UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T X.693

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

Amendement 1
(10/2003)

SÉRIE X: RÉSEAUX DE DONNÉES ET COMMUNICATION ENTRE SYSTÈMES OUVERTS

Réseautage OSI et aspects systèmes – Notation de syntaxe abstraite numéro un (ASN.1)

Technologies de l'information – Règles de codage ASN.1: règles de codage XML (XER)

Amendement 1: Instructions de codage XER et règles de codage XML étendues (EXTENDED-XER)

Recommandation UIT-T X.693 (2001) – Amendement 1

RECOMMANDATIONS UIT-T DE LA SÉRIE X

RÉSEAUX DE DONNÉES ET COMMUNICATION ENTRE SYSTÈMES OUVERTS

RÉSEAUX PUBLICS DE DONNÉES	
Services et fonctionnalités	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalisation et commutation	X.50–X.89
Aspects réseau	X.90–X.149
Maintenance	X.150–X.179
Dispositions administratives	X.180–X.199
INTERCONNEXION DES SYSTÈMES OUVERTS	
Modèle et notation	X.200–X.209
Définitions des services	X.210–X.219
Spécifications des protocoles en mode connexion	X.220–X.229
Spécifications des protocoles en mode sans connexion	X.230–X.239
Formulaires PICS	X.240–X.259
Identification des protocoles	X.260–X.269
Protocoles de sécurité	X.270–X.279
Objets gérés des couches	X.280–X.289
Tests de conformité	X.290–X.299
INTERFONCTIONNEMENT DES RÉSEAUX	
Généralités	X.300–X.349
Systèmes de transmission de données par satellite	X.350–X.369
Réseaux à protocole Internet	X.370–X.399
SYSTÈMES DE MESSAGERIE	X.400–X.499
ANNUAIRE	X.500–X.599
RÉSEAUTAGE OSI ET ASPECTS SYSTÈMES	
Réseautage	X.600–X.629
Efficacité	X.630–X.639
Qualité de service	X.640–X.649
Dénomination, adressage et enregistrement	X.650–X.679
Notation de syntaxe abstraite numéro un (ASN.1)	**X.680–X.699**
GESTION OSI	
Cadre général et architecture de la gestion-systèmes	X.700–X.709
Service et protocole de communication de gestion	X.710–X.719
Structure de l'information de gestion	X.720–X.729
Fonctions de gestion et fonctions ODMA	X.730–X.799
SÉCURITÉ	X.800–X.849
APPLICATIONS OSI	
Engagement, concomitance et rétablissement	X.850–X.859
Traitement transactionnel	X.860–X.879
Opérations distantes	X.880–X.899
TRAITEMENT RÉPARTI OUVERT	X.900–X.999
SÉCURITÉ DES TÉLÉCOMMUNICATIONS	X.1000–

Pour plus de détails, voir la Liste des Recommandations de l'UIT-T.

NORME INTERNATIONALE ISO/CEI 8825-4
RECOMMANDATION UIT-T X.693

Technologies de l'information – Règles de codage ASN.1: règles de codage XML (XER)

Amendement 1

Instructions de codage XER et règles de codage XML étendues (EXTENDED-XER)

Résumé

Le présent Amendement 1 s'applique aux Recommandations UIT-T X.680 | ISO/CEI 8824-1, UIT-T X.681 | ISO/CEI 8824-2, UIT-T X.690 | ISO/CEI 8825-1, UIT-T X.691 | ISO/CEI 8825-2 et UIT-T X.693 | ISO/CEI 8825-4. Il permet de:

- Rectifier une erreur dans les règles canoniques de codage en langage de balisage extensible (CXER, *canonical extensible markup language (XML) encoding rules*), résultant de la présence d'un espace entre le signe moins et la valeur **`INTEGER`** ou **`REAL`** qui le suit (la règle CXER n'est pas canonique). Ceci n'est plus autorisé dans la notation des valeurs, ni dans la notation des valeurs en langage XML, ni dans les règles de codage en langage XML (XER, *XML encoding rules*) et CXER. **Il s'agit d'une modification** (découlant du texte de l'Amendement 1 à la Rec. X.680 | ISO/CEI 8824-1) **et non d'une adjonction**.
- Ajouter des instructions de codage dans un module en notation ASN.1, soit au moyen d'un préfixe de type, soit dans une section de commande de codage, afin de spécifier les variantes de codage selon les règles de codage en langage XML de base (BASIC-XER, *basic XML encoding rule*). Ces instructions de codage visent à prendre en charge des mappages d'une spécification conforme à une description schématique en langage XML (XSD, *XML schema definition*) sur une spécification en notation ASN.1. Cette disposition correspond à une modification de terminologie dans laquelle un type commençant par "`[…]`" est un type préfixé, et la notation "`[…]`" peut être une étiquette ou non. Cette modification de terminologie conduit à des modifications du texte (mais non du contenu) des spécifications selon les règles de codage de base (BER, *basic encoding rules*) et les règles de codage compact (PER, *packed encoding rules*), de sorte qu'un Amendement est aussi prévu pour ces spécifications.
- Ajouter les nouvelles valeurs NaN (*not-a-number*, pas un nombre) et moins zéro pour la valeur **`REAL`** (la prise en charge du codage de ces nouvelles valeurs est assurée au moyen de l'Amendement 1 aux Recommandations UIT-T X.690 | ISO/CEI 8825-1 et UIT-T X.691 | ISO/CEI 8825-2, ainsi que de l'Amendement 1 à la Rec. UIT-T X.693 | ISO/CEI 8825-4).
- Ajouter les nouvelles notations des valeurs en langage XML pour les valeurs **`REAL`**, **`BOOLEAN`**, **`ENUMERATED`** et **`INTEGER`** qui emploient du texte plutôt que des étiquettes contenant l'élément vide pour ces valeurs. Elles s'expriment en langage XML et selon les règles de codage en langage XML étendues (EXTENDED-XER, *extended XML encoding rules*), mais non selon les règles BASIC-XER (pour des raisons de compatibilité avec ce qui précède).
- Modifier la notation des valeurs en langage XML des types sequence-of (et des codages selon les règles XER) afin de déterminer les valeurs pour lesquelles ceux-ci ne sont plus des éléments en langage XML (cela se produit pour les notations supplémentaires des valeurs en langage XML et n'affecte que l'emploi de celles-ci). Cette modification ne concerne que l'emploi des notations des valeurs en langage XML qui ont été ajoutées conformément au présent Amendement, et n'est pas autorisée dans les codages selon les règles BASIC-XER, qui ne sont pas touchés.

Ainsi est assurée la prise en charge de base nécessaire des règles EXTENDED-XER.

Le présent Amendement dépend des Recommandations UIT-T X.680 (2002)/Amd.1 (2003) | ISO/CEI 8824-1:2002/Amd.1:2003 et UIT-T X.681 (2002)/Amd.1 (2003) | ISO/CEI 8824-2:2002/Amd.1:2003 en ce qui concerne la spécification des variantes de codage pour certains types et pour la syntaxe relative à l'insertion dans une spécification en notation ASN.1 des instructions de codage selon les règles XER. De nombreuses références dans le présent Amendement, aux définitions figurant dans ces Recommandations | Normes internationales, ont trait aux définitions introduites par ces Amendements.

La plus grande partie du présent Amendement porte sur la spécification de la syntaxe et de la sémantique des (nouvelles) instructions de codage selon les règles XER, qui peuvent être employées pour exiger des codeurs selon les règles EXTENDED-XER qu'ils fournissent des codages particuliers pour les types en notation ASN.1. Ces codages sont essentiellement destinés à la prise en charge de la Rec. UIT-T X.694 | ISO/CEI 8825-5 (Mappage de spécifications conformes à la description XSD sur des spécifications en notation ASN.1)

Source

L'Amendement 1 de la Recommandation UIT-T X.693 (2001) a été approuvé le 29 octobre 2003 par la Commission d'études 17 (2001-2004) de l'UIT-T selon la procédure définie dans la Recommandation UIT-T A.8. Un texte identique est publié comme Norme internationale ISO/CEI 8825-4, Amendement 1.

AVANT-PROPOS

L'UIT (Union internationale des télécommunications) est une institution spécialisée des Nations Unies dans le domaine des télécommunications. L'UIT-T (Secteur de la normalisation des télécommunications) est un organe permanent de l'UIT. Il est chargé de l'étude des questions techniques, d'exploitation et de tarification, et émet à ce sujet des Recommandations en vue de la normalisation des télécommunications à l'échelle mondiale.

L'Assemblée mondiale de normalisation des télécommunications (AMNT), qui se réunit tous les quatre ans, détermine les thèmes d'étude à traiter par les Commissions d'études de l'UIT-T, lesquelles élaborent en retour des Recommandations sur ces thèmes.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution 1 de l'AMNT.

Dans certains secteurs des technologies de l'information qui correspondent à la sphère de compétence de l'UIT-T, les normes nécessaires se préparent en collaboration avec l'ISO et la CEI.

NOTE

Dans la présente Recommandation, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation reconnue.

Le respect de cette Recommandation se fait à titre volontaire. Cependant, il se peut que la Recommandation contienne certaines dispositions obligatoires (pour assurer, par exemple, l'interopérabilité et l'applicabilité) et considère que la Recommandation est respectée lorsque toutes ces dispositions sont observées. Le futur d'obligation et les autres moyens d'expression de l'obligation comme le verbe "devoir" ainsi que leurs formes négatives servent à énoncer des prescriptions. L'utilisation de ces formes ne signifie pas qu'il est obligatoire de respecter la Recommandation.

DROITS DE PROPRIÉTÉ INTELLECTUELLE

L'UIT attire l'attention sur la possibilité que l'application ou la mise en œuvre de la présente Recommandation puisse donner lieu à l'utilisation d'un droit de propriété intellectuelle. L'UIT ne prend pas position en ce qui concerne l'existence, la validité ou l'applicabilité des droits de propriété intellectuelle, qu'ils soient revendiqués par un Membre de l'UIT ou par une tierce partie étrangère à la procédure d'élaboration des Recommandations.

A la date d'approbation de la présente Recommandation, l'UIT n'avait pas été avisée de l'existence d'une propriété intellectuelle protégée par des brevets à acquérir pour mettre en œuvre la présente Recommandation. Toutefois, comme il ne s'agit peut-être pas de renseignements les plus récents, il est vivement recommandé aux responsables de la mise en œuvre de consulter la base de données des brevets du TSB.

TABLE DES MATIÈRES

NORME INTERNATIONALE
RECOMMANDATION UIT-T

Technologies de l'information – Règles de codage ASN.1: Règles de codage XML (XER)

Amendement 1

Instructions de codage XER et règles de codage XML étendues (EXTENDED-XER)

NOTE – Dans le présent amendement, tout texte nouveau ou modifié a été souligné dans les paragraphes à modifier. Lors de l'incorporation de ce texte dans la Recommandation de base, il convient de supprimer ce soulignement.

Remplacer l'introduction par ce qui suit:

Introduction

Dans l'ensemble des Recommandations UIT-T X.680 | ISO/CEI 8824-1, UIT-T X.681 | ISO/CEI 8824-2, UIT-T X.682 | ISO/CEI 8824-3 et UIT-T X.683 | ISO/CEI 8824-4 est décrite la notation de syntaxe abstraite numéro un (ASN.1, *abstract syntax notation one*), qui permet de définir les messages échangés par des applications homologues.

Dans la présente Recommandation | Norme internationale sont définies les règles de codage, qui pourront être appliquées à des valeurs de types en notation ASN.1, déterminés au moyen de la notation spécifiée dans les Recommandations | Normes internationales susmentionnées. L'application de ces règles de codage produit une syntaxe de transfert pour ces valeurs. Il est implicitement entendu, lors de la spécification de ces règles de codage, que celles-ci pourront aussi servir au décodage.

Plusieurs ensembles de règles de codage peuvent être appliqués à des valeurs de types en notation ASN.1. Dans la présente Recommandation | Norme internationale sont définis trois ensembles de règles de codage utilisant le langage de balisage extensible (XML, *extensible markup language*). Ces règles de codage produisent toutes un document en langage XML, conforme à la version XML 1.0 du Consortium W3C. Le premier ensemble consiste en les règles de codage en langage XML (XER, *XML encoding rule*) de base (BASIC-XER, *basic XML encoding rules*), le deuxième ensemble englobe les règles canoniques de codage en langage XML (CANONICAL-XER ou CXER, *canonical XML encoding rules*), celles-ci ne permettant de coder une valeur en notation ASN.1 que d'une seule manière (les règles canoniques de codage sont généralement employées pour des applications où interviennent des fonctions liées à la sécurité telles que les signatures numériques), tandis que le troisième ensemble regroupe les règles de codage en langage XML étendues (EXTENDED-XER, *extended XML encoding rules*). Ces dernières permettent de disposer de plus d'options pour les codeurs, et de tenir compte des instructions de codage qui spécifient les variantes des codages selon les règles BASIC-XER, de manière à prendre en charge les styles particuliers d'un document en langage XML (voir ci-après). Les règles étendues de codage en langage XML ne sont pas canoniques, et il n'en est pas donné de forme canonique dans la présente Recommandation | Norme internationale.

De nombreux aspects de la représentation des données en langage XML (tels que l'emploi des attributs en langage XML au lieu des éléments descendants, ou l'emploi de listes délimitées par des blancs) sont une question de style et de choix du concepteur en langage XML. Lorsqu'un type défini dans une spécification en notation ASN.1 est codé selon les règles BASIC-XER ou CXER, la représentation en langage XML ne permet d'employer qu'un style déterminé, sans intervention possible de l'utilisateur en ce qui concerne la stylistique. Dans la présente Recommandation UIT-T | Norme internationale sont spécifiées la syntaxe et la sémantique des instructions de codage selon les règles XER, qui définissent la stylistique du langage XML dans un codage selon les règles EXTENDED-XER. Il n'est pas tenu compte de ces instructions de codage selon les règles XER dans les règles BASIC-XER ou CXER, mais uniquement dans les règles EXTENDED-XER.

NOTE – La "stylistique", telle que l'emploi des attributs ou de listes délimitées par des blancs, peut aussi avoir une incidence sur la dimension d'un codage et sur sa facilité de traitement, de sorte que l'emploi de ces caractéristiques n'est pas simplement une question de style. Lorsque ces questions sont importantes, on peut vouloir préférer pour les instructions de codage les règles EXTENDED-XER aux règles BASIC-XER ou CXER.

Le § 8 définit le codage des types en notation ASN.1 selon les règles BASIC-XER.

Le § 9 définit le codage des types en notation ASN.1 selon les règles CXER.

Le § 10 spécifie le codage des types en notation ASN.1 selon les règles EXTENDED-XER, se référant à des paragraphes ultérieurs où sont définies les instructions de codage XER.

Dans les § 11 à 14 sont énumérées et classées les instructions de codage selon les règles XER, et est indiquée la syntaxe permettant de les attribuer à un type ou à une composante en notation ASN.1 au moyen soit d'un préfixe de type conforme aux règles XER (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 30.3) soit d'une section de commande de codage selon les règles XER (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 50).

Le § 15 définit l'ordre de préséance lorsque les instructions de codage selon les règles XER sont présentes tant dans un préfixe de type conforme aux règles XER que dans une section de commande de codage selon les règles XER.

Le § 16 décrit la prise en charge de l'instruction de codage selon les règles XER pour les espaces de noms en langage XML lors de l'emploi des règles EXTENDED-XER.

Le § 17 spécifie les codages selon les règles EXTENDED-XER.

Dans les § 18 à 39 sont définies:

a) la syntaxe pour chacune des instructions de codage selon les règles XER, qui sont employées dans un préfixe de type ou dans une section de commande de codage;

b) les restrictions concernant les instructions de codage selon les règles XER, qui peuvent être associées à un type particulier en notation ASN.1 (résultant de l'héritage ou des attributions multiples);

c) les modifications à apporter aux règles de codage XER, qui sont requises dans un codage selon les règles EXTENDED-XER lorsqu'une instruction de codage selon les règles XER est appliquée.

A l'Annexe A sont donnés, à titre informatif, des exemples des codages selon les règles BASIC-XER et CXER.

A l'Annexe B est donnée, à titre informatif, une description du contenu partiel en langage XML, qui est produit lorsque les étiquettes entourant des entités telles qu'une séquence ou une sequence-of sont enlevées, et que sont appliquées des restrictions sur les spécifications qui permettent une détermination facile de la composante en notation ASN.1 à laquelle un élément du langage XML est associé.

A l'Annexe C sont donnés, à titre informatif, des exemples d'instructions de codage selon les règles XER et les codages selon les règles EXTENDED-XER correspondants.

Remplacer le § 1 par le suivant:

1 Domaine d'application

Dans la présente Recommandation | Norme internationale est défini un ensemble de règles de codage en langage XML de base (BASIC-XER), qui peuvent être utilisées pour élaborer une syntaxe de transfert applicable à des valeurs de types définis dans les Recommandations UIT-T X.680 | ISO/CEI 8824-1 et UIT-T X.681 | ISO/CEI 8824-2. Y est aussi défini un ensemble de règles canoniques de codage en langage XML (CXER), qui impose des contraintes aux règles de codage en langage XML de base de manière à produire un codage unique pour toute valeur donnée en notation ASN.1. Finalement, y est aussi spécifié un ensemble de règles étendues de codage en langage XML (EXTENDED-XER), qui offre aux codeurs d'autres options et permet aussi au responsable de la formulation en notation ASN.1 de modifier le codage par rapport à celui qui serait produit par le codage selon les règles BASIC-XER. Il est implicitement entendu, lors de la spécification de ces règles de codage, que celles-ci pourront aussi servir au décodage.

Les règles de codage définies dans la présente Recommandation | Norme internationale:

- sont utilisées au moment de la communication;
- sont destinées à être employées dans des circonstances où l'affichage de valeurs ou leur traitement au moyen d'outils du langage XML courants (tels que des navigateurs) sont les principales considérations dans le choix des règles de codage;
- permettent l'extension d'une syntaxe abstraite au moyen de l'adjonction de valeurs supplémentaires pour toutes les formes d'extension décrites dans la Rec. UIT-T X.680 | ISO/CEI 8824-1.

Dans la présente Recommandation | Norme internationale sont également spécifiées la syntaxe et la sémantique des instructions de codage selon les règles XER, ainsi que les règles de leur attribution et de leur combinaison. Les instructions de codage selon les règles XER peuvent être employées pour commander le codage selon les règles EXTENDED-XER pour des types particuliers en notation ASN.1.

Remplacer le § 2.1 comme suit:

2.1 Recommandations | Normes internationales identiques

– Recommandation UIT-T X.680 (2002) | ISO/CEI 8824-1:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification de la notation de base.*

– Recommandation UIT-T X.680 (2002)/Amd.1 (2003) | ISO/CEI 8824-1:2002/Amd.1:2003, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification de la notation de base – Amendement 1: prise en charge des règles de codage XML étendues (EXTENDED-XER).*

– Recommandation UIT-T X.681 (2002) | ISO/CEI 8824-2:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des objets informationnels.*

– Recommandation UIT-T X.681 (2002)/Amd.1 (2003) | ISO/CEI 8824-2:2002/Amd.1:2003, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des objets informationnels – Amendement 1: prise en charge des règles de codage XML étendues (EXTENDED-XER).*

– Recommandation UIT-T X.682 (2002) | ISO/CEI 8824-3:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un*: *spécification des contraintes.*

– Recommandation UIT-T X.683 (2002) | ISO/CEI 8824-4:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: paramétrage des spécifications de la notation de syntaxe abstraite numéro un.*

– Recommandation UIT-T X.690 (2002) | ISO/CEI 8825-1:2002, *Technologies de l'information – Règles de codage ASN.1: spécification des règles de codage de base, des règles de codage canoniques et des règles de codage distinctives.*

– Recommandation UIT-T X.690 (2002)/Amd.1 (2003) | ISO/CEI 8825-1:2002/Amd.1:2003, *Technologies de l'information – Règles de codage ASN.1: spécification des règles de codage de base, des règles de codage canoniques et des règles de codage distinctives – Amendement 1: prise en charge des règles de codage XML étendues (EXTENDED-XER).*

– Recommandation UIT-T X.691 (2002) | ISO/CEI 8825-2:2002, *Technologies de l'information – Règles de codage ASN.1: spécification des règles de codage compact*.

– Recommandation UIT-T X.691 (2002)/Amd.1 (2003) | ISO/CEI 8825-2:2002/Amd.1:2003, *Technologies de l'information – Règles de codage ASN.1: spécification des règles de codage compact – Amendement 1: prise en charge des règles de codage XML étendues (EXTENDED-XER).*

– Recommandation UIT-T X.692 (2002) | ISO/CEI 8825-3:2002, *Technologies de l'information – Règles de codage ASN.1: spécification de la notation de contrôle de codage (ECN).*

Remplacer le § 2.2 comme suit:

2.2 Autres références

– IETF RFC 2045 (1996), *Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies.*

– IETF RFC 2141 (1997), *URN Syntax.*

– IETF RFC 2396 (1998), *Uniform Resource Identifiers (URI): Generic Syntax.*

– IETF RFC 3061 (2001), *A URN Namespace of Object Identifiers.*

– ISO/CEI 10646-1:2000, *Technologies de l'information – Jeu universel de caractères codés à plusieurs octets – Partie 1*: *Architecture et table multilingue.*

– The Unicode Standard, Version 3.2.0, The Unicode Consortium. (Reading, MA, Addison-Wesley)

NOTE – Les caractères graphiques (et leur codage) définis dans la référence susmentionnée sont identiques à ceux qui sont spécifiés dans la norme ISO/CEI 10646-1, mais la référence ci-dessus a été incluse parce qu'elle définissait aussi les noms des caractères de commande.

– W3C XML 1.0:2000, *Extensible Markup Language (XML) 1.0 (deuxième édition), Recommandation du Consortium W3C, Copyright © [6 octobre 2000] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2000/REC-xml-20001006.*

– W3C XML Namespaces:1999, *Namespaces in XML, Recommandation du Consortium W3C, Copyright © [14 janvier 1999] World Wide Web Consortium, (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/1999/REC-xml-names-19990114.*

NOTE – La référence à un document dans la présente Recommandation | Norme internationale ne donne pas à ce document en tant que tel le statut d'une Recommandation ou Norme internationale.

