 17.

NOTA – Los tipos conjunto tienen el mismo rótulo que los tipos conjunto de (véase 24.5).

25.3 La notación para definir un valor de un tipo conjunto de será "SetOfValue":

SetOfValue ::= "{" ValueList "}" | "{" "}"

"ValueList" se especifica en 23.3.

La notación "{" "}" se utiliza cuando el SetOfValue es una lista vacía.

25.4 Cada secuencia "Value" de la "ValueList" deberá ser la notación de un valor del "Type" especificado en el "SetOfType".

NOTAS

1 No debe darse significado semántico al orden de estos valores.

2 No es necesario que las reglas de codificación conserven el orden de estos valores.

3 El tipo conjunto de no es un conjunto matemático de valores, para "SET OF INTEGER", los valores "{ 1 }" y "{ 1 1 }" son distintos.

26 Notación para tipos elección

26.1 La notación para definir un tipo elección (choice) (véase 3.8.13) a partir de otros tipos será "ChoiceType":

ChoiceType ::= CHOICE "{" AlternativeTypeList "}"

AlternativeTypeList ::=
NamedType |
AlternativeTypeList "," NamedType

NOTA – T ::= CHOICE { a A } y A no son el mismo tipo y pueden estar codificados diferentemente.

26.2 Los tipos definidos en la "AlternativeTypeList" deberán tener rótulos distintos (véase la cláusula 28).

NOTA – Cuando el "TagDefault" del módulo en que aparece esta notación es "AUTOMATIC TAGS" los rótulos se hacen distintos como resultado de la aplicación de 22.6.

26.3 Cuando la producción "AlternativeTypeList" ocurre dentro de la definición de un módulo para el que se ha seleccionado rotulación automática (véase 10.3), y ninguna de las ocurrencias de "NamedTyped" en ella contiene un "Type" que sea una ocurrencia de "TaggedType", se selecciona la transformación de rotulación automática para la totalidad de la "AlternativeTypeList"; de no ser así, no se selecciona. Cuando se selecciona, se aplica la transformación de rotulación automática de una "AlternativeTypeList" a cada "NamedType" de la "AlternativeTypeList" sustituyendo cada "Type", originalmente en la producción "NamedType", por una ocurrencia del "TaggedType" de sustitución especificado en 22.7.

26.4 El tipo elección contiene valores cuyos rótulos no son todos iguales. (El rótulo depende de la alternativa que aportó el valor del tipo elección.)

26.5 Cuando este tipo se emplea en un lugar en que la presente Recomendación | Norma Internacional exige la utilización de tipos con rótulos distintos (véanse 22.5, 24.3 y 26.2) deben tenerse en cuenta en dicha exigencia todos los posibles rótulos de valores del tipo elección.

Los ejemplos siguientes, en los que se supone que "TagDefault" no es "AUTOMATIC TAGS", ilustran ese requisito.

EJEMPLOS

1

```
A ::= CHOICE
        {b   B,
         c   NULL}

B ::= CHOICE
        {d   [0] NULL,
         e   [1] NULL}
```

2

```
A ::= CHOICE
        {b   B,
         c   C}

B ::= CHOICE
        {d   [0] NULL,
         e   [1] NULL}

C ::= CHOICE
        {f   [2] NULL,
         g   [3] NULL}
```

3 (INCORRECT)

```
A ::= CHOICE
        {b   B,
         c   C}

B ::= CHOICE
        {d   [0] NULL,
         e   [1] NULL}

C ::= CHOICE
        {f   [0] NULL,
         g   [1] NULL}
```

Los ejemplos 1 y 2 presentan una utilización correcta de la notación. La utilización presentada en el ejemplo 3 es incorrecta porque los rótulos de los tipos d y f, y e y g, son idénticos.

26.6 Los "identifier"(identificadores) de todos los "NamedTypes" de la "AlternativeTypeList" deberán ser diferentes de los de otros "NamedTypes" de esa lista.

26.7 La notación para definir el valor de un tipo elección será "ChoiceValue":

```
ChoiceValue ::= identifier ":" Value
```

26.8 "Value" deberá ser una notación para un valor del tipo de la "AlternativeTypeList" que esté nombrado por el "identifier".

27 Notación para tipos selección

27.1 La notación para definir un tipo selección (selection) (véase 3.8.44) será "SelectionType":

```
SelectionType ::= identifier "<" Type
```

donde "Type" denota un tipo choice, e "identifier" es el de algún "NamedType" que aparece en la "AlternativeTypeList" de ese tipo elección.

27.2 Cuando el "SelectionType" se utiliza como un "NamedType", el "identifier" se empleará como el "identifier" de ese "NamedType".

27.3 Cuando el "SelectionType" se utiliza como un "Type", se retiene el "identifier" y el tipo denotado es el de la alternativa seleccionada.

27.4 La notación para un valor de un tipo selection será la notación para un valor del tipo referenciado por el "SelectionType".

28 Notación para tipos rotulado

Un tipo rotulado (tagged) (véase 3.8.53) es un tipo nuevo que es isomórfico con un tipo antiguo, pero que tiene un rótulo (tag) diferente. El tipo rotulado se utiliza principalmente donde esta Recomendación | Norma Internacional requiera el empleo de tipos con rótulos distintos (véanse 22.5, 24.3, 26.2, y 26.4). Mediante el empleo de un "TagDefault" de "AUTOMATIC TAGS" en un módulo puede conseguirse esto sin que aparezca explícitamente en ese módulo una notación de tipo rotulado.

NOTA – Cuando un protocolo determina que pueden transmitirse valores de varios tipos de datos en cualquier momento, pueden necesitarse rótulos distintos para hacer posible que el receptor decodifique correctamente el valor.

28.1 La notación para un tipo rotulado será "TaggedType":

```
TaggedType ::=
        Tag Type                    |
        Tag IMPLICIT Type           |
        Tag EXPLICIT Type

Tag ::= "[" Class ClassNumber "]"

ClassNumber ::=
        number |
        DefinedValue
```

```
Class ::=
        UNIVERSAL       |
        APPLICATION     |
        PRIVATE         |
        empty
```

28.2 La "valuereference" de "DefinedValue" deberá ser de tipo entero y se le asignará un valor no negativo.

28.3 El nuevo tipo es isomórfico con el tipo antiguo, pero tiene un rótulo con la clase "Class" y el número "ClassNumber", a menos que la "Class" sea "empty" (vacío) cuando el rótulo es de clase específica del contexto, número "ClassNumber".

28.4 La "Class" no será "UNIVERSAL" salvo para los tipos definidos en esta Recomendación | Norma Internacional.

NOTAS

1 El uso de rótulos de la clase universal es aprobado cada cierto tiempo por la ISO y el UIT-T.

2 La subcláusula F.2.12 contiene directrices y sugerencias sobre la utilización estilística de las clases de rótulos.

28.5 Toda aplicación de rótulos es rotulación implícita o rotulación explícita. La rotulación implícita indica, para aquellas reglas de codificación que proporciona la opción, que la identificación explícita del rótulo del "Type" del "TaggedType" no se necesita durante la transferencia.

NOTA – Puede ser útil retener el rótulo antiguo cuando sea de clase universal y, por lo tanto, identifique sin ambigüedad el tipo antiguo sin conocer la definición ASN.1 del tipo nuevo. La transferencia mínima de octetos, sin embargo, se consigue normalmente con el uso de IMPLICIT. En la Rec. UIT-T X.690 | ISO/CEI 8825-1 se da un ejemplo de una codificación utilizando IMPLICIT.

28.6 La construcción de la rotulación especifica rotulación explícita si se cumple cualquiera de los siguientes requisitos:

a) se utiliza la alternativa "Tag EXPLICIT Type";

b) se utiliza la alternativa "Tag Type" y el valor de "TagDefault" para el módulo es "EXPLICIT TAGS" o está vacío;

c) se utiliza la alternativa "Tag Type" y el valor de "TagDefault" para el módulo es "IMPLICIT TAGS" o "AUTOMATIC TAGS", pero el tipo definido "Type" es un tipo choice (elección), un tipo open (abierto) o una "DummyReference" (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4, subcláusula 8.3).

En todos los demás casos, la construcción de la rotulación especifica rotulación implícita.

28.7 Si la "Class" es "empty" (vacío), no hay otras restricciones a la utilización del "Tag" que las implicadas por los requisitos de rótulos distintos de 22.5, 24.3 y 26.2.

28.8 La alternativa "IMPLICIT" no deberá utilizarse si el tipo definido por "Type" es un tipo choice (elección) o un tipo open (abierto) o una "DummyReference" (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4, subcláusula 8.3).

28.9 La notación para un valor de un "TaggedType" será "TaggedValue":

```
TaggedValue ::= Value
```

donde "Value" es una notación para un valor del "Type" en el "TaggedType".

NOTA – El "Tag" no aparece en esta notación.

29 Notación para el tipo identificador de objeto

29.1 El tipo identificador de objeto (object identifier) (véase 3.8.36) será referenciado por la notación "ObjectIdentifierType":

```
ObjectIdentifierType ::=
            OBJECT IDENTIFIER
```

29.2 Este tipo tiene un rótulo que es de clase universal, número 6.

29.3 La notación de valor para un identificador de objeto será "ObjectIdentifierValue":

```
ObjectIdentifierValue ::=
        "{" ObjIdComponentList "}"      |
        "{" DefinedValue ObjIdComponentList "}"
```

```
ObjIdComponentList ::=
        ObjIdComponent                  |
        ObjIdComponent ObjIdComponentList

ObjIdComponent ::=      NameForm        |
                        NumberForm      |
                        NameAndNumberForm

NameForm ::= identifier

NumberForm ::= number | DefinedValue

NameAndNumberForm ::=
        identifier "(" NumberForm ")"
```

29.4 La "valuereference" en "DefinedValue" de "NumberForm" será de tipo integer, y se le asignará un valor no negativo.

29.5 La "valuereference" en "DefinedValue" de "ObjectIdentifierValue" será de tipo identificador de objeto.

29.6 La "NameForm" se utilizará solamente para los componentes de identificador de objeto cuyo valor numérico e identificador están especificados en los Anexos B a D, y serán uno de los identificadores especificados en los Anexos B a D.

29.7 El "number" en la "NumberForm" será el valor numérico asignado al componente identificador de objeto.

29.8 Se especificará el "identifier" en el "NameAndNumberForm" cuando se asigne un valor numérico al componente identificador de objeto.

NOTA – Las autoridades que atribuyen valores numéricos a componentes de identificador de objeto figuran en los anexos a esta Recomendación | Norma Internacional.

29.9 La semántica de un valor de object identifier se define por un "**object identifier tree**" (árbol de identificador de objeto). Un árbol identificador de objeto es un árbol cuya raíz corresponde a esta Recomendación | Norma Internacional y cuyos vértices corresponden a autoridades administrativas responsables de la atribución de arcos desde ese vértice. Cada arco del árbol está etiquetado por un componente identificador de objeto que es un valor numérico. A cada objeto se le atribuirá exactamente un vértice (normalmente una hoja) para ser identificado, y ningún otro objeto de información (del mismo tipo o de uno diferente) es atribuido a ese mismo vértice. Así pues, un objeto está unívoca e inequívocamente identificado por la secuencia de valores numéricos (componentes identificador de objeto) que etiquetan los arcos del trayecto que va desde la raíz al vértice atribuido al objeto.

NOTA – Los valores de identificador de objeto contienen al menos dos componentes identificador de objeto, tal como se especifica en los Anexos B a D.

29.10 Semánticamente un valor de identificador de objeto es una lista ordenada de valores componentes identificador de objeto. Empezando con la raíz del árbol de identificador de objeto, cada valor de componente identificador de objeto identifica un arco del árbol de identificador de objeto. El último valor de componente identificador de objeto identifica un arco que conduce a un vértice al que se ha asignado un objeto. Es este objeto el que es identificado por el valor de identificador de objeto. La parte significativa del componente identificador de objeto es la "NameForm" o la "NumberForm" a la que él se reduce y que proporciona el valor numérico para el componente identificador de objeto.

NOTA – En general, un objeto es una clase de información [por ejemplo el formato de file (fichero)] más bien que un elemento de tal clase (por ejemplo un fichero individual). Es pues a la clase de información (definida por alguna especificación referenciable), más bien que la información en sí misma, a la que se asigna un lugar en el árbol.

29.11 Cuando el "ObjectIdentifierValue" incluye un "DefinedValue", la lista de componentes identificador de objeto a la que se refiere, es prefijada a los componentes explícitamente presentes en el valor.

NOTA – Se recomienda que, cuando una Recomendación, o una Norma Internacional u otro documento, asigne valores de tipo OBJECT IDENTIFIER a objetos de información, haya un apéndice o anexo que recapitule las asignaciones hechas en el mismo. Se recomienda también que una autoridad que asigne valores de tipo OBJECT IDENTIFIER a un objeto de información asigne también valores de tipo ObjectDescriptor (véase 4.1) a ese objeto de información.

EJEMPLOS

Con identificadores asignados según se especifica en el Anexo B, los valores:

{ iso standard 8571 pci (1) }

y

{ 1 0 8571 1 }

identificarían cada uno un objeto, "pci", definido en la Norma ISO 8571.

Con la siguiente definición adicional:

ftam OBJECT IDENTIFIER ::= { iso standard 8571 }

el valor siguiente es también equivalente a los anteriores

{ ftam pci(1) }

30 Notación para el tipo pdv incrustado

30.1 El tipo pdv incrustado (embedded PDV) (véase 3.8.21) será referenciado por la notación "EmbeddedPDVType":

EmbeddedPDVType ::= EMBEDDED PDV

30.2 Este tipo tiene un rótulo que es de clase universal, número 11.

NOTA – Cuando se utiliza la negociación de la capa de presentación, la misma funcionalidad que EXTERNAL es proporcionada por EMBEDDED PDV (junto con la funcionalidad añadida), pero los bits en la línea serán diferentes. En este caso se recomienda que los cambios de versiones futuras de los protocolos de aplicación incorporen la sustitución de EXTERNAL por CHOICE{external EXTERNAL, embedded-pdv EMBEDDED PDV}. En la Rec. UIT-T X.681 | ISO/CEI 8824-2, Anexo C, se proponen sustituciones adicionales para utilizar EXTERNAL cuando no se utilice la negociación de la capa de presentación.

30.3 El tipo está constituido por valores que representan:

a) una codificación de un valor de datos único que puede ser, aunque no necesariamente, el valor de un tipo ASN.1; y

b) la identificación (de manera separada o conjunta) de:

1) una clase de valores que contienen ese valor de datos (una sintaxis abstracta); y

2) la codificación utilizada (la sintaxis de transferencia) para distinguir ese valor de datos de otros valores de la misma clase.

NOTAS

1 El valor de dato puede ser el valor de un tipo ASN.1 o puede ser, por ejemplo, la codificación de una imagen fija o una imagen en movimiento. La identificación consta de uno o dos identificadores de objeto o hace referencia a un contexto de presentación OSI para la identificación de las sintaxis abstracta y de transferencia.

2 La identificación de la sintaxis abstracta y/o o la codificación también puede estar determinada por la del diseñador de la aplicación como un valor fijo, en cuyo caso no será codificada en una instancia de comunicación.

30.4 El tipo pdv incrustado tiene un tipo asociado. Este tipo se utiliza para dar precisión a la definición de los valores abstractos del tipo pdv incrustado y también para sustentar las notaciones de valor y subtipo del tipo pdv incrustado.

NOTA – Las reglas de codificación pueden definir un tipo diferente, utilizado para derivar codificaciones, o pueden especificar codificaciones sin referencia a ningún tipo asociado. En particular, la codificación en PER asegura que las ocurrencias múltiples (en el mismo mensaje) de EMBEDDED–PDV con el mismo o los mismos identificadores de objeto codifican el o los identificadores de objeto solamente una vez.

30.5 El tipo asociado para la definición de valores y la subtipificación, suponiendo un entorno de rotulación automática, es (con comentarios normativos):

```
SEQUENCE {
    identification                  CHOICE {
        syntaxes                        SEQUENCE {
            abstract                    OBJECT IDENTIFIER,
            transfer                    OBJECT IDENTIFIER }
                -- Identificadores de objetos de sintaxis abstracta y de transferencia --,
```

```
        syntax                          OBJECT IDENTIFIER
                -- Un solo identificador de objeto para la identificación de la clase y la codificación --,

        presentation-context-id         INTEGER
                -- (Aplicable solamente a entornos OSI)
                -- El contexto de presentación negociado identifica la clase del valor y su codificación --,

        context-negotiation             SEQUENCE {
            presentation-context-id         INTEGER,
            transfer-syntax                 OBJECT IDENTIFIER }
                -- (Aplicable solamente a entornos OSI)
                -- Negociación del contexto en curso para que un contexto identifique la clase del valor
                -- y su codificación --,

        transfer-syntax                 OBJECT IDENTIFIER
                -- La clase del valor (por ejemplo, la especificación de que se trata del valor de un tipo
                -- ASN.1) viene fijada por el diseñador de la aplicación (y la conocen, por tanto, el
                -- emisor y el receptor). Este caso se da sobre todo para sustentar la encripción
                -- selectiva de campo (u otras transformaciones de codificación) de un tipo ASN.1 --,

        fixed                           NULL
                -- El valor de dato es el valor de un tipo ASN.1 fijo (y conocido, por tanto, por el emisor
                -- y el receptor) -- },

    data-value-descriptor               ObjectDescriptor  OPTIONAL
                -- Esto proporciona identificación de la clase del valor legible por los seres humanos --,

    data-value                          CHOICE {
        notation                            ABSTRACT-SYNTAX.&Type
                -- Esta notación de tipo se define en la Rec UIT-T X.681 | ISO/CEI 8824-2 y tiene una
                -- notación de valor que es cualquier definición de tipo ASN.1, seguida de dos puntos (:) y de
                -- la notación del valor para ese tipo. Esta alternativa de choice (elección) se da para
                -- permitir la especificación utilizando una notación, fácil para las personas, de los valores
                -- de los datos que son valores de un tipo ASN.1. --,

        encoded                                 BIT STRING
                -- Esta alternativa de choice (elección) se da para permitir la especificación de valores de
                -- datos que no son valores de un tipo ASN.1. -- } }

( WITH COMPONENTS {
    ... ,
    data-value-descriptor  ABSENT } )
```

NOTA – El tipo pdv incrustado no permite la inclusión de un valor "data-value-descriptor" junto con la "identification". Sin embargo, la definición del tipo asociado que aquí se proporciona sustenta las comunalidades existentes entre el tipo incrustado pdv (embedded-pdv), el tipo externo (external) y el tipo cadena de caracteres no restringida (unrestricted character string).

30.6 Para la alternativa "presentation-context-id", el valor entero será un identificador de contexto de presentación del conjunto de contextos. Esta alternativa no deberá ser utilizada en la petición P-CONEXIÓN ni en la petición P-ALTERACIÓN DE CONTEXTO para un contexto de presentación cuya adición o supresión está siendo propuesta por esas primitivas de petición.

NOTA – Incluso si existe una sola sintaxis de transferencia propuesta para un contexto de presentación en la lista de definiciones de contextos de presentación, la alternativa "presentation-context-id" no puede ser utilizada para ese contexto de presentación.

30.7 La alternativa "context-negotiation" deberá utilizarse solamente en la petición P-CONEXIÓN o en la petición P-ALTERACIÓN DE CONTEXTO y el valor entero será un identificador de contexto de presentación propuesto para su adición al conjunto de contextos definido. El identificador de objeto "transfer-syntax" identificará una sintaxis de transferencia propuesta para el contexto de presentación que se utiliza para codificar el valor.

30.8 La notación para un valor del tipo embedded-pdv será la notación del valor del tipo asociado definido en 30.5.

```
EmbeddedPdvValue ::= SequenceValue      -- valor del tipo asociado definido en 30.5.
```

30.9 EJEMPLO 1 – Cuando un diseñador de aplicación desea que la codificación sea independiente de cualquier entorno de presentación (y, por tanto, que sea posible retransmitirla o almacenarla y recuperarla sin modificación), es necesario prohibir la utilización de las alternativas "presentation-context" y "context-negotiation", lo cual puede hacerse escribiendo:

```
EMBEDDED PDV (WITH COMPONENTS {
        ... ,
        identification  (WITH COMPONENTS {
                ... ,
                presentation-context-id     ABSENT,
                context-negotiation         ABSENT } ) } )
```

30.10 EJEMPLO 2 – Si ha de implementarse una sola opción, tal como la utilización de "syntaxes", esto puede hacerse escribiendo:

```
EMBEDDED PDV (WITH COMPONENTS {
        ... ,
        identification  (WITH COMPONENTS {
          syntaxes  PRESENT } ) } )
```

30.11 EJEMPLO 3 – (En este ejemplo se hace uso del tipo parametrizado especificado en la Rec. UIT-T X.683 | ISO/CEI 8824-4.) Un tipo parametrizado, ENCRYPTED, se define utilizando EMBEDDED PDV de la siguiente manera:

```
ENCRYPTED { ToBeEncrypted } ::= EMBEDDED PDV (WITH COMPONENTS {
                identification (WITH COMPONENTS { fixed PRESENT } ),
                -- El data-value (valor de dato) es cualquier valor deToBeEncrypted.
                -- La sintaxis de transferencia es la sintaxis de transferencia de seguridad.
                data-value (WITH COMPONENTS { notation (ToBeEncrypted) } ) } )
```

Un tipo, como por ejemplo "CONFIDENTIAL", puede ser encriptado ahora escribiendo "ENCRYPTED {CONFIDENTIAL}". En este ejemplo sólo se transmitiría el valor encriptado sin valores de identificadores de objetos adicionales.

31 Notación para el tipo externo

31.1 El tipo externo (external) (véase 3.8.25) será referenciado por la notación "ExternalType":

```
ExternalType ::= EXTERNAL
```

31.2 Este tipo tiene un rótulo que es de clase universal, número 8.

31.3 El tipo está constituido por valores que representan:

a) la codificación de un valor de dato único que puede ser, aunque no necesariamente, el valor de un tipo ASN.1; y

b) la identificación de

   1) una clase de valores que contienen ese valor de dato (una sintaxis abstracta); y

   2) la codificación utilizada (la sintaxis de transferencia) para distinguir ese valor de dato de otros valores en la misma clase; y

c) (opcionalmente) un descriptor de objeto que proporciona una descripción legible por los seres humanos de la clase del valor de dato. El descriptor de objeto opcional no estará presente, a menos que lo permitan de manera explícita comentarios asociados con la utilización de la notación "ExternalType".

NOTA – La Nota 1 de 30.3 también se aplica al tipo external.

31.4 El tipo externo tiene un tipo asociado. Este tipo se utiliza para dar precisión a la definición de los valores abstractos del tipo externo y se utiliza también para sustentar las notaciones de valor y subtipo del tipo externo.

NOTA – Las reglas de codificación pueden definir un tipo diferente, utilizado para derivar codificaciones, o pueden especificar codificaciones sin referencia a ningún tipo asociado. En particular, la codificación en BER utiliza un tipo secuencia (sequence) equivalente, idéntico al que estaba presente en la definición del tipo externo de la Rec. X.208 del CCITT (1988) | ISO/CEI 8824:1990, y las codificaciones de valores externos por BER permanecen inalteradas.

31.5 El tipo asociado para la definición del valor y la subtipificación, suponiendo un entorno de rotulación automática, es (con comentarios normativos):

```
SEQUENCE {
    identification                  CHOICE {
        syntaxes                        SEQUENCE {
            abstract                        OBJECT IDENTIFIER,
            transfer                        OBJECT IDENTIFIER }
                -- Identificadores de objetos de sintaxis abstracta y de transferencia --,

        syntax                      OBJECT IDENTIFIER
                -- Un solo identificador de objeto para la identificación de la clase y la codificación --,

        presentation-context-id     INTEGER
                -- (Aplicable solamente a entornos OSI)
                -- El contexto de presentación negociado identifica la clase del valor y su codificación --,

        context-negotiation         SEQUENCE {
            presentation-context-id         INTEGER
            transfer-syntax                 OBJECT IDENTIFIER }
            -- (Aplicable solamente a entornos OSI)
            -- Negociación del contexto en curso para que un contexto identifique la clase del valor
            -- y su codificación --,

        transfer-syntax             OBJECT IDENTIFIER
                -- La clase del valor (por ejemplo, la especificación de que se trata del valor de un tipo
                -- ASN.1) viene fijada por el diseñador de la aplicación (y la conocen, por tanto, el
                -- emisor y el receptor). Este caso se da sobre todo para sustentar la encripción
                -- selectiva de campo (u otras transformaciones de codificación) de un tipo ASN.1 --,

        fixed                       NULL
                -- El valor de dato es el valor de un tipo ASN.1 fijo (y conocido, por tanto, por el emisor
                -- y el receptor) -- },

    data-value-descriptor           ObjectDescriptor  OPTIONAL
                -- Esto proporciona identificación de la clase del valor legible por los seres humanos --,

    data-value                      CHOICE {
        notation                        ABSTRACT-SYNTAX.&Type
                -- Esta notación de tipo se define en la Rec UIT-T X.681 | ISO/CEI 8824-2 y tiene una
                -- notación de valor que es cualquier definición de tipo ASN.1, seguida de dos puntos (:) y de
                -- la notación del valor para ese tipo. Esta alternativa de elección se proporciona para
                -- permitir la especificación utilizando la notación, fácil para las personas, de los valores
                -- de los datos que son valores de un tipo ASN.1. --,

        encoded                         BIT STRING
                -- Esta alternativa de elección se proporciona para permitir la especificación de valores de
                -- datos que no son valores de un tipo ASN.1. -- } }

( WITH COMPONENTS {
    ... ,
    identification  (WITH COMPONENTS {
        ... ,
        syntaxes                    ABSENT,
        transfer-syntax             ABSENT,
        fixed                       ABSENT } ) } )
```

NOTA – El tipo externo no permite las alternativas "syntaxes", "transfer" y "transfer-syntax" o "fixed" de "identification". Estas alternativas no pueden autorizarse para el tipo externo por la necesidad de mantener la retrocompatibilidad con el tipo externo de la Rec. X.208 del CCITT (1988) | ISO/CEI 8824:1990. Los diseñadores de aplicaciones que exijan estas opciones deberán utilizar el tipo pdv incrustado (embedded-pdv). La definición del tipo asociado que aquí se proporciona sustenta las comunalidades existentes entre el tipo externo, el tipo cadena de caracteres no restringida y el tipo pdv incrustado.

31.6 El texto de 30.6 y 30.7 también se aplica al tipo external.

31.7 La notación para un valor de tipo externo deberá ser la notación de valor del tipo asociado definido en 31.5.

```
ExternalValue ::= SequenceValue     -- valor del tipo asociado definido en 31.5
```

32 Tipos cadena de caracteres

Estos tipos están constituidos por cadenas de caracteres procedentes de algún repertorio de caracteres especificado. Es normal definir un repertorio de caracteres y su codificación mediante el uso de células de una o más tablas, correspondiendo cada célula a un carácter del repertorio. A cada célula se le asigna también, normalmente, un símbolo gráfico y un nombre de carácter, si bien en algunos repertorios se dejan células vacías o tienen nombres pero no formas (ejemplos de células con nombre pero sin forma son los caracteres de control, tales como el EOF de Norma ISO 646, y los caracteres de avance de espacio tales como THIN-SPACE y EN-SPACE de Norma ISO/CEI 10646-1).

El término **abstract character** (carácter abstracto) denota la totalidad de la información asociada con una célula de una tabla repertorio de caracteres. La información asociada con una célula denota un carácter abstracto distinto del repertorio incluso si esa información es nula (no hay un símbolo gráfico o nombre de carácter asignado a esa célula).

La notación de valor ASN.1 para los tipos cadenas de caracteres (character string) tiene tres variantes (que pueden combinarse), especificadas formalmente a continuación:

a) una representación impresa de los caracteres de la cadena utilizando el símbolo gráfico asignado, con la posible inclusión de caracteres con avance de espacio; ésta es la notación "cstring";

NOTAS

1 Una representación como esa puede resultar equívoca cuando se utilice la misma forma de carácter para más de un carácter del repertorio.

2 Una representación como esa puede resultar equívoca cuando se utilicen caracteres con avance de espacio o la especificación esté impresa con un tipo de carácter de espaciamiento proporcional.

b) un listado de los caracteres del valor cadena de caracteres dando una serie de referencias de valores ASN.1 que han sido asignados al carácter; un conjunto de tales referencias de valores se define en el módulo ASN.1-ISO 10646 (cláusula 35) para el repertorio de caracteres de Norma ISO/CEI 10646-1 y para el repertorio de caracteres IA5String; esta forma no está disponible para otros repertorios de caracteres a menos que el usuario efectúe asignaciones a esas referencias de valores utilizando la notación de valor descrita en el inciso anterior a) o en el c) siguiente;

c) un listado de caracteres del valor cadena de caracteres identificando cada uno de los caracteres abstractos por la posición de su célula en la tabla o tablas repertorio de caracteres; esta forma está disponible solamente para IA5String, UniversalString, y BMPString.

33 Notación para tipos cadena de caracteres

33.1 La notación para referenciar un tipo cadena de caracteres (character string) (véase 3.8.11) será:

CharacterStringType ::= RestrictedCharacterStringType | UnrestrictedCharacterStringType

"RestrictedCharacterStringType" es la notación para un tipo cadena de caracteres restringida (restricted character string) y se define en la cláusula 34. "UnrestrictedCharacterStringType" es la notación para un tipo cadena de caracteres no restringidas (unrestricted character string) y se define en 37.1.

33.2 El rótulo de cada tipo cadena de caracteres restringida se especifica en 34.1. El rótulo del tipo cadena de caracteres no restringida se especifica en 37.2.

33.3 La notación para un valor cadena de caracteres será:

CharacterStringValue ::= RestrictedCharacterStringValue | UnrestrictedCharacterStringValue

"RestrictedCharacterStringValue" se define en 34.7 y 37.6. "UnrestrictedCharacterStringValue" es la notación para un valor cadena de caracteres no restringida y se define en 34.7 y 37.6.

34 Definición de tipos cadena de caracteres restringida

En esta cláusula se definen tipos cuyos valores están limitados a secuencias de cero, uno o más caracteres de un repertorio especificado de caracteres. La notación para referenciar un tipo cadena de caracteres restringida será "RestrictedCharacterStringType":

```
RestrictedCharacterStringType ::= BMPString      |
                                  GeneralString  |
                                  GraphicString  |
                                  IA5String      |
                                  ISO646String   |
```

NumericString |
PrintableString |
TeletexString |
T61String |
UniversalString |
VideotexString |
VisibleString

Cada una de las alternativas de "RestrictedCharacterStringType" se define especificando:

a) el rótulo asignado al tipo; y

b) un nombre (por ejemplo, NumericString) por el cual puede referenciarse la definición del tipo; y

c) los caracteres de la colección de caracteres utilizada para definir el tipo, por referencia a una tabla que da los caracteres gráficos, por referencia a un número de registración en el ISO International Register of Coded Character Sets (registro internacional de conjuntos de caracteres codificados de la ISO) (véase el *ISO International Register of Coded Character Sets*) o por referencia a la Norma ISO/CEI 10646-1.

34.1 El Cuadro 3 da el nombre por el cual se referencia cada tipo cadena de caracteres restringida, el número del rótulo de la clase universal asignada al tipo, el número de registración o la tabla que proporcionan la definición o la cláusula del texto que proporciona la definición y, donde sea necesario, la identificación de una nota relativa a la entrada en la tabla (o cuadro). Si la notación contiene la definición de un nombre sinónimo, éste aparecerá entre paréntesis.

NOTA – El rótulo asignado a tipos cadena de caracteres (character string) identifica inequívocamente el tipo. Obsérvese no obstante que si se emplea ASN.1 para definir nuevos tipos a partir de este tipo (en particular, utilizando IMPLICIT), puede resultar imposible reconocer estos tipos si no se conoce la definición del tipo ASN.1.

Cuadro 3 – Lista de tipos cadena de caracteres restringida

Nombre para referenciar el tipo	Número de clave universal	Número de registración para la definición[a], número de cuadro o cláusula de la Rec. UIT-T X.680 | ISO/CEI 8824-1	Notas
NumericString	18	Cuadro 4	(1)
PrintableString	19	Cuadro 5	(1)
TeletexString (T61String)	20	6, 87, 102, 103, 106, 107, 126, 144, 150, 153, 156, 164, 165, 168 + SPACE + DELETE	(2)
VideotexString	21	1, 13, 72, 73, 87, 89, 102, 108, 126, 128, 129, 144, 150, 153, 164, 165, 168 + SPACE + DELETE	(3)
IA5String	22	1, 6 + SPACE + DELETE	
GraphicString	25	Todos los conjuntos G + SPACE	
VisibleString (ISO646String)	26	6 + SPACE	
GeneralString	27	Todos los conjuntos G y todos los conjuntos C + SPACE + DELETE	
UniversalString	28	Véase 34.6	
BMPString	30	Véase 34.12	

[a] Los números de registración que proporcionan la definición se dan en ISO *International Register of Coded Character Sets*.

NOTAS

1 El estilo tipográfico, tamaño, color, intensidad u otras características de visualización no son significativos.

2 Las entradas que corresponden a estos números de registración hacen referencia a la Recomendación T.61 para las reglas relativas a su utilización. Pueden utilizarse las entradas de registro 6 y 156 en vez de las 102 y 103.

3 Las entradas correspondientes a estos números de registración dan la funcionalidad de las Recomendaciones T.100 y T.101.

4 La referencia al registro 6 de «ISO International Register of Coded Character Sets to be used with Escape Sequences» constituye una referencia indirecta a la Norma ISO 646:1991. Esto representa un cambio con respecto a la Rec. X.208 del CCITT (1988) | ISO/CEI 8824:1990, que referenciaba el registro 2 (referencia indirecta a la Norma ISO 646:1973). En las aplicaciones en las que se desee referenciar el número de registración 2 se deberán emplear otras maneras de hacerlo (por ejemplo, utilización de la cadena de caracteres no restringida (véase la cláusula 37) para llevar la definición antigua de VisibleString o referenciar la Rec. X.208 del CCITT | ISO/CEI 8824.

34.2 El Cuadro 4 indica los caracteres que pueden aparecer en el tipo NumericString y en la sintaxis abstracta de caracteres NumericString.

Cuadro 4 – NumericString

Nombre	Símbolo gráfico
Dígitos	0, 1, ... 9
Espacio	(espacio)

34.3 Los siguientes valores de identificador de objeto y de descriptor de objeto se asignan para identificar y describir la sintaxis abstracta de caracteres NumericString:

{ joint-iso-ccitt asn1(1) specification(0) characterStrings(1) numericString(0) }

y

"NumericString character abstract syntax"

NOTAS

1 Este valor de identificador de objeto puede utilizarse en valores de CHARACTER STRING y en otros casos en los que es necesario llevar la identificación del tipo cadena de caracteres separada del valor.

2 Un valor de una sintaxis abstracta de caracteres NumericString puede ser codificado por:

a) Una de las reglas dadas en la Norma ISO/CEI 10646-1 para la codificación de caracteres abstractos. En este caso, la sintaxis de transferencia de caracteres es identificada por el identificador de objeto asociado con las reglas del Anexo M a la Norma ISO/CEI 10646-1.

b) Las reglas de codificación ASN.1 para el tipo NumericString incorporado. En este caso, la sintaxis de transferencia de caracteres es identificada por el valor de identificador de objeto {joint-iso-ccitt asn1(1) basic-encoding(1)}.

34.4 El Cuadro 5 indica los caracteres que pueden aparecer en el tipo PrintableString y en la sintaxis abstracta de caracteres PrintableString.

Cuadro 5 – PrintableString

Nombre	Símbolo
Letras mayúsculas	A, B, ... Z
Letras minúsculas	a, b, ... z
Dígitos	0, 1, ... 9
Espacio	(espacio)
Apóstrofo	'
Paréntesis izquierdo	(
Paréntesis derecho	)
Signo más	+
Coma	,
Guión	-
Punto	.
Barra de fracción	/
Dos puntos	:
Signo igual	=
Signo de interrogación (final)	?

34.5 Los siguientes valores de identificador de objeto y descriptor de objeto se asignan para identificar y describir la sintaxis abstracta de caracteres PrintableString:

{ joint-iso-ccitt asn1(1) specification(0) characterStrings(1) printableString(1) }

y

"PrintableString character abstract syntax"

NOTAS

1 Este valor de identificador de objeto puede utilizarse en valores de CHARACTER STRING y en otros casos en los que es necesario llevar la identificación del tipo cadena de caracteres separada del valor.

2 Un valor de una sintaxis abstracta de caracteres PrintableString puede ser codificado por:

a) Una de las reglas dadas en la Norma ISO/CEI 10646-1 para la codificación de los caracteres abstractos. En este caso, la sintaxis de transferencia de caracteres es identificada por el identificador de objeto asociado con las reglas del Anexo M a la Norma ISO/CEI 10646-1.

b) Las reglas de codificación ASN.1 para el tipo PrintableString incorporado. En este caso, la sintaxis de transferencia de caracteres es identificada por el identificador de objeto {joint-iso-ccitt asn1(1) basic-encoding(1)}.

34.6 Los caracteres que pueden aparecer en el tipo UniversalString son cualesquiera de los autorizados por la Norma ISO/CEI 10646-1, y la utilización de este tipo invoca los requisitos de conformidad especificados en la Norma ISO/CEI 10646-1, especialmente con respecto a la zona de uso restringido de la Norma ISO/CEI 10646-1.

NOTAS

1 La utilización de este tipo no constreñido está desaconsejada, pues la conformidad será generalmente imposible en la práctica.

2 La cláusula 35 define un módulo ASN.1 que contiene varios subtipos de este tipo para los "Collections of graphics characters for subsets" (Colecciones de caracteres gráficos para subconjuntos) definidas en el Anexo A de la Norma ISO/CEI 10646-1.

34.7 La notación de valor para los tipos restricted character string será "cstring" (véase 9.11), "CharacterStringList", Quadruple" o "Tuple". "Quadruple" sólo es capaz de definir una cadena de caracteres de longitud uno y sólo se puede utilizar en notación de valor para tipos UniversalString o BMPString. "Tuple" sólo puede definir una cadena de caracteres de longitud uno y sólo se puede utilizar en notación de valor para tipos IA5String.

```
RestrictedCharacterStringValue ::= cstring | CharacterStringList | Quadruple | Tuple

CharacterStringList ::=  "{" CharSyms "}"
CharSyms ::=  CharsDefn | CharSyms "," CharsDefn
CharsDefn ::=  cstring | DefinedValue

Quadruple ::= "{"  Group  ","  Plane  ","  Row  ","  Cell "}"
Group     ::= number
Plane     ::= number
Row       ::= number
Cell      ::= number

Tuple ::= "{" TableColumn "," TableRow "}"
TableColumn ::= number
TableRow ::= number
```

NOTAS

1 La notación "cstring" sólo puede utilizarse en un medio capaz de visualizar los símbolos gráficos de los caracteres que están presentes en el valor. Por el contrario, si el medio no tiene esa capacidad, la única posibilidad de especificar un valor cadena de caracteres que utiliza esos símbolos gráficos es mediante la notación "CharacterStringList" y solamente si el tipo es UniversalString, BMPString o IA5String y se utiliza la alternativa "DefinedValue" de "CharsDefn" (véase 35.1.2).

2 La cláusula 35 define varias "valuereference" que denotan caracteres únicos (cadenas de tamaño 1) de tipo UniversalString e IA5String.

EJEMPLO – Supóngase que se desea especificar un valor de "abcΣef" para una UniversalString (cadena universal), no siendo representable el carácter "Σ" en el medio disponible; este valor puede expresarse también como:

```
IMPORTS BasicLatin, greekCapitalLetterSigma FROM ASN1-CHARACTER-MODULE

        { joint-iso-ccitt asn1(1) specification(0) modules(0) iso10646(0) };

MyAlphabet ::= UniversalString (FROM (BasicLatin | greekCapitalLetterSigma))

mystring MyAlphabet ::= { "abc" , greekCapitalLetterSigma , "def" }
```

3 Cuando se especifique el valor de un tipo UniversalString deberá emplearse la notación "cstring", a menos que se hayan resuelto las ambigüedades derivadas de los caracteres gráficos diferentes con formas similares.

EJEMPLO – No deberá utilizarse la siguiente notación "cstring" porque las letras "H", "O", "P" y "E" aparecen en los alfabetos LATÍN BÁSICO, CIRÍLICO y GRIEGO BÁSICO por lo que resultan ambiguas.

```
IMPORTS BasicLatin, Cyrillic, BasicGreek FROM ASN1-CHARACTER-MODULE

        { joint-iso-ccitt asn1(1) specification(0) modules(0) iso10646(0) };

MyAlphabet ::= UniversalString (FROM (BasicLatin | Cyrillic | BasicGreek))

mystring MyAlphabet ::= "HOPE"
```

34.8 El "DefinedValue" de "CharsDefn" deberá ser una referencia a un valor de ese tipo.

34.9 El "number" (número) de las producciones "Group", "Plane", "Row" y "Cell" deberá ser inferior a 256.

34.10 El "Group" especifica un grupo en el espacio de codificación del UCS, el "Plane" especifica un plano dentro del grupo, la "Row" especifica una fila dentro del plano y la "Cell" especifica una célula dentro de la fila. El carácter abstracto identificado por esta notación es el carácter abstracto de la célula especificado por los valores de "Group", "Plane", "Row" y "Cell". En todos los casos, el conjunto de caracteres permitidos puede ser restringido por subtipificación.

NOTA – Los diseñadores de aplicación deben analizar cuidadosamente las implicaciones de conformidad cuando se utilizan tipos cadena de caracteres abierta (open–ended character string), tales como GeneralString, GrphicStrig y UniversalString, sin la aplicación de constricciones. Se necesita también un texto cuidadosamente elaborado sobre conformidad para tipos cadena de caracteres que, aunque acotados, son extensos, como TeletexString.

34.11 El "number" (número) de la producción "TableColumn" deberá estar en la gama de cero a siete y el "number" de la producción "TableRow" deberá estar en la gama de cero a quince. La "TableColumn" especifica una columna y la "TableRow" especifica una fila de una tabla de códigos de caracteres, de conformidad con la Figura 1 de la Norma ISO/CEI 2022. Esta notación se emplea solamente para IA5String cuando la tabla de códigos contiene la entrada de registro 1 en las columnas 0 y 1 y la entrada de registro 6 en las columnas 2 a 7 (véase el *ISO International Register of Coded Character Sets to be used with Escape Sequence*).

34.12 BMPString es un subtipo de UniversalString que tiene su propio rótulo exclusivo y modela el plano multilingüe básico (las primeras células 64K-2) de la Norma ISO/CEI 10646-1. Tiene un tipo asociado definido como:

```
UniversalString (Bmp)
```

donde Bmp se define en el módulo ASN.1 "ASN1-ISO 10646" (véase la cláusula 35) como el subtipo de UniversalString correspondiente al nombre de colección "BMP" definido en la Norma ISO/CEI 10646-1, Anexo A.

NOTAS

1 Puesto que BMPString es un tipo incorporado (built-in), no se define en ASN1-ISO 10646.

2 La finalidad de definir BMPString como un tipo incorporado es permitir que las reglas de codificación (tales como BER), que tienen en cuenta las constricciones, utilicen codificaciones de 16 bits en vez de 32 bits.

3 En la notación de valor todos los valores BMPString son valores UniversalString válidos.

35 Denominación de caracteres y colecciones definidas en la Norma ISO/CEI 10646-1

Esta cláusula especifica un módulo incorporado ASN.1 que contiene la definición de un nombre de referencia de valor para cada carácter de la Norma ISO/CEI 10646-1, donde cada carácter referencia un valor UniversalString de tamaño 1. Este módulo contiene también la definición de un nombre de referencia de tipo para cada colección de caracteres de la Norma ISO/CEI 10646-1, donde cada nombre referencia un subconjunto de UniversalString.

NOTA – Estos valores están disponibles para uso en la notación de valor del tipo UniversalString y de los tipos derivados de éste. Todas las referencias de valores y tipos definidas en el módulo especificado en 35.1 son exportadas y tienen que ser importadas por cualquier módulo que las utilice.

35.1 Especificación del módulo ASN.1 "ASN1-CHARACTER-MODULE"

Este módulo no se reproduce aquí en su totalidad; en su lugar se especifica el medio para definirlo.

35.1.1 El módulo comienza de esta manera:

```
ASN1-CHARACTER-MODULE {joint-iso-ccitt asn1(1) specification(0) modules(0) iso10646(0)}
DEFINITIONS ::= BEGIN
-- Todas las referencias de valor y referencias de tipo definidas dentro de este módulo
-- son exportadas explícitamente y están disponibles para su importación por cualquier módulo
-- Caracteres de control de la Norma ISO 646.

nul   IA5String ::= {0, 0}
soh   IA5String ::= {0, 1}
stx   IA5String ::= {0, 2}
etx   IA5String ::= {0, 3}
eot   IA5String ::= {0, 4}
enq   IA5String ::= {0, 5}
ack   IA5String ::= {0, 6}
bel   IA5String ::= {0, 7}
bs    IA5String ::= {0, 8}
ht    IA5String ::= {0, 9}
lf    IA5String ::= {0,10}
vt    IA5String ::= {0,11}
ff    IA5String ::= {0,12}
cr    IA5String ::= {0,13}
so    IA5String ::= {0,14}
si    IA5String ::= {0,15}
dle   IA5String ::= {1, 0}
dc1   IA5String ::= {1, 1}
dc2   IA5String ::= {1, 2}
dc3   IA5String ::= {1, 3}
dc4   IA5String ::= {1, 4}
nak   IA5String ::= {1, 5}
syn   IA5String ::= {1, 6}
etb   IA5String ::= {1, 7}
can   IA5String ::= {1, 8}
em    IA5String ::= {1, 9}
sub   IA5String ::= {1,10}
esc   IA5String ::= {1,11}
is4   IA5String ::= {1,12}
is3   IA5String ::= {1,13}
is2   IA5String ::= {1,14}
is1   IA5String ::= {1,15}
del   IA5String ::= {7,15}
```

35.1.2 Para cada entrada de cada lista de nombres de caracteres de los caracteres gráficos (glyphs) mostradas en las cláusulas 24 y 25 de la Norma ISO/CEI 10646-1, el módulo incluye un enunciado (statement) de la forma:

```
<namedcharacter> BMPString ::= <tablecell>
    -- representa el carácter <iso10646name>, véase la Norma ISO/CEI 10646-1
```

donde:

a) <iso10646name> es el nombre de carácter derivado de uno listado en la Norma ISO/CEI 10646-1;

b) <namedcharacter> es una cadena obtenida aplicando a <iso10646name> los procedimientos especificados en 35.2;

c) <tablecell> es el glyph de la célula de tabla de la Norma ISO/CEI 10646-1 correspondiente a la entrada en la lista.

EJEMPLO

```
latinCapitalLetterA  BMPString  ::=  {0, 0, 0, 65}
    -- representa el carácter LETRA MAYÚSCULA LATINA A (letra A), véase la Norma ISO/CEI 10646-1
greekCapitalLetterSigma BMPString  ::=  {0, 0, 3, 145}
    -- representa el carácter LETRA MAYÚSCULA GRIEGA SIGMA, (letra griega Σ), véase la Norma
       ISO/CEI 10646-1
```

35.1.3 Para cada nombre de una colección de caracteres gráficos especificados en la Norma ISO/CEI 10646-1, Anexo A, se incluye un enunciado (statement) en el módulo de la forma:

```
<namedcollectionstring> ::= BMPString
    (FROM ( <alternativelist>))
    -- representa la colección de caracteres <collectionstring>,
    -- véase la Norma ISO/CEI 10646-1.
```

donde:

a) <collectionstring> es el nombre para la colección de caracteres asignada en la Norma ISO/CEI 10646-1;

b) <namedcollectionstring> se forma aplicando a <collectionstring> los procedimientos de 35.3;

c) <alternativelist> se forma utilizando los <namedcharacter>s generados como se indica en 35.2 para cada uno de los caracteres especificados por la Norma ISO/CEI 10646-1.

La referencia de tipo resultante, <namedcollectionstring>, forma un subconjunto limitado. (Véase el suplemento didáctico del Anexo G.)

NOTA – Un subconjunto limitado es una lista de caracteres de un subconjunto especificado. Un subconjunto seleccionado es, en cambio, una colección de caracteres enumerados en la Norma ISO/CEI 10646-1, Anexo A, más la colección LATÍN BÁSICO.

EJEMPLO (parcial):

```
space BMPString ::= {0, 0, 0, 32}
exclamationMark BMPString ::= {0, 0, 0, 33}
quotationMark BMPString ::= {0, 0, 0, 34}
...          -- y así sucesivamente
tilde BMPString ::= {0, 0, 0, 126}

BasicLatin ::= BMPString
    (FROM (space
    | exclamationMark
    | quotationMark
    | ...          -- y así sucesivamente
    | tilde)
    )
-- representa la colección de caracteres LATÍN BÁSICO, véase la Norma ISO/IEC 10646-1.
-- La elipsis de este ejemplo se utiliza para abreviar y significa "y así sucesivamente";
-- esto no se puede utilizar en un módulo ASN.1 real.
```

35.1.4 La Norma ISO/CEI 10646-1 define tres niveles de realización. Por defecto, todos los tipos definidos en ASN1-CHARACTER-MODULE, excepto para "Level1" y "Level2", se atienen a la realización de nivel 3, porque dichos tipos no tienen restricciones a la utilización de los caracteres combinantes. "Level1" indica que se requiere la realización de nivel 1, "Level2" indica que se requiere la realización de nivel 2 y "Level3" indica que se requiere la realización de nivel 3. Así pues, en ASN1-CHARACTER-MODULE se define lo siguiente:

```
Level1 ::= BMPString (FROM (ALL EXCEPT CombiningCharacters))

Level2 ::= BMPString (FROM (ALL EXCEPT CombiningCharactersB-2))

Level3 ::= BMPString
```

NOTAS

1 Los "CombiningCharacters" y los "CombiningCharactersType-2" son las <namedcollectionstring>s correspondientes a "COMBINING CHARACTERS" y "COMBINING CHARACTERS TYPE-2", respectivamente, definidas en el Anexo A de la Norma ISO/CEI 10646-1.

2 "Level1" y "Level2" se utilizarán siguiendo una "IntersectionMark" (véase la cláusula 44) o como la única constricción en una "ConstraintSpec". Véase un ejemplo en F.2.7.1.

3 Para más información sobre este tema véase G.2.5.

35.1.5 El módulo se termina con el enunciado:

```
END
```

35.1.6 Un equivalente al ejemplo de 35.1.3 definido por el usuario es:

BasicLatin ::= BMPString (FROM (space..tilde))

-- representa la colección de caracteres LATÍN BÁSICO, véase la Norma ISO/CEI 10646-1.

35.2 Un <namedcharacter> es la cadena obtenida tomando un <iso10646name> (véase 35.1.2) y aplicando el siguiente algoritmo:

a) cada letra mayúscula del <ISO10646> name se transforma en la correspondiente letra minúscula, a menos que la letra mayúscula esté precedida por un SPACE (carácter de espacio), en cuyo caso se mantiene la mayúscula;

b) se mantiene sin cambiar cada dígito y cada HYPHEN-MINUS (guión signo menos);

c) se suprime cada SPACE (carácter de espacio).