Remplacer le § 3 comme suit:

3 Définitions

Aux fins de la présente Recommandation | Norme internationale, les définitions de la Rec. UIT-T X.680 | ISO/CEI 8824-1, ainsi que les définitions suivantes s'appliquent.

3.1 Règles de codage de base en notation ASN.1

La présente Recommandation | Norme internationale utilise les termes suivants définis dans la Rec. UIT-T X.690 | ISO/CEI 8825-1:

a) valeur de donnée;

b) conformité dynamique;

c) codage (d'une valeur de donnée);

d) destinataire;

e) expéditeur;

f) conformité statique.

3.2 Autres définitions

Pour les besoins de la présente Recommandation | Norme internationale, les définitions suivantes s'appliquent.

3.2.1 schéma en notation ASN.1: définition du contenu et de la structure des données, utilisant une définition de type en notation ASN.1.

NOTE – Cela permet aux règles de codage de produire des codages binaires des valeurs d'un type en notation ASN.1 ou des codages utilisant le langage XML.

Ajouter les 4 nouveaux § 3.2.1 bis, 3.2.1 ter, 3.2.1 quat et 3.2.1 quin comme suit:

3.2.1 *bis* étiquette associée d'élément vide: étiquette d'élément vide en langage XML, qui peut remplacer, lorsqu'elles sont présentes, une étiquette associée précédente et une étiquette associée suivante.

3.2.1 *ter* instruction associée de codage (pour un type): ensemble d'instructions de codage selon les règles XER, associées à un type.

3.2.1 *quat* étiquette associée suivante: étiquette de fin en langage XML, qui suit la valeur "XMLValue" d'un type en l'absence d'instructions de codage permettant d'enlever les étiquettes associées.

3.2.1 *quin* étiquette associée précédente: étiquette de début en langage XML, qui précède la valeur "XMLValue" d'un type en l'absence d'instructions de codage permettant d'enlever les étiquettes associées.

Ajouter les 16 nouveaux § 3.2.2 bis à 3.2.2 septdec comme suit:

3.2.2 *bis* document canonique valable en langage XML (pour un schéma en notation ASN.1): document en langage XML bien formé (voir la version XML 1.0 du Consortium W3C), dont le contenu est conforme à la spécification des règles CXER pour le codage d'un type en notation ASN.1, défini par un schéma en notation ASN.1.

3.2.2 *ter* type susceptible d'être codé au moyen de caractères: type en notation ASN.1, auquel une instruction de codage **`ATTRIBUTE`** peut être appliquée (voir le § 20.2.1).

3.2.2 *quat* espace de nom de commande: espace de nom employé pour identifier les attributs qui assurent les fonctions ou permettent d'acheminer les valeurs commandant le codage selon les règles EXTENDED-XER.

NOTE 1 – L'attribut d'identification du type en est un exemple. L'espace de nom de commande est par défaut celui en notation ASN.1, qui est spécifié dans le § 16.9, mais peut être modifié par l'instruction de codage **`GLOBAL-DEFAULTS`**.

NOTE 2 – L'espace de nom de commande peut aussi contenir les noms des attributs qui peuvent être présents, mais dont les décodeurs selon les règles EXTENDED-XER ne tiennent pas compte (voir le § 10.2.10). Un tel nom d'attribut est par exemple **schemaLocation**.

3.2.2 ***quin*** **type (en notation ASN.1) contenu**: type en notation ASN.1 dont la valeur "XMLValue" dans un codage selon les règles BASIC-XER est contenue directement dans la valeur "XMLValue" d'un type en notation ASN.1 (le type contenant).

NOTE – Tous les types dans un codage selon les règles BASIC-XER ou EXTENDED-XER sont des types contenus, à moins qu'ils ne soient employés dans un codage comme types de base (voir le § 10.3.1 b).

3.2.2 ***sex*** **élément contenant (un type en notation ASN.1)**: valeurs "ExtendedXMLTypedValue", "ExtendedXMLChoiceValue", "ExtendedXMLNamedValue" ou "ExtendedXMLDelimitedItem" dont la valeur "ExtendedXMLValue" est égale au codage du type de valeur "ExtendedXMLValue" (voir les § 17.1, 17.5, 17.6 et 17.7).

3.2.2 ***sept*** **type contenant (un type en notation ASN.1)**: type en notation ASN.1 dont la valeur "XMLValue" dans un codage selon les règles BASIC-XER contient directement la valeur "XMLValue" d'un type en notation ASN.1 (le type contenu).

NOTE – Le type contenant peut être un type séquence, un type ensemble, un type choix, un type sequence-of, un type set-of, un type ouvert, ou un type octetstring ou bitstring (avec l'indication **`CONTAINING`** mais sans l'indication **`ENCODED BY`**).

3.2.2 ***oct*** **instructions finales de codage (pour un type)**: ensemble d'instructions de codage selon les règles XER, qui sont associées à un type, à la suite de la spécification complète en notation ASN.1, et qui sont appliquées pour produire les codages de ce type.

3.2.2 ***non*** **instructions de codage héritées**: instructions de codage selon les règles XER, qui sont associées au type identifié par une référence à lui-même.

3.2.2 ***dec*** **nom restrictif d'espace de nom**: nom dans un document en langage XML, qui possède un préfixe d'espace de nom en langage XML ou est un nom d'élément en langage XML, dans le cadre d'une déclaration de nom d'espace par défaut en langage XML.

NOTE – Les déclarations d'espace de nom par défaut en langage XML n'affectent que les éléments en langage XML et non les noms des attributs. Un préfixe d'espace de nom peut s'appliquer aux deux.

3.2.2 ***unodec*** **attribut d'identification de valeur nulle**: attribut en langage XML qui peut figurer dans un élément quelconque et permet de préciser si le contenu a une valeur **nulle** (voir le § 33).

3.2.2 ***duodec*** **contenu partiel d'un élément en langage XML**: éléments descendants en langage XML définis par un type en notation ASN.1, qui est un type **`UNTAGGED`** et qui fournit une partie du contenu de l'élément en langage XML, produit par le type contenant.

NOTE – Si le type contenant est lui-même un type **`UNTAGGED`**, il peut aussi ne produire que des contenus partiels d'élément en langage XML.

3.2.2 ***tredec*** **instructions préfixées de codage**: instructions de codage selon les règles XER, qui sont attribuées au moyen d'un préfixe de type.

NOTE – Les instructions préfixées de codage permettent de supprimer, de remplacer ou d'ajouter des instructions associées de codage d'un type.

3.2.2 ***quatdec*** **information restrictive**: information fournie dans le cadre de la spécification d'une liste de cibles dans une section de commande de codage selon les règles XER.

3.2.2 ***quindec*** **instructions ciblées de codage**: instructions de codage selon les règles XER, qui sont attribuées à l'aide d'une liste de cibles dans une section de commande de codage selon les règles XER.

NOTE – Les instructions ciblées de codage permettent de supprimer, de remplacer ou d'ajouter des instructions associées de codage d'un type.

3.2.2 ***sexdec*** **attribut d'identification du type**: attribut en langage XML, qui peut figurer dans un élément quelconque et qui permet d'identifier le type de cet élément (voir le § 37).

3.2.2 ***septdec*** **identificateur uniforme de ressource (URI,** ***uniform resource identifier*****)**: identificateur non ambigu à l'échelle mondiale, attribué conformément à l'un quelconque d'un nombre de schémas URI et employé pour identifier les espaces de nom dans les codages selon les règles EXTENDED-XER.

NOTE – Le schéma URI employé par défaut pour la notation ASN.1 assure qu'une valeur d'identificateur d'objet en notation ASN.1 permettra d'identifier les espaces de nom (voir les § 16.9 et 29.1.5).

Remplacer le § 3.2.3 comme suit:

3.2.3 document valable en langage XML (pour un schéma en notation ASN.1): document en langage XML bien formé (voir la version XML 1.0 du Consortium W3C), dont le contenu est conforme à la spécification des règles BASIC-XER, CXER ou EXTENDED-XER pour le codage d'un type en notation ASN.1, défini par un schéma en notation ASN.1 et comportant éventuellement des instructions de codage selon les règles XER.

Ajouter les 4 nouveaux § 3.2.3 bis à 3.2.3 quin comme suit:

3.2.3 *bis* instructions de codage selon les règles XER: instructions de codage associées à un type en notation ASN.1 (ou à une composante d'un type en notation ASN.1), qui sont attribuées à ce type (ou à cette composante) dans un préfixe de type selon les règles XER (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 30.3) ou dans une section de commande de codage selon les règles XER (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 50).

3.2.3 *ter* attribut en langage XML: partie d'un codage selon les règles EXTENDED-XER consistant en une valeur "XMLValue" placée entre guillemets ou apostrophes et précédée d'un nom (attribut) et d'un signe égal.

3.2.3 *quat* élément en langage XML: partie d'un document en langage XML, spécifiée dans la version XML 1.0 du Consortium W3C.

NOTE – Un élément en langage XML est l'étiquette d'un élément vide, ou commence par une étiquette de début et se termine par une étiquette de fin. Tant l'étiquette de début que l'étiquette d'élément vide peuvent contenir des codages d'attribut.

3.2.3 *quin* nom de l'élément en langage XML: unité lexicale suivant un signe "<" ou "</" dans les étiquettes associées.

Remplacer le § 3.2.4 comme suit:

3.2.4 document en langage XML: séquence de caractères, conforme à la définition de document de la version XML 1.0 du Consortium W3C.

Ajouter les 2 nouveaux § 3.2.5 et 3.2.6 comme suit:

3.2.5 instruction de traitement en langage XML: partie d'un document en langage XML qui permet d'acheminer des informations concernant le traitement en partie ou en totalité de ce document (voir la version XML 1.0 du Consortium W3C).

NOTE – L'instruction de traitement permet d'identifier le type de traitement qui est applicable, et dont il n'est pas tenu compte dans les autres traitements. Elle pourrait être employée pour identifier une feuille de style à appliquer lorsque le document est lu.

3.2.6 prologue en langage XML: partie initiale d'un document en langage XML (qui ne permet pas d'acheminer des informations sur la valeur du type codé en notation ASN.1).

Remplacer le § 4 comme suit:

4 Abréviations

Pour les besoins de la présente Recommandation | Norme internationale, les abréviations suivantes s'appliquent:

ASN.1 notation de syntaxe numéro un (*abstract syntax notation one*)

CXER règles canoniques de codage en langage XML (*canonical XML encoding rules*)

PDU unité de données protocolaire (*protocol data unit*)

UCS jeu universel de caractères codés sur plusieurs octets (*universal multiple-octet coded character set*)

URI identificateur uniforme de ressource (*uniform resource identifier*)

UTC temps universel coordonné (*coordinated universal time*)

UTF-8 format de transformation d'un jeu UCS à 8 bits (*UCS transformation format, 8-bit form*)

XER règles de codage en langage XML (*XML encoding rules)*

XML langage de balisage extensible (*extensible markup language*)

Remplacer le § 5 comme suit:

5 Cette section a été supprimée par l'Amendement 1

Remplacer les § 6.1, 6.2 et 6.3 comme suit:

6.1 La présente Recommandation | Norme internationale définit trois ensembles de règles de codage:

- Règles de codage en langage XML de base (BASIC-XER).
- Règles canoniques de codage en langage XML (CXER).
- Règles étendues de codage en langage XML (EXTENDED-XER).

6.2 L'ensemble de base des règles de codage définies dans la présente Recommandation | Norme internationale est l'ensemble de règles BASIC-XER, qui ne produit habituellement pas de codage canonique, et ne permet pas à un utilisateur de commander le style du langage XML produit.

6.3 Un deuxième ensemble de règles de codage définies dans la présente Recommandation | Norme internationale est l'ensemble de règles CXER, qui produit des codages canoniques. Cela correspond à une restriction des choix en fonction de l'implémentation dans le codage BASIC-XER.

NOTE 1 – Toute implémentation conforme aux règles de codage CXER est conforme aux règles de codage BASIC-XER. Toute implémentation conforme au décodage BASIC-XER est conforme au décodage CXER. Donc, les codages effectués conformément aux règles CXER sont des codages autorisés par les règles BASIC-XER.

NOTE 2 – Les règles CXER produisent des codages qui s'appliquent lorsque l'authentification doit être appliquée à des valeurs abstraites.

Ajouter le nouveau § 6.3 bis comme suit:

6.3 *bis* Le troisième ensemble de règles de codage définies dans la présente Recommandation | Norme internationale est l'ensemble de règles EXTENDED-XER. Cela correspond à des variantes des codages BASIC-XER spécifiés par les instructions de codage XER (voir le § 6 *bis*), associées à un type ASN.1. En l'absence d'instructions de codage XER, un codage EXTENDED-XER ne diffère d'un codage BASIC-XER que par le fait qu'il offre plusieurs options aux codeurs.

Remplacer le § 6.4 par le suivant:

6.4 Si un type codé au moyen des règles CXER contient les types **`EMBEDDED PDV`**, **`EXTERNAL`** ou **`CHARACTER STRING`**, le codage extérieur cesse d'être canonique à moins que le codage employé pour tous les types **`EMBEDDED PDV`**, **`EXTERNAL`** et **`CHARACTER STRING`** ne soit canonique.

Ajouter le nouveau § 6 bis comme suit:

6 *bis* Instructions de codage définies dans la présente Recommandation | Norme internationale

6 *bis*.1 La présente Recommandation | Norme internationale définit la syntaxe et la sémantique des instructions de codage XER (voir les § 11 à 39). Les instructions de codage XER n'affectent que les codages EXTENDED-XER.

6 *bis*.2 La notation ASN.1 est la notation de base du schéma XML. Le schéma ASN.1 sert à définir le contenu et la structure des données au moyen de la notation ASN.1 et des règles de codage BASIC-XER (et CXER). Elle peut être employée sans instruction de codage XER.

6 *bis*.3 Les instructions de codage XER assurent une plus grande souplesse dans les documents XML qui peuvent être spécifiés.

6 *bis*.4 Des instructions de codage XER sont attribuées aux définitions des types ASN.1 ou aux références à des types au moyen d'un ou de deux préfixes de type XER (voir la Rec. UIT.T X.680 | ISO/CEI 8824-1, § 30.3) et d'une section de commande de codage XER (voir la Rec. UIT.T X.680 | ISO/CEI 8824-1, § 50). Si des instructions de codage sont associées à une définition de type, elles sont acheminées avec le type ASN.1 (par l'intermédiaire de la référence qui lui est faite) dans d'autres définitions de type et d'autres modules ASN.1. Lorsqu'un type est codé au moyen des règles EXTENDED-XER et que des instructions de codage XER sont associées à certaines ou à toutes ses parties, ces instructions finales de codage sont appliquées de manière à modifier les codages EXTENDED-XER produits.

NOTE – Les instructions finales de codage sont aussi employées lors de la validation et/ou du décodage d'un codage EXTENDED-XER.

Remplacer le § 7.1 par le suivant:

7.1 La conformité dynamique des règles de codage XML de base est définie dans le § 8, tandis que celle des règles canoniques de codage XML est définie dans le § 9 et que celle des règles étendues de codage XML est définie dans le § 10.

Remplacer le § 7.3 par le suivant:

7.3 Des variantes de codage sont autorisées par les règles de codage XML de base et par les règles étendues de codage XML en tant qu'options pour le codeur. Les décodeurs réputés conformes aux règles BASIC-XER prendront en charge toutes les variantes de ce codage, tandis que les décodeurs réputés conformes aux règles EXTENDED-XER prendront en charge toutes les variantes de ce codage.

NOTE – Ce paragraphe s'applique lorsque des instructions finales de codage sont présentes ou non.

Remplacer le § 8.1 et ses paragraphes comme suit:

8.1 Production d'un codage BASIC-XER complet

8.1.1 Un codage BASIC-XER conforme consiste en un document XML valide qui comportera (dans l'ordre):

a) un prologue XML (qui peut être vide) tel que spécifié dans le § 8.2;

b) un élément de document XML qui consiste en le codage complet d'une valeur d'un type ASN.1 unique tel que spécifié dans le § 8.3.

8.1.2 La spécification dans les § 8.2 à 8.6 définit complètement le codage BASIC-XER.

NOTE – D'autres structures de la version XML 1.0 du Consortium W3C, telles que les instructions de traitement XML, ne sont pas autorisées par ces paragraphes et ne sont jamais produites par un codeur conforme aux règles BASIC-XER.

8.1.3 Le document XML sera codé au moyen du format UTF-8 de manière à produire une chaîne d'octets qui constitue le codage spécifié dans la présente Recommandation | Norme internationale. L'identificateur d'objet ASN.1 pour ces règles de codage est défini dans le § 40.

8.1.4 Lorsque le terme "blanc" est employé dans la présente Recommandation | Norme internationale, cela correspond à un ou à plusieurs des caractères suivants de la norme Unicode: HORIZONTAL TABULATION (9), LINE FEED (10), CARRIAGE RETURN (13), SPACE (32). Les nombres entre parenthèses sont les valeurs décimales des caractères dans cette norme. Le nombre et le choix des caractères correspondant aux blancs sont laissés à l'appréciation du codeur.

Ajouter le nouveau § 8.1.5 comme suit:

8.1.5 Lorsque le terme "blanc avec échappement" est employé dans la présente Recommandation | Norme internationale, cela correspond à l'un ou à plusieurs des caractères énumérés au § 8.1.4, avec en option pour le codeur la possibilité de représenter chacun de ces caractères par une séquence d'échappement de la forme "**&#*n***;" ou "**&#x*n***;" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.15.8).

Remplacer le § 8.3.1 par le suivant:

8.3.1 L'élément de document XML sera une valeur "XMLTypedValue", telle que spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 15.2, avec les modifications et les restrictions définies dans les paragraphes suivants du § 8.3.

Ajouter le nouveau § 8.3.1 bis comme suit:

8.3.1 *bis* Toutes les occurrences de la référence "ExternalTypeReference" dans la valeur "XMLTypedValue" seront remplacées par la référence "typereference" dans cette référence "ExternalTypeReference".

Remplacer le § 8.3.3 par le suivant:

8.3.3 Lorsque la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.1.4, 11.11 et 11.13, autorise l'emploi de blancs ASN.1 entre des unités lexicales ou dans des chaînes "xmlbstring" ou "xmlhstring", les caractères employés se limiteront aux "blancs" spécifiés dans le § 8.1.4.

Ajouter le nouveau § 8.3.3 bis comme suit:

8.3.3 *bis* La valeur "XMLBooleanValue" spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 17.3, ne sera qu'un élément booléen "EmptyElementBoolean" et les valeurs "XMLSequenceOfValue" et "XMLSetOfValue" dont une composante est le type booléen ne seront que la liste "ValueList".

Remplacer le § 8.3.4 par le suivant:

8.3.4 La valeur "XMLIntegerValue" spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 18.9 ne sera qu'un nombre "XMLSignedNumber".

Ajouter les 2 nouveaux § 8.3.4 bis et 8.3.4 ter comme suit:

8.3.4 *bis* La valeur "XMLEnumeratedValue" spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 19.8, ne vaudra que "EmptyElementEnumerated" et les valeurs "XMLSequenceOfValue" et "XMLSetOfValue" dont une composante est le type énuméré ne seront que la liste "ValueList".

8.3.4 *ter* La valeur "XMLSpecialRealValue" spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 20.6, ne vaudra que "EmptyElementReal".

Ajouter un nouveau § 8.5 comme suit:

8.5 Codage du type ouvert

Les deux variantes des valeurs "XMLOpenTypeFieldVal" (voir la Rec. UIT-T X.681 | ISO/CEI 8824-2, § 14.6) peuvent être employées.

NOTE – L'emploi de la variante "xmlhstring" de la valeur "XMLOpenTypeFieldVal" n'est habituellement pas recommandé, parce qu'il n'existe pas de mécanisme permettant d'identifier les règles de codage employées pour produire la chaîne "xmlhstring" dans une instance de codage. Les cas où cette variante peut convenir sont ceux pour lesquels le message codé selon les règles XER (par exemple, dans un but d'affichage) résulte d'un codage binaire précédent et n'a pas encore été complètement décodé, ou lorsqu'il existe des accords bilatéraux.

Ajouter un nouveau § 8.6 comme suit:

8.6 Décodage des types contenant des marqueurs d'extension

8.6.1 Un décodeur BASIC-XER acceptera en tant que document XML valable des codages BASIC-XER des types dont les marqueurs d'extension comportent des extensions inconnues.

8.6.2 Des extensions inconnues dans un type séquence ou ensemble conduisent à des éléments XML imprévus dont les noms diffèrent de ceux de l'élément XML prévu suivant.

NOTE – Il peut y avoir plusieurs noms pour un élément XML connu suivant lorsque le choix est possible, mais les extensions ajoutées auront toujours des noms qui diffèrent de tous ceux-ci.

8.6.3 Des extensions inconnues dans un type choix conduisent à un seul élément XML imprévu au lieu d'un élément correspondant à l'un des choix connus. Son nom d'élément XML différera de celui de tout élément XML servant au codage d'une variante connue du type choix.

8.6.4 Des extensions inconnues dans un type énuméré conduisent à un élément XML de contenu imprévu, mais sans éléments XML imprévus.

8.6.5 Des extensions inconnues provenant de la relaxe d'une contrainte relative à un sous-type conduisent à un codage qui peut être un codage valable de toute valeur du type auquel aucune contrainte n'est imposée. De tels codages peuvent produire un contenu imprévu, mais pas d'éléments XML imprévus.

Remplacer le § 9.1 par le suivant:

9.1 Règles générales pour les règles XER canoniques

Remplacer le § 9.1.2 par le suivant (en conservant la Note):

9.1.2 Toutes les unités lexicales formant la valeur "XMLTypedValue" ne seront pas séparées par des "blancs" (voir le § 8.3.3).

Remplacer le § 9.3.1 par le suivant:

9.3.1 Si la variante "XMLTypedValue" de la valeur "XMLBitStringValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 21.9) peut être employée (telle que spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 21.10), alors elle doit l'être. Sinon la variante "xmlbstring" sera utilisée, tous les "blancs" étant supprimés (voir le § 8.3.3).

Remplacer le § 9.4 par le suivant:

9.4 Valeur de octetstring

Si la variante "XMLTypedValue" de la valeur "XMLOctetStringValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 22.3) peut être employée (telle que spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 22.4), alors elle doit l'être. Sinon la variante "xmlhstring" sera utilisée, tous les "blancs" étant supprimés (voir le § 8.3.3), et toutes les lettres étant en caractères majuscules.