NOTA – Este algoritmo, junto con las directrices relativas a la denominación de caracteres del Anexo K de la Norma ISO/CEI 10646-1, permite obtener una notación de valor inequívoca para todos los nombres de carácter indicados en la Norma ISO/CEI 10646-1.

EJEMPLO – El carácter de la Norma ISO/CEI 10646-1, fila 0, célula 60, que lleva por nombre "LESS-THAN SIGN" (signo menor que) y tienen la representación gráfica "<" puede ser referenciado mediante el "DefinedValue" de:

less-thanSign

35.3 Una <namedcollectionstring> es la cadena obtenida tomando <collectionstring> y aplicándole el siguiente algoritmo:

a) cada letra mayúscula del nombre de colección de la Norma ISO/CEI 10646-1 se transforma en la correspondiente letra minúscula, a menos que la letra mayúscula esté precedida por un SPACE (carácter de espacios), o sea la primera letra del nombre, en cuyo caso se mantiene la mayúscula;

b) se mantiene sin cambiar cada dígito y cada HYPHEN-MINUS (guión signo menos);

c) se suprime cada SPACE (carácter de espacio).

EJEMPLOS

1 La colección identificada en el Anexo A de ISO/CEI 10646-1 como

LATÍN BÁSICO

tiene la referencia de tipo ASN.1

BasicLatin

2 Un tipo character string (cadena de caracteres) constituido por los caracteres pertenecientes a la colección LATÍN BÁSICO, junto con la colección ÁRABE BÁSICO, podría definirse como sigue:

My-Character-String ::= BMPString (FROM (BasicLatin | BasicArabic))

NOTA – La construcción anterior es necesaria porque la construcción, aparentemente más sencilla, de

My-Character-String ::= BMPString (BasicLatin | BasicArabic)

sólo permitiría cadenas que fuesen enteramente LATÍN BÁSICO o ÁRABE BÁSICO, pero no una mezcla de ambos.

36 Orden canónico de los caracteres

36.1 A efectos de subtipificación de "ValueRange" y posible utilización por las reglas de codificación se especifica un ordenamiento canónico de caracteres para UniversalString, BMPString, NumericString, PrintableString, VisibleString, e IA5String.

36.2 Para los fines de esta subcláusula solamente, un carácter es una correspondencia de uno a uno con una célula en una tabla de códigos, tanto si a la célula se le ha asignado un nombre como una forma de carácter y tanto si es un carácter de control como si es un carácter de impresión, un carácter combinante o un carácter no combinante.

36.3 El orden canónico de un carácter abstracto está definido por el orden canónico de su célula.

36.4 Para UniversalString, el orden canónico de las células se define (véase la Norma ISO/CEI 10646-1) como:

256*(256*(256*(número de grupo)+(número de plano))+(número de fila))+(número de célula)

La totalidad del conjunto de caracteres contiene precisamente 256*256*256*256 caracteres. Los puntos extremos de los "ValueRangers" de las notaciones "PermittedAlphabet" (o caracteres individuales) pueden especificarse utilizando la referencia de valor ASN.1 definida en el módulo ASN1-CHARACTER-MODULE o (cuando el símbolo gráfico sea inequívoco en el contexto de la especificación) dando el símbolo gráfico de una "cstring". (ASN1 CHARACTER-MODULE se define en 35.1). No es posible especificar una célula como un punto extremo de una gama o identificar un carácter individual cuando no hayan sido asignados nombres o símbolos gráficos a dicha célula.

36.5 Para BMPString, el orden canónico de las células se define (véase la Norma ISO/CEI 10646-1) como:

256*(número de fila)+(número de célula)

El conjunto de caracteres contiene, en su totalidad, exactamente 256*256 caracteres. Los puntos extremos de las "ValueRanges" de las notaciones "PermittedAlphabet" (o caracteres individuales) pueden especificarse utilizando la referencia de valor ASN.1 definida en el módulo ASN1-CHARACTER-MODULE o (cuando el símbolo gráfico sea inequívoco en el contexto de la especificación) dando el símbolo gráfico de una "cstring". No es posible especificar una célula como un punto extremo de una gama o identificar un carácter individual cuando no se hayan asignado nombres o símbolos gráficos a esa célula.

36.6 Para NumericString, el orden canónico, creciente de izquierda a derecha, se define (véase el Cuadro 4 de la cláusula 34) como:

(espacio) 0 1 2 3 4 5 6 7 8 9

El conjunto de caracteres contiene, en su totalidad, exactamente 11 caracteres. El punto extremo de una "ValueRange" (o caracteres individuales) puede especificarse utilizando el símbolo gráfico de una "cstring".

NOTA – Este orden es el mismo que el de los caracteres correspondientes de una colección LATÍN BÁSICO de la Norma ISO/CEI 10646-1.

36.7 Para PrintableString, el orden canónico, creciente de izquierda a derecha y de arriba abajo, se define (véase el Cuadro 5 de la cláusula 34) como:

(espacio) (apóstrofo) (paréntesis izquierdo) (paréntesis derecho) (signo más) (coma) (guión) (punto) (barra de fracción) 0123456789 (dos puntos) (signo igual) (signo de interrogación final) ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz

El conjunto de caracteres contiene, en su totalidad, exactamente 74 caracteres. El punto extremo de una "ValueRange" (o caracteres individuales) puede especificarse utilizando el símbolo gráfico de una "cstring".

NOTA – Este orden es el mismo que el de los caracteres correspondientes de una colección LATÍN BÁSICO de la Norma ISO/CEI 10646-1.

36.8 Para VisibleString, el orden canónico de las células se define a partir de la codificación de la Norma ISO 646 (llamada ISO 646 ENCODING) de la siguiente manera:

(ISO 646 ENCODING) - 32

NOTA – Es decir, el orden canónico es el mismo que el de los caracteres de las células 2/0 - 7/14 de la tabla de códigos de la Norma ISO 646.

El conjunto de caracteres contiene, en su totalidad, exactamente 95 caracteres. El punto extremo de una "ValueRange" (o caracteres individuales) puede especificarse utilizando el símbolo gráfico de una "cstring".

36.9 Para IA5String, el orden canónico de las células se define a partir de la codificación ISO 646 de la siguiente manera:

(ISO 646 ENCODING)

El conjunto de caracteres contiene, en su totalidad, exactamente 128 caracteres. El punto extremo de una "ValueRange" (o caracteres individuales) puede especificarse utilizando el símbolo gráfico de una "cstring" o una referencia de valor de carácter de control de la Norma ISO 646 definida en 35.1.1.

37 Definición de tipos cadena de caracteres no restringida

Esta cláusula define un tipo cuyos valores son los valores de cualquier sintaxis abstracta de caracteres. Esta sintaxis abstracta puede formar parte del conjunto de contexto definido en un ejemplar de comunicación, o ser referenciada directamente para cada ejemplar de uso del tipo cadena de caracteres no restringida.

NOTAS

1 Una sintaxis abstracta de caracteres (y una o más sintaxis de transferencia de caracteres correspondientes) puede ser definida por cualquier organización capaz de atribuir los OBJECT IDENTIFIER de ASN.1.

2 Perfiles producidos por comunidades de interés normalmente determinarán las sintaxis abstractas de caracteres y las sintaxis de transferencia de caracteres que son soportadas para instancias específicas o grupos de instancias de CHARACTER STRING. Por lo general, la referencia a las sintaxis soportadas se incluirá en un formulario de enunciado de conformidad de realización de protocolo (PICS, *protocol implementation conformance statement*). Obsérvese que la agrupación de casos para fines de especificación de la capa aplicación puede conseguirse utilizando diferentes referencias de tipo ASN.1 (todas las cuales serían referencias para el tipo CHARACTER STRING).

37.1 El tipo cadena de caracteres no restringida (unrestricted character string) (véase 3.8.58) será referenciado por la notación "CharacterStringType":

UnrestrictedCharacterStringType ::= CHARACTER STRING

37.2 Este tipo tiene un rótulo que es de clase universal, número 29.

37.3 El tipo cadena de caracteres no restringida tiene un tipo asociado que se proporciona solamente para especificar las notaciones de valores y subtipos del tipo.

NOTA – Las reglas de codificación pueden definir un tipo diferente que se utiliza para derivar codificaciones para valores del tipo en cuestión.

37.4 El tipo asociado para la definición de valor y la subtipificación, suponiendo un entorno de rotulación automática, es (con comentarios normativos):

```
SEQUENCE {
    identification                  CHOICE {
        syntaxes                        SEQUENCE {
            abstract                        OBJECT IDENTIFIER,
            transfer                        OBJECT IDENTIFIER }
                -- Identificadores de objeto de sintaxis abstracta y de transferencia --,

        syntax                          OBJECT IDENTIFIER
                -- Un solo identificador de objeto para la identificación de la clase y la codificación --,
        presentation-context-id         INTEGER
                -- (Aplicable solamente a entornos OSI)
                -- El contexto de presentación negociado identifica la clase del valor y su codificación --,

        context-negotiation             SEQUENCE {
            presentation-context-id         INTEGER,
            transfer-syntax                 OBJECT IDENTIFIER }
                -- (Aplicable solamente a entornos OSI)
                -- Negociación de contexto en curso para que un contexto identifique la clase del valor
                -- y su codificación --,

        transfer-syntax                 OBJECT IDENTIFIER
                -- La clase del valor (por ejemplo, la especificación de que se trata del valor de
                -- un tipo ASN.1) viene fijada por el diseñador de la aplicación (y la conocen, por
                -- tanto, el emisor y el receptor). Este caso se da sobre todo para sustentar la encriptación
                -- selectiva de campo (u otras transformaciones de codificación) de un tipo ASN.1 --,

        fixed                           NULL
                -- El valor de dato es el valor de un tipo ASN.1 fijo (y conocido, por tanto, por el
                -- emisor y el receptor) -- },

    data-value-descriptor           ObjectDescriptor  OPTIONAL
                -- Esto proporciona identificación de la clase del valor legible por los seres humanos --,
```

```
string-value                    CHOICE {
    notation                        ABSTRACT-SYNTAX.&Type
        -- Esta notación de tipo se define en la Rec. UIT-T X.681 | ISO/IEC 8824-2, y tiene una
        -- notación de valor que es cualquier definición de tipo ASN.1 seguida de dos puntos (:) y
        -- de la notación de valor para ese tipo. Esta alternativa de elección se proporciona para
        -- permitir la especificación utilizando la notación, fácil para las personas, de los valores de
        -- datos que son valores de un tipo ASN.1. --,
    encoded                         OCTET STRING
        -- Esta alternativa de elección se proporciona para permitir la especificación de valores de
        -- datos que no son valores de un tipo ASN.1. -- } }
( WITH COMPONENTS {
    ... ,
    data-value-descriptor  ABSENT } )
```

NOTA – El tipo cadena de caracteres no restringida (unrestricted character string) no permite la inclusión de un valor "data-value-descriptor" (descriptor de valor de dato) junto con la "identification" (identificación). Sin embargo, la definición del tipo asociado que aquí se da sustenta las comunalidades que existen entre el tipo pdv incrustado (embedded-pdv), el tipo externo (external) y el tipo cadena de caracteres no restringida (unrestricted character string).

37.5 El componente "string-value" deberá ser designado por medio de "notation" (solamente si la sintaxis abstracta se ha definido como un tipo cadena de caracteres (character string) o un subconjunto de dicho tipo), o por medio de "encoded". En este último caso, el componente "identification" identifica una sintaxis de transferencia de caracteres para la sintaxis abstracta de caracteres, y el componente "encoded" es una representación del valor cadena de octetos del valor cadena de caracteres que utiliza la sintaxis de transferencia de caracteres "identification".

37.6 La notación de valor debe ser la notación de valor del tipo asociado.

```
UnrestrictedCharacterStringValue ::= SequenceValue      -- valor de tipo asociado definido en 37.4
```

37.7 En F.2.8 figura un ejemplo de tipo cadena de caracteres no restringida (unrestricted character string).

38 Notación para tipos definidos en las cláusulas 39-41

38.1 La notación para referenciar un tipo definido en las cláusulas 39-41 será:

```
UsefulType ::= typereference
```

donde "typereference" es una de las definidas en las cláusulas 39-41 utilizando la notación ASN.1.

38.2 El rótulo de cada "UsefulType" se especifica en las cláusulas 39-41.

39 Generalized time (tiempo generalizado) (u hora generalizada)

39.1 Este tipo será referenciado por el nombre

```
GeneralizedTime
```

39.2 Está constituido por valores que representan:

a) una fecha calendario (calendar date), definida en la Norma ISO 8601; y

b) una hora del día, con cualquiera de las precisiones definidas en la Norma ISO 8601, excepto que el valor 24 para las horas no se utilizará; y

c) el factor diferencial de hora local, definido en la Norma ISO 8601.

39.3 En ASN.1, el tipo se define como sigue:

```
GeneralizedTime ::=
    [UNIVERSAL 24] IMPLICIT VisibleString
```

con los valores de la "VisibleString" restringidos a cadenas de caracteres que son o bien:

a) una cadena que representa la fecha calendario, como se especifica en la Norma ISO 8601, con una representación de cuatro dígitos para el año, una representación de dos dígitos para el mes, y una representación de dos dígitos para el día, sin separadores, seguida de una cadena que representa la hora del día, especificada en la Norma ISO 8601, sin otros separadores que la coma o punto decimal (como se prescribe en la Norma ISO 8601), y sin ninguna Z de terminación (como se prescribe en la Norma ISO 8601); o

b) los caracteres señalados en el inciso a) anterior seguidos por una letra Z mayúscula; o

c) los caracteres señalados en el inciso a) anterior seguidos por una cadena que representa un diferencial de hora local, como se especifica en la Norma ISO 8601, sin separadores.

En el caso a), la hora representará la hora local. En el caso b), la hora representará la hora de tiempo universal coordinado. En el caso c), la parte de la cadena formada como en el caso a) representa la hora local (t_1), y el diferencial de hora (t_2) permite determinar la hora de tiempo universal coordinado como sigue:

Hora de tiempo universal coordinado: $t_1 - t_2$

EJEMPLOS

Caso a)

"19851106210627.3"
hora local 21 horas, 6 minutos, 27,3 segundos del 6 de noviembre de 1985.

Caso b)

"19851106210627.3Z"
hora UTC igual que la anterior.

Caso c)

"19851106210627.3-0500"
hora local como en el ejemplo a) con la hora local retrasada 5 horas con relación a la hora de tiempo universal coordinado.

39.4 El rótulo será el definido en 39.3.

39.5 La notación de valor será la notación de valor para la "VisibleString" definida en 39.3.

40 Tiempo universal

40.1 Este tipo será referenciado por el nombre:

UTCTime

40.2 Está constituido por valores que representan:

a) una fecha calendario; y

b) una hora con una precisión de un minuto o un segundo; y

c) (opcionalmente) un diferencial de hora local con respecto a la hora de tiempo universal coordinado.

40.3 Puede definirse el tipo, utilizando la ASN.1, de la siguiente manera:

UTCTime ::= [UNIVERSAL 23] IMPLICIT VisibleString

con los valores de la "VisibleString" restringidos a cadenas de caracteres que son la yuxtaposición de:

a) los seis dígitos YYMMDD donde YY son los dos dígitos de orden inferior del año cristiano, MM es el mes (contando enero como 01), y DD es el día del mes (01 a 31); y

b) o bien

1) los cuatro dígitos hhmm donde hh son la hora (00 a 23) y mm son los minutos (00 a 59); o

2) los seis dígitos hhmmss donde hh y mm son como en 1), y ss son los segundos (00 a 59); y

c) o bien

1) el carácter Z; o

2) uno de los caracteres + o –, seguidos de hhmm, donde hh es hora y mm es minutos.

La alternativa de b) permite variar la precisión en la especificación de la hora.

En la alternativa c) 1), la hora es hora de tiempo universal coordinado. En la alternativa c) 2), la hora (t_1) especificada en a) y b) es la hora local; el diferencial de hora (t_2) especificado en c) 2) permite expresar la hora de tiempo universal coordinado de la forma siguiente:

Hora de tiempo universal coordinado: $t_1 - t_2$

EJEMPLO 1 – Si la hora local es las 7 de la mañana del 2 de enero de 1982 y la hora de tiempo universal coordinado es las doce del mediodía del 2 de enero de 1982, el valor de UTCTime es uno de los siguientes:

"8201021200Z", o
"8201020700-0500".

EJEMPLO 2 – Si la hora local es las 7 de la mañana del 2 de enero de 2001 y la hora de tiempo universal coordinado es las doce del mediodía del 2 de enero de 2001, el valor de UTCTime es uno de los siguientes:

"0101021200Z", o
"0101020700-0500".

40.4 El rótulo será como se define en 40.3.

40.5 La notación de valor será la notación de valor para la "VisibleString" definida en 40.3.

41 Tipo descriptor de objeto

41.1 Este tipo será referenciado por el nombre

ObjectDescriptor

41.2 El tipo descriptor de objeto (object descriptor) está constituido por texto legible por los seres humanos que sirve para describir un objeto. El texto no es una identificación inequívoca del objeto, si bien deberá ser poco habitual un mismo texto para objetos diferentes.

NOTA – Se recomienda que una autoridad que asigne valores de tipo "OBJECT IDENTIFIER" a un objeto asigne también valores de tipo "ObjectDescriptor" a ese objeto.

41.3 El tipo puede definirse, utilizando la notación ASN.1, de la forma siguiente:

```
ObjectDescriptor ::= [UNIVERSAL 7] IMPLICIT GraphicString
```

La "GraphicString" contiene el texto que describe el objeto.

41.4 El rótulo será como se define en 41.3.

41.5 La notación de valor será la notación de valor para la "GraphicString" definida en 41.3.

42 Tipos constreñidos

42.1 La notación "ConstrainedType" (tipo constreñido) permite aplicar una constricción a un tipo progenitor (parent), sea para restringir su conjunto de valores a algún subtipo del progenitor, sea [dentro de un tipo conjunto (set) o un tipo secuencia (sequence)] para especificar que unas relaciones de componentes se aplican a valores del tipo progenitor y a valores de algún otro componente en el mismo valor conjunto o secuencia. Permite también asociar un identificador de excepción a una constricción.

```
ConstrainedType ::=
    Type Constraint |
    TypeWithConstraint
```

En la primera alternativa, el tipo progenitor es "Type", y la constricción es especificada por "Constraint" como se define en 42.5. La segunda alternativa se define en 42.4.

42.2 Cuando la notación "Constraint" sigue a una notación de tipo conjunto de (set-of) o secuencia de (sequence-of), se aplica al "Type" en la notación set-of o sequence-of (más interior), no al tipo conjunto de (set-of) o secuencia de (sequence-of).

NOTA – En lo que sigue, la constricción "(SIZE(1..64))" se aplica a la VisibleString, no a la SEQUENCE OF:

```
NamesOfMemberNations ::= SEQUENCE OF VisibleString (SIZE(1..64))
```

42.3 Cuando la notación "Constraint" sigue a una notación "TaggedType", la interpretación de la notación global es la misma independientemente de que el "TaggedType" o el "Type" se considere como el tipo progenitor.

42.4 Como consecuencia de la interpretación especificada en 42.2, se proporciona una notación especial para permitir aplicar una constricción a un tipo conjunto de o secuencia de. Esta notación especial es "TypeWithConstraint":

TypeWithConstraint ::=

SET Constraint OF Type |

SET SizeConstraint OF Type |

SEQUENCE Constraint OF Type |

SEQUENCE SizeConstraint OF Type

En las alternativas primera y segunda, el tipo progenitor es "SET OF Type", mientras que en las alternativas tercera y cuarta es "SEQUENCE OF Type". En las alternativas primera y tercera, la constricción es "Constraint" (véase 42.5), mientras que en las alternativas segunda y cuarta es "SizeConstraint" (véase 45.5).

NOTA – Aunque las alternativas "Constraint" comprenden las correspondientes alternativas "SizeConstraint", estas últimas, que no van encerradas entre corchetes, se proporcionan a efectos de retrocompatibilidad con la Rec. X.208 del CCITT (1988) | ISO/CEI 8824:1990.

42.5 Una constricción se especifica por la notación "Constraint":

Constraint ::= "(" ConstraintSpec ExceptionSpec ")"

ConstraintSpec ::=

SubtypeConstraint |

GeneralConstraint

Si se requiere cualquier forma de constricción que no sea "SubtypeConstraint", deberá utilizarse la alternativa "GeneralConstraint", especificada en 8.1 de la Rec. UIT-T X.682 | ISO/CEI 8824-3; en otros casos deberá utilizarse la alternativa "SubtypeConstraint" especificada en 42.6. La "ExceptionSpec" se define en la cláusula 43. Sólo deberá estar presente si la "ConstraintSpec" incluye una ocurrencia de "DummyReference" (véase 8.3 de la Rec. UIT-T X.683 | ISO/CEI 8824-4) o es una "UserDefinedConstraint" (véase la cláusula 9 de la Rec. UIT-T X.682 | ISO/CEI 8824-3).

42.6 La notación "SubtypeConstraint" es la notación "ElementSetSpec" de propósito general (cláusula 44):

SubtypeConstraint ::= ElementSetSpec

En este contexto, los elementos son valores del tipo progenitor [el gobernador (governor) del conjunto de elementos es el tipo progenitor]. Deberá haber por lo menos un elemento del conjunto.

43 Identificador de excepción

43.1 En una especificación ASN.1 compleja hay un cierto número de sitios en los que se reconoce específicamente que los codificadores han de tratar material que no está completamente especificado en ella. Estos casos surgen, en particular, como consecuencia del empleo de una constricción que se ha definido utilizando un parámetro de la sintaxis abstracta (véase la cláusula 10 de la Rec. UIT-T X.683 | ISO/CEI 8824-4).

43.2 En tales casos, el diseñador de aplicaciones tiene que identificar las acciones que han de tomarse cuando se viole alguna constricción dependiente de la realización. El identificador de excepción se proporciona como una manera inequívoca de referirse a partes de una especificación ASN.1 para indicar las acciones que deben efectuarse. El identificador consiste en un carácter "!", seguido de un tipo ASN.1 opcional y un valor de ese tipo. En ausencia del tipo, se supone que INTEGER es el tipo del valor.

43.3 Si está presente un identificador de excepción, ello indica que hay texto en el cuerpo de la norma en el que se dice cómo tratar la violación de la constricción asociada con el "!". Si está ausente, los implementadores necesitarán identificar texto que describa la acción que han de efectuar o realizarán acciones dependientes de la realización cuando se produzca una violación de la constricción.

43.4 La notación "ExceptionSpec" se define como sigue:

```
ExceptionSpec ::= "!"  ExceptionIdentification | empty

ExceptionIdentification ::= SignedNumber |
                            DefinedValue |
                            Type ":" Value
```

Las tres primeras alternativas detonan identificadores de excepción de tipo integer (entero). La cuarta alternativa denota un identificador de excepción ("Value") de tipo arbitrario ("Type").

44 Especificación de conjunto de elementos

44.1 En algunas notaciones puede especificarse un conjunto de elementos de alguna clase de elemento identificada (el gobernador). En esos casos se utiliza la notación "ElementSetSpec":

```
ElementSetSpec ::= Unions |
                   ALL  Exclusions

Unions ::= Intersections |
           UElems UnionMark Intersections

UElems ::= Unions

Intersections ::= IntersectionElements |
                  IElems IntersectionMark IntersectionElements

IElems ::= Intersections

IntersectionElements ::= Elements | Elems Exclusions

Elems ::= Elements

Exclusions ::= EXCEPT Elements

UnionMark  ::=  "|"  |  UNION

IntersectionMark  ::=  "^"  |  INTERSECTION
```

NOTAS

1 El carácter signo de intercalación "^" y la palabra INTERSECTION son sinónimos. El carácter "|" y la palabra UNION son sinónimos. Se recomienda que, por cuestión de estilo, a lo largo de la especificación se utilicen o bien los caracteres o bien las palabras. EXCEPT puede utilizarse con cualquiera de los estilos.

2 El orden de precedencia de mayor a menor es: "EXCEPT", "^", "|". Se señala que se ha especificado "ALL EXCEPT" por lo que no puede entremezclarse con las otras limitaciones sin la utilización del paréntesis entorno a "ALL EXCEPT xxx".

3 Dondequiera que aparezca "Elements", puede aparecer una constricción sin paréntesis (por ejemplo, INTEGER (1..4)) o una limitación de subtipo entre paréntesis (por ejemplo, INTEGER ((1..4 | 9))).

4 Se señala que dos operadores "EXCEPT" deben tener "|", "^", "(" o ")" separándoles, por lo que no está permitido (A EXCEPT B EXCEPT C) que debe cambiarse a ((A EXCEPT B) EXCEPT C) o (A EXCEPT (B EXCEPT C)).

5 Se señala que ((A EXCEPT B) EXCEPT C) es lo mismo que (A EXCEPT (B | C))

44.2 Los elementos que forman el conjunto son:

a) si se selecciona la primera alternativa de la "ElementSetSpec", los especificados en las "Unions" [véase b)]; de no ser así, todos los elementos del gobernador excepto los especificados en la notación "Elements" de las "Exclusions";

b) si se selecciona la primera alternativa de "Unions", los especificados en las "Intersections" [véase c)]; de no ser así, los especificados por lo menos una vez en los "UElems" o en las "Intersections";

c) si se selecciona la primera alternativa de "Intersections", los especificados en los "IntersectionElements" [véase d)]; de no ser así, los especificados por "IElems" que también son especificados por "IntersectionElements";

d) si se selecciona la primera alternativa de "IntersectionElements", los especificados en los "Elements"; de no ser así, los especificados en los "Elems" excepto los especificados en las "Exclusions".

44.3 La notación "Elements" se define como sigue:

```
Elements ::=
    SubtypeElements     |
    ObjectSetElements   |
    "(" ElementSetSpec ")"
```

Los elementos especificados por esta notación son:

a) los descritos en la cláusula 45, si se utiliza la alternativa "SubtypeElements". Esta notación sólo se utilizará cuando el gobernador sea un tipo, y el tipo que efectivamente interviene constreñirá más aún las posibilidades notacionales. En este contexto, se hace referencia al gobernador como el tipo progenitor;

b) los descritos en 12.3 de la Rec. UIT-T X.681 | ISO/CEI 8824-2, si se utiliza la notación "ObjectSetElements". Esta notación sólo se utilizará cuando el gobernador sea una clase de objeto de información;

c) los especificados por la "ElementSetSpec" si se utiliza la tercera alternativa.

45 Elementos subtipo

45.1 Generalidades

Se proporciona un cierto número de formas diferentes de notación para "SubtypeElements". Estás formas se identifican más adelante, y sus sintaxis y semántica se definen en las subcláusulas siguientes. El Cuadro 6 muestra de forma resumida cuáles son las notaciones que pueden aplicarse y a qué tipos parent (progenitor) pueden aplicarse.

```
SubtypeElements ::=
    SingleValue             |
    ContainedSubtype        |
    ValueRange              |
    PermittedAlphabet       |
    SizeConstraint          |
    TypeConstraint          |
    InnerTypeConstraints
```

45.2 Single Value (valor único)

45.2.1 La notación "SingleValue" será:

```
SingleValue ::= Value
```

donde "Value" es la notación de valor para el tipo progenitor.

45.2.2 Un "SingleValue" especifica el valor único del tipo progenitor especificado por "Value". Esta notación puede aplicarse a todos los tipos progenitores.

45.3 Contained Subtype (subtipo contenido)

45.3.1 La notación "ContainedSubtype" será:

```
ContainedSubtype ::= Includes Type
Includes ::= INCLUDES | empty
```

La alternativa "empty" (vacío) de la producción "Includes" no deberá utilizarse cuando "Type", en "ContainedSubtype", sea la notación para el tipo nulo (null).

45.3.2 Un "ContainedSubtype" especifica todos los valores del tipo progenitor (parent) resultantes de la intersección del tipo progenitor y "Type". "Type" ha de derivarse del mismo tipo insertado que el tipo progenitor.

45.4 Value Range (gama de valores)

45.4.1 La notación "ValueRange" será:

```
ValueRange ::= LowerEndpoint ".." UpperEndpoint
```

Cuadro 6 – Aplicabilidad de conjuntos de valores subtipo

Tipo	Valor único	Subtipo contenido	Gama de valores	Limitación de tamaño	Alfabeto permitido	Limitación de tipo	Subtipif. interior
Cadena de bits (bit string)	Sí	Sí	No	Sí	No	No	No
Booleano (boolean)	Sí	Sí	No	No	No	No	No
Elección (choice)	Sí	Sí	No	No	No	No	Sí
pdv incrustado (embedded-pdv)	Sí	No	No	No	No	No	Sí
Enumerado (enumerated)	Sí	Sí	No	No	No	No	No
Externo (external)	Sí	No	No	No	No	No	Sí
Ejemplar de (instance-of)	Sí	Sí	No	No	No	No	Sí
Entero (integer)	Sí	Sí	Sí	No	No	No	No
Nulo (null)	Sí	Sí	No	No	No	No	No
Campo clase de objeto (object class field)	Sí	Sí	No	No	No	No	No
Identificador de objeto (object identifier)	Sí	Sí	No	No	No	No	No
Cadena de octetos (octet string)	Sí	Sí	No	Sí	No	No	No
Abierto (open)	No	No	No	No	No	Sí	No
Real (real)	Sí	Sí	Sí	No	No	No	Sí
Cadena de caracteres restringida (restricted character string)	Sí	Sí	Sí[a)]	Sí	Sí	No	No
Secuencia (sequence)	Sí	Sí	No	No	No	No	Sí
Secuencia de (sequence-of)	Sí	Sí	No	Sí	No	No	Sí
Conjunto (set)	Sí	Sí	No	No	No	No	Sí
Conjunto de (set-of)	Sí	Sí	No	Sí	No	No	Sí
Cadena de caracteres no restringida (unrestricted character string)	Sí	No	No	Sí	No	No	Sí

a) Sólo se permite dentro del "PermittedAlphabet" de BMPString, IA5String, NumericString, PrintableString, VisibleString y UniversalString.

45.4.2 Una "ValueRange" especifica todos los valores de una gama de valores que se designan especificando los valores de los puntos extremos de la gama. Esta notación sólo puede aplicarse a tipos enteros, al PermittedAlphabet (alfabeto permitido) de determinados tipos cadena de caracteres restringida (IA5String, NumericString, PrintableString, VisibleString, BMPString y UniversalString) y a tipos reales.

NOTA – A los fines de la subtipificación, "PLUS-INFINITY" es mayor que todos los valores "NumericReal" y "MINUS-INFINITY" es menor que todos los valores "NumericReal".

45.4.3 Cada punto extremo de la gama es o bien cerrado (en cuyo caso el punto extremo está especificado) o abierto (en cuyo caso el punto extremo no está especificado). Cuando es abierto, la especificación del punto extremo incluye un símbolo menor que ("<"):

LowerEndpoint ::= LowerEndValue | LowerEndValue "<"

UpperEndpoint ::= UpperEndValue | "<" UpperEndValue

45.4.4 Un punto extremo puede también no estar especificado, en cuyo caso la gama se extiende en ese sentido tanto como lo permita el tipo progenitor:

LowerEndValue ::= Value | MIN

UpperEndValue ::= Value | MAX

45.5 Size Constraint (constricción de tamaño)

45.5.1 La notación "SizeConstraint" será:

SizeConstraint ::= SIZE Constraint

45.5.2 Una "SizeConstraint" sólo puede aplicarse a tipos bitstring, octetstring, character string, set-of o sequence-of, o a tipos formados por rotulación a partir de estos tipos.

45.5.3 La "Constraint" especifica los valores enteros permitidos para la longitud de los valores especificados, y adopta la forma de cualquier "Constraint" (constricción) que pueda aplicarse al siguiente tipo progenitor:

INTEGER (0 .. MAX)

La "Constraint" utilizará la alternativa "SubtypeConstraint" de "ConstraintSpec".

45.5.4 La unidad de medida depende del tipo progenitor, como sigue:

Tipo	*Unidad de medida*
bit string	bit
octet string	octeto
character string	carácter
set-of	valor componente
sequence-of	valor componente

NOTA – La cuenta del número de caracteres especificado en esta subcláusula para determinar el tamaño de un valor cadena de caracteres deberá distinguirse claramente de una cuenta de octetos. Una cuenta de caracteres se interpretará de acuerdo con la definición de la colección de caracteres utilizada en el tipo, en particular, en relación con las referencias a las normas, cuadros o números de registración de un registro que puedan aparecer en esa definición.

45.6 Type Constraint (constricción de tipo)

45.6.1 La notación "TypeConstraint" será:

TypeConstraint ::= Type

45.6.2 Esta notación sólo se aplica a una notación de tipo abierto y constriñe el tipo abierto a valores de "Type".

45.7 Permitted Alphabet (alfabeto permitido)

45.7.1 La notación "PermittedAlphabet" será:

PermittedAlphabet ::= FROM Constraint

45.7.2 Un "PermittedAlphabet" especifica todos los valores que pueden construirse utilizando un subalfabeto de la cadena progenitora. Esta notación sólo puede aplicarse a tipos cadena de caracteres restringida.

45.7.3 La "Constraint" es cualquiera que pueda aplicarse al tipo progenitor (véase el Cuadro 6), con la salvedad de que deberá utilizar la alternativa "SubtypeConstraint" de "ConstraintSpec". El subalfabeto incluye exactamente los caracteres que aparecen en uno o más de los valores del tipo cadena progenitor autorizados por la "Constraint".

45.8 Inner Subtyping (subtipificación interior)

45.8.1 La notación "InnerTypeConstraint" será:

InnerTypeConstraints ::=
WITH COMPONENT SingleTypeConstraint |
WITH COMPONENTS MultipleTypeConstraints

45.8.2 Un "InnerTypeConstraints" especifica solamente los valores que satisfacen una colección de constricciones sobre la presencia y/o valores de los componentes del tipo progenitor. Un valor del tipo progenitor no se especifica, a menos que satisfaga todas las constricciones expresadas o implicadas (véase 45.8.6). Esta notación puede aplicarse a los tipos set-of, sequence-of, set, sequence y choice, o a los tipos formados por rotulación a partir de ellos.

45.8.3 Para los tipos que se definen en términos de otro tipo (interior) único (set-of y sequence of), se proporciona una constricción en forma de una especificación de valor de subtipo. La notación para esta constricción es "SingleTypeConstraint":

SingleTypeConstraint ::= Constraint

La "Constraint" define un subtipo del otro tipo (interior) único. Se especifica un valor del tipo progenitor únicamente si cada valor interior pertenece al subtipo obtenido al aplicar la "Constraint" al tipo interior.

45.8.4 Para los tipos que se definen en términos de múltiples otros tipos (interiores) (choice, set y sequence) puede proporcionarse un número de constricciones sobre estos tipos interiores. La notación para estas constricciones es "MultipleTypeConstraint":

MultipleTypeConstraints ::= FullSpecification | PartialSpecification

FullSpecification ::= "{" TypeConstraints "}"

PartialSpecification ::= "{" "..." "," TypeConstraints "}"

TypeConstraints ::=
NamedConstraint |
NamedConstraint "," TypeConstraints

NamedConstraint ::=
identifier ComponentConstraint

45.8.5 "TypeConstraint" contiene una lista de constricciones sobre los tipos componentes del tipo progenitor. Para un tipo sequence, las constricciones tienen que aparecer en orden. El tipo interior a que se aplica la constricción se identifica por medio de su identificador. Para un componente dado deberá haber por lo menos una "NamedConstraint".

45.8.6 "MultipleTypeConstraints" comprende una "FullSpecification" o una "PartialSpecification". Cuando se utiliza "FullSpecification", hay una constricción de presencia implicada, de "ABSENT", en todos los tipos interiores que puedan ser constreñidos a estar ausentes (véase 45.8.9) y que no está explícitamente listada. Cuando se emplea "PartialSpecification" y el tipo progenitor es un tipo conjunto o secuencia, no hay constricciones implicadas y cualquier tipo interior puede ser omitido de la lista. Cuando se emplea "PartialSpecification" y el tipo parent es un tipo elección, hay una constricción de presencia implicada de "ABSENT" en todos los tipos interiores que no están listados explícitamente.

45.8.7 Un determinado tipo interior puede ser constreñido en cuanto a su presencia (en valores de tipo progenitor) o su valor, o en cuanto a ambos. La notación es "ComponentConstraint":

ComponentConstraint ::= ValueConstraint PresenceConstraint

45.8.8 Una constricción sobre el valor de un tipo interior se expresa por la notación "ValueConstraint":

ValueConstraint ::= Constraint | empty

La constricción se satisface por un valor del tipo progenitor únicamente si el valor interior pertenece al subtipo especificado por la "Constraint" aplicada al tipo interior.

45.8.9 Una constricción sobre la presencia de un tipo interior se expresará por la notación "PresenceConstraint":

PresenceConstraint ::= PRESENT | ABSENT | OPTIONAL | empty

El significado de estas alternativas, en la situaciones en que se permiten, se define en 45.8.9.1 a 45.8.9.3.

45.8.9.1 Si el tipo progenitor es sequence o set, un tipo componente marcado "OPTIONAL" puede ser constreñido a estar "PRESENT" (en cuyo caso la constricción se satisface únicamente si el valor componente correspondiente está presente) o a estar "ABSENT" (en cuyo caso la constricción se satisface únicamente si el valor componente está ausente) o a ser "OPTIONAL" (en cuyo caso no se imponen constricciones a la presencia del valor componente correspondiente).

45.8.9.2 Si el tipo progenitor es una elección, un tipo componente puede ser constreñido a estar "ABSENT" (en cuyo caso la constricción se satisface únicamente si el tipo componente correspondiente no se utiliza en el valor), o "PRESENT" (en cuyo caso la constricción se satisface únicamente si el tipo componente correspondiente se utiliza en el valor); deberá haber a lo sumo una palabra clave "PRESENT" en "MultipleTypeConstraints".

NOTA – Véase un ejemplo aclarador en F.4.5.

45.8.9.3 El significado de una "PresenceConstraint" vacía depende de que se esté empleando una "FullSpecification" o una "PartialSpecification":

a) en una "FullSpecification", equivale a una constricción de "PRESENT" para un componente set o sequence marcado OPTIONAL y, en otro caso, no impone ninguna otra constricción;

b) en una "PartialSpecification", no se imponen constricciones.

Anexo A

Utilización de la notación ASN.1-88/90

(Este texto es parte integrante de la presente Recomendación | Norma Internacional)

A.1 Mantenimiento

La expresión **notación ASN.1-88/90** se utiliza para referirse a la notación ASN.1 especificada en la Rec. X.208 del CCITT (1988) | ISO/CEI 8824:1990. La expresión **notación ASN.1 actual** se utiliza para referirse a la especificada en la presente Recomendación | Norma Internacional.

En el momento de publicarse la presente Recomendación | Norma Internacional se mantenía todavía la Rec. X.208 | ISO/CEI 8824. Este mantenimiento continuado depende de una Resolución anual por parte del JTC1/SC21 de la Norma ISO/CEI y no cabe esperar que sea indefinido. Se proporciona para dar tiempo a los usuarios de ASN.1 a sustituir las características (sobre todo ANY y la utilización de la notación macro) de la notación ASN.1-88/90 por la notación ASN.1 actual. (Esto puede hacerse sin cambios en los bits de la línea.)

A.2 Combinación de la notación ASN.1-88/90 con la notación ASN.1 actual

Tanto la notación ASN.1-88/90 como la notación ASN.1 actual especifican una construcción sintáctica de nivel superior que es un módulo ASN.1. Un usuario de ASN.1 escribe una colección de módulos ASN.1 y puede importar definiciones de otros módulos ASN.1.

Para cualquier módulo dado, la notación utilizada ha de estar (completamente) conforme con la notación ASN.1-88/90 o con la notación ASN.1 actual y una especificación de usuario deberá identificar claramente qué notación se está utilizando (por referencia a la Recomendación | Norma Internacional apropiada) para cada módulo incluido textualmente en la especificación del usuario.

Obsérvese que podría ocurrir que un usuario deseara modificar parte de un módulo para utilizar la nueva notación, pero dejando otras partes en la notación antigua. Esto puede conseguirse (solamente) dividiendo el módulo en dos módulos.

Cuando un módulo esté conforme con la notación ASN.1-88/90, las referencias de tipo y valor pueden importarse desde un módulo que se definió utilizando la notación ASN.1 actual. Esos tipos y valores deben estar asociados con tipos que pueden ser definidos utilizando solamente la notación ASN.1-88/90. Por ejemplo, un módulo escrito utilizando la notación ASN.1-88/90 no puede importar un valor del tipo UniversalString, ya que este tipo está definido en la notación actual pero no en ASN.1-88/90; puede sin embargo, importar valores cuyos tipos sean, por ejemplo, INTEGER, IA5String, etc.

Cuando un módulo es conforme con la notación ASN.1 actual, las referencias de tipo y valor pueden importarse desde un módulo que se definió utilizando la notación ASN.1-88/90. No se importará macro ASN.1. La notación de valor de un tipo importado solamente se utilizará en el módulo importador si están presentes identificadores para valores de SET, SEQUENCE y CHOICE utilizados en la notación de valor, y son distintos, y si en la notación de valor no se exige ningún valor del tipo ANY. No se aplicará una constricción de tipo interior a un tipo importado si el componente que se constriñe no tiene un identificador.

A.3 Paso a la notación ASN.1 actual

Cuando se modifique un módulo (escrito originalmente para cumplir con la notación ASN.1-88/90) de modo que esté conforme con la notación actual, deberán tenerse en cuenta los siguientes puntos:

a) A todos los componentes de SET y SEQUENCE y CHOICE se les deberá dar identificadores que sean inequívocos dentro de ese SET, SEQUENCE o CHOICE y esos identificadores deberán ser incluidos en la notación de valor.

NOTA – La notación de valor para un tipo CHOICE contiene un signo de dos puntos (":").

b) Todas las utilizaciones de ANY y ANY DEFINED BY deberán ser sustentadas mediante una definición de clase de objeto de información adecuada, con el ANY y el ANY DEFINED BY (y los componentes referenciados) sustituidos por referencias apropiadas a los campos de esa clase de objeto. En la mayoría de los casos, la especificación puede mejorarse en buena medida fijándose atentamente en la inserción de constricciones de tabla y relación de componentes. En muchos casos, la especificación puede mejorarse más aún si en la constricción de tabla o relación de componentes se hace un parámetro del tipo.

c) Todas las definiciones de macros serán sustituidas por la definición de una clase de objeto de información, un tipo parametrizado o un valor parametrizado. Si la cláusula WITH SYNTAX (con sintaxis) se diseña cuidadosamente en la definición de una clase de objeto de información, la notación utilizada para definir un objeto de esa clase puede hacerse muy similar a la notación definida por la utilización antigua de la notación de macro.

d) Todos los casos de utilización de un macro deberán ser sustituidos por definiciones de objetos de información equivalentes o por referencias a "ObjectClassFieldType" equivalentes, tipos parametrizados o valores parametrizados. En la mayoría de los casos, la especificación de objetos de información puede mejorarse en gran medida agrupando esas informaciones en conjuntos de objetos de información e indicando claramente si es obligatorio sustentar todos los objetos de información del conjunto y si las ampliaciones, dependientes de la realización, de ese conjunto de objetos de información han de ser acomodadas por las realizaciones receptoras, y si es así, cómo deben tratar la recepción de valores "unknown" (desconocidos). También puede ser conveniente considerar la posibilidad de que una versión posterior de la especificación del usuario amplíe el conjunto de objetos de información y orientar a los realizadores actuales respecto a cómo deben tratarse esas ampliaciones.

e) Todas las ocurrencias de EXTERNAL deben examinarse atentamente; si bien esa notación sigue siendo lícita en la ASN.1 actual, una especificación de usuario podría mejorarse, probablemente, haciendo lo siguiente:

1) Considerar la utilización de la notación INSTANCE OF (ejemplar de) (preferiblemente con una constricción de tabla que puede ser un parámetro del tipo, según se examinó anteriormente para ANY y ANY DEFINED BY) en lugar de la notación EXTERNAL (externo); en muchos casos esto no cambiará los bits en la línea.

2) Cuando se retenga EXTERNAL, la utilización de la subtipificación interior del tipo asociado (véase 31.5) puede ayudar a dar precisión a la especificación de si el empleo de identificadores del contexto de presentación está o no permitido. También son aplicables aquí comentarios anteriores (véase la cláusula 31) en los que se dan directrices respecto a qué valores de EXTERNAL se han de sustentar y qué realizaciones deben hacerlo si se reciben valores no sustentados.

3) Considerar un cambio a

CHOICE {external EXTERNAL, embedded-pdv EMBEDDED PDV}

(de nuevo con subtipificación interior, si procede) para permitir una migración por fases de las aplicaciones pares distribuidas hacia la notación actual. Esto puede influir en los bits de la línea y normalmente debería hacerse como parte de un cambio de versión en el protocolo. La utilización de EMBEDDED PDV (pdv incrustado) (sobre todo para especificaciones nuevas) dará normalmente una mayor flexibilidad, según puede verse por comparación de los tipos asociados; además, EMBEDDED PDV es codificado más eficazmente que EXTERNAL por todas las reglas de codificación especificadas en la Rec. UIT-T X.690 | ISO/CEI 8825-1.

f) Es posible mejorar la legibilidad de la notación en los módulos ASN.1 existentes (sin cambios en los bits en la línea) mediante la inserción de AUTOMATIC TAGS (rótulos automáticos) en el encabezamiento del módulo y la supresión de algunos rótulos o de la totalidad de los mismos.

NOTA – Esto es algo que hay que hacer con cuidado y con conocimiento del funcionamiento de la rotulación automática ya que, si se aplica de manera incorrecta, se producirá un cambio de los bits de la línea.

g) Si no se aplica AUTOMATIC TAGS, como se describe en f), a los módulos existentes, normalmente convendrá no añadir nuevas definiciones de tipos al módulo existente sino más bien crear un nuevo módulo (con rotulación automática) para las definiciones de tipos nuevos. De esta manera es posible aprovechar las ventajas de la rotulación automática sin afectar a los bits de la línea.

h) Deberá prestarse atención a los campos que contengan cadenas de caracteres para ver si debe emplearse la notación CHARACTER STRING, BMPString o UniversalString. Sin embargo esto cambiará, normalmente, los bits de la línea y debería hacerse como parte de un cambio de versión.

i) Los identificadores "mantissa", "base" y "exponent" han de añadirse a cualquier notación de valor real que utilice la alternativa "NumericRealValue" de la producción "RealValue". Deberá considerarse la restricción de la "base" a 2 ó 10 en la notación de tipo.

Por lo general, pueden producirse mejoras importantes en lo tocante a legilibilidad, eficacia, precisión y flexibilidad al pasar a utilizar la nueva notación ASN.1 [sobre todo si se aprovechan plenamente las ventajas de la utilización de las constricciones de tabla y relación de componentes y la parametrización, así como de la utilización de los nuevos tipos cadena de caracteres (character string)]. Se insta encarecidamente a todos los usuarios de ASN.1-88/90 a que acometan el cambio cuando revisen una especificación, o como una actividad independiente, cuando no se prevean revisiones durante algún tiempo.

Por lo general se considera un error efectuar adiciones a módulos existentes utilizando una notación no conforme con la especificación ASN.1 actual, incluso si se retienen referencias a las especificaciones ASN.1-88/90 para esos módulos. Deberán evitarse, en particular, nuevas utilizaciones de macros y de ANY o ANY DEFINED BY o nuevas construcciones de SET, SEQUENCE o CHOICE sin identificadores inequívocos.

Anexo B

Asignación por la ISO de valores de componentes OBJECT IDENTIFIER

(Este anexo es parte integrante de la presente Recomendación | Norma Internacional)

B.1 Se especifican tres arcos desde el nodo de la raíz. La asignación de valores e identificadores, y la autoridad para la asignación de valores de componentes subsiguientes, son como sigue:

Valor	*Identificador*	*Autoridad para asignaciones subsiguientes*
0	itu-t	UIT-T
1	iso	ISO
2	joint-iso-itu-t	Véase el Anexo D

NOTA – El resto de este anexo concierne solamente a la asignación de valores por la ISO.

B.2 Cada uno de los identificadores "itu-t", "iso" y "joint-iso-itu-t", asignados anteriormente, puede utilizarse como una "NameForm".

B.3 Los identificadores "itu-t" y "joint-iso-itu-t" son sinónimos de "ccitt" y "joint-iso-ccitt", respectivamente, y pueden aparecer por tanto en valores de identificadores de objeto.

B.4 Se especifican cuatro arcos desde el nodo identificado por "iso". La asignación de valores e identificadores es:

Valor	*Identificador*	*Autoridad para asignaciones subsiguientes*
0	standard	Véase B.5
2	member-body	Véase B.6
3	identified-organization	Véase B.7

Estos identificadores pueden ser usados como una "NameForm".

NOTA – El arco 1 (registration-authority, autoridad de registración) no se utiliza. En versiones anteriores estaba reservado a utilización futura, pero su empleo ha sido revocado.

B.5 Los arcos situados debajo de "standard" tendrán cada uno el valor del número de una Norma Internacional (International Standard). Cuando la Norma Internacional sea multiparte, habrá un arco adicional para el número de parte, a menos que esto esté explícitamente excluido en el texto de la Norma Internacional. Los demás arcos tendrán los valores definidos en dicha Norma Internacional.

NOTA – Si una Norma Internacional no multiparte atribuye identificadores de objeto, y después se convierte en una Norma Internacional multiparte, continuará atribuyendo identificadores de objeto como si fuera una Norma Internacional con una sola parte.

B.6 Los arcos situados inmediatamente debajo de "member-body" (organismo miembro) tendrán valores de un indicativo de país numérico de tres cifras, tal como se especifica en la Norma ISO 3166, que identifica el Organismo miembro de la ISO de ese país (véase la nota). La "NameForm" de un componente de identificador de objeto no se permite con esos identificadores. Los arcos situados debajo del "country code" no se definen en esta Recomendación (o Norma Internacional).

NOTA – La existencia de un indicativo de país en la Norma ISO 3166 no implica necesariamente que haya un Organismo miembro de la ISO que represente a ese país o que ese Organismo miembro de la ISO para dicho país administre un esquema para la atribución de componentes de identificador de objeto.

B.7 Los arcos situados inmediatamente debajo de "identified-organization" (organización identificada) tendrán valores de un designador de indicativo internacional (ICD, *international code designator*) atribuido por la autoridad de registración para la Norma ISO 6523 que identifica una organización emisora registrada específicamente por esa autoridad como atribuidora de componentes de identificador de objeto (véanse las Notas 1 y 2). Los arcos situados inmediatamente debajo de ICD tendrán valores de un "organization code" (código de organización) atribuido por la organización emisora de acuerdo con la Norma ISO 6523. Los arcos situados debajo de "organization code" (código de organización) no se definen en esta Recomendación (véase la Nota 3).

NOTAS

1 El requisito de que las organizaciones emisoras estén registradas por la autoridad de registración para la Norma ISO 6523 como atribuidoras de códigos de organización para componentes de identificador de objeto asegura que solamente se atribuyan valores numéricos de acuerdo con esta Recomendación (o Norma Internacional).

2 La declaración de que una organización emisora atribuye códigos de organización para códigos de componentes de identificador de objeto no excluye el uso de esos códigos para otros propósitos.

3 Se da por supuesto que las organizaciones identificadas por "organization code" (código de organización) definirán los arcos ulteriores de forma que se asegure la atribución única de valores.