Remplacer le § 9.6.1 par le suivant:

9.6.1 Les éléments de la liste "RootComponentTypeList" du type ensemble doivent être classés selon l'ordre canonique spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 8.6 et, en outre, aux fins de déterminer l'ordre dans lequel les composantes sont codées lorsqu'une ou plusieurs composantes sont le type choix sans étiquette ASN.1, chacun de ces types choix est classé comme s'il avait une étiquette identique à la plus petite étiquette dans la liste "RootAlternativeTypeList" de ce type choix ou de tout type choix qui y est emboîté.

Remplacer le § 9.7.1 par le suivant:

9.7.1 L'ordre des éléments d'une valeur "XMLSetOfValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 27.3) sera déterminé en triant les chaînes de caractères qui représentent le codage CXER pour chaque élément spécifié dans les § 9.7.2 et 9.7.3.

Ajouter un nouveau § 9.12 comme suit:

9.12 Valeur du type ouvert

La variante "xmlhstring" de la valeur "XMLOpenTypeFieldVal" ne sera pas employée (voir le § 8.5).

Ajouter les 30 nouveaux paragraphes 10 à 39 avant le § 10 existant (auquel est attribuée par le présent Amendement la nouvelle numérotation 40):

10 Règles étendues de codage en langage XML

10.1 Généralités

10.1.1 Les règles étendues de codage XML (EXTENDED-XER) élargissent et modifient les règles BASIC-XER. Elles permettent de définir au moyen de la notation ASN.1 la forme et le contenu d'un éventail beaucoup plus large de documents XML.

10.1.2 Les règles EXTENDED-XER élargissent les règles BASIC-XER, notamment concernant les trois points principaux suivants:

a) elles offrent des options supplémentaires au codeur (par exemple, concernant l'insertion d'instructions de traitement XML ou de commentaires XML, et concernant l'emploi d'identificateurs pour les bits nommés dans une valeur bitstring);

b) elles définissent un ensemble d'instructions de codage qui peuvent être employées pour spécifier la modification du codage BASIC-XER d'un type ASN.1, y compris une instruction de codage permettant d'utiliser du simple texte au lieu d'étiquettes d'éléments vides pour les valeurs booléennes, entières (avec des nombres nommés), énumérées ou spéciales des types réels et bitstring (avec des bits nommés);

c) elles exigent que les décodeurs ne tiennent pas compte (en l'absence d'instructions de codage) des attributs de l'espace de nom de commande, qui sont inconnus (par exemple, un attribut **schemaLocation**), et de certains attributs connus pouvant être introduits par d'autres outils XML, dont les valeurs peuvent différer de celles qu'un codeur réputé conforme peut insérer (par exemple, l'emploi d'un attribut d'identification de type) (voir le § 10.2.10).

10.1.3 Si une spécification ASN.1 ne contient aucune instruction de codage XER, alors tout codage BASIC-XER d'une valeur abstraite quelconque d'un type ASN.1 est aussi un codage EXTENDED-XER de la même valeur abstraite de ce type.

NOTE – Le contraire n'est pas vrai. Même en l'absence d'instructions de codage XER, certains codages EXTENDED-XER ne sont pas conformes aux codages BASIC-XER (voir les § 10.1.2 a et 10.1.2 c).

10.1.4 Toutes les occurrences de la notation "Type" ASN.1 incluent un ensemble associé (éventuellement vide) d'instructions de codage XER (les instructions associées finales de codage). Les instructions de codage sont associées à un "Type" au moyen:

a) (D'instructions de codage héritées) Présence d'instructions de codage associées, appliquées au "Type" employé dans la définition d'une référence "typereference" utilisée comme un "Type"; et

b) (D'instructions de codage ciblées) Attribution d'une ou de plusieurs instructions de codage XER à une occurrence de "Type" au moyen d'une section de commande de codage XER (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 50); et

NOTE – Un module ASN.1 ne peut contenir qu'une seule section de commande de codage XER, et donc une seule liste "EncodingInstructionAssignmentList" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 50.2).

c) (D'instructions de codage préfixées) Attribution d'une ou de plusieurs instructions de codage XER à une occurrence de "Type" au moyen des préfixes de type XER (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 30.3); et

d) (D'instructions de codage de liste d'importation) Attribution d'une ou de plusieurs instructions de codage à toutes les références aux types importées d'un module ASN.1 identifié.

10.1.5 L'effet de l'attribution d'une instruction de codage XER est d'ajouter, de supprimer ou de remplacer les instructions de codage associées (voir le § 15 pour les règles qui s'appliquent à des attributions multiples d'instructions de codage XER).

10.1.6 L'ordre (ou la manière) selon lequel (ou laquelle) les instructions de codage deviennent partie intégrante (ou sont éliminées) de l'ensemble des instructions de codage associées n'est pas important(e) en ce qui concerne l'application des instructions finales de codage.

10.1.7 Les instructions finales de codage affectent le codage EXTENDED-XER des types. Elles n'ont pas d'autres effets et, en particulier, elles ne sont pas associées à une référence aux valeurs, définie au moyen du type, et elles n'affectent pas les mappages des valeurs, ni les autres règles de codage.

NOTE – Certaines prescriptions existent toutefois, en ce qui concerne la non-ambiguïté des noms qui sont affectés par la présence d'une instruction finale de codage **NAME**, **NAMESPACE**, ou **UNTAGGED**. Ces prescriptions peuvent être interprétées soit comme des restrictions sur la manière dont les types peuvent être employés avec de telles instructions finales de codage, soit comme des restrictions sur l'emploi de ces instructions de codage.

10.2 Conformité avec les règles EXTENDED-XER

10.2.1 Si une spécification ASN.1 attribue des instructions de codage XER conformément aux § 11 à 17 de manière qu'un type ou une composante ASN.1 comporte des instructions finales de codage qui violent les restrictions spécifiées dans les paragraphes suivant le § 18, alors cette spécification ASN.1 n'est pas conforme à la présente Recommandation | Norme internationale, même si (sans les instructions de codage XER) elle est conforme à la Rec. UIT-T X.680 | ISO/CEI 8824-1.

NOTE – Il n'est qu'occasionnellement illicite d'attribuer une instruction de codage à un "Type", puisqu'elle peut être reniée (supprimée de l'ensemble des instructions de codage associées) au moyen d'une autre attribution. Ce sont les instructions finales de codage qui déterminent normalement la conformité de la spécification. Dans certains cas (mais pas dans tous), il n'est pas tenu compte d'une instruction finale de codage qui ne s'applique pas au type auquel elle a été appliquée. Si les parageraphes définissant la syntaxe et l'application des instructions de codage recensent des circonstances dans lesquelles il n'est pas tenu compte d'une instruction de codage lors de l'application des instructions finales de codage, les paragraphes spécifiant les codages n'indiquent normalement pas la présence éventuelle de cette instruction finale de codage.

10.2.2 Un codage EXTENDED-XER réputé conforme d'un type ASN.1 ne comportant pas d'instructions finales de codage sera le codage produit par les règles de codage XML de base (BASIC-XER) spécifiées dans le § 8, avec les options supplémentaires pour le codeur définies dans les § 10.2.5 et 10.2.6.

NOTE – Les décodeurs EXTENDED-XER doivent en vertu du § 10.2.4 accepter et traiter les déclarations d'un type de document XML du Consortium W3C, bien que celles-ci ne soient pas produites par des codeurs réputés conformes et ne fassent pas partie des codages EXTENDED-XER.

10.2.3 Le codage EXTENDED-XER d'un type ASN.1 comportant des instructions finales de codage, ou des composantes (à une profondeur quelconque et après la résolution de toutes les références au type) auxquelles sont associées des instructions de codage, sera le codage spécifié dans le § 17.

NOTE – Les instructions finales de codage sont appliquées dans un codage EXTENDED-XER et sont aussi employées par les décodeurs et les processeurs chargés de la validation des codages EXTENDED-XER.

10.2.4 Les décodeurs EXTENDED-XER (que les codages **MODIFIED-ENCODINGS** aient été employés ou non – voir le § 26) traiteront toute déclaration présente de type de document (voir la version XML 1.0 du Consortium W3C, § 2.8), conformément aux prescriptions pour les processeurs XML non chargés de la validation (voir la version XML 1.0 du Consortium W3C, § 5.1). Ce traitement sera effectué (en principe) avant d'appliquer une quelconque autre prescription de décodage de la présente Recommandation | Norme internationale. Les codeurs EXTENDED-XER n'incluront pas de déclaration de type de document.

10.2.5 Un codeur EXTENDED-XER peut introduire (en vertu des options dont il dispose) des instructions de traitement XML ou des commentaires XML (en plus de ceux qui pourraient être exigés conformément au § 30) dans l'élément de document XML ou dans le prologue XML, partout où la version XML 1.0 du Consortium W3C le permet. La forme syntaxique et la sémantique des instructions de traitement XML sont spécifiées dans la version XML 1.0 du Consortium W3C, au § 2.6. La forme syntaxique et la sémantique des commentaires XML sont spécifiées dans la version XML 1.0 du Consortium W3C, au § 2.5.

10.2.6 Si aucune instruction de codage **GLOBAL-DEFAULTS** n'inclut de mot-clé **MODIFIED-ENCODINGS** (voir le § 26) dans la section de commande de codage XER, alors:

a) la valeur "XMLIntegerValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 18.9) peut être soit le nombre "XMLSignedNumber" soit l'entier "EmptyElementInteger", en fonction du choix du codeur; et

b) la valeur "XMLBitStringValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 21.9) peut être l'une des variantes de cette production, en fonction du choix du codeur. Si la liste "XMLIdentifierList" est employée, il s'agira de la liste "EmptyElementList".

10.2.7 Si une instruction de codage **GLOBAL-DEFAULTS** inclut un mot-clé **MODIFIED-ENCODINGS** (voir le § 26) dans la section de commande de codage XER, alors:

a) la valeur "XMLBooleanValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 17.3) sera le nombre booléen "TextBoolean"; et

b) la valeur "ExtendedXMLIntegerValue" (voir le § 17.4) sera la variante "ModifiedXMLIntegerValue" définie dans le § 17.8; et

NOTE – Cela permet l'emploi d'une valeur de texte pour les nombres "NamedNumber" d'un type entier, au choix pour le codeur, mais cela modifie aussi la syntaxe des codages numériques d'une valeur entière.

c) la valeur "ExtendedXMLEnumeratedValue" (voir le § 34.3) ne sera pas énumérée "EmptyElementEnumerated"; et

NOTE – En l'absence d'une instruction de codage **GLOBAL-DEFAULTS** dans les codages **MODIFIED-ENCODINGS**, elle ne peut être énumérée "TextEnumerated" (voir les § 8.3.4 *bis* et 34.3).

d) la valeur "ExtendedXMLRealValue" (voir le § 17.4) sera la variante "ModifiedXMLRealValue" définie dans le § 17.9; et

e) la valeur "XMLSpecialRealValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 20.6) sera la variante "TextReal"; et

f) la variante de la liste "XMLIdentifierList" dans la valeur "XMLBitStringValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 21.9) sera la liste "TextList" (voir le § 10.2.8 b); et

g) les valeurs "XMLSequenceOfValue" et "XMLSetOfValue" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 25.3 et 27.3) seront les éléments "XMLDelimitedItem" pour tous les types de composantes, sans tenir compte du Tableau 5 (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 25.5); et

h) la chaîne "xmlhstring" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.13) ne contiendra pas de "blancs" (voir le § 8.1.4); et

i) tous les "blancs" qui sont présents en dehors des étiquettes XML ou à l'intérieur des valeurs des attributs XML peuvent être des "blancs avec échappement" (voir le § 8.1.5), en fonction du choix du codeur.

NOTE – Certaines instructions de codage (telles que l'instruction **UNTAGGED**) ne peuvent être utilisées s'il n'y a pas une instruction **GLOBAL-DEFAULTS** dans les codages **MODIFIED-ENCODINGS**.

10.2.8 Si une instruction de codage **GLOBAL-DEFAULTS** avec un mot-clé **MODIFIED-ENCODINGS** (voir le § 26) figure dans la section de commande de codage XER, alors un codeur **EXTENDED-XER** peut (en vertu des options dont il dispose):

a) employer la variante "TextInteger" de la valeur "ModifiedXMLIntegerValue" (voir le § 17.8), à condition qu'il y ait un nombre "NamedNumber" pour la valeur entière dans la définition du type (voir aussi le § 10.2.7 b).

NOTE – L'emploi de ce codage avec des valeurs nommées qui ont été ajoutées dans une version ultérieure peut rendre illisible la valeur abstraite dans l'implémentation d'une version antérieure de la spécification.

b) employer la liste "XMLIdentifierList" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 21.9) pour une valeur "XMLBitStringValue", à condition que la valeur de bitstring à coder ne contienne pas de bits "un" qui ne soient pas des bits nommés (voir aussi le § 10.2.7 f);

NOTE – L'emploi de ce codage avec des bits nommés qui ont été ajoutés dans une version ultérieure peut rendre illisible la valeur abstraite dans l'implémentation d'une version antérieure de la spécification.

10.2.9 Lorsque les options pour le codeur sont autorisées dans un codage EXTENDED-XER, les décodeurs et les processeurs chargés de la validation, réputés conformes, accepteront toutes les options.

10.2.10 Les décodeurs et les processeurs chargés de la validation, réputés conformes, accepteront, mais peuvent négliger, la présence d'un attribut de l'espace de nom de commande dans un quelconque élément XML d'un codage, à moins que sa présence et que son emploi ne soient tels que spécifiés dans les § 37 et 38. Les codeurs ne produiront de tels attributs que dans les cas spécifiés dans les § 37 et 38.

NOTE – Ces attributs peuvent être introduits au moyen d'autres outils XML. En général, un décodeur EXTENDED-XER ne peut facilement déterminer la valeur et la signification permises de certains attributs de l'espace de nom de commande. Plutôt que de ne pas en tenir compte ou de produire une erreur fatale de décodage, leur présence et leur valeur peuvent être utiles à une application si (par exemple) des éléments descendants XML présents (en fonction du choix du décodeur) sont transmis à une application.

10.2.11 Une spécification ASN.1 est illicite jusqu'à ce qu'un décodeur puisse déterminer sans ambiguïté, pour toutes les valeurs abstraites (au moyen seulement du nom de l'étiquette XML et du contenu de tout élément XML précédent), la composante ASN.1 (ou le marqueur d'extension) avec laquelle (ou lequel) l'élément XML est associé.

NOTE 1 – L'association ne peut dépendre du contenu de l'élément XML ou de ses attributs ou de tout élément XML suivant.

NOTE 2 – Cette condition est toujours satisfaite lorsque aucune instruction de codage XER n'est présente, mais l'application inappropriée de l'instruction **UNTAGGED** pour supprimer la répétition (sequence-of ou set-of) en boucle (par exemple) des étiquettes associées et les variantes (choix), ainsi que l'application inappropriée de l'instruction **NAME**, peuvent conduire à des spécifications illicites.

NOTE 3 – Le § 10.2.11 contient une condition nécessaire pour que les codages soient valables, mais il est admis qu'en général il n'est pas possible qu'un outil ASN.1 (ou une personne) vérifie la régularité sur la base de cette déclaration de haut niveau seulement. A l'Annexe B figure un modèle qui décrit l'effet de l'application de l'instruction **UNTAGGED**, et les règles qui, si elles sont appliquées, peuvent assurer la régularité de la spécification, telle que décrite au § 10.2.11.

10.2.12 Si une spécification ASN.1 contient les types "ObjectClassFieldType" ouverts (voir la Rec. UIT-T X.681 | ISO/CEI 8824-2, § 14.2), avec des contraintes de tables ou des contraintes de types, il ne sera pas tenu compte de ces contraintes en appliquant le § 10.2.11.

10.3 Structure d'un codage EXTENDED-XER

10.3.1 Un codage EXTENDED-XER complet consiste en un document XML bien formé qui comportera (dans l'ordre):

a) un prologue XML (qui peut être vide, en fonction du choix du codeur) tel que spécifié dans le § 8.2;

b) un élément de document XML qui est le codage complet d'une valeur d'un type ASN.1 unique, nommé type de base, tel que spécifié dans le § 17.

10.3.2 Les codages de valeurs "XMLValue" employés pour les codages BASIC-XER sont modifiés pour les codages EXTENDED-XER par les instructions finales de codage pour les "Type" qu'ils codent, et par les instructions finales de codage pour leurs composantes (à une profondeur quelconque), en même temps que par toutes les instructions de codage **GLOBAL-DEFAULTS**.

NOTE – Dans un cas extrême, le contenu entier de l'élément de document XML pour une structure ASN.1 fortement imbriquée peut (au moyen de l'application de l'instruction de codage **UNTAGGED**) ne comporter qu'une séquence linéaire d'éléments XML où seul l'élément de base a des éléments descendants. L'application de l'instruction **UNTAGGED** est limitée afin d'assurer que toutes les séquences linéaires résultantes des éléments XML peuvent être mappées sans ambiguïté sur les composantes d'une valeur abstraite d'un type de base ASN.1 (voir le § 10.2.11).

10.3.3 L'élément de document XML dans un codage EXTENDED-XER consiste en un élément XML qui aura une valeur "ExtendedXMLTypedValue" pour le type codé (le type de base). Il peut intégrer des attributs dans son étiquette de début ou dans son étiquette d'élément vide, et peut avoir un contenu qui intègre tant les éléments descendants (voir la version XML du Consortium W3C) que du texte non étiqueté. Les éléments peuvent eux-mêmes avoir aussi bien des attributs qu'un contenu qui intègre les éléments descendants ainsi que du texte non étiqueté.

10.3.4 Les valeurs abstraites des composantes d'un type contenant sont codées sous la forme de valeurs "ExtendedXMLValue" (voir le § 17.4), éventuellement modifiées par des instructions de codage qui leur sont appliquées ou qui le sont à leurs composantes. Ces valeurs "ExtendedXMLValue":

a) peuvent être précédées d'une étiquette de début XML et suivies d'une étiquette de fin XML (nommées les étiquettes associées) pour former un élément au sein de la valeur "ExtendedXMLValue" du type contenant; ou

b) peuvent (à l'aide de l'application d'une instruction de codage **UNTAGGED** à un type qui n'est pas susceptible d'être codé au moyen de caractères) former le contenu partiel d'un élément XML pour la valeur "ExtendedXMLValue" du type contenant; ou

NOTE – A L'Annexe B est décrit le résultat de l'application de l'instruction **UNTAGGED,** qui consiste en la production d'un contenu partiel de l'élément XML, pouvant être combiné avec d'autres codages pour former le contenu de l'élément XML dans le cas de certains éléments contenant dont le type n'a pas été soumis à l'instruction **UNTAGGED**.

c) peuvent (à l'aide de l'application d'une instruction de codage **UNTAGGED** à un type qui est susceptible d'être codé au moyen de caractères) former la valeur complète "ExtendedXMLValue" de la composante; ou

d) peuvent (à l'aide de l'application d'une instruction de codage **ATTRIBUTE** à un type qui est susceptible d'être codé au moyen de caractères) former la valeur "CharacterEncodableValue" dans la valeur "QuotedValue" d'un "Attribute" (voir le § 20.3.3).

10.3.5 Si une valeur "ExtendedXMLValue" est vide, et que ses étiquettes associées n'ont pas été enlevées par l'application d'une instruction de codage **UNTAGGED**, les étiquettes associées précédentes et suivantes peuvent (en fonction du choix du codeur) être remplacées par une étiquette d'élément vide XML (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 16.8). Elle s'appelle étiquette associée d'élément vide.

10.3.6 La transformation spécifiée dans le § 10.3.5 est effectuée en principe après l'achèvement du processus de codage entier et elle peut être évitée par une instruction de codage **PI-OR-COMMENT** (voir le § 30) produisant une ou plusieurs instructions de traitement XML ou d'éléments de commentaire XML entre les étiquettes de début et de fin.

10.3.7 L'étiquette associée précédente, l'étiquette associée suivante et l'étiquette associée d'élément vide sont toutes trois désignées comme étant les étiquettes associées. Les noms des éléments XML dans les étiquettes associées sont nommés noms des étiquettes associées et sont (en l'absence d'instructions finales de codage **NAME** et **NAMESPACE**) des identificateurs, des noms de référence au type ou des noms "xmlasn1typename" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.25).

11 Notation, ensemble de caractères et unités lexicales employés dans les instructions de codage XER

11.1 La notation employée pour définir la syntaxe de l'instruction "EncodingInstruction" dans un préfixe de type XER (voir le § 13) et dans une liste "EncodingInstructionAssignmentList" dans une section de commande XER (voir le § 14) est définie dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 5.

11.2 La Rec. UIT-T X.680 | ISO/CEI 8824-1, § 10, s'applique à une instruction "EncodingInstruction" XER et à une liste "EncodingInstructionAssignmentList" XER.

NOTE – Des caractères blancs ASN.1 arbitraires peuvent en particulier figurer entre des unités lexicales dans les deux structures de syntaxe, à moins que la notation "&" ne soit employée (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 5.4).

11.3 Les règles générales spécifiées dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.1, s'appliquent aussi à une instruction "EncodingInstruction" XER et à une liste "EncodingInstructionAssignmentList" XER.

NOTE – Un commentaire ASN.1 peut en particulier être inséré dès qu'un "blanc" est admis, et les prescriptions relatives à un "blanc" ou à un commentaire entre des unités lexicales qui pourraient sinon prêter à confusion sont spécifiées dans la Rec. UIT-T X.680 | ISO/CEI 8824-1.

11.4 Les unités lexicales suivantes sont employées dans la présente Recommandation | Norme internationale:

comment	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.6)
cstring	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.14)
identifier	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.3)
modulereference	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.5)
number	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.8)
typereference	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.2)
"{"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
"}"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
"."	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
":"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
","	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
";"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
"\""	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
"*"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
"::="	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.16)
"<"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
">"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.26)
"</"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.21)
"/>"	(voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.22)

Des unités lexicales supplémentaires (les nombres "modifiedXMLNumber" et "modifiedXMLRealNumber") sont définies et employées dans les § 17.8.3 et 17.9.