Anexo C

Asignación por el UIT-T de valores de componentes OBJECT IDENTIFIER

(Este anexo es parte integrante de la presente Recomendación | Norma Internacional)

C.1 Se especifican tres arcos desde el nodo raíz. La asignación de valores e identificadores, y la autoridad para la asignación de valores de componentes subsiguientes, son como sigue:

Valor	*Identificador*	*Autoridad para asignaciones subsiguientes*
0	itu-t	UIT-T
1	iso	ISO
2	joint-iso-itu-t	Véase el Anexo D

NOTA – El resto de este anexo concierne solamente a la asignación de valores por el UIT-T.

C.2 Cada uno de los identificadores "itu-t", "iso" y "joint-iso-itu-t", asignados anteriormente, puede utilizarse, como una "NameForm".

C.3 Los identificadores "itu-t" y "joint-iso-itu-t" son sinónimos de "ccitt" y "joint-iso-ccitt", respectivamente, y pueden aparecer por tanto en valores de identificadores de objeto.

C.4 Se especifican cinco arcos desde el nodo identificado por "itu-t". La asignación de valores e identificadores es:

Valor	*Identificador*	*Autoridad para asignaciones subsiguientes*
0	recommendation	Véase C.5
1	question	Véase C.6
2	administration	Véase C.7
3	network-operator	Véase C.8
4	identified-organization	Véase C.9

Estos identificadores pueden ser usados como una "NameForm".

C.5 Los arcos situados debajo de "recommendation" tienen el valor de 1 a 26 con identificadores asignados de la a a la z. Los arcos situados debajo de éstos tienen los números de las Recomendaciones UIT-T y CCITT de las series identificadas por la letra. Los arcos situados debajo de éstos se determinan según sea necesario por las Recomendaciones UIT-T y CCITT. Los identificadores a a z pueden utilizarse como "NameForm".

C.6 Los arcos situados debajo de "question" tienen valores correspondientes a las Comisiones de Estudio del UIT-T, calificadas por Periodos de Estudios. El valor se calcula mediante la fórmula:

study group number + (Period * 32)

donde "Period" tiene el valor 0 para 1984-1988, 1 para 1988-1992, etc., y el multiplicador es el decimal 32.

Los arcos situados debajo de cada Comisión de Estudio tienen los valores correspondientes a las Cuestiones asignadas a dicha Comisión de Estudio. Los arcos situados debajo de éstos están determinados como lo exija el Grupo (por ejemplo, Grupo de Trabajo o Grupo de Relator especial) asignado al estudio de la Cuestión.

C.7 Los arcos situados debajo de "administration" tienen los valores de los indicativos de país para datos (DCC, *data country codes*) de la Recomendación X.121. Los arcos situados debajo de éstos se determinan como lo exija la administración del país identificado por el DCC de la Recomendación X.121.

C.8 Los arcos situados debajo de "network-operator" tienen el valor de los códigos de identificación de red de datos (DNIC, *data network identification code*) de la Recomendación X.121. Los arcos situados debajo de éstos están determinados como lo exija la administración o empresa de explotación reconocida (EER) identificada por el DNIC.

C.9 Los arcos debajo de "identified-organization" son valores asignados por la Oficina de Normalización de las Telecomunicaciones (TSB) de la UIT. Los arcos debajo de éste son determinados, según se necesiten, por las organizaciones identificadas por el valor de la UIT.

NOTA – Cabe esperar razonablemente que entre los tipos de organizaciones que puedan encontrar de utilidad este arco figuren las siguientes:

- EER que no exploten una red pública de datos;
- organizaciones científicas e industriales;
- organizaciones regionales de normalización; y
- organizaciones multinacionales.

Anexo D

Asignación conjunta de valores de componentes OBJECT IDENTIFIER

(Este anexo es parte integrante de la presente Recomendación | Norma Internacional)

D.1 Se especifican tres arcos desde el nodo raíz. La asignación de valores e identificadores, y la autoridad para la asignación de valores de componentes subsiguientes son como sigue:

Valor	*Identificador*	*Autoridad para asignaciones subsiguientes*
0	itu-t	UIT-T
1	iso	ISO
2	joint-iso-itu-t	Véase el Anexo D

NOTA – El resto de este Anexo concierne solamente a la asignación conjunta de valores por ISO/UIT-T.

D.2 Cada uno de los identificadores "itu-t", "iso" y "joint-iso-itu-t", asignados anteriormente, puede utilizarse como una "NameForm".

D.3 Los identificadores "itu-t" y "joint-iso-itu-t" son sinónimos de "ccitt" y "joint-iso-ccitt", respectivamente, y pueden aparecer por tanto en valores de identificadores de objeto.

D.4 Los arcos situados debajo de "joint-iso-itu-t" tienen valores, que están asignados y acordados cada cierto tiempo por la ISO y el UIT-T para identificar áreas de actividad normalizadora de ISO/UIT-T, de acuerdo con los "procedimientos para la asignación de valores de componentes de identificador de objeto para uso conjunto ISO/UIT-T[2)]".

D.5 Los arcos situados debajo de cada arco identificado por el mecanismo de D.4 serán atribuidos de acuerdo con los mecanismos establecidos cuando se atribuye el arco.

NOTA – Se espera que esto incluirá la delegación de autoridad para el acuerdo conjunto de los Relatores del UIT-T y de la ISO para el ámbito conjunto de trabajo.

D.6 Las Normas Internacionales y las Recomendaciones iniciales en sectores de actividad conjunta ISO/UIT-T requieren atribuir OBJECT IDENTIFIERS con anterioridad al establecimiento de los procedimientos de D.4, y a partir de ahí proceder a la atribución de acuerdo con los Anexos B y C. Los objetos codificados de esta forma por Normas Internacionales o Recomendaciones no tendrán que cambiar sus OBJECT IDENTIFIERS cuando se establezcan los procedimientos de D.4.

---

2) La Registration Authority (autoridad de registración) para la asignación de valores de componentes de identificador de objeto para uso conjunto ISO/UIT-T es el American National Standards Institute (ANSI), 91 West 42nd St., New York, NY 10036, Estados Unidos de América.

Anexo E

Asignación de valores de identificador de objeto

(Este anexo es parte integrante de la presente Recomendación | Norma Internacional)

En esta Recomendación | Norma Internacional se asignan los siguientes valores:

Subcláusula	**Valor de identificador de objeto**
34.3	{ joint-iso-ccitt asn1(1) specification(0) characterString(1) numericString(0) }
	Valor de descriptor de objeto
	"NumericString ASN.1 type"
Subcláusula	**Valor de identificador de objeto**
34.5	{ joint-iso-ccitt asn1(1) specification(0) characterString(1) printableString(1) }
	Valor de descriptor de objeto
	"PrintableString ASN.1 type"
Subcláusula	**Valor de identificador de objeto**
35.1	{ joint-iso-ccitt asn1(1) specification(0) modules(0) iso10646(0) }
	Valor de descriptor de objeto
	"ASN1 Character Module"

Anexo F

Ejemplos y sugerencias

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

Este anexo contiene ejemplos de la utilización de ASN.1 en la descripción de estructuras de datos (hipotéticas). Contiene también sugerencias, o directrices, para el uso de diversas prestaciones (features) de ASN.1. Salvo que se indique otra cosa, se supone un entorno de AUTOMATIC TAGS (rótulos automáticos).

F.1 Ejemplo de un registro de personal

La utilización de ASN.1 se ilustra por medio de un registro de personal hipotético simple.

F.1.1 Descripción informal de registro de personal

A continuación se presenta la estructura del registro de personal (personnel record) y su valor para un determinado individuo.

Name:	John P Smith
Title:	Director
Employee Number:	51
Date of Hire:	17 September 1971
Name of Spouse:	Mary T Smith
Number of Children:	2
Child Information	
Name:	Ralph T Smith
Date of Birth	11 November 1957
Child Information	
Name:	Susan B Jones
Date of Birth	17 July 1959

F.1.2 Descripción ASN.1 de la estructura del registro

La estructura de cada registro de personal se describe formalmente a continuación utilizando la notación normalizada para tipos de datos.

```
PersonnelRecord ::= [APPLICATION 0] SET
{ name            Name,
  title           VisibleString,
  number          EmployeeNumber,
  dateOfHire      Date,
  nameOfSpouse    Name,
  children        SEQUENCE OF ChildInformation DEFAULT { }
}

ChildInformation ::= SET
{ name            Name,
  dateOfBirth     Date
}

Name ::= [APPLICATION 1] SEQUENCE
{ givenName       VisibleString,
  initial         VisibleString,
  familyName      VisibleString
}

EmployeeNumber ::= [APPLICATION 2] INTEGER

Date ::= [APPLICATION 3] VisibleString  -- YYYY MMDD
```

Este ejemplo ilustra un aspecto de la descomposición analítica (parsing) de la sintaxis ASN.1. La construcción sintáctica "DEFAULT" (por defecto) sólo puede aplicarse a un componente de una "SEQUENCE" (secuencia) o un "SET" (conjunto); no puede aplicarse a un elemento de una "SEQUENCE OF" (secuencia de). Así, el "DEFAULT { }" en "PersonnelRecord" se aplica a "children", no a "ChildInformation".

F.1.3 Descripción ASN.1 de un valor de registro

El valor del registro de personal de John Smith se describe formalmente a continuación utilizando la notación normalizada para valores de datos.

```
{ name            {givenName "John", initial "P", familyName "Smith"},
  title           "Director",
  number          51,
  dateOfHire      "19710917",
  nameOfSpouse    {givenName "Mary", initial "T", familyName "Smith"},
  children
    { {name  {givenName "Ralph", initial "T", familyName "Smith"} ,
       dateOfBirth "19571111"},
       {name  {givenName "Susan", initial "B", familyName "Jones"} ,
       dateOfBirth "19590717" }
    }
}
```

F.2 Directrices para la utilización de la notación

Los tipos de datos y la notación formal definidos en la presente Recomendación | Norma Internacional son flexibles; su utilización permite diseñar una amplia gama de protocolos. Esta flexibilidad, sin embargo, puede llevar a veces a confusión, especialmente cuando la notación se utiliza por primera vez. Este anexo procura minimizar la confusión dando directrices y ejemplos para el empleo de la notación. Se ofrecen una o más directrices de uso para cada uno de los tipos de datos incorporados (built-in). Los tipos cadena de caracteres (por ejemplo, VisibleString) y los tipos definidos en las cláusulas 39 a 41 no se tratan aquí.

F.2.1 Booleano

F.2.1.1 Se utilizará un tipo booleano (boolean) para modelar los valores de una variable lógica (es decir, de dos estados), por ejemplo la respuesta a una pregunta del tipo sí o no.

EJEMPLO

```
Employed ::= BOOLEAN
```

F.2.1.2 Al asignar un nombre de referencia a un tipo boolean se elegirá uno que describa el estado **true** (verdadero).

EJEMPLO

```
Married ::= BOOLEAN
```

no

```
MaritalStatus ::= BOOLEAN
```

F.2.2 Entero

F.2.2.1 Se utilizará un tipo entero (integer) para modelar los valores (para todos los fines prácticos, sin limitación de magnitud) de una variable cardinal o entera.

EJEMPLO

```
CheckingAccountBalance ::= INTEGER      -- en centavos; negativo significa saldo deudor

balance CheckingAccountBalance ::= 0
```

F.2.2.2 Se definirán los valores máximo y mínimo permitidos de un tipo integer (entero) como valores distinguidos.

EJEMPLO

```
DayOfTheMonth ::= INTEGER {first(1), last(31)}

today DayOfTheMonth ::= first

unknown DayOfTheMonth ::= 0
```

Se señala que los números denominados "first" (primero) y "last" (último) se eligieron debido a su significado semántico para el lector y no excluyen la posibilidad de que DayOfTheMonth tenga otros valores que pueden ser menores que 1, superiores a 31 o entre 1 y 31.

Para restringir el valor de DayOfTheMonth a "first" y "last" simplemente, habría que escribir:

```
DayOfTheMonth ::= INTEGER {first(1), last(31)} (first | last)
```

y para restringir el valor de DayOfTheMonth a todos los valores entre 1 y 31, inclusive, habría que escribir:

```
DayOfTheMonth ::= INTEGER {first(1), last(31)} (first .. last)

dayOfTheMonth DayOfTheMonth ::= 4
```

F.2.3 Enumerado

F.2.3.1 Se utilizará un tipo enumerado (enumerated) para modelar los valores de una variable con tres o más estados. Se asignarán valores a partir de cero si la única restricción es la distinción.

EJEMPLO

```
DayOfTheWeek ::= ENUMERATED {sunday(0), monday(1), tuesday(2),
                             wednesday(3), thursday(4), friday(5), saturday(6)}

firstDay DayOfTheWeek ::= sunday
```

Se señala que, si bien las enumeraciones "sunday", "monday", etc., se eligieron debido a su significado semántico para el lector, DayOfTheWeek está limitado a suponer uno de estos valores y no otro. Además, sólo el nombre "sunday", "monday", etc., puede ser asignado a un valor; no se permiten los valores enteros equivalentes.

F.2.3.2 Se utilizará un tipo enumerado (enumerated) para modelar los valores de una variable que tiene en ese momento dos estados, pero que puede tener más estados en una futura versión del protocolo.

EJEMPLO

```
MaritalStatus ::= ENUMERATED {single(0), married(1)}
```

en anticipación de

```
MaritalStatus ::= ENUMERATED {single(0), married(1), widowed(2)}
```

F.2.4 Real

F.2.4.1 Se utilizará un tipo real para modelar un número aproximado.

EJEMPLO

```
AngleInRadians ::= REAL

pi REAL ::= {mantissa 314159265358979323846264338327 9, base 10, exponent –30}
```

F.2.4.2 Los diseñadores de aplicaciones quizás deseen asegurar el pleno interfuncionamiento con valores reales a pesar de las diferencias entre equipos físicos de coma flotante y entre decisiones relativas a la realización respecto a la utilización, por ejemplo, de coma flotante de longitud simple o longitud doble para una aplicación. Esto puede conseguirse como a continuación se indica:

```
App-X-Real ::= REAL (WITH COMPONENTS {
                mantissa (–16777215..16777215),
                base (2),
                exponent (–125..128) } )
-- Los emisores no deberán transmitir valores fuera de estas gamas y los
-- receptores conformes deberán ser capaces de recibir y tratar todos
-- los valores de estas gamas.

girth App-X-Real ::= {mantissa 16, base 2, exponent 1}
```

F.2.5 Cadena de bits

F.2.5.1 Se utilizará un tipo de cadena de bits (bit string) para modelar datos binarios cuyo formato y longitud no están especificados, o lo están en alguna otra parte, y cuya longitud en bits no es necesariamente un múltiplo de ocho.

EJEMPLO

```
G3FacsimilePage ::= BIT STRING
-- una secuencia de bits conforme con la Recomendación T.4.

image G3FacsimilePage ::= '100110100100001110110'B

trailer BIT STRING ::= '0123456789ABCDEF'H

body1 G3FacsimilePage ::= '1101'B

body2 G3FacsimilePage ::= '1101000'B
```

Obsérvese que "body1" y "body2" son valores abstractos distintos, porque los bits 0 finales son significativos (ya que no hay "NamedBitList" en la definición de G3FacsimilePage).

F.2.5.2 Se utilizará un tipo cadena de bits (bit string) con una constricción de tamaño para modelar los valores de un campo de bits de tamaño fijo.

EJEMPLO

```
BitField ::= BIT STRING (SIZE (12))

map1 BitField ::= '100110100100'B

map2 BitField ::= '9A4'H

map3 BitField ::= '1001101001'B     -- Ilegal - viola la constricción de tamaño
```

Obsérvese que "map1" y "map2" son el mismo valor abstracto, porque los cuatro bits finales de "map2" no son significativos.

F.2.5.3 Se utilizará un tipo cadena de bits (bit string) para modelar los valores de un **bit map** (mapa de bits), una colección ordenada de variables lógicas que indican si una codificación particular se cumple para cada colección de objetos correspondientemente ordenada.

```
DaysOfTheWeek ::=       BIT STRING {
                        sunday(0), monday (1), tuesday(2),
                        wednesday(3), thursday(4), friday(5),
                        saturday(6) } (SIZE (0..7))

sunnyDaysLastWeek1 DaysOfTheWeek ::= {sunday, monday, wednesday}
sunnyDaysLastWeek2 DaysOfTheWeek ::= '1101'B
sunnyDaysLastWeek3 DaysOfTheWeek ::= '1101000'B

sunnyDaysLastWeek4 DaysOfTheWeek ::= '11010000'B        -- Ilegal - viola la constricción de tamaño
```

Obsérvese que si el valor de la cadena de bits tiene una longitud inferior a 7 bits, los bits que faltan indican día nublado para esos días, por lo que los tres primeros valores anteriores tienen el mismo valor abstracto.

F.2.5.4 Se utilizará un tipo cadena de bits (bit string) para modelar los valores de un **bit map** (mapa de bits), una colección ordenada de tamaño fijo de variables lógicas que indican si una condición particular se cumple para cada colección de objetos correspondientemente ordenada.

```
DaysOfTheWeek ::=       BIT STRING {
                        sunday(0), monday (1), tuesday(2),
                        wednesday(3), thursday(4), friday(5),
                        saturday(6) } (SIZE (7))
sunnyDaysLastWeek1 DaysOfTheWeek ::= {sunday, monday, wednesday}
sunnyDaysLastWeek2 DaysOfTheWeek ::= '1101'B        -- Ilegal - viola la constricción de tamaño
sunnyDaysLastWeek3 DaysOfTheWeek ::= '1101000'B

sunnyDaysLastWeek4 DaysOfTheWeek ::= '11010000'B        -- Ilegal - viola la constricción de tamaño
```

Obsérvese que los valores primero y tercero tienen el mismo valor abstracto.

F.2.5.5 Se utilizará un tipo cadena de bits (bit string) con bits denominados para modelar los valores de una colección de variables lógicas relacionadas.

EJEMPLO

```
PersonalStatus ::= BIT STRING
    {married(0), employed(1), veteran(2), collegeGraduate(3)}

billClinton PersonalStatus ::= {married, employed, collegeGraduate}

hillaryClinton PersonalStatus ::= '110100'B
```

Obsérvese que "billClinton" y "hillaryClinton" tienen los mismos valores abstractos.

F.2.6 Cadena de octetos

F.2.6.1 Se utilizará un tipo cadena de octetos (octet string) para modelar datos binarios cuyo formato y longitud no están especificados, o están especificados en alguna otra parte, y cuya longitud en bits es un múltiplo de ocho.

EJEMPLO

```
G4FacsimileImage ::= OCTET STRING
-- una secuencia de octetos de acuerdo con las
-- Recomendaciones T.5 y T.6

image G4FacsimilePage ::= '3FE2EBAD471005'H
```

F.2.6.2 Se utilizará un tipo cadena de caracteres (character string) con preferencia a un tipo cadena de octetos cuando se disponga de uno apropiado.

EJEMPLO

```
Surname ::= PrintableString

president Surname ::= "Clinton"
```

F.2.7 UniversalString y BMPString

Se utilizará el tipo BMPString para modelar cualquier cadena de información que conste únicamente de caracteres procedentes del plano multiidioma básico (BMP, *basic multilingual plane*) de la Norma ISO/CEI 10646-1, y UniversalString para modelar cualquier cadena que conste de caracteres de la Norma ISO/CEI 10646-1 no limitados al BMP.

F.2.7.1 Se utilizará "Level1" o "Level2" para denotar que el nivel de realización impone restricciones a la utilización de caracteres combinantes.

EJEMPLO

```
RussianName ::= Cyrillic (Level1)                 -- RussianName no emplea caracteres combinantes

SaudiName ::= BasicArabic (SIZE (1..100) ^ Level2)  -- SaudiName emplea un subconjunto de
                                                    -- caracteres combinantes
```

F.2.7.2 Puede expandirse una colección para que sea un subconjunto seleccionado (es decir, incluir todos los caracteres en la colección LATÍN BÁSICO) utilizando la "UnionMark" (véase la cláusula 44).

EJEMPLO

```
KatakanaAndBasicLatin ::= UniversalString (FROM(Katakana | BasicLatin))
```

F.2.8 CHARACTER STRING

Se utilizará el tipo cadena de caracteres no restringida (unrestricted character string) para modelar cualquier cadena de información que no pueda modelarse utilizando uno de los tipos cadena de caracteres restringida. Hay que asegurarse de especificar el repertorio de caracteres y su codificación en octetos.

EJEMPLO

```
PackedBCDString ::= CHARACTER STRING ( WITH COMPONENTS {
                                          identification  ( WITH COMPONENTS {
                                                              fixed  PRESENT } )
                                       -- La sintaxis abstracta y de transferencia deberán ser
                                       -- packedBCDStringAbstractSyntax y
                                       -- packedBCDStringTransferSyntax definidas a continuación
                                     } )

-- valor de identificador de objeto para mi sintaxis abstracta de caracteres (conjunto de
-- caracteres) cuyo alfabeto es los dígitos 0 a 9
packedBCDStringAbstractSyntaxId  OBJECT  IDENTIFIER ::=
    { joint-iso-ccitt xxx(999) yyy(999) zzz(999) packedBCD(999) charSet(0) }

-- valor de identificador de objeto para mi sintaxis de transferencia de caracteres que empaqueta
-- dos dígitos por octeto, cada dígito codificado como 0000 a 1001, 1111₂ se utiliza para relleno.
packedBCDStringTransferSyntaxId  OBJECT IDENTIFIER ::=
    { joint-iso-ccitt xxx(999) yyy(999) zzz(999) packedBCD(999) characterTransferSyntax(1) }
```

```
-- La codificación de PackedBCDString contendrá solamente la codificación definida de
-- los caracteres, con cualquier campo de longitud que se necesite y, en el caso de BER,
-- con un campo que lleve el rótulo. Los valores de identificadores de objeto no se llevan
-- ya que se ha especificado "fixed" (fijo).
```

NOTA – Las reglas de codificación no necesariamente codifican valores del tipo CHARACTER STRING en una forma tal que incluya siempre los valores de identificadores de objeto, si bien garantizan la preservación del valor abstracto en la codificación.

F.2.9 Nulo

Se utilizará un tipo nulo (null) para indicar la ausencia efectiva de un componente de una secuencia.

EJEMPLO

```
PatientIdentifier ::= SEQUENCE {
    name            VisibleString,
    roomNumber      CHOICE {
        room                INTEGER,
        outPatient          NULL  -- if an out-patient  --
    }
}

lastPatient PatientIdentifier ::= {
    name                            "Jane Doe",
    roomNumber                      outPatient : NULL
}
```

F.2.10 Secuencia y secuencia de

F.2.10.1 Se utilizará un tipo secuencia de (sequence-of) para modelar una colección de variables del mismo tipo, de número elevado o impredecible, y cuyo orden sea significativo.

EJEMPLO

```
NamesOfMemberNations ::= SEQUENCE OF VisibleString
-- en orden alfabético

firstTwo  NamesOfMemberNations ::= {"Australia", "Austria"}
```

F.2.10.2 Se utilizará un tipo secuencia (sequence) para modelar una colección de variables del mismo tipo, de número conocido y pequeño, y cuyo orden es significativo, siempre que sea poco probable que la constitución de la colección cambie de una versión de protocolo a la siguiente.

EJEMPLO

```
NamesOfOfficers ::= SEQUENCE {
    president       VisibleString,
    vicePresident   VisibleString,
    secretary       VisibleString}

acmeCorp NamesOfOfficers ::= {
    president               "Jane Doe",
    vicePresident           "John Doe",
    secretary               "Joe Doe"}
```

F.2.10.3 Se utilizará un tipo secuencia (sequence) para modelar una colección de variables de tipos diferentes, de número conocido y pequeño, y cuyo orden es significativo, siempre que sea poco probable que la constitución de la colección cambie de una versión del protocolo a la siguiente.

EJEMPLO

```
Credentials ::= SEQUENCE {
    userName        VisibleString,
    password        VisibleString,
    accountNumber   INTEGER}
```

F.2.11 Conjunto y conjunto de

F.2.11.1 Se utilizará un tipo conjunto (set) para modelar una colección de variables cuyo número es conocido y pequeño y cuyo orden no es significativo. Si la rotulación automática no está en vigor, se identificará cada variable con un rótulo específico al contexto como se muestra a continuación. (Con rotulación automática, los rótulos no son necesarios.)

EJEMPLO

```
UserName ::= SET {
    personalName        [0] VisibleString,
    organizationName    [1] VisibleString,
    countryName         [2] VisibleString}

user UserName ::= {
    countryName         "Nigeria",
    personalName        "Jonas Maruba",
    organizationName    "Meteorology, Ltd."}
```

F.2.11.2 Se utilizará un tipo conjunto (set) con "OPTIONAL" para modelar una colección de variables que es un subconjunto (propio o impropio) de otra colección de variable de número conocido y razonablemente pequeño y cuyo orden no es significativo. Si la rotulación automática no está en vigor, se identificará cada variable con un rótulo específico al contexto como se indica a continuación. (Con rotulación automática, los rótulos no son necesarios.)

EJEMPLO

```
UserName ::= SET {
    personalName        [0] VisibleString,
    organizationName    [1] VisibleString OPTIONAL
        -- por defecto el de la organización en cuestión --,
    countryName         [2] VisibleString OPTIONAL
        -- por defecto el del país en cuestión -- }
```

F.2.11.3 Se utilizará un tipo conjunto (set) para modelar una colección de variables cuya constitución pueda presumiblemente cambiar de una versión del protocolo a la siguiente. Se identificará cada variable rotulándola específicamente según el contexto para retener el control de los rótulos utilizados.

EJEMPLO

```
UserName ::= SET {
    personalName        [0] VisibleString,
    organizationName    [1] VisibleString OPTIONAL ,
        -- por defecto el de la organización en cuestión
    countryName         [2] VisibleString OPTIONAL
        -- por defecto el del país en cuestión
        -- otros atributos opcionales quedan en estudio --}

user UserName ::= { personalName  "Jonas Maruba" }
```

F.2.11.4 Se utilizará un tipo conjunto de (set-of) para modelar una colección de variables cuyos tipos son los mismos y cuyo orden no es significativo.

EJEMPLO

```
Keywords ::= SET OF VisibleString  -- en orden arbitrario

someASN1Keywords Keywords ::= {"INTEGER", "BOOLEAN", "REAL"}
```

F.2.12 Rotulado

Antes de la introducción de la construcción AUTOMATIC TAGS (rótulos automáticos), las especificaciones ASN.1 contenían rótulos frecuentemente. En las subcláusulas que siguen se describe la manera según la cual se aplicaba la rotulación típicamente. Con la introducción de AUTOMATIC TAGS, las nuevas especificaciones ASN.1 de usuario no necesitan hacer uso de la notación de rótulo, si bien aquellas que modifiquen notación antigua quizá tengan que ocuparse de los rótulos.

F.2.12.1 Los rótulos de clase universal sólo se utilizan en esta Recomendación | Norma Internacional. La notación [UNIVERSAL 30] (por ejemplo) se proporciona únicamente para facilitar la precisión de la definición de los tipos útiles normalizados internacionalmente. No deberá utilizarse en otro sitio.

F.2.12.2 Un estilo de utilización de los rótulos, que se da frecuentemente, consiste en asignar un rótulo de clase de aplicación exactamente una vez en la totalidad de la especificación, utilizándolo para especificar un tipo que se emplea de una manera amplia y dispersa dentro de la especificación. También se utiliza frecuentemente (solamente una vez) un rótulo de clase de aplicación para rotular los tipos de la CHOICE (elección) más externa de una aplicación, proporcionando la identificación de mensajes individuales mediante el rótulo de clase de aplicación. Lo que sigue es un ejemplo de utilización en el primer caso.

EJEMPLO

```
FileName ::= [APPLICATION 8] SEQUENCE {
    directoryName              VisibleString,
    directoryRelativeFileName  VisibleString}
```

F.2.12.3 La rotulación específica al contexto se aplica frecuentemente de manera algorítmica a todos los componentes de un SET, SEQUENCE o CHOICE. Se señala, no obstante, que la prestación AUTOMATIC TAGS facilita esto al usuario.

EJEMPLO

```
CustomerRecord ::= SET {
    name              [0] VisibleString,
    mailingAddress    [1] VisibleString,
    accountNumber     [2] INTEGER,
    balanceDue        [3] INTEGER  --  en centavos  --}

CustomerAttribute ::= CHOICE {
    name              [0] VisibleString,
    mailingAddress    [1] VisibleString,
    accountNumber     [2] INTEGER,
    balanceDue        [3] INTEGER  --  en centavos  --}
```

F.2.12.4 La rotulación de clase privada no deberá utilizarse, normalmente, en las especificaciones normalizadas internacionalmente (aunque esto no puede prohibirse). Las aplicaciones producidas por una empresa utilizarán normalmente clases de rótulos de aplicación y específicos al contexto. No obstante, puede haber alguna vez casos en los que una especificación específica a la empresa trate de ampliar una especificación normalizada internacionalmente y, en tal caso, la utilización de rótulos de clase privada puede resultar ventajosa al proteger parcialmente la especificación específica a la empresa frente a los cambios de la especificación normalizada internacionalmente.

EJEMPLO

```
AcmeBadgeNumber ::= [PRIVATE 2] INTEGER

badgeNumber AcmeBadgeNumber ::= 2345
```

F.2.12.5 La utilización textual de IMPLICIT en cada uno de los rótulos se encuentra, por lo general, únicamente en las especificaciones antiguas. BER produce una representación menos compacta cuando se emplea la rotulación explícita que cuando se emplea la implícita. PER produce la misma codificación compacta en ambos casos. Con BER y rotulación explícita, hay más visibilidad del tipo subyacente (INTEGER, REAL, BOOLEAN, etc.) en los datos codificados. En estas directrices se ha utilizado la rotulación implícita en los ejemplos en que está autorizado a hacerlo. Dependiendo de la regla de codificación, esto puede dar como resultado una representación compacta, muy conveniente en algunas aplicaciones. En otras aplicaciones, la compacidad puede ser menos importante que, por ejemplo, la aptitud para efectuar una verificación de tipos fuerte (strong type-checking). En el último caso, se utilizará la rotulación explícita.

EJEMPLO

```
CustomerRecord ::= SET {
    name              [0] IMPLICIT VisibleString,
    mailingAddress    [1] IMPLICIT VisibleString,
    accountNumber     [2] IMPLICIT INTEGER,
    balanceDue        [3] IMPLICIT INTEGER  --  en centavos  --}

CustomerAttribute ::= CHOICE {
    name              [0] IMPLICIT VisibleString,
    mailingAddress    [1] IMPLICIT VisibleString,
    accountNumber     [2] IMPLICIT INTEGER,
    balanceDue        [3] IMPLICIT INTEGER  --  en centavos  --}
```

F.2.12.6 El criterio respecto a la utilización de rótulos en las nuevas especificaciones ASN.1 de usuario que hagan referencia a esta Recomendación | Norma Internacional es muy simple: NO UTILIZAR RÓTULOS. Poner AUTOMATIC TAGS (rótulos automáticos) en el encabezamiento del módulo y olvidarse a continuación de los rótulos. Si se necesita añadir componentes nuevos a los SET, SEQUENCE o CHOICE en una versión posterior, se añaden al final.

F.2.13 Elección

F.2.13.1 Se utilizará un CHOICE (elección) para modelar una variable seleccionada de una colección de variables cuyo número es conocido y reducido.

EJEMPLO

```
FileIdentifier ::= CHOICE {
    relativeName        VisibleString,
        -- nombre de file (fichero) (por ejemplo, "MarchProgressReport")
    absoluteName        VisibleString,
        -- nombre de file (fichero) y directorio contenedor
        -- (por ejemplo, "<Williams>MarchProgressReport")
    serialNumber        INTEGER
        -- identificador asignado por el sistema para file (fichero) --}

file FileIdentifier ::= serialNumber : 106448503
```

F.2.13.2 Se utilizará un CHOICE (elección) para modelar una variable seleccionada de una colección de variables cuya constitución vaya a cambiar probablemente de una versión del protocolo a la siguiente.

EJEMPLO

```
FileIdentifier ::= CHOICE {
    relativeName    VisibleString,
        -- nombre de file (fichero) (por ejemplo, "MarchProgressReport")
    absoluteName    VisibleString
        -- nombre de file (fichero) y directorio contenedor
        -- (por ejemplo, "<Williams>MarchProgressReport")
        -- otras formas de identificadores de file (fichero) quedan en estudio  --}
```

F.2.13.3 Cuando la rotulación implícita sea la norma para una determinada aplicación de esta Recomendación | Norma Internacional se utilizará un CHOICE de un solo tipo donde se haya previsto la posibilidad de permitir más de un tipo en el futuro.

EJEMPLO

```
Greeting ::= [APPLICATION 12] CHOICE {
    postCard    VisibleString}
```

en anticipación de:

```
Greeting ::= [APPLICATION 12] CHOICE {
    postCard    VisibleString,
    recording   Voice }
```

F.2.13.4 Cuando un valor elección (choice) está anidado dentro de otro valor elección, se requieren múltiples caracteres dos puntos (:)

EJEMPLO

```
Greeting ::= [APPLICATION 12] CHOICE {
    postCard    VisibleString,
    recording   Voice }

Voice ::= CHOICE {
    english             OCTET STRING,
    swahili             OCTET STRING }

myGreeting Greeting ::= recording : english : '019838547E0'H
```

F.2.14 Tipo selección

F.2.14.1 Se utilizará un tipo selección (selection) para modelar una variable cuyo tipo sea el de algunas alternativas determinadas de un CHOICE definido anteriormente.

F.2.14.2 Considérese la definición:

```
FileAttribute ::= CHOICE {
    date-last-used      INTEGER,
    file-name           VisibleString}
```

es posible, entonces, la siguiente definición:

```
AttributeList ::= SEQUENCE {
    first-attribute     date-last-used  < FileAttribute,
    second-attribute    file-name < FileAttribute }
```

con una posible notación de valor de

```
listOfAttributes AttributeList ::=  {
    first-attribute     27,
    second-attribute    "PROGRAM" }
```

F.2.15 Tipo campo de clase de objeto

F.2.15.1 Se utilizará un tipo campo de clase de objeto (object class field), es decir, un tipo abierto, para identificar un tipo definido mediante una clase de objeto de información (véase la Rec. UIT-T X.681 | ISO/CEI 8824-2). Por ejemplo, campos de la clase de objeto de información ATTRIBUTE pueden ser utilizados en la definición de un tipo, Attribute.

EJEMPLO

```
ATTRIBUTE ::= CLASS
{
    &AttributeType,
    &attributeId        OBJECT IDENTIFIER UNIQUE
}

Attribute ::= SEQUENCE {
    attributeID  ATTRIBUTE.&attributeId,                -- esto está constreñido normalmente
    attributeValue      ATTRIBUTE.&AttributeType         -- esto está constreñido normalmente
}
```

Tanto ATTRIBUTE.&attributeID como ATTRIBUTE.&AttributeType son tipos campo de clase de objeto puesto que son tipos definidos por referencia a una clase de objeto de información (ATTRIBUTE). El tipo ATTRIBUTE.&attributeId está fijo porque está definido explícitamente en ATTRIBUTE como un OBJECT IDENTIFIER (identificador de objeto). Sin embargo, el tipo ATTRIBUTE.&AttributeType puede llevar un valor de cualquier tipo definido utilizando ASN.1, puesto que su tipo no está fijo en la definición de ATTRIBUTE. Las notaciones que poseen la propiedad de poder llevar un valor de cualquier tipo reciben el nombre de "notaciones de tipo abierto", por lo que ATTRIBUTE.&AttributeType es de tipo abierto.

F.2.16 pdv incrustado

F.2.16.1 Se utilizará un tipo pdv incrustado (embedded-pdv) para modelar una variable cuyo tipo no se ha especificado o lo ha sido en otro sitio sin limitación en la notación utilizada para especificar el tipo.

EJEMPLO

```
FileContents ::= EMBEDDED PDV

DocumentList ::= SEQUENCE OF EMBEDDED PDV
```

F.2.17 Externo

El tipo externo (external) es similar al tipo pdv incrustado, pero tiene menos opciones de identificación. En las especificaciones nuevas se preferirá, por lo general, utilizar pdv incrustado debido a su mayor flexibilidad y al hecho de que algunas reglas de codificación codifican su valor de una manera más eficiente.

F.2.18 Ejemplar de

F.2.18.1 Se utilizará un ejemplar de (instance-of) para especificar un tipo que contenga un campo de identificador de objeto y un tipo open (abierto) cuyo valor es un tipo determinado por el identificador de objeto. El tipo ejemplar de (instance-of) se limita a llevar un valor de la clase TYPE-IDENTIFIER (véanse los Anexos A y C a la Rec. UIT-T X.681 | ISO/CEI 8824-2).

EJEMPLO

```
ACCESS-CONTROL-CLASS ::= TYPE-IDENTIFIER

Get-Invoke ::= SEQUENCE {
    objectClass         ObjectClass,
    objectInstance      ObjectInstance,
    accessControl       INSTANCE OF ACCESS-CONTROL-CLASS,
                            -- esto está constreñido normalmente
    attributeID         ATTRIBUTE.&attributeId
}
```

Get-Invoke entonces es equivalente a:

```
Get-Invoke ::= SEQUENCE {
    objectClass         ObjectClass,
    objectInstance      ObjectInstance,
    accessControl       [UNIVERSAL 8] IMPLICIT SEQUENCE {
        type-id                 ACCESS-CONTROL-CLASS.&id,
                                    -- esto está constreñido normalmente
        value                   [0] ACCESS-CONTROL-CLASS.&Type
                                    -- esto está constreñido normalmente
    },
    attributeID         ATTRIBUTE.&attributeId
}
```

La verdadera utilidad del tipo ejemplar de (instance-of) no se ve sino hasta que se constriñe utilizando un conjunto de objetos de información, pero ese ejemplo va más allá del alcance de la presente Recomendación | Norma Internacional. Véase en la Rec. UIT-T X.682 | ISO/CEI 8824-3 la definición de conjunto de objetos de información, y en el Anexo A a ese documento, cómo utilizar un conjunto de objetos de información para constreñir un tipo ejemplar de. Se señala que la codificación de la INSTANCE OF ACCESS-CONTROL-CLASS (ejemplar de clase de control de acceso) es la misma que para un valor EXTERNAL (externo) que tiene sólo un identificador de objeto y un valor de datos.

F.3 Identificación de sintaxis abstractas

F.3.1 El uso del servicio de presentación (Rec. UIT-T X.216 | ISO/CEI 8822) requiere la especificación de valores denominados valores de datos de presentación y la agrupación de dichos valores de datos de presentación en conjuntos que se denominan sintaxis abstractas. Cada uno de estos conjuntos recibe un nombre de sintaxis abstracta como identificador de objeto tipo ASN.1.

F.3.2 Puede utilizarse la notación ASN.1 como un instrumento general en la especificación de los valores de datos de presentación y su agrupación en sintaxis abstractas denominadas.

F.3.3 En el uso más sencillo, existe un tipo ASN.1 único tal que cada valor de datos de presentación en la sintaxis abstracta denominada es un valor de dicho tipo ASN.1. Este tipo normalmente será un tipo elección (choice) y cada valor de datos de presentación será un tipo alternativo de este tipo elección. En este caso se recomienda que se utilice la notación modular ASN.1 para contener dicho tipo elección como el primer tipo definido, seguido por la definición de aquellos tipos (no universales) referenciados directa o indirectamente por este tipo elección.

NOTA – No se tiene el propósito de excluir las referencias a los tipos definidos en otros módulos.

F.3.4 Se recomienda que la asignación de un identificador de objeto y un descriptor de objeto a una sintaxis abstracta se haga utilizando la clase de objeto de información útil ABSTRACT-SINTAX (sintaxis abstracta) definida en la Rec. UIT-T X.681 | ISO/CEI 8824-2. Se recomienda también que todas las utilizaciones de la ABSTRACT-SYNTAX se agrupen en un solo módulo "root" (raíz) que identifique todas las sintaxis abstractas utilizadas por una norma de aplicación.

F.3.5 A continuación se presenta un ejemplo de texto que podría aparecer en una aplicación normal.

EJEMPLO

```
ISOxxxx-yyyy {iso standard xxxx asn1-modules(...) yyyy-pdu(...)} DEFINITIONS ::=
BEGIN
    EXPORTS YYYY-PDU;

    YYYY-PDU ::= CHOICE {
        connect-pdu    ..... ,
        data-pdu    CHOICE {
```

```
                ..... ,
                .....
            },
            .....
        }
        ......
END

ISOxxxx-yyyy-Abstract-Syntax-Module {iso standard xxxx asn1-modules(...) } DEFINITIONS ::=
BEGIN
    IMPORTS YYYY-PDU FROM ISOxxxx-yyyy {iso standard xxxx asn1-modules(...) yyyy-pdu(...)};

-- Esta Recomendación | Norma Internacional define la siguiente sintaxis abstracta:

    YYYY-Abstract-Syntax ABSTRACT-SYNTAX ::=
        { YYYY-PDU  IDENTIFIED BY  yyyy-abstract-syntax-object-id }

    yyyy-abstract-syntax-object-id OBJECT IDENTIFIER ::= {iso standard yyyy(xxxx) abstract-syntax(...) }

-- El descriptor de objeto correspondiente es:

    yyyy-abstract-syntax-descriptor ObjectDescriptor ::= "....................."

-- Los valores del identificador de objeto y del descriptor de objeto ASN.1:
    .................   -- identificador de objeto de regla de codificación
    .................   -- descriptor de objeto de regla de codificación
-- asignados a reglas de codificación en las Rec. UIT-T X.690 y 691 | Norma ISO/CEI 8825-1 y 2,
-- pueden utilizarse como identificador de sintaxis de transferencia junto con la sintaxis de transferencia,
-- ISOxxxx-yyyy-Abstract-Syntax.

END
```

F.3.6 Para garantizar el interfuncionamiento, la norma puede además hacer obligatoria la admisión de la sintaxis de transferencia obtenida aplicando las reglas de codificación mencionadas en su módulo de sintaxis abstracta.

F.4 Subtipos

F.4.1 Se utilizarán subtipos para limitar los valores de un tipo existente que se admitan en una situación particular.

EJEMPLOS

```
AtomicNumber    ::=  INTEGER (1..104)

TouchToneString ::=  IA5String
        (FROM ("0123456789" | "*" | "#")) (SIZE (1..63))

ParameterList   ::=  SET SIZE (1..63) OF Parameter

SmallPrime   ::=  INTEGER (2|3|5|7|11|13|17|19|23|29)
```

F.4.2 Cuando dos o más tipos relacionados tengan en común aspectos significativos (dícese "comunalidad significativa"), se considerará la definición explícita de su progenitor común como un tipo y se utilizará la subtipificación para los tipos individuales. Este enfoque permite aclarar la relación y la comunalidad, a la vez que alienta (aunque no obliga) a que continúe esta práctica conforme evolucionan los tipos. Así, facilita el uso de los enfoques de realización comunes para el tratamiento de valores de estos tipos.

EJEMPLO

```
Envelope   ::=  SET {
                typeA  TypeA,
                typeB  TypeB   OPTIONAL,
                typeC  TypeC   OPTIONAL}
        -- el parent (progenitor) común

ABEnvelope  ::=  Envelope (WITH  COMPONENTS
                {... ,
                  typeB   PRESENT, typeC   ABSENT})
                  -- donde typeB tiene que aparecer siempre y typeC nunca puede aparecer
```

```
ACEnvelope ::= Envelope (WITH COMPONENTS
                  {... ,
                    typeB ABSENT, typeC PRESENT})
                    -- donde typeC tiene que aparecer siempre y typeB nunca puede aparecer
```

Las últimas definiciones podrían expresarse, de forma alternativa, como:

```
ABEnvelope ::= Envelope (WITH COMPONENTS {typeA, typeB})

ACEnvelope ::= Envelope (WITH COMPONENTS {typeA, typeC})
```

La elección entre las alternativas debe basarse en factores tales como el número de componentes en el tipo progenitor, y el número de aquellos que son opcionales, el grado de diferencia entre los tipos individuales y la posibilidad de una estrategia de evolución.

F.4.3 Se utilizará la subtipificación para definir parcialmente un valor, por ejemplo, una unidad de datos de protocolo (PDU) que será sometida a una prueba de conformidad, cuando la prueba afecte únicamente a algunos de los componentes de la PDU.

EJEMPLO

Dado:

```
PDU ::= SET
 {alpha      INTEGER,
  beta       IA5String OPTIONAL,
  gamma      SEQUENCE OF Parameter,
  delta      BOOLEAN}
```

cuando se componga, entonces, una prueba que requiera que el booleano sea false (falso) y el entero negativo, escríbase:

```
TestPDU ::= PDU (WITH COMPONENTS
   {... ,
    delta (FALSE),
    alpha (MIN..<0)})
```

y si, además, debe estar presente la cadena AI5 (AI5String), beta, con una longitud de 5 ó 12 caracteres, escríbase:

```
FurtherTestPDU ::= TestPDU (WITH COMPONENTS {... , beta (SIZE (5|12)) PRESENT } )
```

F.4.4 Si un tipo de datos para propósito general se ha definido como SEQUENCE OF, se utilizará la subtipificación para definir un subtipo restringido del tipo general:

EJEMPLO

```
Text-block ::= SEQUENCE OF VisibleString

Address ::= Text-block (SIZE (1..6)) (WITH COMPONENT (SIZE (1..32)))
```

F.4.5 Si un tipo de datos para propósito general se ha definido como CHOICE, se utilizará la subtipificación para definir un subtipo restringido del tipo general:

EJEMPLO

```
Z ::= CHOICE {
      a          A,
      b          B,
      c          C,
      d          D,
      e          E
}

V ::= Z (WITH COMPONENTS { ..., a ABSENT, b ABSENT }) -- 'a' y 'b' deben estar ausentes, y 'c',
                                                      -- 'd' o 'e' pueden estar presentes en un valor

W ::= Z (WITH COMPONENTS { ..., a PRESENT })          -- solamente 'a' puede estar presente
                                                      -- (véase 45.8.9.2)

X ::= Z (WITH COMPONENTS { a PRESENT })               -- solamente 'a' puede estar presente
                                                      -- (véase 45.8.9.2)

Y ::= Z (WITH COMPONENTS { a ABSENT, b, c })          -- 'a', 'd' y 'e' deben estar ausentes, y
                                                      -- 'b' o 'c' pueden estar presentes en un valor
```

NOTA – W y X son semánticamente idénticas.

F.4.6 Se utilizarán subtipos contenidos para formar nuevos subtipos a partir de los subtipos existentes:

EJEMPLO

```
Months      ::= ENUMERATED {
                    january         (1),
                    february        (2),
                    march           (3),
                    april           (4),
                    may             (5),
                    june            (6),
                    july            (7),
                    august          (8),
                    september       (9),
                    october         (10),
                    november        (11),
                    december        (12) }

First-quarter ::= Months (
                        january   |
                        february  |
                        march )

Second-quarter ::= Months (
                        april  |
                        may    |
                        june )

Third-quarter ::= Months (
                        july      |
                        august    |
                        september )

Fourth-quarter ::= Months (
                        october         |
                        november        |
                        december )

First-half ::= Months ( First-quarter | Second-quarter )

Second-half ::= Months ( Third-quarter | Fourth-quarter )
```

Anexo G

Suplemento didáctico sobre cadenas de caracteres ASN.1

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

G.1 Soporte de cadenas de caracteres en ASN.1

G.1.1 ASN.1 soporta cuatro grupos de cadenas de caracteres, a saber:

a) tipos cadena de caracteres (character string) basados en el *ISO International Register of Coded Character Sets to be used with Escape Sequence* (registro internacional ISO de juegos de caracteres codificados que han de utilizarse con secuencia de escape) (es decir, la estructura de la Norma ISO 646) y el International Register of Coded Character Sets (registro internacional de juegos de caracteres codificados) asociado, y proporcionados por los tipos VisibleString, IA5String, TeletexString, VideotexString, GraphicString y GeneralString;

b) tipos cadena de caracteres (character string) basados en ISO/CEI 10646-1 y proporcionados creando subconjuntos del tipo UniversalString o BMPString, estando los subconjuntos definidos en ISO/CEI 10646-1, o utilizando caracteres que llevan un nombre (caracteres denominados).

NOTAS

1 La utilización del tipo UniversalString no constreñido conduce a una violación de los requisitos de conformidad para el intercambio de información especificados en la Norma ISO/CEI 10646-1, ya que no se ha especificado ningún subconjunto adoptado.

2 No obstante lo antes expuesto, la utilización de este tipo con una constricción de subtipo simple que utiliza un parámetro de la sintaxis abstracta (restringido a un subtipo definido de UniversalString) puede proporcionar un medio potente y flexible para el tratamiento de caracteres, basándose en perfiles para determinar el valor del parámetro a fin de satisfacer las necesidades de comunidades particulares de interés. Sin embargo, por lo general, debe preferirse la utilización de CHARACTER STRING en Recomendaciones/Normas Internacionales (véase más adelante).

c) tipos character string (cadena de caracteres) que proporcionan una colección pequeña y simple de caracteres especificados en esta Especificación y destinados a uso especializado; estos tipos son NumericString y PrintableString;

d) utilización del tipo CHARACTER STRING, con negociación del conjunto de caracteres que habrá de utilizarse (o anuncio del conjunto de caracteres que se está utilizando); esto permite a una realización utilizar cualquier colección de caracteres y codificaciones para las cuales se hayan asignado OBJECT IDENTIFIERs, incluidos los de *ISO International Register of Coded Character Sets to be used with Escape Sequence*, Norma ISO 7350, Norma ISO/CEI 10646-1, y colecciones privadas de caracteres y codificaciones (unos perfiles pueden imponer requisitos o restricciones a los conjuntos de caracteres – las sintaxis abstractas de caracteres – que habrán de utilizarse).

G.2 Los tipos UniversalString y BMPString

G.2.1 El tipo UniversalString lleva cualquier carácter de la Norma ISO/CEI 10646-1. El conjunto de caracteres de ISO/CEI 10646-1 es generalmente demasiado grande para que tenga sentido exigir una conformidad y, normalmente, se recurre a un subconjunto del mismo formado por una combinación de las colecciones de caracteres normalizadas en el Anexo A de la Norma ISO/CEI 10646-1.

G.2.2 El tipo BMPString lleva cualquier carácter del plan multilingüe básico de la Norma ISO/CEI 10646-1 (los primeros 62K caracteres). Normalmente se recurre a un subconjunto del plan multilingüe básico formado por una combinación de colecciones de caracteres normalizadas en el Anexo A de la Norma ISO/CEI 10646-1.

G.2.3 Para las colecciones definidas en el anexo A de la Norma ISO/CEI 10646-1 hay referencias de tipos definidas en el módulo ASN.1 incorporado "ASN1-ISO10646" (véase la cláusula 35). El mecanismo "subtype constraint" (constricción de subtipo) permite definir nuevos subtipos de UniversalString que son combinaciones de subtipos existentes.