12 Mots-clés

12.1 Les mots spécifiés dans les § 12.3 et 12.4 ci-après sont employés dans l'une ou dans les deux instructions "EncodingInstruction" XER et dans les listes "EncodingInstructionAssignmentList" XER (en plus de certains mots réservés ASN.1), et ils ne peuvent figurer dans de telles structures syntaxiques qu'avec la signification qui leur est attribuée dans les paragraphes suivants de la présente Recommandation | Norme internationale, sauf comme spécifié dans le § 12.2.

12.2 Les mots-clés ne sont pas des mots réservés, mais s'il est besoin dans une instruction "EncodingInstruction" XER ou dans une liste "EncodingInstructionAssignmentList" XER d'une référence "typereference" ASN.1 qui soit la même qu'un mot-clé figurant dans la liste au § 12.3, alors il faut employer la production de la référence "ModuleAndTypeReference" (voir le § 14.2.2).

12.3 Les mots-clés sont les suivants:

```
AFTER-TAG                DEFAULT-FOR-EMPTY        REPLACE
AFTER-VALUE              ELEMENT                  TEXT
ANY-ATTRIBUTES           EMBED-VALUES             UNCAPITALIZED
ANY-ELEMENT              GLOBAL-DEFAULTS          UNTAGGED
AS                       IN                       UPPERCASED
ATTRIBUTE                LIST                     USE-NIL
BASE64                   LOWERCASED               USE-NUMBER
BEFORE-TAG               MODIFIED-ENCODINGS       USE-ORDER
BEFORE-VALUE             NAME                     USE-QNAME
CAPITALIZED              NAMESPACE                USE-TYPE
COLLAPSE                 NOT                      USE-UNION
CONTROL-NAMESPACE        PI-OR-COMMENT            WHITESPACE
DECIMAL                  PREFIX
```

12.4 Des mots-clés supplémentaires sont employés dans la production du nom "BuiltInTypeName" (voir le § 14.2.3), mais ce sont tous des mots réservés ASN.1 (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.27), qui ne peuvent jamais être employés en notation ASN.1 comme référence "typereference".

13 Attribution d'une instruction de codage XER à un type ASN.1 au moyen d'un préfixe de type

13.1 Les instructions finales de codage pour un type peuvent:

a) nécessiter l'emploi de variantes d'une valeur "ExtendedXMLValue", qui ne sont pas des variantes d'une valeur "XMLValue" pour ce type; ou

NOTE – Des variantes de la production d'une valeur "ExtendedXMLValue" comportent aussi bien des variantes de la production d'une valeur "XMLValue" (inchangée), employées dans les règles BASIC-XER, que des variantes de production choisies par les instructions de codage XER.

b) modifier le nom de l'étiquette associée "AttributeName", ou la valeur de l'attribut d'identification du type pour le codage de ce type; ou

c) impliquer qu'une valeur "ExtendedXMLValue" d'une composante d'un type ASN.1 soit insérée en tant que valeur "CharacterEncodableValue" dans la valeur "QuotedValue" d'un "Attribute" (voir le § 20.3.3); ou

d) spécifier le nom de l'espace de nom XML pour les noms de référence et les identificateurs des types définis dans un module ASN.1 et recommander un préfixe d'espace de nom à employer avec cet espace de nom; ou

e) spécifier si un nom restrictif d'espace de nom (au lieu d'un nom non restrictif) est à employer dans un élément XML ou en tant que nom d'un attribut XML; ou

f) spécifier l'enlèvement des étiquettes associées, conduisant généralement à un texte non étiqueté ou à un contenu partiel d'élément XML (qui peuvent être précédés ou suivis d'un autre contenu partiel d'élément XML – voir l'Annexe B); ou

g) spécifier l'insertion d'une ou de plusieurs instructions de traitement XML ou de commentaires XML (voir le § 30):

1) avant l'étiquette associée précédente ou l'étiquette associée d'élément vide; ou

2) entre l'étiquette associée précédente et la valeur "ExtendedXMLValue"; ou

NOTE 1 – Cela interdit l'emploi d'une étiquette associée d'élément vide.

3) entre la valeur "ExtendedXMLValue" et l'étiquette associée suivante; ou

NOTE 2 – Cela interdit l'emploi d'une étiquette associée d'élément vide.

4) après l'étiquette associée suivante.

NOTE 3 – Tout ce qui précède interdit l'application de l'instruction **UNTAGGED** pour enlever les étiquettes associées (voir le § 30.2.2).

13.2 Les instructions de codage XER peuvent être attribuées aux types ASN.1 en employant soit la production d'une instruction "EncodingInstruction" dans un préfixe de type XER soit la production d'une liste "EncodingInstructionAssignmentList" dans une section de commande de codage XER. L'attribution d'un préfixe de type est spécifiée dans le présent paragraphe. L'attribution au moyen d'une section de commande de codage XER est spécifiée dans le § 14.

NOTE – L'effet de l'attribution multiple d'instructions de codage de la même catégorie est spécifié dans le § 15.

13.3 La production de l'instruction "EncodingInstruction" XER est la suivante:

```
EncodingInstruction::=
        PositiveInstruction
        | NegatingInstruction

PositiveInstruction ::=
        AnyAttributeInstruction
        | AnyElementInstruction
        | AttributeInstruction
        | Base64Instruction
        | DecimalInstruction
        | DefaultForEmptyInstruction
        | EmbedValuesInstruction
        | GlobalDefaultsInstruction
        | ListInstruction
        | NameInstruction
        | NamespaceInstruction
        | PIOrCommentInstruction
        | TextInstruction
        | UntaggedInstruction
        | UseNilInstruction
        | UseNumberInstruction
        | UseOrderInstruction
        | UseQNameInstruction
        | UseTypeInstruction
        | UseUnionInstruction
        | WhitespaceInstruction


NegatingInstruction ::=
        NOT PositiveInstruction
        | ElementInstruction
```

13.4 L'instruction "ElementInstruction" (voir le § 24) est strictement synonyme de l'instruction **NOT UNTAGGED**, et son examen n'est pas poursuivi dans le présent paragraphe.

NOTE 1 – Le synonyme **ELEMENT** est donné pour éviter la double négation et pour assurer la lisibilité des spécifications. Il sera normalement employé (contrairement à l'instruction de codage **ATTRIBUTE**) pour identifier la nature des types de haut niveau dans le module ASN.1. Les types de haut niveau qui ne comportent ni des instructions finales de codage **ELEMENT** ni des instructions finales de codage **ATTRIBUTE** prendront en charge des types qui ne correspondent pas directement aux éléments ou aux attributs XML, et seront habituellement soumis à l'instruction **UNTAGGED**.

NOTE 2 – Il n'existe pas d'instruction d'annulation de codage pour l'instruction **ELEMENT**. Une instruction de codage **ELEMENT** peut être annulée par une instruction de codage **UNTAGGED**, mais il n'est pas recommandé de le faire.

13.5 Toute utilisation d'une instruction "PositiveInstruction" dans un préfixe de type XER ou dans une section de commande de codage attribue cette instruction de codage XER au "Type" correspondant.

13.6 Si le "Type" dans une attribution "TypeAssignment" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 15.1) comporte des instructions finales de codage, toutes les utilisations de la référence "typereference" correspondante (dans le module contenant l'attribution "TypeAssignment" ou dans un autre module) acquièrent leurs instructions associées finales de codage par héritage, à l'exception des instructions finales de codage **NAME** et **NAMESPACE**, qui ne sont pas susceptibles d'être héritées.

NOTE – Ces deux instructions de codage affectent le nom XML employé au lieu du nom de référence au type. Lorsque le nom de référence au type est employé pour définir le type dans une attribution de type ou dans une composante, il ne convient pas d'utiliser sa définition pour acquérir par héritage ces instructions finales de codage.

13.7 Une instruction de codage dans un préfixe de type ou dans une section de commande de codage peut être une instruction affirmative, employée pour ajouter ou remplacer une instruction de codage (emploi de l'instruction "PositiveInstruction"), ou une instruction d'annulation, employée pour annuler (emploi de l'instruction "NegatingInstruction") une ou plusieurs instructions de codage associées.

13.8 Les instructions de codage XER comportent quatre parties (dont certaines peuvent être vides):

a) la mention **`NOT,`** indiquant la négation ou la suppression des instructions de codage d'une catégorie donnée; et

NOTE 1 – Cette mention est indiquée pour les instructions négatives (à l'exception de l'instruction "ElementInstruction") mais ne l'est pas pour les instructions affirmatives.

b) un mot-clé identifiant la catégorie de l'instruction de codage; et

NOTE 2 – Ce mot-clé est toujours présent.

c) l'indication d'une liste de cibles pour l'attribution de l'instruction de codage (éventuellement avec des informations restrictives limitant son application à un sous-ensemble de valeurs du type); et

NOTE 3 – Lorsqu'elle est employée dans un préfixe de type, la liste de cibles est toujours la production "vide", puisque la cible de l'attribution est toujours le type associé au préfixe de type (voir le § 13.12). La liste de cibles est aussi toujours "vide" pour l'instruction de codage **`GLOBAL-DEFAULTS`**.

d) la syntaxe, propre à chacune des catégories d'instruction de codage, donnant des détails concernant les instructions de codage dans cette catégorie.

NOTE 4 – Lorsqu'elle est employée dans une instruction négative, la syntaxe est toujours une production "vide". Elle ne figure pas non plus dans certaines instructions de codage XER pour lesquelles le mot-clé est une définition suffisante.

13.9 Certaines instructions de codage XER nécessitent la spécification de la valeur abstraite d'un type. Cette spécification emploie la production de la valeur "Value" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 16.7). Si une référence "valuereference" est employée en tant que valeur "Value", elle sera définie (ou sera importée) dans le module ASN.1 qui contient l'instruction de codage XER.

NOTE – Cela veut dire que la valeur peut être spécifiée soit directement au moyen d'une notation de valeur ASN.1 de base, soit au moyen d'une référence à une valeur, qui a été spécifiée soit au moyen d'une notation de valeur ASN.1 de base, soit au moyen d'une notation de valeur XML.

13.10 Dans le Tableau 1 sont énumérées dans la colonne 1 les variantes des productions de l'instruction "PositiveInstruction". Dans la colonne 2 sont donnés les paragraphes où sont spécifiés les prescriptions relatives à l'application de ces instructions de codage et les codages modifiés qu'elles produisent. Dans la colonne 3 est indiquée la catégorie de l'instruction de codage.

NOTE – Ces catégories ont été introduites afin que soit clairement établi le résultat de l'attribution multiple des instructions de codage de la même catégorie.

Tableau 1 – Instructions de codage, paragraphes où elles sont définies et catégories

Instruction de codage	Paragraphe où elle est définie	Catégorie
AnyAttributesInstruction	§ 18	Instruction "tout-attribut"
AnyElementInstruction	§ 19	Instruction "tout-élément"
AttributeInstruction	§ 20	Instruction "attribut"
Base64Instruction	§ 21	Instruction "base64"
DecimalInstruction	§ 22	Instruction "décimal"
DefaultForEmptyInstruction	§ 23	Instruction "valeur-par-défaut-pour-le-vide"
ElementInstruction	§ 24	Instruction "élément"
EmbedValuesInstruction	§ 25	Instruction "imbrication-des-valeurs"
GlobalDefaultsInstruction	§ 26	Instruction "valeurs-globales-par-défaut" (mais voir le § 15.3)
ListInstruction	§ 27	Instruction "liste"
NameInstruction	§ 28	Instruction "nom" (mais voir le § 15.3)
NamespaceInstruction	§ 29	Instruction "espace de nom"

Tableau 1 – Instructions de codage, paragraphes où elles sont définies et catégories

Instruction de codage	Paragraphe où elle est définie	Catégorie
PiOrCommentInstruction	§ 30	Instruction "pi-(processing instruction, traitement)-ou-commentaire" (mais voir le § 15.3)
TextInstruction	§ 31	Instruction "texte" (mais voir le § 15.3)
UntaggedInstruction	§ 32	Instruction "non étiqueté"
UseNilInstruction	§ 33	Instruction "emploi-de-zéro"
UseNumberInstruction	§ 34	Instruction "emploi-d'un-nombre"
UseOrderInstruction	§ 35	Instruction "emploi-d'un-ordre"
UseQNameInstruction	§ 36	Instruction "emploi-d'un-nom Q"
UseTypeInstruction	§ 37	Instruction "emploi-d'un-type"
UseUnionInstruction	§ 38	Instruction "emploi-de-l'union"
WhitespaceInstruction	§ 39	Instruction "blanc"

13.11 Chacune des variantes de la production de l'instruction "PositiveInstruction" appartient à une catégorie bien définie d'instructions de codage (ou dans certains cas, concerne plusieurs catégories). La catégorie de chacune des instructions de codage est spécifiée dans la colonne 3 du Tableau 1 (voir aussi le § 15.3 pour les instructions de codage qui concernent plusieurs catégories).

NOTE – Les catégories des instructions de codage sont employées au § 15.4 pour déterminer l'effet des attributions multiples des instructions de codage.

13.12 La liste "TargetList" dans toutes les structures d'instruction "EncodingInstruction" qui figurent dans un préfixe de type sera vide ("empty") et la cible sera le "Type" associé au préfixe de type.

13.13 Une instruction négative appartient à la même catégorie que l'instruction affirmative correspondante.

13.14 Un type ASN.1 ne peut jamais se voir associer plus d'une instruction de codage XER dans une catégorie donnée (voir les § 15.3 et 15.4), quelle que soit la manière dont elles ont été attribuées. Le résultat des attributions multiples d'une instruction de codage XER d'une catégorie donnée est spécifié au § 15.

14 Attribution d'une instruction de codage XER au moyen d'une section de commande de codage

14.1 Liste des attributions des instructions de codage

14.1.1 Les instructions de codage XER peuvent être attribuées aux types ASN.1 au moyen soit de la production de l'instruction "EncodingInstruction" dans un préfixe de type XER soit de la production de la liste "EncodingInstructionAssignmentList" dans une section de commande de codage XER. L'attribution au moyen d'un préfixe de type est spécifiée au § 13. L'attribution au moyen d'une section de commande de codage est spécifiée dans le présent paragraphe.

14.1.2 La production de la liste "EncodingInstructionAssignmentList" XER est la suivante:

EncodingInstructionAssignmentList ::=
EncodingInstruction
EncodingInstructionAssignmentList ?

14.1.3 La production de l'instruction "EncodingInstruction" est définie au § 13.3.

14.1.4 Chacune des utilisations d'une instruction "EncodingInstruction" dans une section de commande de codage attribue cette instruction de codage XER aux occurrences du "Type" qui sont identifiées dans la liste "TargetList" de l'instruction de codage, ou aux références dans une liste d'importation. La production de la liste "TargetList" et les cibles qu'elle identifie sont spécifiées au § 14.2.

14.1.5 Les § 13.4 à 13.14 s'appliquent aussi aux instructions de codage dans une section de commande de codage. Les paragraphes définissant la syntaxe détaillée pour chacune des catégories d'instructions de codage sont énumérés dans le Tableau 1. Les catégories des instructions de codage XER sont également mentionnées dans le Tableau 1.

14.2 Identification des cibles d'une instruction de codage XER à l'aide d'une liste de cibles

14.2.1 Règles générales

14.2.1.1 Les variantes de l'instruction "EncodingInstruction" spécifient l'instruction de codage XER qui est attribuée, et la ou les cibles de cette attribution dans le module ASN.1. Toutes les cibles sont une occurrence de la production du "Type" dans le module ASN.1.

NOTE – Des cibles multiples de la même attribution de type ASN.1 ou d'attributions différentes peuvent être spécifiées. Une cible correspondant au module entier, ou toutes les occurrences dans le module d'un type intégré ou d'un constructeur peuvent aussi être spécifiées. Donc (en employant la section de commande de codage), une seule instruction "EncodingInstruction" peut être utilisée pour attribuer une instruction de codage XER particulière à tous les types dans un module ASN.1, auxquels cette instruction de codage doit être associée.

14.2.1.2 Pour identifier la ou les cibles d'attribution d'une instruction de codage XER, on emploie la production de la liste "TargetList". La définition en est donnée dans les paragraphes suivants.

NOTE 1 – Il est renvoyé à la production de la liste "TargetList" dans les paragraphes après le § 18.

NOTE 2 – La production de la liste "TargetList" a une variante "empty". C'est la seule variante permise lorsque l'instruction "EncodingInstruction" est employée dans un préfixe de type (voir le § 13.12). Dans le § 14.2, il n'est examiné que l'emploi de la liste "TargetList" dans une section de commande de codage.

14.2.1.3 La production de la liste "TargetList" est la suivante:

```
TargetList ::=
        Targets "," +
    | empty

Targets ::=
        TypeIdentification
        | BuiltInTypeIdentification
        | IdentifiersInContext
        | ImportedTypesIdentification
```

14.2.1.4 Si dans la liste "TargetList" sont énumérées une ou plusieurs productions de cibles "Targets", chacune d'elles identifie une ou plusieurs cibles (les "Type" auxquels l'instruction de codage est attribuée), mais peut aussi fournir des informations restrictives, limitant son application aux codages employant un identificateur particulier dans la définition du type de cible, ou à l'utilisation des étiquettes d'élément vide pour les caractères de commande spécifiés dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.5.

NOTE – Les informations restrictives ne sont présentes que si la cible définit un type booléen, bitstring, énuméré, entier ou chaîne de caractères limités (voir le § 14.2.2.9).

14.2.1.5 Une liste "TargetList" de valeurs "empty" n'est autorisée que dans un préfixe de type (lorsque c'est la seule variante autorisée) et dans l'instruction de codage **`GLOBAL-DEFAULTS`**. Dans un préfixe de type, elle identifie le type associé au préfixe. Dans l'instruction de codage **`GLOBAL-DEFAULTS,`** elle identifie tous les types "Type" dans le module.

14.2.1.6 L'instruction de codage XER (éventuellement avec les informations restrictives associées) est attribuée à tous les types identifiés dans la liste "TargetList" telle que spécifiée aux § 14.2.1.10 à 14.2.1.16.

NOTE – Il serait inhabituel, mais pas illicite, pour un "Type" donné d'être identifié plus d'une fois dans la liste. Dans ces cas, le § 15 s'applique.

14.2.1.7 L'identification (à titre d'explication) (et les éventuelles informations restrictives) par la production de cibles "Targets" emploie l'une des cinq formes de base suivantes:

a) une référence "typereference" (voir le § 14.2.2), éventuellement suivie d'une liste d'identificateurs séparés par des points, spécifiant:

   1) le "Type" dans une attribution de type (aucun identificateur présent); ou

   2) le "Type" dans une composante d'une définition de type (qui peut comprendre des composantes de haut niveau introduites par la structure **`COMPONENTS OF`** – voir le § 14.2.1.12); ou

   3) l'un des types 1) ou 2), plus un identificateur final (précédé d'une virgule et non d'un point) pour un identificateur employé dans la définition de type de cible, donnant les informations restrictives;

b) l'indication **`ALL`** en tant que dernier identificateur de la forme a), spécifiant tous les "Type" contextuellement présents dans la définition du type (qui est identifié par la référence au type précédent et une liste d'identificateurs séparés par des points), ou des informations restrictives (précédées d'une virgule et non d'un point) spécifiant tous les identificateurs employés pour les valeurs d'une définition de type booléen, bitstring, énuméré ou entier (qui est identifié par la référence au type précédent et une liste d'identificateurs séparés par des points) ou spécifiant toutes les utilisations des étiquettes d'élément vide

XML employées pour représenter certains caractères de commande (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.15.5);

c) un nom "BuiltInTypeName" (voir le § 14.2.3), identifiant tous les "Type" dans le module, qui sont définis à travers l'emploi du nom du type intégré ou du constructeur correspondant, éventuellement (uniquement dans le cas de types **BOOLEAN**, **BIT STRING**, **ENUMERATED, INTEGER** et chaîne de caractères limités) suivi d'informations restrictives;

d) une liste d'identificateurs "identifier" suivis de l'indication **IN** (ou **ALL** suivi de **IN,** ou **COMPONENTS** suivi de **IN)** et la forme a) ci-dessus (voir aussi le § 14.2.4), identifiant:

1) le "Type" des composantes identifiées de la forme a); ou

2) (indication **ALL**) tous les "Type" qui figurent contextuellement dans le "Type" identifié par la forme a); ou

3) (indication **COMPONENTS**) tous les "Type" qui sont les composantes de haut niveau du "Type" identifié par la forme a);

e) l'identification "ImportedTypesIdentification" (voir le § 14.2.5) spécifiant toutes les références "typereference" dans la liste **IMPORTS**, qui sont importées d'un module spécifié.

NOTE 1 – Le terme "type definition" employé aux alinéas a) et b) ci-dessus met l'accent sur le fait que seuls des identificateurs contextuellement présents peuvent être employés. Les identificateurs ne peuvent pas être employés si le type correspond à une référence au "Type".

NOTE 2 – En général, il peut être renvoyé à une composante au moyen des alinéas a) ou d) ci-dessus. S'il doit être renvoyé vers plus d'une composante d'un type, on préférera l'alinéa d) parce qu'il est moins prolixe, sinon on préférera l'alinéa a). C'est une question de style.

14.2.1.8 Un type bitstring ou octetstring avec une restriction relative au contenu, qui porte sur un type, sera considéré comme un type à une seule composante, employant "*****" comme identificateur de la composante, aux fins de l'attribution d'une instruction ciblée au "Type" dans la restriction relative au contenu.

14.2.1.9 Une définition d'un type sequence-of ou set-of indiquera un type à une seule composante, employant "*****" comme identificateur de la composante, aux fins de l'attribution d'une instruction ciblée au "Type" qui est la composante du type sequence-of ou set-of.

NOTE – Il est aussi posssible d'identifier cette unique composante au moyen de l'identificateur de composante (s'il est présent).

14.2.1.10 Si une cible consiste en l'emploi d'un paramètre muet d'un type paramétrisé, la cible acquiert par héritage les instructions finales de codage du paramètre en vigueur avant que des instructions de codage visant le paramètre muet ne soient attribuées. La spécification est licite seulement si les instructions finales de codage résultantes pour toutes les instanciations du type paramétrisé sont licites.

NOTE 1 – Si le type paramétrisé est exporté, les instructions finales de codage pour ses paramètres muets seront acheminées avec lui.

NOTE 2 – Aucun mécanisme n'est fourni pour attribuer des instructions de codage directement au "Type" d'un paramètre en vigueur dans une intanciation d'un type paramétrisé.

14.2.1.11 Si la cible est un type "SelectionType", elle acquiert par héritage les instructions finales de codage des variantes choisies du type choix auquel il est fait référence par le type choix, puis l'attribution des instructions de codage au type "SelectionType" a lieu.

14.2.1.12 Si la cible est une composante produite à la suite de la transformation **COMPONENTS OF**, elle acquiert par héritage les instructions finales de codage de la composante du type auquel il est fait référence par la transformation **COMPONENTS OF,** puis l'attribution des instructions de codage aux composantes produites par la transformation **COMPONENTS OF** a lieu. Il ne sera pas tenu compte des instructions de codage pour le "Type" dont les composantes sont extraites.

14.2.1.13 Si la production de cibles "Targets" est l'identification "TypeIdentification", alors les cibles qu'elle identifie sont celles spécifiées au § 14.2.2.

14.2.1.14 Si la production de cibles "Targets" est l'identification "BuiltInTypeIdentification", alors les cibles qu'elle identifie sont celles spécifiées au § 14.2.3.

14.2.1.15 Si la production de cibles "Targets" est le contexte "IdentifiersInContext", alors les cibles qu'il identifie sont celles spécifiées au § 14.2.4.

14.2.1.16 Si la production de cibles "Targets" est l'identification "ImportedTypesIdentification", alors les cibles qu'elle identifie sont celles spécifiées au § 14.2.5.

14.2.1.17 EXEMPLE: L'exemple ci-après montre une définition de type ASN.1, suivie de l'attribution de deux manières différentes d'instructions de codage XER dans une section de commande de codage, et finalement, la même définition de type ASN.1 et les instructions de codage XER qui sont attribuées au moyen des préfixes de type. Les trois démarches conduisent au même codage EXTENDED-XER.

La définition du type est la suivante:

```
My-Type ::= SEQUENCE {
      field1  INTEGER,
      field2  CHOICE {
                 first   SEQUENCE OF INTEGER,
                 second  SEQUENCE OF OBJECT IDENTIFIER } }
```

Les instructions de codage XER dans la section de commande de codage devraient être les suivantes:

```
      ATTRIBUTE   field1 IN My-Type
      LIST        first IN My-Type.field2
```

Elles pourraient aussi être les suivantes:

```
      ATTRIBUTE   My-Type.field1
      LIST        My-Type.field2.first
```

La définition du type avec les préfixes de type est la suivante:

```
My-Type ::= SEQUENCE {
      field1  [ATTRIBUTE] INTEGER,
      field2  CHOICE {
                 first   [LIST] SEQUENCE OF INTEGER,
                 second  SEQUENCE OF OBJECT IDENTIFIER } }
```

14.2.2 Identification des cibles au moyen d'une référence au type ASN.1 et des identificateurs

14.2.2.1 La production de l'identification "TypeIdentification" est la suivante:

TypeIdentification ::=
ALL
| ModuleAndTypeReference
ComponentReference ?
QualifyingInformationPart ?

ModuleAndTypeReference ::=
typereference
| modulereference "." typereference

ComponentReference ::=
"."
ComponentIdList

ComponentIdList ::=
ComponentId "." +

ComponentId ::=
identifier
| "*"
| ALL

QualifyingInformationPart ::=
":"
QualifyingInformation

QualifyingInformation
identifier
| ALL

14.2.2.2 Une identification "TypeIdentification" de l'indication **ALL** identifie tous les "Type" des attributions "TypeAssignment" dans le module.

14.2.2.3 La production de la référence "ModuleAndTypeReference" identifie le "Type" qui est attribué à la référence "typereference". La référence "modulereference" dans la référence "ModuleAndTypeReference" sera la référence au module contenant la liste "EncodingInstructionAssignmentList", et la référence "typereference" sera une référence au type défini dans le module. Elle ne sera employée que si et seulement si la référence "typereference" contient les mêmes six caractères que l'un des mots-clés spécifiés dans le § 12.3, sinon seule la référence "typereference" sera employée.

14.2.2.4 Un symbole "*****" identifie le "Type" de la (seule) composante d'un type sequence-of ou set-of, ou le type dans une contrainte relative au contenu qui contient un "Type".

NOTE – Cette forme peut être employée même si la composante sequence-of ou set-of inclut un identificateur, mais l'emploi de l'identificateur devrait être préféré.

14.2.2.5 Si l'indication **ALL** est employée comme identification "ComponentId", elle sera la dernière identification "ComponentId" dans la liste "ComponentIdList" et ne sera pas suivie d'une information "QualifyingInformation".

14.2.2.6 Si la première identification "ComponentId" dans la liste "ComponentIdList" (lorsqu'elle est présente) est un identificateur qui est contextuellement présent (ou qui résulte de l'emploi de la structure **COMPONENTS OF**) en tant qu'identificateur de composante dans le "Type" identifié par la référence "ModuleAndTypeReference", alors elle identifie le "Type" de cette composante. Si elle n'est pas un identificateur qui est contextuellement présent (ou qui résulte de l'emploi de la structure **COMPONENTS OF**) en tant qu'identificateur de composante dans le "Type" identifié par la référence "ModuleAndTypeReference", alors cette occurrence de l'identification "TypeIdentification" n'est pas illicite, mais elle n'identifie aucune cible.

NOTE – Cela exige que le type auquel il est fait référence dans la référence "ModuleAndTypeReference" soit une définition du type séquence, ensemble, choix, sequence-of ou set-of, ou du type bitstring ou octetstring avec une contrainte relative au contenu qui contient un "Type".

14.2.2.7 Si une identification suivante "ComponentId" (à l'exception de la dernière) dans la liste "ComponentIdList" (lorsqu'elle est présente) est un identificateur qui est contextuellement présent en tant qu'identificateur de composante dans le "Type" identifié par la précédente identification "ComponentId", alors elle identifie le "Type" de cette composante. Si elle n'est pas un identificateur de composante qui est contextuellement présent dans le "Type" identifié par la précédente identification "ComponentId", alors cette occurrence de l'identification "TypeIdentification" n'est pas illicite, mais elle n'identifie aucune cible.

NOTE – Le premier emploi de l'identification "ComponentId" peut se référer aux composantes introduites par une structure **COMPONENTS OF**. Les composantes de ces composantes ne peuvent être identifiées par des identifications suivantes "ComponentId".

14.2.2.8 Si la dernière identification "ComponentId" dans la liste "ComponentIdList" (lorsqu'elle est présente) est:

a) un identificateur qui est contextuellement présent en tant qu'identificateur de composante dans le "Type" identifié par la précédente identification "ComponentId", alors elle identifie le "Type" de cette composante; l'instruction de codage sera attribuée à ce "Type"; ou

b) le mot-clé **ALL**; l'instruction de codage sera attribuée à tous les "Type" qui sont contextuellement présents dans la définition du type identifié par la précédente identification "ComponentId", qui sera un type avec une ou plusieurs composantes.

14.2.2.9 La partie "QualifyingInformationPart" ne sera pas présente à moins que les références "ModuleAndTypeReference" avec "ComponentReference" (lorsqu'elle est présente) identifie la ou les cibles, à savoir:

a) les types booléens; ou

b) les types bitstring avec des bits nommés; ou

c) les types énumérés; ou

d) les types entiers avec des nombres nommés; ou

e) les types chaînes de caractères limités.

14.2.2.10 La variante "identifier" de l'information "QualifyingInformation" ne sera pas employée à moins que la référence "ModuleAndTypeReference" dans la référence "ComponentReference" (lorsqu'elle est présente) identifie une seule cible qui n'est pas de type chaîne de caractères limités, ou identifie une liste de cibles dont tous les types sont booléens. L'identificateur "identifier" sera un identificateur dans la définition du type de cible si celle-ci n'est pas de type booléen, ou aura la valeur **true** ou **false**. L'identificateur "identifier" donne une information restrictive qui indique que l'instruction de codage ne s'applique qu'aux codages employant cet identificateur.

14.2.2.11 Les variantes **true** et **false** de l'information "QualifyingInformation" pour un type booléen donnent des informations restrictives qui indiquent que l'instruction de codage ne s'applique qu'au codage des valeurs abstraites **TRUE** ou **FALSE**, respectivement.

14.2.2.12 La variante **ALL** de l'information "QualifyingInformation" ne sera pas employée à moins que la cible ne donne (seulement) une définition ou plusieurs définitions pour tous les types énumérés dans le § 14.2.2.9. Elle ne sera pas employée si la cible identifie une ou plusieurs cibles à chaîne de caractères limités, à moins que l'instruction de codage ne soit l'espace **NAMESPACE**. Elle donne des informations restrictives qui indiquent que l'instruction de codage s'applique à tous les identificateurs dans les définitions de type, ou, dans le cas d'un type chaîne de caractères limités, à toutes les utilisations des étiquettes d'élément vide XML, employées pour représenter les caractères de commande énumérés dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.5.

NOTE – Il n'est pas possible d'employer des informations restrictives avec un identificateur "identifier" pour intervenir de manière sélective sur la représentation des caractères de commande. Seule la variante **ALL** est possible dans ce cas.

14.2.3 Identification des cibles au moyen d'un nom de type intégré

14.2.3.1 La production de l'identification "BuiltInTypeIdentification" est la suivante:

BuiltInTypeIdentification ::=
BuiltInTypeName
BuiltInTypeQualifyingInformationPart ?

BuiltInTypeName::=
BIT STRING
| **BOOLEAN**
| **CHARACTER STRING**
| **CHOICE**
| **EMBEDDED PDV**
| **ENUMERATED**
| **EXTERNAL**
| **GeneralizedTime**
| **INSTANCE OF**
| **INTEGER**
| **NULL**
| **ObjectDescriptor**
| **OBJECT IDENTIFIER**
| **OCTET STRING**
| **REAL**
| **RELATIVE-OID**
| **SEQUENCE**
| **SEQUENCE OF**
| **SET**
| **SET OF**
| **UTCTime**
| **RestrictedCharacterStringType**

BuiltInTypeQualifyingInformationPart ::=
":"
BuiltInTypeQualifyingInformation

BuiltInTypeQualifyingInformation
identifier
| **ALL**

14.2.3.2 La production de l'identification "BuiltInTypeIdentification" indique que l'instruction de codage doit être appliquée à toutes les occurrences textuelles dans le module du type intégré correspondant ou d'un type défini à l'aide du constructeur correspondant.

14.2.3.3 Le type "RestrictedCharacterStringType" est défini dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 37.

14.2.3.4 La partie "BuiltInTypeQualifyingInformationPart" ne sera pas présente à moins que le nom ne soit "BuiltInTypeName" de type **BOOLEAN**, **BIT STRING, ENUMERATED**, **INTEGER**, ou chaîne de caractères limités.

NOTE – Seule la forme **ALL** de l'information "BuiltInTypeQualifyingInformation" est autorisée pour un type chaîne de caractères limités (voir le § 14.2.2.10 et le paragraphe suivant).

14.2.3.5 La variante "identifier" de l'information "BuiltInTypeQualifyingInformation" ne sera pas employée à moins que le nom "BuiltInTypeName" ne soit de type **BOOLEAN**, et aura alors la valeur **true** ou **false**. Elle donne des informations restrictives qui indiquent que l'instruction de codage ne s'applique qu'au codage des valeurs abstraites **TRUE** ou **FALSE**, respectivement.

14.2.3.6 La variante **ALL** de l'information "BuiltInQualifyingInformation" donne des informations restrictives qui indiquent que l'instruction de codage s'applique à tous les identificateurs employés dans chacune des utilisations du nom "BuiltInTypeName" dans le module (ou à toutes les valeurs de la définition du type **BOOLEAN**, ou à toutes les étiquettes d'élément vide employées dans les valeurs du type chaîne de caractères limités spécifié – voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.15.5).

14.2.4 Emploi des identificateurs dans le contexte

14.2.4.1 La production du contexte "IdentifiersInContext" est la suivante:

```
IdentifiersInContext ::=
        IdentifierList
        IN
        TypeIdentification

IdentifierList ::=
        identifier "," +
        | ALL
        | COMPONENTS
```

14.2.4.2 L'identification "TypeIdentification" est définie au § 14.2.2, et elle spécifie un type défini dans une déclaration d'attribution de type dans le module ou une composante ou sous-composante d'un type défini dans un module. La partie "QualifyingInformationPart" sera absente.

14.2.4.3 Le "Type" défini par l'identification "TypeIdentification" sera un type séquence, ensemble ou choix, qui est nommé aux fins du présent paragraphe le "Type" identifié.

NOTE – L'identification "TypeIdentification" dans le contexte "IdentifiersInContext" ne peut être employée pour un type sequence-of ou set-of. Son utilisation est interdite par souci de clarté, parce qu'elle n'est pas moins prolixe que l'utilisation directe de l'identification "TypeIdentification" dans les cibles "Targets".

14.2.4.4 Chaque identificateur "identifier" dans la liste "IdentifierList" sera l'identificateur "identifier" d'une composante du "Type" identifié. L'instruction de codage XER est attribuée au "Type" de toutes les composantes du "Type" identifié, qui ont une composante identificateur "identifier" dans la liste "IdentifierList".

14.2.4.5 L'emploi de la variante **ALL** pour la liste "IdentifierList" spécifie que toutes les composantes contextuellement présentes (et toutes les composantes contextuellement présentes de ces composantes, à une profondeur quelconque) dans le type "Type" identifié sont des cibles auxquelles l'instruction de codage XER est attribuée.

14.2.4.6 L'emploi de l'indication **COMPONENTS** pour la liste "IdentifierList" spécifie que toutes les composantes (au premier niveau) du "Type" identifié sont des cibles auxquelles l'instruction de codage XER est attribuée.

14.2.5 Emploi de l'identification des types importés

14.2.5.1 La production de l'identification "ImportedTypesIdentification" est la suivante:

```
ImportedTypesIdentification ::=
        ALL IMPORTS FROM modulereference
```

14.2.5.2 La référence "modulereference" sera l'une de celles qui sont employées dans l'une des références "GlobalModuleReferences" de la clause d'importation du module.

14.2.5.3 L'instruction de codage XER est attribuée à chacune des références "typereference" dans la liste correspondante "SymbolList", après que les instructions finales de codage, produites par l'attribution dans le module exportant, ont été attribuées.

14.2.5.4 Si une référence importée "typereference" est exportée de ce module, les instructions finales de codage héritées par cette référence "typereference" dans un module qui l'importe sont celles qui y ont été héritées, et qui ne sont pas affectées par l'attribution des instructions de codage au moyen de l'identification "ImportedTypesIdentification". Cette attribution n'affecte que l'emploi de la référence au type dans ce module.

15 Attributions multiples des instructions de codage XER

15.1 Ordre d'examen des attributions multiples

15.1.1 Un "Type" qui n'est pas une référence "typereference" comporte initialement des instructions de codage associées qui forment un ensemble vide.

15.1.2 Un "Type" qui est une référence "typereference" (pouvant être importée) comporte initialement l'ensemble d'instructions finales de codage du "Type" qui lui a été attribué lorsqu'il a été défini (éventuellement modifiées par les instructions de codage qui lui ont été attribuées dans la liste d'importation d'un module d'importation – voir le § 14.2.5).

15.1.3 Des instructions de codage ciblées pour un "Type" (au moyen d'une section de commande de codage) sont ensuite attribuées, dans l'ordre d'apparition des instructions de codage ciblées dans la section de commande de codage. Si le "Type" est identifié par plus d'un élément d'une liste "TargetList" (voir le § 14.2), cela sera considéré comme des attributions multiples de la même instruction de codage à ce type "Type", dans l'ordre d'apparition des éléments dans la liste "TargetList".

NOTE – Les § 15.1.2 et 15.1.3 impliquent que l'attribution ciblée à un "Type" dans une attribution "TypeAssignment" est toujours supplantée par une attribution ciblée à un "Type" défini au moyen de la référence "typereference" correspondante, quelle que soit l'attribution ciblée qui apparaît en premier dans la section de commande de codage. Toutefois, si une attribution ciblée est faite à toutes les composantes d'un type, et aussi à une composante distincte de ce type, l'effet dépendra de l'ordre des instructions de codage dans la section de commande de codage.

15.1.4 Des instructions de codage préfixées (au moyen d'un préfixe de type) qui sont attribuées à un type sont ensuite examinées, l'instruction de codage préfixée examinée en premier étant celle qui est située le plus à droite (tout à l'intérieur), tandis que l'instruction de codage préfixée examinée en dernier est celle qui est située le plus à gauche (tout à l'extérieur).

15.1.5 Comme spécifié au § 14.2.1.10, les instructions de codage ne sont attribuées à un paramètre muet qu'après que les instructions finales de codage ont été déterminées pour les paramètres en vigueur.

15.1.6 Comme spécifié aux § 14.2.1.11 et 14.2.1.12, un type "SelectionType" et les composantes produites par une transformation **COMPONENTS OF** acquièrent d'abord par héritage des instructions finales de codage du type original, puis se voient appliquer les instructions de codage les visant.

15.1.7 Chaque attribution d'une instruction de codage produit un nouvel ensemble d'instructions de codage associées comme spécifié dans les § 15.2 à 15.4.

15.2 Effet de l'attribution d'une instruction négative de codage

15.2.1 Toutes les attributions d'une instruction négative de codage conduisent à la suppression (de l'ensemble des instructions de codage associées) de toutes les instructions de codage de la même catégorie. En l'absence d'instructions de codage associées d'une catégorie différente, l'ensemble devient vide.

15.2.2 L'instruction de codage **NOT GLOBAL-DEFAULTS** ne sera jamais attribuée.

15.2.3 Pour les instructions de codage de diverses catégories (voir le § 15.3), une instruction négative de codage supprimera toutes les instructions de codage dans chacune de ces catégories.

NOTE – Une instruction négative de codage ne fait jamais partie de l'ensemble des instructions de codage associées.

15.3 Attribution multiple d'instructions de codage de diverses catégories

15.3.1 Les instructions de codage **NAME** et **TEXT** (voir les § 28 et 31) peuvent être attribuées à un type afin de:

a) modifier le nom de l'étiquette associée (sans information "QualifyingInformation"); ou

NOTE – Cela ne s'applique qu'à l'instruction de codage **NAME**.

b) modifier le codage de la valeur "ExtendedXMLValue" en donnant un nouveau nom à l'identificateur "identifier" spécifié, figurant dans la définition du type (avec une information "QualifyingInformation" qui n'est pas l'indication **ALL**); ou

c) modifier le codage de la valeur "ExtendedXMLValue" en communiquant une modification à appliquer à tous les identificateurs "identifier" présents dans la définition du type (avec une information "QualifyingInformation" qui est l'indication **ALL**, la cible n'étant pas de type chaîne de caractères limités).

15.3.2 Dans le cas mentionné à l'alinéa 15.3.1 b), l'instruction de codage pour un identificateur "identifier" spécifié est considérée comme étant d'une catégorie différente de celle d'une instruction de codage pour tout autre identificateur "identifier", ainsi que de celle d'une instruction de codage correspondant à l'alinéa 15.3.1 a).

15.3.3 Dans le cas mentionné à l'alinéa 15.3.1 c), l'instruction de codage est étendue en un ensemble d'instructions de codage du type correspondant à celui de l'alinéa 15.3.1 b), avec une instruction de codage pour chaque identificateur "identifier" figurant dans la définition du type.

15.3.4 L'instruction de codage **PI-OR-COMMENT** (voir le § 30) comporte quatre catégories, correspondant aux quatre variantes pour la "Position".

15.3.5 Sous réserve des § 15.3.3 à 15.3.4, le § 15.4 spécifie les règles d'attribution multiple des instructions de codage.

15.3.6 Chacune des variantes de l'instruction de codage **GLOBAL-DEFAULTS** correspond à une catégorie distincte, chacune des catégories de cette instruction de codage n'étant toutefois pas attribuée plus d'une fois.

15.4 Attribution multiple d'instructions de codage XER d'une même catégorie

NOTE – L'attribution multiple d'instructions de codage XER de la même catégorie devrait normalement être occasionnelle, sauf lorsqu'une instruction de codage XML est attribuée globalement, et qu'une instruction de codage (éventuellement négative) prépondérante est attribuée à des types ou à des composantes particuliers. Dans le présent paragraphe sont données les règles d'attribution multiple des instructions de codage XER de même catégorie. Le § 15.3.5 y renvoie aussi en ce qui concerne le traitement des attributions multiples des instructions de codage **NAME**, **PI-OR-COMMENT**, et **TEXT**.

15.4.1 Les attributions des instructions affirmatives de codage conduisent à l'adjonction (à l'ensemble des instructions de codage associées) de l'instruction de codage XER lorsqu'il n'existe pas d'autres instructions de codage associées de la même catégorie.

15.4.2 L'attribution d'une instruction de codage **ELEMENT** est toujours équivalente à l'attribution d'une instruction de codage **NOT UNTAGGED**.

15.4.3 S'il existe une instruction de codage de la même catégorie dans l'ensemble des instructions de codage associées, alors l'instruction de codage est supprimée de l'ensemble et l'instruction de codage XER attribuée est ajoutée.

NOTE – Si les instructions de codage sont attribuées globalement dans une section de commande de codage, dans le but de les supplanter dans des cas particuliers, ceci devra se faire en employant dans la section de commande de codage soit un préfixe soit une instruction de codage ultérieure, et non antérieure.

15.4.4 Si un type qui figure dans une contrainte "ContentsConstraint" ou "TypeConstraint" doit être codé au moyen des règles EXTENDED-XER, les instructions finales de codage (telles que déterminées par les règles ci-dessus) sont employées pour déterminer le codage de ce type. Si un type figure dans une autre contrainte ASN.1, alors toutes les instructions de codage associées sont écartées.

15.5 Combinaisons permises des instructions finales de codage

15.5.1 Dans le Tableau 2 sont spécifiées les combinaisons permises des instructions finales de codage pour un "Type" lorsque l'instruction **GLOBAL-DEFAULTS** pour le codage **MODIFIED-ENCODINGS** est employée. Dans la colonne 1 sont énumérées toutes les instructions de codage, tandis que dans la colonne 2 sont énumérées les instructions de codage qui peuvent être utilisées, en tant qu'instructions finales de codage, en combinaison avec l'instruction de codage figurant dans la colonne 1. Dans de nombreux cas, des restrictions s'appliquent. Elles sont énumérées dans les paragraphes applicables.