G.2.4 Ejemplos de referencias de tipo definidas en ASN1-ISO10646 y sus correspondientes de colecciones de la Norma ISO/CEI 10646 son:

BasicLatin	BASIC LATIN	(LATÍN BÁSICO)
Latin-1Supplement	LATIN-1 SUPPLEMENT	(SUPLEMENTO DE LATÍN 1)
LatinExtended-A	LATIN EXTENDED-A	(LATÍN AMPLIADO A)
LatinExtended-B	LATIN EXTENDED-B	(LATÍN AMPLIADO B)

IpaExtensions	IPA EXTENSIONS	(AMPLIACIONES DEL ALFABETO FONÉTICO INTERNACIONAL) (IPA, International Phonetic Alphabet)
SpacingModifierLetters	SPACING MODIFIER LETTERS	(LETRAS MODIFICADORAS DEL ESPACIAMIENTO)
CombiningDiacriticalMarks	COMBINING DIACRITICAL MARKS	(PUNTOS DIACRÍTICOS COMBINANTES)

G.2.5 La Norma ISO/CEI 10646-1 especifica tres "niveles de realización" y exige que todas las utilizaciones de ISO/CEI 10646-1 especifiquen el nivel de realización.

El nivel de realización está relacionado con el grado de soporte que se da a los *caracteres combinantes* en el repertorio de caracteres y, por consiguiente, en términos de ASN.1, define un subconjunto de los tipos cadena de caracteres restringida (restricted character string) UniversalString y BMPString.

En el nivel 1 de realización, no se permiten los caracteres combinantes y normalmente, existe una correspondencia biunívoca entre caracteres abstractos (referencias a células) de las cadenas de caracteres ASN.1 y caracteres impresos de una reproducción física de la cadena.

En el nivel 2 de realización están disponibles para su utilización determinados caracteres combinantes (indicados en el Anexo B de la Norma ISO/CEI 10646-1), pero hay otros cuya utilización está prohibida.

En el nivel 3 de realización no hay restricciones a la utilización de los caracteres combinantes.

G.3 Requisitos de conformidad de la Norma ISO/CEI 10646-1

La utilización de UniversalString (codificación de 4 octetos) o de BMPString (codificación de 2 octetos) (o subtipos de UniversalString o BMPString) en una definición de tipo ASN.1 exige que se tengan en cuenta los requisitos de conformidad de la Norma ISO/CEI 10646-1.

Según dichos requisitos de conformidad, los implementadores de una norma (digamos, X) que utilice esos tipos ASN.1, han de proporcionar (en los enunciados de conformidad de realización de protocolos) una declaración del subconjunto adoptado de la Norma ISO/CEI 10646-1 para su realización de la norma X, y del nivel (soporte de caracteres combinantes) de la realización.

La utilización de un subtipo ASN.1 de UniversalString o BMPString en una especificación exige que la realización soporte todos los caracteres de la Norma ISO/CEI 10646-1 incluidos en ese subtipo ASN.1 y, por consiguiente, que esos caracteres (por lo menos) estén presentes en el subconjunto adoptado para la realización. También se requiere el soporte del nivel establecido para todos esos subtipos ASN.1.

NOTA – Una especificación ASN.1 (en ausencia de parámetros de la sintaxis abstracta y de especificaciones de excepción) determina el conjunto (máximo) de caracteres que puede ser transmitido y el conjunto (mínimo) de caracteres que ha de ser tratado en recepción. El conjunto adoptado de la Norma ISO/CEI 10646-1 exige la no transmisión de los caracteres que excedan de este conjunto y que todos los caracteres del mismo sean soportados en recepción. Es necesario, por consiguiente, que el conjunto adoptado sea precisamente el conjunto de todos los caracteres permitidos por la especificación ASN.1. El caso en que está presente un parámetro de la sintaxis abstracta se examina más adelante.

G.4 Recomendaciones a los usuarios de la ASN.1 sobre conformidad de la Norma ISO/CEI 10646-1

Los usuarios de ASN.1 deben establecer claramente el conjunto de caracteres de la Norma ISO/CEI 10646-1 que formará el subconjunto adoptado de las realizaciones (y el nivel de realización requerido) para que se satisfagan los requisitos de su norma.

Esto puede hacerse convenientemente definiendo un subtipo ASN.1 de UniversalString o BMPString que contenga todos los caracteres necesarios para la norma y restringiéndolo a "Level1" (nivel 1) o "Level2" (nivel 2) si procede. Un nombre adecuado para este tipo podría ser "ISO-10646-String".

EJEMPLO

```
ISO-10646-String ::= BMPString
                     (FROM(Level2 INTERSECTION (BasicLatin UNION HebrewExtended UNION Hiragana)))
                     -- Este es el tipo que define el conjunto mínimo de caracteres en el subconjunto
                     -- adoptado para una realización de esta norma
                     -- Es preciso que el nivel de la realización sea, por lo menos, el nivel 2.
```

El PICS contendría entonces una simple declaración de que el subconjunto adoptado de ISO/CEI 10646-1 es el subconjunto limitado (y el nivel) definido por "ISO-10646-String", y de que se utilizaría "ISO-10646-String" (posiblemente subtipificados) en todas las normas en que se tuvieran que incluir cadenas de ISO/CEI 10646-1.

EJEMPLO DE PICS:

El subconjunto adoptado de ISO/CEI 10646-1 es el subconjunto limitado formado por todos los caracteres del tipo ASN.1 "ISO-10646-String" definido en el módulo <your module name goes here>, con un nivel de realización de 2.

EJEMPLO DE UTILIZACIÓN EN PROTOCOLO

```
Message ::= SEQUENCE {
    first-field ISO-10646-String,
            -- pueden aparecer todos los caracteres del subconjunto adoptado
    second-field   ISO-10646-String (FROM (latinSmallLetterA .. latinSmallLetterZ)),
            -- solamente letras minúsculas del alfabeto latino
    third-fieldISO-10646-String (FROM (digitZero .. digitNine))
            -- solamente dígitos

}
```

G.5 Subconjuntos adoptados como parámetros de la sintaxis abstracta

La Norma ISO/CEI 10646-1 exige que el subconjunto adoptado y el nivel de una realización se definan *explícitamente*. Si un usuario de la ASN.1 no desea constreñir la gama de caracteres ISO/CEI 10646-1 en alguna parte de la norma que se define, esto puede expresarse definiendo "ISO-10646-String" (por ejemplo) como un subtipo de UniversalString (codificación de 4 octetos) o de BMPString (codificación de 2 octetos) con una constricción de subtipo que conste de (o que incluya) "ImplementorsSubset" que se deja como un parámetro de la sintaxis abstracta.

Se advierte a los usuarios de ASN.1 que, en este caso, un emisor conforme puede transmitir a un receptor conforme caracteres que no pueden ser tratados por el receptor, porque quedan fuera del subconjunto adoptado (dependiente de la realización) o del nivel del receptor, y se recomienda que, se incluya, entonces, una especificación de tratamiento de excepción en la definición de "ISO-10646-String".

EJEMPLO

```
ISO-10646-String {UniversalString : ImplementorsSubset, ImplementationLevel} ::=
                    UniversalString (FROM((ImplementorsSubset UNION BasicLatin)
                                        INTERSECTION ImplementationLevel) !characterSetProblem)
                    -- El subconjunto adoptado de la Norma ISO/CEI 10646-1 incluirá "BasicLatin", pero
                    -- también puede incluir cualesquiera caracteres adicionales especificados en
                    -- "ImplementorsSubset", que es un parámetro de la sintaxis abstracta.
                    -- "ImplementationLevel", que es un parámetro de la sintaxis abstracta, define el nivel
                    -- de realización. Un receptor conforme debe estar preparado para recibir
                    -- caracteres fuera de su subconjunto adoptado y del nivel de realización. En
                    -- este caso, se invoca el tratamiento de excepción especificado en la cláusula
                    -- <add your clause number here> para "characterSetProblem". Se señala que
                    -- esto nunca puede ser invocado por un receptor conforme si los caracteres que
                    -- de hecho se utilizan en una instancia de comunicación están limitados a "BasicLatin".

My-Level2-String ::= ISO-10646-String { { HebrewExtended UNION Hiragana }, Level2 }
```

G.6 El tipo CHARACTER STRING

G.6.1 El tipo CHARACTER STRING (cadena de caracteres) da completa flexibilidad en la elección del conjunto de caracteres y del método de codificación. Cuando una conexión única proporcione transferencia de datos de extremo a extremo (es decir, cuando no se trate una aplicación relevadora), la negociación de los conjuntos de caracteres que han de utilizarse y de la codificación puede efectuarse como parte de la definición de los contextos de presentación para sintaxis abstractas de caracteres.

G.6.2 Es importante comprender que una sintaxis abstracta de caracteres es una sintaxis abstracta ordinaria con algunas restricciones en cuanto a los posibles valores (todos son cadenas de caracteres y, por cierto, cadenas de caracteres formadas a partir de alguna colección de caracteres). Así, la registración de esas sintaxis, y la negociación de un contexto de presentación se efectúan de manera normal.

G.6.3 La codificación de CHARACTER STRING permite también el anuncio de la sintaxis abstracta y de la sintaxis de transferencia utilizadas, sin negociación, en el caso de entornos en que esto sea apropiado.

NOTAS

1 En lo que respecta a la utilización de negociación de presentación para estos campos, los diseñadores de aplicaciones pueden prohibirla, exigirla, o dejarla a la voluntad del emisor.

2 Cuando se emplee anuncio en lugar de negociación, el diseñador de la aplicación debe considerar dos cosas: en primer lugar, la manera en que el emisor puede determinar qué sintaxis abstractas de caracteres (y qué sintaxis de transferencia) podrían ser aceptables por el receptor (por ejemplo, mediante la utilización del Directory Service (servicio de directorio) o como resultado del establecimiento de perfiles) y, en segundo lugar, las acciones que deberá ejecutar el receptor cuando reciba un valor CHARACTER STRING de una sintaxis abstracta de caracteres no soportada por él.

G.6.4 Si se utiliza negociación, el diseñador (de capa) de aplicación puede controlar dicha negociación especificando cuándo deberán establecerse esos contextos de presentación, y el parámetro de datos de usuario de las primitivas P-ALTERACIÓN DE CONTEXTO, o, simplemente suponer que algún perfil ha determinado qué sintaxis de caracteres habrá de utilizarse, y establecer un contexto de presentación para la misma en el momento de P-CONEXIÓN.

G.6.5 Las facilidades de gestión de contexto del servicio de presentación permiten a un iniciador (en una P-CONEXIÓN o dentro de una conexión establecida utilizando P-ALTERACIÓN DE CONTEXTO) proponer una lista de nuevas sintaxis abstractas (que pueden incluir sintaxis abstractas de caracteres), o retirar del uso sintaxis abstractas, de modo que el respondedor seleccione dentro de esa lista.

G.6.6 El iniciador puede expresar preferencia por medio del orden de la sintaxis abstracta en la lista, o de la utilización del parámetro user-data (datos de usuario), que puede ser empleado por el diseñador de aplicación para aclarar la finalidad de proponer la utilización de la nueva sintaxis abstracta. Podría indicar, por ejemplo, que todas las sintaxis abstractas (de caracteres) se están proponiendo para uso con una sola finalidad determinada, o que se tiene el propósito de permitir la selección de una sola sintaxis abstracta (de caracteres) para uso con varios fines.

G.6.7 Se han definido sintaxis abstractas de caracteres (y las correspondientes sintaxis de transferencia de caracteres) en Recomendaciones UIT-T y Normas Internacionales, y pueden ser definidas sintaxis abstractas de caracteres adicionales (y/o sintaxis de transferencia de caracteres) por cualquier organización capaz de atribuir identificadores de objetos.

G.6.8 En la Norma ISO/CEI 10646-1 se define una sintaxis abstracta de caracteres (y se asignan identificadores de objetos) para la colección completa de caracteres, para cada una de las colecciones definidas de caracteres para subconjuntos (LATÍN BÁSICO, SÍMBOLOS BÁSICOS, etc.), y para toda posible combinación de las colecciones definidas de caracteres. Se definen también dos sintaxis de transferencia de caracteres para identificar las diversas opciones (en particular, 16 bits y 32 bits) en la Norma ISO/CEI 10646-1.

Anexo H

Prestaciones reemplazadas

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

Un número de prestaciones que fueron incluidas en ediciones anteriores de esta Recomendación | Norma Internacional (a saber, la Rec. X.208 del CCITT (1988) | ISO/CEI 8824: 1990) han sido reemplazadas y no forman parte de ASN.1. No obstante pueden encontrarse en algunos módulos ASN.1 existentes. Este anexo describe estas prestaciones (features), así como la forma de obtener sus capacidades utilizando las que han venido a reemplazarlas.

H.1 Utilización de identificadores ahora obligatoria

La notación para un NamedType y un NamedValue que inicialmente era:

NamedType ::= identifier Type | Type | SelectionType
NamedValue ::= identifier Value | Value

es ahora obligatoria al haber sido reemplazada por:

NamedType ::= identifier Type
NamedValue ::= identifier Value

porque la primera podría dar lugar a una gramática ambigua.

Pueden añadirse identificadores a los "NamedType" de las especificaciones ASN.1 antiguas sin afectar a la codificación de tipo (aunque serán necesarios cambios en la ASN.1 para cualquier utilización de notación de valor relacionado). Esa modificación deberá introducirse en el marco de un informe de faltas o como parte de una nueva revisión de la norma que se modifica.

H.2 El valor de elección

La notación de valor para el tipo elección (choice) era inicialmente:

ChoiceValue ::= NamedValue

NamedValue ::= identifier Value | Value

pero ha sido reemplazada por:

ChoiceValue ::= Identifier ":" Value

porque la primera podría dar lugar a una gramática ambigua.

H.3 El tipo cualquiera

El tipo cualquiera (any) fue definido en ediciones anteriores de la presente Recomendación | Norma Internacional. Para información, la parte de la anterior edición de esta Recomendación | Norma Internacional que especificaba esta capacidad se une a esta especificación como Anexo I.

El uso normal del tipo cualquiera (any) tenía por finalidad dejar un "hueco", en la especificación, que sería llenado por alguna otra especificación. La notación era "AnyType", permitida como una alternativa para "Type", y se especificaba así:

AnyType ::= ANY | ANY DEFINED BY identifier

La notación de valor para el tipo cualquiera (any) era inicialmente.

AnyValue ::= Type Value

aunque después se cambió por:

AnyValue ::= Type : Value

porque la primera conducía a dificultades en el procesamiento por máquina de ASN.1.

Se recomendaba insistentemente utilizar la segunda alternativa de la notación. En esta alternativa, que sólo estaba permitida donde el tipo cualquiera (any) era uno de los tipos componentes de un tipo conjunto (set) o secuencia (sequence), algún otro componente del conjunto o de la secuencia (aquel cuyo "identifier" hubiese sido referenciado) indicaría por su valor entero o de identificador de objeto (o una opción de estos) el tipo que gobernaría en efecto al componente any. La correspondencia entre esos valores y tipos ASN.1 particulares podría visualizarse como una especie de "table" (tabla, o cuadro) que formaría parte de la sintaxis abstracta. En ausencia del "DEFINED BY identifier" (la primera alternativa notacional), no había en la notación ninguna indicación sobre la forma en que podría determinarse el tipo. Esto conducía frecuentemente a especificaciones en las que el "hueco" continuaba existiendo incluso en la etapa en que se esperaban realizaciones.

El tipo cualquiera (any) está ahora reemplazado por la aptitud para especificar clases de objeto de información y, seguidamente, referir a los campos de objetos de información desde el interior de definiciones de tipo (véase la Rec. UIT-T X.681 | ISO/CEI 8824-2). Puesto que pueden definirse campos para permitir un tipo ASN.1 arbitrario, se proporciona la aptitud básica para dejar "huecos". Sin embargo, la nueva prestación permite también la especificación de una "table constraint" (constricción por tabla), en la cual se indica explícitamente que un determinado "information object set" (conjunto de objetos de información) (un conjunto de objetos de información de la clase apropiada) constriñe el tipo. Esta última capacidad comprende la ofrecida por "**ANY DEFINED BY** identifier".

Se proporcionan asimismo algunos usos predefinidos de las nuevas capacidades (véase la Rec. UIT-T X.681 | ISO/CEI 8824-2), que corresponden a diversos patrones de utilización del tipo cualquiera (any) que suelen presentarse con frecuencia. Por ejemplo, una secuencia que contiene un identificador de objeto y un any, que solía utilizarse anteriormente para transportar algún valor arbitrario junto con una indicación de su tipo, puede ahora describirse como:

```
INSTANCE OF MUMBLE
```

donde **MUMBLE** se define como una clase de objeto de información (no un tipo ASN.1):

```
MUMBLE ::= TYPE-IDENTIFIER
```

Esta notación hace que "INSTANCE OF MUMBLE" sea sustituida por un identificador de objeto para un objeto de clase MUMBLE, junto con el tipo identificado por el identificador de objeto. Véase un ejemplo en F.2.18.

Emparejamientos particulares de identificador de objeto y tipo se definen como objetos de información de clase **MUMBLE**, y, si es necesario, conjuntos particulares de éstos pueden también definirse y utilizarse para constreñir la construcción **INSTANCE OF** de modo que en el conjunto sólo puedan aparecer esos objetos.

La capacidad macro se utilizaba a menudo como una manera semiformal de definir tablas de objetos de información para gobernar un uso asociado de un tipo cualquiera (any), y ha sido también reemplazada por las nuevas capacidades.

H.4 La capacidad macro

La capacidad macro permitía al usuario de ASN.1 ampliar la notación definiendo macros. Para información, la parte de la anterior edición de esta Recomendación | Norma Internacional que especificaba la capacidad se adjunta a la presente Recomendación | Norma Internacional como Anexo J.

La capacidad macro se ha utilizado sobre todo para definir una notación para la especificación de objetos de información. Esta capacidad ha sido ahora incluida en ASN.1 directamente (véase la Rec. UIT-T X.681 | ISO/CEI 8824-2), por lo que no es necesario recurrir a la solución general de una notación definida por el usuario (con los peligros consiguientes).

Aparte de esto, la única utilización que pueden tener las macros parece ser la de definir expresiones a las que deben suministrarse algunos parámetros antes de que se conviertan en tipos ASN.1 completamente definidos. Esto se consigue ahora mediante la capacidad más general de parametrización (véase la Rec. UIT-T X.683 | ISO/CEI 8824-4).

Anexo I

La notación de tipo cualquiera (any)

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

Este anexo se incluye como una referencia histórica; la notación aquí descrita ya no forma parte de la ASN.1.

I.1 Notación para el tipo cualquiera (any)

I.1.1 La notación para un tipo cualquiera (any) es "AnyType":

AnyType ::= ANY | ANY DEFINED BY identificateur

NOTA – La utilización de "ANY" en una norma ISO o Recomendación produce una especificación incompleta a menos que vaya suplementada por una especificación adicional. La construcción "ANY DEFINED BY" proporciona el medio de especificar en un caso de comunicación el tipo que llena el ANY, así como un puntero (pointer) a su semántica. Si se siguen las reglas para su utilización, puede proporcionar una especificación completa. La utilización de ANY sin la construcción DEFINED BY está desaconsejada.

I.1.2 La alternativa "DEFINED BY" sólo deberá utilizarse cuando el tipo cualquiera (any), o un tipo derivado de él, sea uno de los tipos componentes de un tipo secuencia (sequence) o de un tipo conjunto (set) (el tipo contenedor).

I.1.3 El "identifier" en la alternativa "DEFINED BY" deberá también aparecer en un "NamedType" que especifique otro componente no opcional del tipo contenedor. El "Type" del "NamedType" será un tipo derivado de un tipo entero (integer), de un tipo identificador de objeto (object identifier) o de la elección de uno de estos tipos.

I.1.4 Cuando el "Type" del "NamedType" es de tipo entero, el documento que emplea la notación "DEFINED BY" debe contener, o referenciar explícitamente, una lista única que especifique el tipo ASN.1 que habrá de ser transportado por el ANY para cada valor permitido del tipo entero. Deberá haber exactamente una lista tal que comprenda todos los casos de comunicación del tipo contenedor.

I.1.5 Cuando el "Type" del "NamedType" es de tipo identificador de objeto, se necesitan registros que, para cada valor de identificador de objeto atribuido, asocien un solo tipo ASN.1 (que puede ser un tipo CHOICE) que habrá de ser transportado por el ANY.

NOTAS

1 Puede haber un número arbitrario de registros que asocien un valor de identificador de objeto a un tipo ASN.1, para esta finalidad.

2 Se espera que la registración de valores para interconexión abierta se efectúe dentro de Normas de la ISO y de Recomendaciones que utilizan la notación. Cuando se pretenda que haya una Autoridad de Registración Internacional (International Registration Authority) distinta para cualquier ejemplar de "ANY DEFINED BY", deberá indicarse así en el documento que emplea la notación.

3 La diferencia principal entre definiciones de entero y de identificador de objeto es que las referencias a enteros se hacen en una sola lista, contenida en la norma que se está empleando, mientras que la utilización de identificador de objeto permite un conjunto abierto de tipos determinado por cualquier autoridad capaz de atribuir identificadores de objeto.

I.1.6 Este tipo tiene un rótulo indeterminado, y no se utilizará cuando esta Recomendación | Norma Internacional requiera rótulos distintos (véanse 22.5, 24.3, 26.2 y 26.4).

I.1.7 La notación para el valor de un tipo cualquiera (any) se definirá utilizando ASN.1, y es "AnyValue":

AnyValue ::= Type : Value

donde "Type" es la notación para el tipo elegido, y "Value" es una notación válida para un valor de este tipo.

Anexo J

La notación macro

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

Este anexo se incluye como una referencia histórica; la notación descrita en la misma ya no forma parte de ASN.1.

J.1 Introducción

En la ASN.1 se proporciona al usuario de ASN.1 un mecanismo para definir una nueva notación con el cual se pueden construir y referenciar tipos ASN.1 o especificar valores de tipos. La nueva notación se define utilizando la notación "MacroDefinition". Una "MacroDefinition" especifica simultáneamente una nueva notación para construir y referenciar un tipo y también una nueva notación para especificar un valor.

Con una "MacroDefinition" (definición de macro), el usuario de ASN.1 especifica la nueva notación por medio de un conjunto de producciones de una manera similar a la de esta Norma Internacional. El escritor de la definición de macro:

a) especifica la sintaxis completa que habrá de utilizarse para definir todos los tipos soportados por la macro (esta especificación de sintaxis es invocada para análisis de sintaxis por cualquier ocurrencia del nombre de macro en la notación de tipo ASN.1); y

b) especifica la sintaxis completa que habrá de utilizarse para un valor de uno de estos tipos (esta especificación de sintaxis es invocada para análisis de sintaxis cuando se espera un valor del tipo macro); y

c) especifica, como el valor de un tipo ASN.1 normalizado (de complejidad arbitraria), el tipo y valor resultantes para todos los ejemplares de la notación de valor de macro.

Un ejemplar de la sintaxis definida por la definición de macro puede contener ejemplares de tipos o valores (cuando se emplea la notación ASN.1 normalizada). Estos tipos o valores (que aparecen al utilizar la notación de macro) pueden ser asociados, a todo lo largo del análisis de sintaxis, con una referencia de tipo local o una **referencia de valor local** por enunciados (statements) apropiados en la definición de macro. También es posible incrustar (embed) en la definición de macro, asignaciones de tipo ASN.1 normalizadas. Estas asignaciones son activadas cuando la categoría sintáctica asociada es concordada (matched) con uno o más ítems del ejemplar de la nueva notación que se está analizando. Su tiempo de vida está limitado al del análisis.

Cuando se analiza un valor de la nueva notación, las asignaciones hechas durante el análisis de la notación de tipo correspondiente están disponibles. Se considera que tal análisis precede lógicamente al análisis de cada ejemplar de la notación de valor.

El tipo y valor resultantes de un ejemplar de utilización de la nueva notación de valor se determinan por el valor (y el tipo del valor) asignado finalmente a la referencia de valor local distinguido identificada por el ítem palabra clave VALUE de acuerdo con el procesamiento de la definición de macro para el nuevo tipo de notación, seguido por el de la nueva notación de valor.

Cada "MacroDefinition" (definición de macro) define una notación (una sintaxis) para definición de tipo y una notación (una sintaxis) para definición de valor. El tipo ASN.1 que es definido por un ejemplar de la nueva notación de tipo puede, pero no tiene necesariamente que, depender del ejemplar de la notación de valor con la cual está asociado el tipo. En esta medida, la utilización de la nueva notación de tipo es similar a un CHOICE (el rótulo está indeterminado). De esta forma, la nueva notación no puede utilizarse, en este caso, en aquellos lugares en que se requiera un rótulo conocido, ni puede tampoco ser implícitamente rotulada.

J.2 Ampliaciones del conjunto de caracteres ASN.1 y de los ítems

Los caracteres | y > se utilizan en la notación de macro.

También se utilizan los ítems especificados en las siguientes subcláusulas.

J.2.1 Macroreference (referencia de macro)

Nombre de ítem – macroreference

Una "macroreference" consistirá en la secuencia de caracteres especificadas para una "typereference" en 9.2, con la salvedad de que todos los caracteres deberán ser letras mayúsculas. Dentro de un solo módulo no se utilizará la misma secuencia de caracteres para las dos referencias, es decir para una typereference y una macroreference.

J.2.2 Productionreference (referencia de producción)

Nombre de ítem – productionreference

Una "productionreference" consistirá en la secuencia de caracteres especificada para una "typereference" en 9.2.

J.2.3 Localtypereference (referencia de tipo local)

Nombre de ítem – localtypereference

Una "localtypereference" consistirá en la secuencia de caracteres especificada para una "typereference" en 9.2. Una localtypereference" se utiliza como un identificador para tipos que son reconocidos durante el análisis de sintaxis de un ejemplar de la nueva notación de tipo o valor.

J.2.4 Localvaluereference (referencia de valor local)

Nombre de ítem – localvaluereference

Una "localvaluereference" consistirá en la secuencia de caracteres especificada para una "typereference" en 9.2. Una "localvaluereference" se utiliza como un identificador para valores que son reconocidos durante el análisis de sintaxis de una instancia del nuevo ejemplar de tipo o valor.