NOTE – L'instruction **GLOBAL-DEFAULTS** ne figure pas dans le tableau, parce qu'elle n'est pas attribuée à un type.

Tableau 2 – Combinaisons permises des instructions finales de codage avec le codage MODIFIED-ENCODINGS

Instruction de codage	Autres instructions de codage permises
ANY-ATTRIBUTES *(voir § 18)*	**ELEMENT, NAME, NAMESPACE**
ANY-ELEMENT *(voir § 19)*	**ELEMENT, NAME, NAMESPACE**
ATTRIBUTE *(voir § 20)*	**BASE64, DECIMAL, ELEMENT, LIST, NAME, NAMESPACE, TEXT, USE-NUMBER, USE-QNAME, USE-UNION, WHITESPACE**
BASE64 *(voir § 21)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
DECIMAL *(voir § 22)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
DEFAULT-FOR-EMPTY *(voir § 23)*	**BASE64, DECIMAL, ELEMENT, EMBED-VALUES, LIST, NAME, NAMESPACE, PI-OR-COMMENT, TEXT, USE-NIL, USE-NUMBER, USE-ORDER, USE-QNAME, USE-UNION, WHITESPACE**
ELEMENT *(voir § 24)*	*Equivalent à* **NOT UNTAGGED**
EMBED-VALUES *(voir § 25)*	**DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, USE-NIL, USE-ORDER**
LIST *(voir § 27)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
NAME *(voir § 28)*	*Sans restrictions*
NAMESPACE *(voir § 29)*	*Sans restrictions*
PI-OR-COMMENT *(voir § 30)*	**BASE64, DECIMAL, DEFAULT-FOR-EMPTY, ELEMENT, EMBED-VALUES, LIST, NAME, NAMESPACE, TEXT, USE-NIL, USE-NUMBER, USE-ORDER, USE-QNAME, USE-TYPE, USE-UNION, WHITESPACE**

Tableau 2 – Combinaisons permises des instructions finales de codage avec le codage MODIFIED-ENCODINGS

Instruction de codage	Autres instructions de codage permises
TEXT *(voir § 31)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
UNTAGGED *(voir § 32)*	**BASE64, DECIMAL, LIST, NAME, NAMESPACE, TEXT, USE-NUMBER, USE-QNAME, USE-UNION, WHITESPACE**
USE-NIL *(voir § 33)*	**DEFAULT-FOR-EMPTY, ELEMENT, EMBED-VALUES, NAME, NAMESPACE, PI-OR-COMMENT, USE-ORDER**
USE-NUMBER *(voir § 34)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
USE-ORDER *(voir § 35)*	**DEFAULT-FOR-EMPTY, ELEMENT, EMBED-VALUES, NAME, NAMESPACE, PI-OR-COMMENT, USE-NIL.**
USE-QNAME *(voir § 36)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
USE-TYPE (*voir § 37)*	**ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT**
USE-UNION *(voir § 38*)	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**
WHITESPACE *(voir § 39)*	**ATTRIBUTE, DEFAULT-FOR-EMPTY, ELEMENT, NAME, NAMESPACE, PI-OR-COMMENT, UNTAGGED**

15.5.2 Dans le Tableau 3 sont spécifiées les combinaisons permises des instructions finales de codage lorsque l'instruction **GLOBAL-DEFAULTS** pour le codage **MODIFIED-ENCODINGS** n'est pas employée. Dans la colonne 1 sont énumérées toutes les instructions de codage qui sont permises, en tant qu'instructions finales de codage, lorsque l'instruction **GLOBAL-DEFAULTS** pour le codage **MODIFIED-ENCODINGS** n'est pas employée, tandis que dans la colonne 2 est indiqué "*Non permis*" ou sont énumérées toutes les instructions de codage qui peuvent être utilisées, en tant qu'instructions finales de codage, en combinaison avec l'instruction de codage figurant dans la colonne 1. Dans de nombreux cas, des restrictions s'appliquent. Elles sont énumérées dans les paragraphes applicables. L'indication "*Non permis*" veut dire que l'instruction de codage ne peut être utilisée comme instruction finale de codage lorsque l'instruction **GLOBAL-DEFAULTS** pour le codage **MODIFIED-ENCODINGS** n'est pas employée.

NOTE – L'instruction **GLOBAL-DEFAULTS** ne figure pas dans le tableau, parce qu'elle n'est pas attribuée à un type.

Tableau 3 – Combinaisons permises des instructions finales de codage sans le codage MODIFIED-ENCODINGS

Instruction de codage	Autres instructions de codage permises
ANY-ATTRIBUTES	*Non permis*
ANY-ELEMENT	*Non permis*
ATTRIBUTE	**BASE64, LIST, NAME, TEXT, USE-NUMBER, WHITESPACE**
BASE64	**ATTRIBUTE, NAME, PI-OR-COMMENT**
DECIMAL	*Non permis*
DEFAULT-FOR-EMPTY	*Non permis*
ELEMENT	*Non permis*
EMBED-VALUES	*Non permis*
LIST	**ATTRIBUTE, NAME, PI-OR-COMMENT**
NAME	**ATTRIBUTE, BASE64, LIST, PI-OR-COMMENT, TEXT, USE-NUMBER, WHITESPACE**
NAMESPACE	*Non permis*
PI-OR-COMMENT	**BASE64, LIST, NAME, TEXT, USE-NUMBER, WHITESPACE**
TEXT	**ATTRIBUTE, NAME, PI-OR-COMMENT**
UNTAGGED	*Non permis*
USE-NIL	*Non permis*
USE-NUMBER	**ATTRIBUTE, NAME, PI-OR-COMMENT**

Tableau 3 – Combinaisons permises des instructions finales de codage sans le codage MODIFIED-ENCODINGS

Instruction de codage	Autres instructions de codage permises
USE-ORDER	*Non permis*
USE-QNAME	*Non permis*
USE-TYPE	*Non permis*
USE-UNION	*Non permis*
WHITESPACE	**ATTRIBUTE, NAME, PI-OR-COMMENT**

16 Prise en charge par l'instruction de codage XER des espaces de nom XML et des noms restrictifs

16.1 Les espaces de nom XML du Consortium W3C définissent des concepts et des règles régissant les qualificatifs et les mécanismes nécessaires pour qu'un nom d'élément ou un nom d'attribut XML puisse correctement être identifié avec une spécification correspondante de la sémantique associée.

16.2 Les espaces de nom XML du Consortium W3C définissent un espace de nom XML comme un ensemble de noms non ambigus, identifiés par un identificateur uniforme de ressource (URI, *uniform resource identifier*), qui sont employés dans des documents XML comme types d'élément et noms d'attribut. L'identificateur URI qui identifie un espace de nom est appelé nom de l'espace de nom. Dans la présente Recommandation | Norme internationale, les espaces de nom sont aussi employés pour préciser les valeurs d'un type qui inclut une instruction finale de codage pour l'instruction **USE-QNAME** (voir le § 36) et qui représente un nom QName XML (voir le schéma XML du Consortium W3C, partie 2, § 3.2.18).

16.3 Les identificateurs et les noms de référence au type peuvent (mais pas impérativement) se voir attribuer un espace de nom.

NOTE – Dans la présente Recommandation | Norme internationale est employé un nom d'espace de nom qui, par défaut, est une forme d'un identificateur URI, fondée sur des identificateurs d'objets ASN.1 (voir le § 29). Toutes les autres formes des identificateurs URI peuvent être employées pour attribuer un nom d'espace de nom aux noms dans un module ASN.1.

16.4 La question de savoir si un type fait partie d'un espace de nom ou non (et si oui, son nom d'espace de nom) est fonction de la présence (ou de l'absence) d'une instruction finale de codage **NAMESPACE**.

NOTE – Une instruction de codage **NAMESPACE** ne peut être présente que si une instruction de **GLOBAL-DEFAULTS** pour le codage **MODIFIED-ENCODINGS** figure aussi dans le paragraphe de commande de codage (voir le § 29.2.1).

16.5 Un espace de nom est identifié par la production de la spécification "NameSpaceSpecification" qui indique l'identificateur URI pour l'espace de nom, et éventuellement un préfixe d'espace de nom recommandé. La production de la spécification "NameSpaceSpecification" est définie au § 29.

16.6 Les noms d'élément XML et d'attribut dans un codage EXTENDED-XER proviennent de plusieurs sources. Dans le § 16.8 sont énumérées les sources des noms d'élément XML et d'attribut, est identifié l'espace de nom auquel ils appartiennent, et est spécifié s'ils doivent être des noms restrictifs d'espace de nom ou non.

16.7 Un nom d'élément XML, un nom d'attribut XML ou une valeur d'un attribut d'identification du type peuvent (mais pas impérativement) inclure une instruction finale de codage **NAMESPACE** s'appliquant au "Type" qui produit le nom. Si c'est le cas, le nom sera un nom restrictif d'espace de nom dans le codage. (La restriction en ce qui concerne l'espace de nom dans un codage peut se faire soit explicitement en employant un préfixe d'espace de nom XML défini, soit indirectement en établissant un espace de nom XML par défaut pour une application qui prévoit l'utilisation du nom ou de la valeur.) Si aucune instruction de codage **NAMESPACE** ne s'applique au "Type" qui produit un nom, celui-ci n'est pas un nom restrictif d'espace de nom. Les noms qui ne sont pas des noms restrictifs d'espace de nom, nommés noms non restrictifs, ne doivent pas figurer lors de l'application d'un espace de nom XML établi par défaut.

NOTE – Les règles BASIC-XER ne prennent pas en charge les espaces de nom XML, et les noms restrictifs d'espace de nom ne figurent jamais dans les codages BASIC-XER.

16.8 Dans les paragraphes suivants, le terme "espace de nom ASN.1" se réfère à l'espace de nom dont le nom et le préfixe sont spécifiés au § 16.9. Le terme "espace de nom attribué" se réfère à l'espace de nom attribué par l'instruction de codage **NAMESPACE** à un type. Si les noms produits ne proviennent pas de l'espace de nom ASN.1, et qu'aucune attribution de nom d'espace de nom n'est faite, alors les noms d'élément XML, les noms d'attribut et les valeurs des attributs d'identification du type sont des noms non restrictifs.

16.8.1 Dans tous les paragraphes du présent § 16.8, les noms d'élément et les noms d'attribut dans les étiquettes XML (étiquettes d'élément vide XML ou étiquettes de début) sont des noms restrictifs d'espace de nom dans un codage, si et seulement si le "Type" chargé de la production inclut une instruction finale de codage **NAMESPACE**.

16.8.2 Les noms d'élément dans les étiquettes d'élément vide XML employées pour les caractères de commande (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.15.5) n'ont pas d'espace de nom à moins qu'il ne leur en est attribué à la suite de l'application de l'instruction de codage **NAMESPACE** au type chaîne de caractères limités, avec l'information restrictive **ALL**.

16.8.3 Les noms d'élément dans les étiquettes d'élément vide XML employées pour les valeurs des types entier, énuméré, bitstring et les valeurs spéciales des types réels (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 18.9, 19.8, 20.6 et 21.9) seront toujours des noms non restrictifs (voir le § 16.7) dans un codage de ces types.

16.9 L'espace de nom de l'attribut d'identification du type (voir le § 37) et de l'attribut d'identification de valeur nulle (voir le § 33) est l'espace de nom de commande, qui est par défaut l'espace de nom ASN.1, à moins qu'un espace de nom de commande différent ne soit spécifié par une instruction de codage **GLOBAL-DEFAULTS** (voir le § 26). L'espace de nom ASN.1 a le nom **"urn:oid:2.1.5.2.0.1"** (voir le § 40.3), et un préfixe d'espace de nom recommandé **"asn1"** (voir aussi le § 26.3.2).

16.10 Pour un type octetstring soumis à une contrainte relative au contenu, qui spécifie un codage EXTENDED-XER, toute valeur abstraite du type octetstring sera donnée par le codage EXTENDED-XER complet d'une valeur de type ASN.1 (voir la Rec. UIT-T X.682 | ISO/CEI 8824-3, § 11.5 et 11.6), et contiendra toutes les déclarations d'espace de nom nécessaires pour tous les noms restrictifs préfixés et non préfixés figurant dans la valeur abstraite de la chaîne octetstring.

NOTE – Ce type octetstring est codé comme une chaîne "xmlhstring" ou une valeur "Base64OctetStringValue". Les déclarations d'espace de nom figurant dans le document XML qui contient la chaîne "xmlhstring" ou la valeur "Base64OctetStringValue" n'incluent pas dans leur application les noms figurant dans la chaîne octetstring.

16.11 Lorsqu'un type ouvert est codé comme une chaîne "xmlhstring" ou une valeur "Base64XMLOpenTypeFieldVal", et que les règles de codage employées pour le type contenu sont les règles EXTENDED-XER, la chaîne "xmlhstring" ou la valeur "Base64XMLOpenTypeFieldVal" s'exprimeront comme la représentation hexadécimale ou base64 (respectivement) d'une chaîne d'octets, qui est le codage EXTENDED-XER complet d'une valeur du type contenu, et contiendra toutes les déclarations d'espace de nom nécessaires à l'ensemble des noms restrictifs préfixés et non préfixés qui y sont présents.

NOTE – Toutes les déclarations figurant dans le document XML qui contient la chaîne "xmlhstring" ou la valeur "Base64XMLOpenTypeFieldVal" n'incluent pas dans leur application les noms figurant dans la chaîne d'octets.

17 Spécification des codages EXTENDED-XER

La spécification des codages EXTENDED-XER emploie les productions définies dans les paragraphes suivants. Ces productions admettent l'ensemble de la syntaxe des productions correspondantes employées par le codage BASIC-XER (de même nom mais sans l'indication "Extended"), mais fournissent une syntaxe supplémentaire qui est permise dans les codages EXTENDED-XER. L'emploi de cette syntaxe supplémentaire est déterminé par l'application des instructions de codage XER et est spécifié dans les § 18 à 39.

NOTE – Les variantes des productions disponibles sont fréquemment limitées par l'emploi ou le non-emploi d'une instruction de codage **GLOBAL-DEFAULTS** avec le mot-clé **MODIFIED-ENCODINGS** (voir les § 10.2.7 et 10.2.8). Celle-ci commande en particulier l'emploi des codages d'élément vide ou de texte pour certains types intégrés.

17.1 Elément de document en langage XML

17.1.1 L'élément de document XML aura la valeur "ExtendedXMLTypedValue".

17.1.2 La valeur "ExtendedXMLTypedValue" est la suivante:

ExtendedXMLTypedValue ::=
"<" & TypeNameOrModifiedTypeName AttributeList ">"
ExtendedXMLValue
"</" & TypeNameOrModifiedTypeName ">"
| "<" & TypeNameOrModifiedTypeName "/>"

NOTE – Les différences avec la production de la valeur "XMLTypedValue" sont l'inclusion d'une liste "AttributeList", éventuellement vide, et l'emploi d'une valeur "ExtendedXMLValue" au lieu d'une valeur "XMLValue" dans le contenu de l'élément XML.

17.1.3 Le nom "TypeNameOrModifiedTypeName" est défini dans le § 17.2.

17.1.4 La liste "AttributeList" est définie dans le § 17.3.

17.1.5 La valeur "ExtendedXMLValue" est définie dans le § 17.4 et sera la valeur "ExtendedXMLValue" du type identifié par le nom "TypeNameOrModifiedTypeName".

17.1.6 La deuxième variante de la valeur "XMLTypedValue" (emploi d'une étiquette d'élément vide XML) ne peut être utilisée que si une occurrence de la production de la valeur "ExtendedXMLValue" est vide.

NOTE – Si la production de la valeur "ExtendedXMLValue" était une chaîne "xmlcstring" ne contenant que des "blancs", elle ne serait pas vide, et la deuxième variante ne pourrait pas être utilisée.

17.2 Production du nom "TypeNameOrModifiedTypeName"

17.2.1 Le nom "TypeNameOrModifiedTypeName" est le suivant:

TypeNameOrModifiedTypeName ::=
NonParameterizedTypeName
| QualifiedOrUnqualifiedName

17.2.2 Le nom "NonParameterizedTypeName" est défini dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 13.2, et est employé (comme spécifié dans ce paragraphe et dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 13.4 à 13.7) en tant que nom d'élément XML qui identifie un type ASN.1.

17.2.3 Le nom "QualifiedOrUnqualifiedName" est spécifié dans le § 29.3.2. La variante du nom "QualifiedOrUnqualifiedName" sera utilisée si et seulement si une instruction finale de codage **NAME** ou **NAMESPACE** est appliquée au type (voir le § 28), sinon le nom "NonParameterizedTypeName" sera employé.

17.3 Production de la liste "AttributeList"

17.3.1 La liste "AttributeList" est la suivante:

AttributeList ::=
Attribute AttributeList
| empty

17.3.2 L'"Attribute" est défini dans le § 20.3.3.

17.3.3 La liste "AttributeList" sera vide jusqu'à ce que l'application des instructions finales de codage exigent son emploi (voir les § 20, 33 et 37).

17.3.4 Les "Attribute" dans la liste "AttributeList" seront précédés de "blancs" (voir le § 8.1.4).

17.4 Production de la valeur "ExtendedXMLValue"

17.4.1 La valeur "ExtendedXMLValue" est la suivante:

ExtendedXMLValue ::=
ExtendedXMLBuiltinValue
| ExtendedXMLObjectClassFieldValue
| empty

ExtendedXMLBuiltinValue ::=
XMLBitStringValue
| XMLBooleanValue
| ExtendedXMLCharacterStringValue
| ExtendedXMLChoiceValue
| XMLEmbeddedPDVValue
| ExtendedXMLEnumeratedValue
| XMLExternalValue
| XMLInstanceOfValue
| ExtendedXMLIntegerValue
| XMLNullValue
| XMLObjectIdentifierValue
| ExtendedXMLOctetStringValue
| ExtendedXMLRealValue
| XMLRelativeOIDValue
| ExtendedXMLSequenceValue
| ExtendedXMLSequenceOfValue
| ExtendedXMLSetValue
| ExtendedXMLSetOfValue

```
        | ExtendedXMLPrefixedValue

    ExtendedXMLCharacterStringValue ::=
        ExtendedXMLRestrictedCharacterStringValue
        | XMLUnrestrictedCharacterStringValue

    ExtendedXMLRestrictedCharacterStringValue ::=
        XMLRestrictedCharacterStringValue
        | Base64XMLRestrictedCharacterStringValue

    ExtendedXMLObjectClassFieldValue ::=
        ExtendedXMLOpenTypeFieldVal
        | XMLFixedTypeFieldVal

    ExtendedXMLOpenTypeFieldVal ::=
        ExtendedXMLTypedValue
        | Base64XMLOpenTypeFieldVal
        | xmlhstring

    ExtendedXMLOctetStringValue ::=
        ExtendedXMLTypedValue
        | Base64XMLOctetStringValue
        | xmlhstring

    ExtendedXMLRealValue ::=
        XMLRealValue
        | ModifiedXMLRealValue

    ExtendedXMLIntegerValue ::=
        XMLIntegerValue
        | ModifiedXMLIntegerValue

    ExtendedXMLPrefixedValue ::=
        ExtendedXMLValue
```

17.4.2 Les variantes de la valeur "ExtendedXMLBuiltinValue" dont les noms de production ne commencent pas par "Extended", et dont l'emploi pour coder des valeurs abstraites est entièrement spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1 (voir les § 16.10 et 16.2 de la présente Recommandation | Norme internationale) et (pour la valeur "XMLFixedTypeFieldVal" et la troisième variante de la valeur "ExtendedXMLOpenTypeFieldVal") dans la Rec. UIT-T X.681 | ISO/CEI 8824-2, au § 14.6.

17.4.3 La valeur "Base64XMLRestrictedCharacterStringValue" est définie dans le § 21.3.5 et ne sera employée que comme spécifié dans celui-ci.

17.4.4 La valeur "ExtendedXMLChoiceValue" est définie dans le § 17.5 et ne sera employée que comme spécifié dans celui-ci.

17.4.5 La valeur "ExtendedXMLEnumeratedValue" est définie dans le § 34.3 et ne sera employée que comme spécifié dans celui-ci.

17.4.6 Les valeurs "ExtendedXMLSequenceValue" et "ExtendedXMLSetValue" sont définies dans le § 17.6 et ne seront employées que comme spécifié dans celui-ci.

17.4.7 Les valeurs "ExtendedXMLSequenceOfValue" et "ExtendedXMLSetOfValue" sont définies dans le § 17.7 et ne seront employées que comme spécifié dans celui-ci.

17.4.8 Les valeurs "Base64XMLOctetStringValue" et "Base64XMLOpenTypeFieldVal" sont définies dans les § 21.3.2 et 21.3.4 et ne seront employées que comme spécifié dans ceux-ci.

17.4.9 La valeur "ModifiedXMLIntegerValue" est définie dans le § 17.8 et ne sera employée que comme spécifié dans celui-ci.

17.4.10 La valeur "ModifiedXMLRealValue" est définie dans le § 17.9 et ne sera employée que comme spécifié dans celui-ci.

17.4.11 La variante vide "empty" de la valeur "ExtendedXMLValue" ne sera emloyée que comme spécifié dans le § 23.

NOTE – Les autres variantes de la valeur "ExtendedXMLValue" peuvent aussi produire une unité lexicale vide "empty". Ce paragraphe n'affecte pas l'emploi de telles occurrences.

17.5 Production de la valeur "ExtendedXMLChoiceValue"

17.5.1 La valeur "ExtendedXMLChoiceValue" est la suivante:

ExtendedXMLChoiceValue ::=
"<" & TagName AttributeList ">"
ExtendedXMLValue
"</" & TagName ">"
| ExtendedXMLValue

TagName ::=
IdentifierOrModifiedIdentifier

IdentifierOrModifiedIdentifier ::=
identifier
| QualifiedOrUnqualifiedName

17.5.2 Le nom "QualifiedOrUnqualifiedName" est défini dans le § 29.3.2. Il sera utilisé si et seulement si une instruction finale de codage **NAME** (voir le § 28) ou **NAMESPACE** est appliquée au type (voir le § 29), sinon l'identificateur "identifier" sera employé.

NOTE – Si l'identificateur "identifier" est employé, le codage ne peut inclure une déclaration d'espace de nom XML par défaut dans l'application qui prévoit l'emploi de cet identificateur "identifier" (voir le § 16.7).

17.5.3 La liste "AttributeList" et son utilisation sont définies dans le § 17.3 et dans ceux auxquels elle renvoie.

17.5.4 La valeur "ExtendedXMLValue" dans les deux variantes des valeurs "ExtendedXMLChoiceValue" sera la valeur "ExtendedXMLValue" de la variante choisie du type choix.

17.5.5 La deuxième variante de la valeur "ExtendedXMLChoiceValue" sera employée dans l'un des deux cas suivants:

a) la variante choisie du type choix inclut une instruction finale de codage **UNTAGGED** (voir le § 32); ou

b) le type choix inclut une instruction finale de codage **USE-TYPE** ou **USE-UNION** (voir les § 37 et 38).

NOTE – Cela veut dire que la présence de ces instructions finales de codage conduit à l'omission des étiquettes XML, qui permettent de faire un choix, et que le choix doit se faire d'une autre façon (voir les § 37 et 38, ainsi que l'Annexe B).

17.6 Production des valeurs "ExtendedXMLSequenceValue" et "ExtendedXMLSetValue"

17.6.1 Les valeurs "ExtendedXMLSequenceValue" et "ExtendedXMLSetValue" sont les suivantes:

ExtendedXMLSequenceValue ::=
ExtendedXMLComponentValueList
| empty

ExtendedXMLSetValue ::=
ExtendedXMLComponentValueList
| empty

ExtendedXMLComponentValueList ::=
ExtendedXMLNamedValue
| ExtendedXMLComponentValueList ExtendedXMLNamedValue

ExtendedXMLNamedValue ::=
"<" & TagName AttributeList ">"
ExtendedXMLValue
"</" & TagName ">"
| ExtendedXMLValue

17.6.2 Les variantes vides "empty" des valeurs "ExtendedXMLSequenceValue" et "ExtendedXMLSetValue" ne sont employées que si aucune composante du type séquence ou ensemble (à une profondeur quelconque) ne produit une valeur "ExtendedXMLNamedValue", après résolution de toutes les références aux types et application de toutes les instructions finales de codage.

NOTE – Cela inclut (sans s'y limiter) les cas où toutes les composantes sont marquées comme **DEFAULT** ou **OPTIONAL** et toutes les valeurs sont omises; où elles incluent une instruction finale de codage **UNTAGGED** dont les valeurs codées sont nulles; où elles incluent une instruction finale de codage **ATTRIBUTE**. Cela porte aussi sur les combinaisons de ce qui précède, et sur le cas dans lequel la notation du type est **SEQUENCE {}** ou **SET {}**.

17.6.3 Le nom "TagName" est défini dans le § 17.5.1. Le nom "QualifiedOrUnqualifiedName" dans la forme "IdentifierOrModifiedIdentifier" du nom "TagName" doit être employé si et seulement si une instruction finale de codage **NAME** ou **NAMESPACE** est appliquée au type (voir le § 29), sinon l'identificateur "identifier" sera employé.

17.6.4 La liste "AttributeList" et son emploi sont définis dans le § 17.3 et dans ceux auxquels elle renvoie.

17.6.5 La valeur "ExtendedXMLValue" dans les deux variantes de la valeur "ExtendedXMLNamedValue" sera la valeur "ExtendedXMLValue" de la composante du type séquence ou ensemble.

17.6.6 La deuxième variante des valeurs "ExtendedXMLSequenceValue" et "ExtendedXMLSetValue" sera employée si et seulement si la variante inclut une instruction finale de codage **UNTAGGED** (voir le § 32).

17.7 Production des valeurs "ExtendedXMLSequenceOfValue" et "ExtendedXMLSetOfValue"

17.7.1 Les valeurs "ExtendedXMLSequenceOfValue" et "ExtendedXMLSetOfValue" sont les suivantes:

```
ExtendedXMLSequenceOfValue ::=
        ExtendedXMLValueList
        | ExtendedXMLDelimitedItemList
        | empty
        | ExtendedXMLListValue

ExtendedXMLSetOfValue ::=
        ExtendedXMLValueList
        | ExtendedXMLDelimitedItemList
        | empty
        | ExtendedXMLListValue

ExtendedXMLValueList ::=
        ExtendedXMLValueOrEmpty
        | ExtendedXMLValueOrEmpty ExtendedXMLValueList

ExtendedXMLValueOrEmpty ::=
        ExtendedXMLValue
        | "<" & TypeNameOrModifiedTypeName "/>"

ExtendedXMLDelimitedItemList ::=
        ExtendedXMLDelimitedItem
        | ExtendedXMLDelimitedItem ExtendedXMLDelimitedItemList

ExtendedXMLDelimitedItem ::=
        "<" & TypeNameOrModifiedTypeName AttributeList ">"
        ExtendedXMLValue
        "</" & TypeNameOrModifiedTypeName ">"
        | "<" & IdentifierOrModifiedIdentifier AttributeList ">"
        ExtendedXMLValue
        "</" & IdentifierOrModifiedIdentifier ">"
        | ExtendedXMLValue
```

17.7.2 L'emploi des variantes des valeurs "ExtendedXMLSequenceOfValue", "ExtendedXMLSetOfValue" et de la liste "ExtendedXMLValueList" sera conforme à l'emploi des variantes correspondantes des valeurs "XMLSequenceOfValue", "XMLSetOfValue" et de la liste "XMLValueList" (respectivement) comme spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, dans les § 25 et 27, sauf que, si une instruction de codage **GLOBAL-DEFAULTS** avec un mot-clé **MODIFIED-ENCODINGS** est présente, la liste "ExtendedXMLValueList" ne sera jamais employée (voir aussi le § 10.2.7 g).

17.7.3 La valeur "ExtendedXMLListValue" est définie dans le § 27.3.2. Ces variantes de valeurs "ExtendedXMLSequenceOfValue" et "ExtendedXMLSetOfValue" ne seront utilisées que si et seulement si une instruction finale de codage **LIST** (voir le § 27) est appliquée au type sequence-of ou set-of.

17.7.4 La première variante de l'unité "ExtendedXMLDelimitedItem" sera employée si et seulement si le type sequence-of ou set-of ne contient pas d'identificateur "identifier" et que la composante n'inclut pas d'instruction finale de codage **UNTAGGED**. Les paragraphes suivants s'appliquent.

17.7.4.1 Si la composante du type sequence-of ou set-of est une référence "typereference" ou "ExternalTypeReference" (éventuellement avec un ou plusieurs préfixes "TypePrefix"), alors le nom "TypeNameOrModifiedTypeName" sera la référence "typereference" ou la référence "typereference" dans la référence

"ExternalTypeReference", respectivement, éventuellement modifiée conformément aux instructions fianales de codage **NAME** et **NAMESPACE** appliquées à la composante (voir le § 28).

17.7.4.2 Si la composante du type sequence-of ou set-of (après avoir négligé toute occurrence du "TypePrefix") n'est pas une référence "typereference" ou "ExternalTypeReference", alors le nom "TypeNameOrModifiedTypeName" sera le nom "xmlasn1typename" spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, dans le Tableau 4, correspondant au type intégré de la composante, éventuellement modifié conformément aux instructions finales de codage **NAMESPACE** appliquées à la composante (voir le § 29).

17.7.5 La deuxième variante de l'unité "ExtendedXMLDelimitedItem" sera employée si et seulement si le type sequence-of ou set-of contient un identificateur "identifier" et que la composante n'inclut pas d'instruction finale de codage **UNTAGGED**. L'identificateur "IdentifierOrModifiedIdentifier" sera cet identificateur "identifier", éventuellement modifié conformément aux instructions finales de codage **NAME** et **NAMESPACE** appliquées à la composante (voir les § 28 et 29).

17.7.6 La troisième variante de l'unité "ExtendedXMLDelimitedItem" sera employée si et seulement si la composante du type sequence-of ou set-of inclut une instruction finale de codage **UNTAGGED** (voir le § 32).

17.7.7 La valeur "ExtendedXMLValue" dans toutes les variantes de l'unité "ExtendedXMLDelimitedItem" sera la valeur "ExtendedXMLValue" de la composante répétée du type sequence-of ou set-of.

17.7.8 Le nom "TypeNameOrModifiedTypeName" dans la valeur "ExtendedXMLValueOrEmpty" sera le nom "xmlasn1typename" spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, dans le Tableau 4, correspondant au type intégré de la composante, éventuellement modifié conformément aux instructions finales de codage **NAMESPACE** appliquées à la composante (voir le § 29).

17.8 Production de la valeur "ModifiedXMLIntegerValue"

17.8.1 La valeur "ModifiedXMLIntegerValue" est la suivante:

ModifiedXMLIntegerValue ::=
ModifiedXMLSignedNumber
| TextInteger

ModifiedXMLSignedNumber ::=
modifiedXMLNumber
| "-" & modifiedXMLNumber
| "+" & modifiedXMLNumber

17.8.2 Cette variante de la valeur "ExtendedXMLIntegerValue" (voir le § 17.4) ne sera employée que si une instruction de codage **GLOBAL-DEFAULTS** avec un mot-clé **MODIFIED-ENCODINGS** est attribuée.

17.8.3 L'unité lexicale "modifiedXMLNumber" comportera plus d'un chiffre.

NOTE 1 – L'unité lexicale "modifiedXMLnumber" est mappée sur une valeur entière en l'interprétant comme une notation décimale.

NOTE 2 – Cette unité lexicale diffère de l'unité lexicale "number" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.8), uniquement parce qu'elle admet un nombre quelconque de chiffres de poids le plus fort "0".

17.8.4 Toute valeur entière positive peut être codée en employant la première ou la troisième variante du nombre "ModifiedXMLSignedNumber", en fonction du choix du codeur. Une valeur entière négative sera codée en employant la deuxième variante. La valeur entière zéro peut être codée en employant indiffèremment l'une des trois variantes, en fonction du choix du codeur.

17.8.5 L'entier "TextInteger", défini dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, dans le § 18.9, fournit une variante de codage (en fonction du choix du codeur) pour les valeurs entières qui possèdent une définition "NamedNumber".

17.9 Production de la valeur "ModifiedXMLRealValue"

17.9.1 La valeur "ModifiedXMLRealValue" est la suivante:

ModifiedXMLRealValue ::=
ModifiedXMLNumericRealValue
| XMLSpecialRealValue
| XMLDecimalMinusZeroRealValue

ModifiedXMLNumericRealValue ::=
modifiedXMLRealNumber
| "-" & modifiedXMLRealNumber

| "+" & modifiedXMLRealNumber

17.9.2 Cette variante de la valeur "ExtendedXMLRealValue" (voir le § 17.4) ne sera employée que si une instruction de codage **GLOBAL-DEFAULTS** avec un mot-clé **MODIFIED-ENCODINGS** est attribuée.

17.9.3 L'unité lexicale "modifiedXMLRealNumber" comportera une partie entière qui est une suite de un ou de plusieurs chiffres, avec éventuellement un point décimal (.). Ce point décimal peut éventuellement être suivi d'une partie fractionnaire comportant un ou plusieurs chiffres. La partie entière, le point décimal ou la partie fractionnaire (la partie retenue étant la dernière présente) peut éventuellement être suivie de la lettre e ou E et d'un exposant éventuellement avec signe comportant un ou plusieurs chiffres.

NOTE – Cette unité lexicale diffère de l'unité lexicale "realnumber" (voir la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 11.9) uniquement parce qu'elle admet un nombre quelconque de zéros de poids le plus fort dans l'exposant.

17.9.4 Toute valeur réelle positive et plus zéro peut être codée en employant la première ou la troisième variante de la valeur "ModifiedXMLNumericRealValue", en fonction du choix du codeur. Toute valeur négative sera codée au moyen de la deuxième variante de la valeur "ModifiedXMLNumericRealValue". La valeur réelle moins zéro sera codée au moyen de la deuxième variante.

17.9.5 La valeur "XMLDecimalMinusZeroRealValue", définie dans le § 22.3.2, ne sera employée que comme spécifié dans ce paragraphe.

NOTE – L'instruction de codage **DECIMAL,** définie dans le § 22.3.2, fournit cette production en tant que variante de la représentation pour la valeur réelle abstraite plus zéro, mais elle exige que la valeur abstraite moins zéro soit exclue du type auquel elle s'applique.

18 Instruction de codage ANY-ATTRIBUTES

18.1 Généralités

18.1.1 L'instruction "AnyAttributesInstruction" est la suivante:

AnyAttributesInstruction ::=
ANY-ATTRIBUTES
TargetList
NamespaceRestriction ?

NamespaceRestriction ::=
FROM URIList
| EXCEPT URIList

URIList ::=
QuotedURIorAbsent
| URIList QuotedURIorAbsent

QuotedURIorAbsent ::=
QuotedURI
| ABSENT

18.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

18.1.3 L'identificateur "QuotedURI" est défini dans le § 29.1.1.

18.1.4 Cette instruction de codage est attribuée à un type ASN.1 sequence-of ou set-of avec une composante **UTF8String** dont la valeur indique zéro, un ou plusieurs noms d'attribut et des valeurs (une par chaîne **UTF8String**), chacune d'elles étant soumises aux restrictions "NamespaceRestriction" présentes (voir le § 18.2).

NOTE – Bien que le type sequence-of puisse être employé pour la spécification, cet emploi n'implique pas que l'ordre soit important du point de vue de la sémantique, ni que le processus de codage/décodage puisse conduire à un ordre différent des composantes du type sequence-of.

18.1.5 Le contenu de chaque chaîne **UTF8String** est codé comme un "Attribute" de l'élément XML contenant. Le nom de la composante du type sequence-of ou set-of est négligé.

18.1.6 Les indications **FROM** et **EXCEPT** (si elles sont présentes) identifient les noms d'espace de nom, ou le mot-clé spécial **ABSENT**.

18.1.7 L'indication **FROM** oblige les noms d'attribut à être des noms restrictifs des noms d'espace de nom de l'un des espaces de nom spécifiés. Si l'indication **ABSENT** figure dans la liste "URIList", des noms non restrictifs peuvent aussi être employés.

18.1.8 L'indication **EXCEPT** admet les noms restrictifs d'espace de nom de tous les espaces de nom à l'exception de ceux qui figurent dans la liste. Elle admet aussi des noms non restrictifs à moins que l'indication **ABSENT** ne figure dans la liste "URIList".

18.2 Restrictions

18.2.1 Un type ASN.1 n'inclura pas cette instruction finale de codage à moins qu'il ne soit un type set-of ou sequence-of avec une composante qui est un type **UTF8String**.

18.2.2 Un type comportant cette instruction finale de codage ne sera employé que comme une composante d'un type séquence ou ensemble contenant, et la composante ne sera pas marquée comme étant **OPTIONAL** ou **DEFAULT**. Seule une telle composante figurera dans le type contenant.

18.2.3 Un type sequence-of ou set-of comportant cette instruction finale de codage doit être soumis à une contrainte qui lui impose le format et le contenu, spécifiés dans les § 18.2.6 à 18.2.11, de chaque occurrence de la chaîne **UTF8String**, en référence au présent paragraphe ou d'une façon générale.

NOTE – Il est recommandé que la contrainte sur le type sequence-of ou set-of s'exprime comme suit:

```
(CONSTRAINED BY
  {/* Chaque UTF8String doit être conforme au "AnyAttributeFormat" spécifié dans la
      Rec. UIT-T X.693 | ISO/CEI 8825-4, § 18. */})
```

18.2.4 Aucune instruction finale de codage **UNTAGGED** ne s'appliquera au type qui inclut cette instruction finale de codage ou au type contenant.

18.2.5 Chacune des listes "URIList" contiendra au plus une occurrence de l'indication **ABSENT** et ne contiendra pas deux identificateurs identiques "QuotedURI".

18.2.6 Le format de chacune des chaînes **UTF8String** sera conforme à la production du format "AnyAttributeFormat":

AnyAttributeFormat ::=
URI ?
NCName & "=" & xmlcstring

18.2.7 Voir le § 29.1.4 en ce qui concerne la définition de la production de l'identificateur "URI" et le § 29.1.7 en ce qui concerne la définition de la production du nom "NCName". L'unité lexicale "xmlcstring" est définie dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.

18.2.8 S'il existe une restriction "NamespaceRestriction" en ce qui concerne l'indication **FROM**, alors l'identificateur "URI" dans un format "AnyAttributeFormat" sera l'identificateur "URI" dans un identificateur "QuotedURI" dans la liste "URIList", et il peut être absent uniquement si le mot-clé **ABSENT** figure dans la liste "URIList".

18.2.9 S'il existe une restriction "NamespaceRestriction" en ce qui concerne l'indication **EXCEPT**, alors l'identificateur "URI" dans un format "AnyAttributeFormat" ne sera pas l'identificateur "URI" dans un identificateur "QuotedURI" dans la liste "URIList", et il ne sera pas absent si le mot-clé **ABSENT** figure dans la liste "URIList".

18.2.10 La chaîne "xmlcstring" sera une valeur d'attribut XML correcte du point de vue de la syntaxe (définie dans la version XML du Consortium W3C, au § 3), précédée et suivie par un unique caractère APOSTROPHE (39) ou QUOTATION MARK (34).

18.2.11 L'application de cette instruction de codage et de l'instruction de codage **ATTRIBUTE** aux différentes composantes du type contenant ne violera pas le § 20.3.11.

18.2.12 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULT MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

18.2.13 Un type comportant cette instruction finale de codage n'inclura aucune des instructions finales de codage **LIST**, **PI-OR-COMMENT** ou **UNTAGGED**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce leur application au type est interdite: **ANY-ELEMENT**, **ATTRIBUTE**, **BASE64**, **DECIMAL, DEFAULT-FOR-EMPTY**, **EMBED-VALUES**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

18.2.14 Aucune information restrictive ne figurera dans la liste "TargetList".

18.3 Incidence sur les codages

18.3.1 Si le type est codé comme un type de haut niveau, il ne sera pas tenu compte de cette instruction de codage.

18.3.2 La valeur "ExtendedXMLNamedValue" pour cette composante ne figurera pas dans les valeurs "ExtendedXMLSequenceValue" ou "ExtendedXMLSetValue" du type séquence ou ensemble contenant. Au lieu de cela, la valeur du type contenant sera codée au moyen des chaînes **UTF8String** comme un "Attribute" (voir le § 20) de l'élément contenant, conformément à ce qui est spécifié ci-après.

18.3.3 Le codeur:

a) traitera l'identificateur "URI" qui figure dans une chaîne **UTF8String** comme exigeant que le nom suivant "NCName" (le nom de l'attribut) soit un nom restrictif d'espace de nom, l'espace de nom étant spécifié par l'identificateur "URI", et traitera l'absence d'un identificateur "URI" dans une chaîne **UTF8String** comme spécifiant que le nom suivant "NCName" ne soit pas un nom restrictif d'espace de nom, puis supprimera l'identificateur "URI" de la chaîne **UTF8String**; et

b) insérera dans le codage toute déclaration d'espace de nom nécessaire dans l'application comportant les attributs insérés, afin que la restriction d'espace de nom exigée dans les noms "NCName"s identifiés à l'alinéa a) ci-dessus puisse être appliquée; et

c) insérera dans l'élément contenant chacune des chaînes **UTF8String** (après suppression de l'identificateur "URI") en tant qu'attribut, en introduisant des préfixes d'espace de nom selon les besoins avant chacun de noms "NCName" afin que les prescriptions à l'alinéa a) ci-dessus puissent être appliquées.

18.3.4 L'ordre de tous les attributs dans l'élément contenant (qui découle de la présence d'une ou de plusieurs composantes du type contenant avec une instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES**) est laissé au choix du codeur.

18.3.5 Un décodeur EXTENDED-XER produira une chaîne **UTF8String** dans le format du § 18.2.6 pour chaque attribut dans l'élément contenant, qui ne provient pas de l'espace de nom de commande, et dont le nom n'est pas celui de l'identificateur (éventuellement modifié conformément aux instructions finales de codage **NAME** ou **NAMESPACE**) d'une autre composante du type contenant qui inclut une instruction finale de codage **ATTRIBUTE**.

19 Instruction de codage ANY-ELEMENT

19.1 Généralités

19.1.1 L'instruction "AnyElementInstruction" est la suivante:

AnyElementInstruction ::=
ANY-ELEMENT
TargetList
NamespaceRestriction ?

19.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

19.1.3 La restriction "NamespaceRestriction" est définie dans le § 18.1.

19.1.4 Cette instruction de codage permet au type ASN.1 qui est une chaîne **UTF8String** de fournir la spécification d'un seul élément XML.

NOTE – Le contenu et les attributs de l'élément XML ne sont pas limités. Celui-ci peut avoir des attributs ou des éléments descendants, dont les noms peuvent être restrictifs ou non, sans être affectés par une restriction "NamespaceRestriction".

19.1.5 Si une restriction "NamespaceRestriction" est présente, le nom de l'élément est exigé pour satisfaire à cette restriction (voir les § 18.1.6 à 18.1.8), mais il est par ailleurs non restrictif.

19.1.6 La chaîne **UTF8String** avec cette instruction finale de codage peut être le type de base du codage, ou peut être une composante du type choix, séquence, ensemble, sequence-of ou set-of. S'il s'agit d'un type de haut niveau, il n'est pas tenu compte du nom de référence au type. S'il s'agit d'une composante, il n'est pas tenu compte de son nom.

19.2 Restrictions

19.2.1 Un type ASN.1 n'inclura pas cette instruction finale de codage à moins qu'il ne soit un type **UTF8String**. La composante doit être soumise à une contrainte qui lui impose le format et le contenu spécifiés dans les § 19.2.4 à 19.2.9, en référence au présent paragraphe ou d'une façon générale.

NOTE – Il est recommandé que la contrainte sur le type **UTF8String** s'exprime comme suit:

```
(CONSTRAINED BY
  {/* Doit être conforme au "AnyElementFormat" spécifié dans la
      Rec. UIT-T X.693 | ISO/CEI 8825-4, § 19. */})
```

19.2.2 Aucune instruction finale de codage **UNTAGGED** ne s'appliquera au type.

19.2.3 Chacune des listes "URIList" contiendra au plus une occurrence de l'indication **ABSENT** et ne contiendra pas deux identificateurs identiques "QuotedURI".