NOTA – Una "localvaluereference" comienza con una letra mayúscula.

J.2.5 Item alternation (alternación)

Nombre de ítem –"|"

Este ítem consistirá en el carácter único | .

J.2.6 Item definition terminator (terminador de definición)

Nombre de ítem – >

Este ítem consistirá en el carácter único > .

NOTA – El ítem < para el comienzo de definiciones se define en 9.15.

J.2.7 Item syntactic terminal (terminal sintáctico)

Nombre de ítem – astring

Una "astring" consistirá en un número arbitrario (que puede ser cero) de caracteres pertenecientes al conjunto de caracteres ASN.1 [véase la cláusula 8 encerrados entre comillas (")]. Dentro de una "astring", el carácter se representará por un par de ".

NOTA – La utilización de "astring" en la notación de macro especifica la ocurrencia, en el punto correspondiente de la sintaxis que se está analizando, de los caracteres encerrados entre comillas (").

J.2.8 Items syntactic category keyword (palabra clave de categoría sintáctica)

Nombres de ítems –

"string"
"identifier"
"number"
"empty"

Los ítems con los nombres antes indicados consistirán (en la notación de macro) en las secuencias de caracteres que forman el nombre, excluidos los símbolos (''). Estos ítems se utilizan en la notación de macro para especificar la ocurrencia, en un ejemplar de la nueva notación, de ciertas secuencias de caracteres. Las secuencias de la nueva notación especificada por cada ítem se indican en el Cuadro J.1 por referencia a una cláusula de esta Norma Internacional que define la secuencia de caracteres que aparecen en la nueva notación.

NOTA – La notación de macro no admite la distinción entre identificadores y referencias basada en que la letra inicial sea mayúscula o minúscula. Esto se explica por razones históricas.

Cuadro J.1 – Secuencia especificada por ítems

Nombre del ítem	Cláusula que proporciona la definición
"string"	Cualquier secuencia de caracteres
"identifier"	9.3 – Identificadores
"number"	9.8 – Números
"empty"	9.7 – Vacío

J.2.9 Ítems palabra clave adicionales

Nombres de ítems –

MACRO
TYPE
NOTATION
VALUE
value
type

Los ítems que llevan estos nombres consistirán en la secuencia de caracteres que forman el nombre.

Los ítems especificados en J.2.2 a J.2.4 inclusive no serán ninguno de los que figuran en las secuencias de J.2.9, a menos que se utilicen como se especifica más adelante.

La palabra clave "MACRO" se utilizará para introducir una definición de macro. La palabra clave "TYPE NOTATION" se utilizará como el nombre de la producción que define la nueva notación de tipo. La palabra clave "VALUE NOTATION" se utilizará como el nombre de la producción que define la nueva notación de valor. La palabra clave "VALUE" se utilizará como la "localvaluereference" a la cual se asigna el valor resultante. La palabra clave "value" se utilizará para especificar que cada ejemplar de la nueva notación contiene en este punto, en notación ASN.1 normalizada, algún valor de un tipo (especificado en la definición de macro). La palabra clave "type" se utilizará para especificar que cada ejemplar de la nueva notación contiene en este punto, en notación ASN.1 normalizada, algún "Type".

J.3 Notación de definición de macro

J.3.1 Una macro se definirá utilizando la notación "MacroDefinition":

```
MacroDefinition ::=
    macroreference
    MACRO
    "::="
    MacroSubstance

MacroSubstance ::=
    BEGIN MacroBody END     |
    macroreference          |
    Externalmacroreference

MacroBody ::=
    TypeProduction
    ValueProduction
    SupportingProductions

TypeProduction  ::=
    TYPE NOTATION
    "::="
    MacroAlternativeList

ValueProduction ::=
    VALUE NOTATION
    "::="
    MacroAlternativeList
```

```
SupportingProductions ::=
    ProductionList      |
    empty

ProductionList ::=
    Production          |
    ProductionList Production

Production ::=
    productionreference
    "::="
    MacroAlternativeList

Externalmacroreference ::=
    modulereference "." macroreference
```

J.3.2 Si se elige la alternativa "macroreference" de "MacroSubstance", el módulo que contiene la definición de macro deberá:

a) contener otra definición de macro que defina esa "macroreference"; o

b) contener la "macroreference" en su "SymbolsImported".

J.3.3 Si se elige la alternativa "Externalmacroreference" de "MacroSubstance", el módulo denotado por "modulereference" contendrá una definición de macro que defina la "macroreference". La definición asociada también está entonces asociada con la "macroreference" que se está definiendo.

J.3.4 La cadena de definiciones que puede surgir de aplicaciones repetidas de las reglas indicadas en J.3.2 a J.3.3 terminará con una "MacroDefinition" que utilice la alternativa "BEGIN MacroBody END".

J.3.5 Cada "productionreference" que ocurra en un "SymbolDefn" (véase J.3.9) deberá ocurrir exactamente una vez como el primer ítem de una "Production".

J.3.6 Cada ejemplar de la nueva notación de tipo comenzará por la secuencia de caracteres en la "macroreference", seguida por una de las secuencias de caracteres referenciadas por "TYPE NOTATION" después de aplicar las producciones especificadas en la definición de macro.

J.3.7 Cada ejemplar de la nueva notación de valor consistirá en una secuencia de caracteres referenciada por "VALUE NOTATION" después de aplicar las producciones especificadas en la definición de macro.

J.3.8 La "MacroAlternativeList" de una producción especifica los posibles conjuntos de secuencias de caracteres referenciados por esa producción. Se especifica por:

```
MacroAlternativeList ::=
    MacroAlternative    |
    MacroAlternativeList "|" MacroAlternative
```

El conjunto de secuencias de caracteres referenciado por la "MacroAlternativeList" consiste en todas las secuencias de caracteres que son referenciadas por cualquiera de las producciones "MacroAlternative" en la "MacroAlternativeList".

J.3.9 La notación para una "MacroAlternative" será:

```
MacroAlternative ::= SymbolList

SymbolList ::=
    SymbolElement       |
    SymbolList SymbolElement

SymbolElement ::=
    SymbolDefn|
    EmbeddedDefinitions

SymbolDefn ::=
    astring             |
    productionreference |
    "string"            |
    "identifier"        |
    "number"            |
    "empty"             |
```

type |
type(localtypereference) |
value(MacroType) |
value(localvaluereference MacroType) |
value(VALUE MacroType)

MacroType ::=
localtypereference |
Type

NOTA – En una macro, cualquier "MacroType" definido en esa macro puede aparecer en cualquier punto en el que ASN.1 especifica un "Type".

Una "MacroAlternative" referencia todas las cadenas de caracteres que se forman tomando cualquiera de las cadenas de caracteres referenciadas por la primera "SymbolDefn" en la "SymbolList", seguida por cualquiera de las cadenas de caracteres referenciadas por la segunda "SymbolDefn" en la "SymbolList", y así sucesivamente, hasta la última "SymbolDefn" inclusive en la "SymbolList".

NOTA – Las "EmbeddedDefinitions" (si existen) no entran en juego directamente para la determinación de estas cadenas.

J.3.10 Una "astring" referencia la secuencia de caracteres en la "astring", excluido el par de " que enmarca la cadena.

J.3.11 Una "productionreference" referencia cualquier secuencia de caracteres especificada por la "Production" que ella especifica.

J.3.12 La secuencias de caracteres referenciadas por las siguientes cuatro alternativas para "SymbolDefn" se especifican en el Cuadro J.1.

NOTA – Las secuencias de caracteres referenciadas por la "string" deben ser terminadas en un ejemplar de la notación de macro por la aparición de una secuencia referenciada por la "SymbolDefn" siguiente en la "SymbolList".

J.3.13 Un "type" referencia cualquier secuencia de símbolos que forma una notación de "Type" como se especifica en 14.1.

NOTA – El "ReferencedType" de 14.1 puede en este caso contener una "localtypereference" que referencia un tipo definido en la notación de macro.

J.3.14 Un "type(localtypereference)" referencia cualquier secuencia de símbolos que forma un "Type" como se especifica en 14.1, pero, además, asigna ese tipo a la "localtypereference". Puede ocurrir una ulterior asignación a la misma "localtypereference".

J.3.15 Un "value(MacroType)" referencia cualquier secuencia de símbolos que forma una notación "Value" (como se especifica en 14.7) para el tipo especificado por "MacroType".

J.3.16 Un "value(localvaluereference MacroType)" referencia cualquier secuencia de símbolos que forma una notación de "Value" (como se especifica en 14.7) para el tipo especificado por "MacroType", pero, además, asigna el valor especificado por la notación de valor a la "localvaluereference". Puede ocurrir una ulterior asignación a la "localvaluereference".

J.3.17 Un "value(VALUE MacroType)" referencia cualquier secuencia de símbolos que forma una notación de "Value" (como se especifica en 14.7) para el tipo especificado por "MacroType", pero, además, retorna el valor como el valor especificado por la notación de valor. El tipo del valor retornado es el tipo referenciado por "MacroType".

J.3.18 En el análisis de cualquier ejemplar correcto de la nueva notación ocurre exactamente una asignación a VALUE (como se especifica en J.3.17 o en J.3.19).

J.3.19 La notación para un "EmbeddedDefinitions" deberá ser:

EmbeddedDefinitions ::= "<" EmbeddedDefinitionList ">"

EmbeddedDefinitionList ::=
EmbeddedDefinition |
EmbeddedDefinitionList
EmbeddedDefinition

EmbeddedDefinition ::=
LocalTypeassignment |
LocalValueassignment

LocalTypeassignment ::=
localtypereference
"::="
MacroType

```
LocalValueassignment ::=
        localvaluereference
        MacroType
        "::="
        MacroValue

MacroValue ::=
        Value           |
        localvaluereference
```

La asignación de un "MacroType" a una "localtypereference" (o de un "MacroValue" a una "localvaluereference") dentro de un "EmbeddedDefinitions" se efectúa durante el análisis de sintaxis de un ejemplar de la nueva notación en el momento en que se encuentra el "EmbeddedDefinitions", y subsiste hasta que ocurre una redefinición de la "localtypereference" o de la "localvaluereference".

NOTAS

1 La utilización de la "localtypereference" o de la "localvaluereference" asociadas en otro lugar en la "Alternative" implica ciertos supuestos en cuanto a la naturaleza del algoritmo de análisis (parsing algorithm). Estos supuestos deben ser indicados mediante comentarios. Por ejemplo, la utilización de la "localtypereference" que sigue textualmente al "EmbeddedDefinitions" implica un parsing de izquierda a derecha.

2 Se puede asignar un valor a la "localvaluereference" "VALUE" ya sea por la construcción "value(VALUE MacroType)" o por una "EmbeddedDefinition". En ambos casos, el valor es retornado como se especifica en J.3.17.

J.4 Utilización de la nueva notación

Cada vez que esta Norma Internacional invoca una notación de "Type" (o "Value"), se puede utilizar un ejemplar de la notación de tipo (o de la notación de valor) definida por una macro, siempre que la macro haya sido:

a) definida dentro del mismo módulo; o

b) importada al módulo, para lo cual se hace aparecer la "macroreference" en el "SymbolsImported" del módulo.

Para permitir esta última posibilidad, una "macroreference" puede aparecer como un "Symbol" en 10.1.

NOTAS

1 Esta ampliación a la notación ASN.1 normalizada no se indica en el cuerpo de esta Norma Internacional.

2 Es posible construir módulos que incluyan secuencias de asignación de tipo y definiciones de macro que hagan arbitrariamente compleja la descomposición analítica de la sintaxis de valor en los valores DEFAULT.

Anexo K

Sumario de la notación ASN.1

(Este anexo no es parte integrante de la presente Recomendación | Norma Internacional)

Los siguientes ítems se definen en la cláusula 9:

typereference	BEGIN	ISO646String
identifier	BIT	MAX
valuereference	BMPString	MIN
modulereference	BOOLEAN	MINUS-INFINITY
comment	BY	NULL
empty	CHARACTER	NumericString
number	CHOICE	OBJECT
bstring	CLASS	ObjectDescriptor
hstring	COMPONENT	OCTET
cstring	COMPONENTS	OF
"::="	CONSTRAINED	OPTIONAL
".."	DEFAULT	PDV
"..."	DEFINITIONS	PLUS-INFINITY
"{"	EMBEDDED	PRESENT
"}"	END	PrintableString
"<"	ENUMERATED	PRIVATE
","	EXCEPT	REAL
"."	EXPLICIT	SEQUENCE
"("	EXPORTS	SET
")"	EXTERNAL	SIZE
"["	FALSE	STRING
"]"	FROM	SYNTAX
"-"	GeneralizedTime	T61String
":"	GeneralString	TAGS
";"	GraphicString	TeletexString
"@"	IA5String	TRUE
"|"	TYPE-IDENTIFIER	UNION
"!"	IDENTIFIER	UNIQUE
"^"	IMPLICIT	UNIVERSAL
ABSENT	IMPORTS	UniversalString
ABSTRACT-SYNTAX	INCLUDES	UTCTime
ALL	INSTANCE	VideotexString
APPLICATION	INTEGER	VisibleString
AUTOMATIC	INTERSECTION	WITH

En esta Recomendación | Norma Internacional se utilizan las siguientes producciones con los ítems mencionados como símbolos terminales:

```
ModuleDefinition ::= ModuleIdentifier
                     DEFINITIONS
                     TagDefault
                     "::="
                     BEGIN
                     ModuleBody
                     END

ModuleIdentifier ::= modulereference
                     DefinitiveIdentifier

DefinitiveIdentifier ::= "{" DefinitiveObjIdComponentList "}" |
                         empty

DefinitiveObjIdComponentList ::=
        DefinitiveObjIdComponent |
        DefinitiveObjIdComponent DefinitiveObjIdComponentList

DefinitiveObjIdComponent ::=
        NameForm |
        DefinitiveNumberForm |
        DefinitiveNameAndNumberForm

DefinitiveNumberForm ::= number

DefinitiveNameAndNumberForm ::= identifier "(" DefinitiveNumberForm ")"

TagDefault ::=      EXPLICIT TAGS |
                    IMPLICIT TAGS |
                    AUTOMATIC TAGS |
                    empty

ModuleBody ::=      Exports Imports AssignmentList |
                    empty

Exports ::=         EXPORTS SymbolsExported ";" |
                    empty

SymbolsExported ::=     SymbolList |
                        empty

Imports ::=         IMPORTS SymbolsImported ";" |
                    empty

SymbolsImported ::=     SymbolsFromModuleList |
                        empty

SymbolsFromModuleList ::=
                    SymbolsFromModule |
                    SymbolsFromModuleList SymbolsFromModule

SymbolsFromModule ::= SymbolList FROM GlobalModuleReference

GlobalModuleReference ::= modulereference AssignedIdentifier

AssignedIdentifier ::=  ObjectIdentifierValue |
                        DefinedValue |
                        empty

SymbolList ::= Symbol | Symbol "," SymbolList

Symbol ::= Reference | ParameterizedReference

Reference ::=
        typereference |
        valuereference |
        objectclassreference |
        objectreference |
        objectsetreference

AssignmentList ::= Assignment | AssignmentList Assignment
```

```
Assignment ::=
        TypeAssignment |
        ValueAssignment |
        ValueSetTypeAssignment |
        ObjectClassAssignment |
        ObjectAssignment |
        ObjectSetAssignment |
        ParameterizedAssignment

Externaltypereference ::=
            modulereference
            "."
            typereference

Externalvaluereference ::=
            modulereference
            "."
            valuereference

DefinedType ::=
        Externaltypereference |
        typereference |
        ParameterizedType |
        ParameterizedValueSetType

DefinedValue ::=
        Externalvaluereference |
        valuereference |
        ParameterizedValue

AbsoluteReference ::= "@" GlobalModuleReference
        "."
        ItemSpec

ItemSpec ::=
        typereference |
        ItemId "." ComponentId

ItemId ::= ItemSpec

ComponentId ::=
        identifier | number | "*"

TypeAssignment ::=  typereference
                    "::="
                    Type

ValueAssignment ::=     valuereference
                        Type
                        "::="
                        Value

ValueSetTypeAssignment ::=  typereference
                            Type
                            "::="
                            ValueSet

ValueSet ::= "{" ElementSetSpec "}"

Type ::= BuiltinType | ReferencedType | ConstrainedType

BuiltinType ::=
    BitStringType |
    BooleanType |
    CharacterStringType |
    ChoiceType |
    EmbeddedPDVType |
    EnumeratedType |
    ExternalType |
    InstanceOfType |
    IntegerType |
    NullType |
```

```
        ObjectClassFieldType |
        ObjectIdentifierType |
        OctetStringType |
        RealType |
        SequenceType |
        SequenceOfType |
        SetType |
        SetOfType |
        TaggedType

NamedType ::= identifier Type | SelectionType

ReferencedType ::=
        DefinedType |
        UsefulType |
        SelectionType |
        TypeFromObject |
        ValueSetFromObjects

Value  ::=  BuiltinValue | ReferencedValue

BuiltinValue ::=
        BitStringValue |
        BooleanValue |
        CharacterStringValue |
        ChoiceValue |
        EmbeddedPDVValue |
        EnumeratedValue |
        ExternalValue |
        InstanceOfValue |
        IntegerValue |
        NullValue |
        ObjectClassFieldValue |
        ObjectIdentifierValue |
        OctetStringValue |
        RealValue |
        SequenceValue |
        SequenceOfValue |
        SetValue |
        SetOfValue |
        TaggedValue

ReferencedValue ::=
        DefinedValue |
        ValueFromObject

NamedValue ::= identifier Value

BooleanType  ::= BOOLEAN

BooleanValue::=  TRUE | FALSE

IntegerType ::=
                INTEGER                         |
                INTEGER "{" NamedNumberList "}"

NamedNumberList ::=
                NamedNumber                     |
                NamedNumberList "," NamedNumber

NamedNumber ::=
                identifier "(" SignedNumber ")"         |
                identifier "(" DefinedValue ")"

SignedNumber ::= number | "-" number

IntegerValue ::=  SignedNumber | identifier

EnumeratedType ::=
            ENUMERATED "{" Enumeration "}"

Enumeration ::=
            EnumerationItem | EnumerationItem "," Enumeration
```

```
EnumerationItem ::=
        identifier | NamedNumber

EnumeratedValue  ::=
        identifier

RealType ::=  REAL

RealValue ::=
        NumericRealValue | SpecialRealValue

NumericRealValue ::= 0 |
        SequenceValue  --  Value of the associated sequence type

SpecialRealValue ::=
        PLUS-INFINITY | MINUS-INFINITY

BitStringType   ::=        BIT STRING | BIT STRING "{" NamedBitList "}"

NamedBitList::=            NamedBit | NamedBitList "," NamedBit

NamedBit     ::=       identifier "(" number ")" |
                       identifier "(" DefinedValue ")"

BitStringValue  ::=        bstring | hstring | "{" IdentifierList "}" | "{"  "}"

IdentifierList  ::=    identifier | IdentifierList "," identifier

OctetStringType  ::=       OCTET STRING

OctetStringValue ::=       bstring | hstring

NullType     ::=           NULL

NullValue   ::=  NULL

SequenceType ::=           SEQUENCE "{" ComponentTypeList "}"  |
                           SEQUENCE "{"  "}"

ComponentTypeList  ::=  ComponentType  |
                        ComponentTypeList "," ComponentType

ComponentType  ::=         NamedType  |
                           NamedType OPTIONAL  |
                           NamedType DEFAULT Value  |
                           COMPONENTS OF Type

SequenceValue   ::=        "{" ComponentValueList "}" | "{"  "}"

ComponentValueList ::=  NamedValue       |
                  ComponentValueList "," NamedValue

SequenceOfType  ::=        SEQUENCE OF Type

SequenceOfValue  ::=       "{" ValueList "}" | "{"  "}"

ValueList   ::= Value | ValueList "," Value

SetType  ::=       SET "{" ComponentTypeList "}" | SET "{"  "}"

SetValue    ::="{" ComponentValueList "}" | "{"  "}"

SetOfType   ::=  SET OF Type

SetOfValue   ::= "{" ValueList "}"  |  "{"  "}"

ChoiceType   ::= CHOICE "{" AlternativeTypeList "}"

AlternativeTypeList ::=    NamedType        |
                           AlternativeTypeList "," NamedType

ChoiceValue  ::= identifier ":" Value

SelectionType   ::=        identifier "<" Type

TaggedType   ::=  Tag Type  |
                  Tag IMPLICIT Type  |
                  Tag EXPLICIT Type
```

```
Tag      ::=          "[" Class ClassNumber "]"

ClassNumber  ::= number | DefinedValue

Class    ::=      UNIVERSAL        |
                  APPLICATION      |
                  PRIVATE          |
                  empty

TaggedValue  ::= Value

EmbeddedPDVType ::=    EMBEDDED PDV

EmbeddedPDVValue ::=   SequenceValue

ExternalType ::= EXTERNAL

ExternalValue ::= SequenceValue

ObjectIdentifierType  ::=  OBJECT IDENTIFIER

ObjectIdentifierValue ::=  "{" ObjIdComponentList "}"  |
                                "{" DefinedValue ObjIdComponentList "}"

ObjIdComponentList  ::=  ObjIdComponent     |
                         ObjIdComponent ObjIdComponentList

ObjIdComponent  ::=      NameForm          |
                         NumberForm        |
                         NameAndNumberForm

NameForm     ::=         identifier

NumberForm     ::=       number | DefinedValue

NameAndNumberForm ::=          identifier "(" NumberForm ")"

CharacterStringType ::= RestrictedCharacterStringType | UnrestrictedCharacterStringType

RestrictedCharacterStringType ::= BMPString     |

                                  GeneralString       |
                                  GraphicString       |
                                  IA5String           |
                                  ISO646String        |
                                  NumericString       |
                                  PrintableString     |
                                  TeletexString       |
                                  T61String           |
                                  UniversalString     |
                                  VideotexString      |
                                  VisibleString

RestrictedCharacterStringValue ::= cstring | CharacterStringList | Quadruple | Tuple

CharacterStringList  ::=  "{" CharSyms "}"
CharSyms  ::=  CharsDefn | CharSyms "," CharsDefn
CharsDefn  ::=  cstring | DefinedValue

Quadruple      ::= "{"  Group  ","  Plane  ","  Row  ","  Cell "}"
Group          ::= number
Plane          ::= number
Row            ::= number
Cell           ::= number

Tuple ::= "{" TableColumn "," TableRow "}"
TableColumn ::= number
TableRow ::= number

UnrestrictedCharacterStringType ::= CHARACTER STRING

CharacterStringValue ::= RestrictedCharacterStringValue | UnrestrictedCharacterStringValue

UnrestrictedCharacterStringValue ::= SequenceValue

UsefulType    ::= typereference
```

Los siguientes tipos cadena de caracteres se definen en 34.1:

NumericString VisibleString

PrintableString ISO646String

TeletexString IA5String

T61String GraphicString

VideotexString GeneralString

UniversalString BMPString

Los siguientes tipos útiles se definen en 39-41:

GeneralizedTime

UTCTime

ObjectDescriptor

Las siguientes producciones se utilizan en 42-45:

```
ConstrainedType ::=
    Type   Constraint |
    TypeWithConstraint

TypeWithConstraint ::=
    SET Constraint OF Type |
    SET SizeConstraint OF Type |
    SEQUENCE Constraint OF Type |
    SEQUENCE SizeConstraint OF Type

Constraint ::= "(" ConstraintSpec  ExceptionSpec ")"

ConstraintSpec ::=
    SubtypeConstraint |
    GeneralConstraint

ExceptionSpec ::= "!"  ExceptionIdentification | empty

ExceptionIdentification ::= SignedNumber    |
                DefinedValue            |
                Type ":" Value

SubtypeConstraint ::= ElementSetSpec

ElementSetSpec ::= Unions | ALL  Exclusions

Unions ::= Intersections |
            UElems UnionMark Intersections

UElems ::= Unions

Intersections ::= IntersectionElements |
            IElems IntersectionMark IntersectionElements

IElems ::= Intersections

IntersectionElements ::= Elements | Elems Exclusions

Elems ::= Elements

Exclusions ::= EXCEPT Elements

UnionMark  ::=  "|"    |    UNION

IntersectionMark  ::=  "^"     |    INTERSECTION

Elements  ::=
    SubtypeElements |
    ObjectSetElements |
    "(" ElementSetSpec ")"
```

```
SubtypeElements ::=
        SingleValue             |
        ContainedSubtype        |
        ValueRange              |
        PermittedAlphabet       |
        SizeConstraint          |
        TypeConstraint          |
        InnerTypeConstraints

SingleValue    ::=  Value

ContainedSubtype ::= Includes Type

Includes ::= INCLUDES | empty

ValueRange ::= LowerEndpoint ".." UpperEndpoint

LowerEndpoint  ::=  LowerEndValue | LowerEndValue "<"

UpperEndpoint  ::=  UpperEndValue | "<" UpperEndValue

LowerEndValue ::= Value | MIN

UpperEndValue ::= Value | MAX

SizeConstraint ::= SIZE Constraint

PermittedAlphabet ::= FROM Constraint

TypeConstraint ::= Type

InnerTypeConstraints ::=
        WITH COMPONENT SingleTypeConstraint |
        WITH COMPONENTS MultipleTypeConstraints

SingleTypeConstraint::= Constraint

MultipleTypeConstraints ::= FullSpecification | PartialSpecification

FullSpecification   ::= "{" TypeConstraints "}"

PartialSpecification ::= "{"  "..."   ","   TypeConstraints "}"

TypeConstraints ::=
        NamedConstraint |
        NamedConstraint "," TypeConstraints

NamedConstraint ::=
        identifier ComponentConstraint

ComponentConstraint ::= ValueConstraint PresenceConstraint

ValueConstraint ::=  Constraint | empty

PresenceConstraint  ::= PRESENT | ABSENT | OPTIONAL | empty
```