19.2.4 Le format de chacune des valeurs abstraites de la chaîne **UTF8String** sera conforme à la production du format "AnyElementFormat":

AnyElementFormat ::=
xmlcstring

19.2.5 La chaîne "xmlcstring" sera un élément XML correct du point de vue de la syntaxe, tel que défini dans la version XML 1.0 et dans les espaces de nom XML du Consortium W3C.

19.2.6 Elle n'emploiera que les préfixes d'espace de nom qui figurent dans des déclarations d'espace de nom présentes dans la chaîne "xmlcstring". Si des noms restrictifs sans préfixe sont présents, une déclaration d'espace de nom par défaut correspondante doit être présente aussi.

19.2.7 La valeur de la chaîne **UTF8String** n'entraînera pas la violation du § 10.2.11.

19.2.8 S'il existe une restriction "NamespaceRestriction" en ce qui concerne **FROM**, alors le nom d'élément (tout à l'extérieur) dans un format "AnyElementFormat" sera l'identificateur "URI" dans un identificateur "QuotedURI" dans la liste "URIList", et il peut être absent uniquement si le mot-clé **ABSENT** figure dans la liste "URIList".

19.2.9 S'il existe une restriction "NamespaceRestriction" en ce qui concerne **EXCEPT**, alors l'élément de nom (tout à l'extérieur) dans un format "AnyElementFormat" ne sera pas l'identificateur "URI" dans un identificateur "QuotedURI" dans la liste "URIList", et il ne sera pas absent si le mot-clé **ABSENT** figure dans la liste "URIList".

19.2.10 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

19.2.11 Un type comportant cette instruction finale de codage n'inclura aucune des instructions finales de codage **ATTRIBUTE**, **BASE64**, **DEFAULT-FOR-EMPTY**, **PI-OR-COMMENT**, **UNTAGGED** ou **WHITESPACE**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**.

19.2.12 Aucune information restrictive ne figurera dans la liste "TargetList".

19.3 Incidence sur les codages

19.3.1 Un codeur EXTENDED-XER inclura dans le codage la valeur abstraite de la chaîne **UTF8String** en tant qu'élément XML qui serait produit sinon pour cette composante (en ne connaissant pas l'identificateur de cette composante), ou pour le type de base. Cet élément inséré sera identique à la valeur abstraite de la chaîne **UTF8String,** sauf comme spécifié dans le § 19.3.2.

19.3.2 Toute déclaration d'espace de nom qui figure dans la première étiquette de début (ou dans une étiquette d'élément vide) de l'élément et qui est identique aux déclarations d'espace de nom qui sont d'application au point d'insertion peut (mais pas impérativement) être supprimée, en fonction du choix du codeur.

NOTE – La modification, le déplacement ou la suppression d'autres déclarations de nom d'espace de nom dans la chaîne **UTF8String** ne sont pas admis, parce que de telles actions peuvent affecter l'espace de nom et la restriction des noms QNames XML dans le contenu d'un élément ou dans des valeurs d'attribut, et qu'un codeur ne peut généralement pas déterminer si un tel contenu ou si de telles valeurs d'attribut correspondent à des noms QNames ou non.

19.3.3 Un décodeur EXTENDED-XER produira le format du § 19.2.4 à partir du document XML entrant, en tant que valeur abstraite de la chaîne **UTF8String**.

19.3.4 Le décodeur inclura, dans la première étiquette de début (ou dans une étiquette d'élément vide) dans la valeur abstraite de la chaîne **UTF8String,** des attributs de déclaration d'espace de nom pour toutes les déclarations d'espace de nom qui sont d'application pour l'élément codé mais qui ne figurent pas dans l'étiquette de début de cet élément.

20 Instruction de codage ATTRIBUTE

20.1 Généralités

20.1.1 L'instruction "AttributeInstruction" est la suivante:

AttributeInstruction ::=
ATTRIBUTE
TargetList

20.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

20.1.3 Cette instruction de codage spécifie qu'un type ASN.1 susceptible d'être codé au moyen de caractères doit être codé comme un attribut XML.

NOTE – Un cas particulier (mais important) d'un type susceptible d'être codé au moyen de caractères est le type choix (dont toutes les variantes sont des types susceptibles d'être codés au moyen de caractères) qui inclut une instruction finale de codage **USE-UNION**.

20.2 Restrictions

20.2.1 Un type ASN.1 n'inclura pas cette instruction finale de codage à moins qu'il n'inclue au moins un codage de la valeur "ExtendedXMLValue" (tenant compte des options du codeur), pour chacune de ces valeurs abstraites, qui ne contienne aucune étiquette XML et ne repose pas, pour réaliser ceci, sur l'emploi de la chaîne "xmlhstring" (si le type est un type ouvert ou un type octetstring) ou de la chaîne "xmlbstring" (si le type est un type bitstring) ou sur une instruction finale de codage **UNTAGGED**, **ATTRIBUTE**, ou **ANY-ATTRIBUTES,** appliquée à ses composantes (si le type est un type séquence ou un type ensemble).

NOTE 1 – Cela implique qu'un type chaîne de caractères limités comportant une instruction finale de codage **ATTRIBUTE** doit être limité de manière à ne contenir aucun des caractères de commande énumérés dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au Tableau 3 (séquences d'échappement pour les caractères de commande dans une chaîne "xmlcstring"), ou doit inclure une instruction finale de codage **BASE64**.

NOTE 2 – Cela n'inclut pas les types ouverts, les types octetstring et bitstring comportant l'indication **CONTAINING** sans la mention **ENCODED BY**, parce que leurs valeurs "ExtendedXMLValue" peuvent contenir des étiquettes à moins qu'ils ne soient codés comme une chaîne "xmlhstring".

NOTE 3 – Il est admis que certains outils ASN.1 peuvent ne pas être en mesure de vérifier statiquement que la restriction susmentionnée est satisfaite pour toutes les valeurs abstraites, mais les codeurs réputés conformes ne peuvent produire des codages dans lesquels la valeur "ExtendedXMLValue" viole cette restriction (voir le § 20.3.14).

20.2.2 Un type comportant cette instruction finale de codage ne sera employé que comme une composante d'un type séquence ou d'un type ensemble.

NOTE – La composante peut avoir la caractéristique **OPTIONAL** ou **DEFAULT**.

20.2.3 Aucune instruction finale de codage **UNTAGGED** ne s'appliquera au type qui inclut cette instruction finale de codage ou au type contenant qui la contient en tant que composante.

20.2.4 Si les instructions finales de codage qui s'appliquent aux autres composantes du type contenant incluent soit cette instruction de codage soit l'instruction de codage **ANY-ATTRIBUTES**, le § 20.3.11 ne sera pas violé.

20.2.5 Un type comportant cette instruction finale de codage n'inclura aucune des instructions finales de codage **ANY-ELEMENT**, **DEFAULT-FOR-EMPTY**, **PI-OR-COMMENT** ou **UNTAGGED**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **EMBED-VALUES**, **USE-NIL**, **USE-ORDER**, **USE-TYPE**.

20.2.6 Aucune information restrictive ne figurera dans la liste "TargetList".

20.3 Incidence sur les codages

20.3.1 Si le type codé est un type de haut niveau, il ne sera pas tenu compte de cette instruction de codage.

20.3.2 La valeur "ExtendedXMLNamedValue" de cette composante ne figurera pas dans les valeurs "ExtendedXMLSequenceValue" ou "ExtendedXMLSetValue" du type séquence ou ensemble contenant. Au lieu de cela, la valeur de la composante (si elle est présente) sera codée comme un "Attribute" (voir les § 20.3.3 à 20.3.15) de l'élément contenant.

20.3.3 La production de l'"Attribute" est la suivante:

Attribute::=
AttributeName
"="
QuotedValue

AttributeName::=
IdentifierOrModifiedIdentifier
| ControlAttributeName

QuotedValue::=
DoubleQuotedValue
| SingleQuotedValue

DoubleQuotedValue ::=
""" & CharacterEncodableValue & """

SingleQuotedValue ::=
"'" & CharacterEncodableValue & "'"

ControlAttributeName ::= QualifiedName

CharacterEncodableValue ::= ExtendedXMLValue

20.3.4 La production de l'identificateur "IdentifierOrModifiedIdentifier" est définie dans le § 17.5.1, et son emploi dans le contexte de cette instruction de codage est défini dans le § 17.6.3.

20.3.5 La production du nom "ControlAttributeName" n'est pas directement employée dans le présent paragraphe. Tous les noms "QualifiedName" dans cette production proviennent de l'espace de nom de commande (voir le § 16.9). Ces attributs ne sont généralement produits qu'en conformité avec les § 33 et 37, mais les décodeurs doivent impérativement accepter des attributs de commande non prévus (voir le § 10.2.10).

20.3.6 Le nom "QualifiedName" est défini dans le § 29.3.2.

20.3.7 La valeur "ExtendedXMLValue" est définie dans le § 17.4.

20.3.8 Le nom "AttributeName" sera soit l'identificateur "identifier" de la composante qui inclut cette instruction finale de codage, soit, si des instructions finales de codage **`NAME`** ou **`NAMESPACE`** sont présentes, le nom "QualifiedOrUnqualifiedName" déterminé par ces instructions de codage comme spécifié dans les § 28 et 29.

20.3.9 La valeur "CharacterEncodableValue" dans la valeur "QuotedValue" de l'attribut (voir le § 20.3.3) sera la valeur "ExtendedXMLValue" de ce type, éventuellement modifiée comme spécifié dans les § 20.3.12 à 20.3.15.

20.3.10 L'ordre d'apparition des "Attribute" dans une liste "AttributeList" est laissé au choix du décodeur, que ceux-ci soient produits par cette instruction de codage ou par l'instruction de codage **`ANY-ATTRIBUTES`**.

NOTE – Aucune sémantique ne peut être appliquée à l'ordre des attributs dans les codages EXTENDED-XER. Cette restriction est exigée par la version XML 1.0 du Consortium W3C, § 3.1.

20.3.11 Lorsqu'une liste "AttributeList" dans une instance de codage contient de multiples attributs, alors pour deux quelconques "Attribute" dans la liste:

a) si les noms "AttributeName" des deux attributs sont tous les deux des noms non restrictifs, ils seront différents;

b) si les noms "AttributeName" des deux attributs sont tous les deux des noms restrictifs d'espace de nom, alors ils auront soit des noms d'espace de nom différents, soit des noms différents dans le même espace de nom.

Il est illicite d'employer cette instruction de codage si cette condition est violée par l'application des instructions finales de codage pour une quelconque valeur abstraite du type de haut niveau qui est codé.

20.3.12 Si la valeur "QuotedValue" est une valeur "DoubleQuotedValue", et que la valeur "ExtendedXMLValue" dans la valeur "CharacterEncodableValue" contient un caractère QUOTATION MARK (34), ce caractère sera remplacé par les caractères suivants:

`&quot;`

ou en fonction du choix du codeur, par une séquence d'échappement de la forme **`&#n;`** ou **`&#xn;`**, spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.8.

20.3.13 Si la valeur "QuotedValue" est une valeur "SingleQuotedValue" et que la valeur "ExtendedXMLValue" dans la valeur "CharacterEncodableValue" contient un caractère APOSTROPHE (39), alors ce caractère sera remplacé par les caractères suivants:

'

ou en fonction du choix du codeur, par une séquence d'échappement **&#n;** ou **&#xn;**, spécifiée dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.8.

20.3.14 La valeur "ExtendedXMLValue" dans la valeur "CharacterEncodableValue" sera l'un des codages du type susceptible d'être codé au moyen de caractères, qui ne contient pas d'étiquettes XML.

20.3.15 Si la valeur "ExtendedXMLValue" contient les caractères HORIZONTAL TABULATION (9), LINE FEED (10), ou CARRIAGE RETURN (13), alors ces caractères seront remplacés dans la valeur "ExtendedXMLValue" par les séquences d'échappement de la forme "**&#n;**" ou "**&#xn;**" spécifiées dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.8.

21 Instruction de codage BASE64

21.1 Généralités

21.1.1 L'instruction "Base64Instruction" est la suivante:

Base64Instruction ::=
BASE64
TargetList

21.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

21.1.3 Cette instruction de codage peut être attribuée au type **OCTET STRING**, à un type ouvert ou à un quelconque type chaîne de caractères limités.

21.1.4 L'application de cette instruction finale de codage à un type octet string ou à un type ouvert supprime l'option d'un codage hexadécimal, mais autorise l'option d'un codage Base64 (comme spécifié dans la norme IETF RFC 2045, § 6.8). L'application de cette instruction finale de codage à un type chaîne de caractères limités exige que la valeur de ce type soit codée comme un codage Base64.

21.2 Restrictions

21.2.1 Si les instructions finales de codage d'un type ASN.1 contiennent une instruction de codage **BASE64** alors le type sera:

a) un type **OCTET STRING**; ou

b) un type ouvert; ou

c) un type chaîne de caractères limités.

21.2.2 Un type dans cette instruction finale de codage ne contiendra aucune des instructions finales de codage suivantes **ANY-ELEMENT** ou **WHITESPACE**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**.

21.2.3 Aucune information restrictive ne figurera dans la liste "TargetList".

21.3 Incidence sur les codages

21.3.1 Cette instruction de codage n'affecte que la valeur "ExtendedXMLValue" du type auquel elle s'applique. Elle nécessite l'emploi de la première ou de la deuxième variante des valeurs "ExtendedXMLOctetStringValue" et "ExtendedXMLOpenTypeFieldVal" (en fonction du choix du codeur), qui interdit la troisième variante (voir le § 17.4). Elle nécessite aussi l'emploi de la valeur "ExtendedXMLRestrictedCharacterStringValue" (voir le § 17.4).

21.3.2 La valeur "Base64XMLOctetStringValue" est la suivante:

Base64XMLOctetStringValue ::=
XMLBase64String

La chaîne "XMLBase64String" est définie dans le § 21.3.6.

21.3.3 La Rec. UIT-T X.680 | ISO/CEI 8824-1, § 22.4, s'applique.

21.3.4 La valeur "Base64XMLOpenTypeFieldVal" est la suivante:

Base64XMLOpenTypeFieldVal ::=
XMLBase64String

21.3.5 La valeur "Base64XMLRestrictedCharacterStringValue" est la suivante:

Base64XMLRestrictedCharacterStringValue ::=
XMLBase64String

21.3.6 La chaîne "XMLBase64String" est la suivante:

XMLBase64String ::=
XMLRestrictedCharacterStringValue

La valeur "XMLRestrictedCharacterStringValue" correspondra au codage Content-Transfer-Encoding défini dans la norme IETF RFC 2045, au § 6.8, sauf que la limite relative aux 76 caractères ne s'applique pas, et que les "blancs avec échappement" (voir le § 8.1.5) sont autorisés en toute position de la chaîne "XMLBase64String".

NOTE – La norme IETF RFC 2045 exige la présence de sauts de ligne subdivisant le codage en lignes d'au plus 76 caractères, mais ceci n'est pas prescrit dans les codages EXTENDED-XER. Elle admet aussi que des "blancs" soient insérés en toute position dans le codage base64.

21.3.7 Chaque caractère, dans le cas du type chaîne de caractères limités, dans la chaîne de caractères sera codé dans le format UTF-8 (voir la norme ISO 10646, Annexe D). Les octets résultant pour la chaîne entière de caractères seront ensuite codés sous la forme de caractères tels que spécifiés dans la norme IETF RFC 2045, au § 6.8, et les caractères résultants formeront la valeur "ExtendedXMLValue".

22 Instruction de codage `DECIMAL`

22.1 Généralités

22.1.1 L'instruction "DecimalInstruction" est la suivante:

DecimalInstruction ::=
`DECIMAL`
TargetList

22.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

22.1.3 Cette instruction de codage a pour objet de modifier le codage d'un type réel de manière à exclure la notation exponentielle et à faire correspondre à un tiret suivi d'un "0" la valeur plus zéro au lieu de la valeur moins zéro.

NOTE – La valeur moins zéro ne peut être représentée.

22.2 Restrictions

22.2.1 Cette instruction de codage ne sera attribuée qu'à un type réel.

22.2.2 Le type réel auquel cette instruction de codage est appliquée sera limité de telle manière que les valeurs moins zéro, `MINUS-INFINITY`, `PLUS-INFINITY`, et `NOT-A-NUMBER` ne soient pas permises et que la `base` soit 10.

NOTE – Il est recommandé que cela se fasse en appliquant les contraintes suivantes:

```
(WITH COMPONENTS {..., base(10)})
(ALL EXCEPT (-0 | MINUS-INFINITY | PLUS-INFINITY | NOT-A-NUMBER))
```

22.2.3 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage `GLOBAL-DEFAULTS MODIFIED-ENCODINGS` ne figure dans la section de commande de codage.

22.2.4 Un type comportant cette instruction finale de codage peut inclure toute autre instruction finale de codage qui est autorisée pour ce type.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: `ANY-ATTRIBUTES, ANY-ELEMENT, BASE64, EMBED-VALUES, LIST, TEXT, USE-NIL, USE-NUMBER, USE-ORDER, USE-QNAME, USE-TYPE, USE-UNION, WHITESPACE`.

22.2.5 Aucune information restrictive ne figurera dans la liste "TargetList".

22.3 Incidence sur les codages

22.3.1 Le nombre "modifiedXMLRealNumber" (voir le § 17.9.3) ne contiendra pas de caractère e ou E suivi d'un exposant.

NOTE – Toutes les valeurs abstraites, y compris celles qui sont des nombres réels très grands ou très petits, sont pour ce motif codées comme une partie entière éventuellement suivie d'un point décimal et d'une partie fractionnaire.

22.3.2 La valeur zéro peut être codée, en fonction du choix du codeur, comme la valeur "XMLDecimalMinusZeroRealValue", définie comme suit:

XMLDecimalMinusZeroRealValue ::=
"-" & modifiedXMLRealNumber

où le nombre "modifiedXMLRealNumber", limité en vertu du § 22.3.1, ne contient pas de chiffres autres que le chiffre zéro.

NOTE – Ce qui précède ne peut être confondu avec la valeur réelle moins zéro, parce que celle-ci a été supprimée par la contrainte obligatoire qui s'applique au type réel (voir le § 22.2.2).

23 Instruction de codage DEFAULT-FOR-EMPTY

23.1 Généralités

23.1.1 L'instruction "DefaultForEmptyInstruction" est la suivante:

DefaultForEmptyInstruction::=
DEFAULT-FOR-EMPTY
TargetList
AS Value

23.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

23.1.3 Cette instruction de codage définit une valeur abstraite qui peut être codée dans un codage EXTENDED-XER (au choix pour le codeur) comme la variante vide "empty" de la valeur "ExtendedXMLValue" pour un type (voir le § 17.4), dont le codage constitue à lui seul le contenu d'un élément XML.

NOTE – Ce mécanisme de repli prend en charge la présence d'un élément XML sans contenu (généralement, mais non nécessairement, codé comme une étiquette d'élément vide). Il est différent de l'emploi de l'indication ASN.1 **DEFAULT**, qui a trait à l'absence d'une valeur "ExtendedXMLNamedValue" d'une composante d'une séquence ou d'un ensemble.

23.1.4 La liste "TargetList" n'emploiera pas le mot-clé **ALL** et identifiera une cible unique.

23.1.5 Les cinq cas distincts d'utilisation de cette instruction de codage sont décrits ci-après.

23.1.5.1 Le premier cas est celui de l'attribution directe à un type susceptible d'être codé au moyen de caractères, qui ne porte pas l'indication **UNTAGGED** (voir le § 32). Si le contenu de l'élément contenant est vide, il correspond à la valeur "Value" spécifiée du type susceptible d'être codé au moyen de caractères (qui est le régulateur pour cette valeur "Value").

23.1.5.2 Le deuxième cas est celui de l'attribution à un type séquence (**NOT UNTAGGED**, **NOT EMBED-VALUES** et **NOT USE-NIL**) qui contient une composante **UNTAGGED**, susceptible d'être codée au moyen de caractères et dont le codage constitue à lui seul le contenu (pour toutes les valeurs abstraites du type séquence) de l'élément contenant du type séquence. Si le contenu de l'élément contenant du type séquence est vide, il correspond à la valeur "Value" spécifiée de la composante susceptible d'être codée au moyen de caractères (qui est la régulatrice pour la valeur "Value").

NOTE – Le contenu ne peut consister qu'en la composante susceptible d'être codée au moyen de caractères, parce qu'elle est la seule ou parce que toutes les autres composantes comportent une instruction finale de codage **ATTRIBUTE** (voir le § 20) ou **ANY-ATTRIBUTES** (voir le § 18).

23.1.5.3 Le troisième cas est celui de l'attribution à un type séquence (**NOT UNTAGGED** et **NOT USE-NIL**) avec une instruction finale de codage **EMBED-VALUES** (voir le § 25.3.1.4). Si le contenu de l'élément contenant du type séquence est vide, il représente une valeur abstraite du type séquence qui pourrait sinon produire un contenu consistant en la seule valeur "Value" spécifiée d'une seule chaîne **UTF8String** dans la sequence-of **EMBED-VALUES** (la chaîne **UTF8String** est la régulatrice pour la valeur "Value").

23.1.5.4 Le quatrième cas est celui de l'attribution à un type séquence (**NOT UNTAGGED**, **NOT EMBED-VALUES**) avec une instruction finale de codage **USE-NIL** (voir le § 33) dont la composante **OPTIONAL** est un type susceptible d'être codé au moyen de caractères. Si l'élément contenant du type séquence possède un attribut d'identification de valeur nulle **true**, l'instruction **DEFAULT-FOR-EMPTY** n'affecte pas la signification du codage. Si l'élément contenant du type séquence possède un attribut d'identification de valeur nulle **false** (ou ne possède pas d'attribut d'identification de

valeur nulle), et un contenu vide, alors celui-ci consiste en la valeur "Value" de la composante **OPTIONAL** (dont le type est le régulateur pour la valeur "Value").

23.1.5.5 Le cinquième cas est celui de l'attribution à un type séquence (**NOT UNTAGGED**) avec des instructions finales de codage **EMBED-VALUES** (voir le § 25.3.1.4) et **USE-NIL** (voir le § 33) dont la composante **OPTIONAL** est un type séquence. Si l'élément contenant du type séquence possède un attribut d'identification de valeur nulle **true**, l'instruction **DEFAULT-FOR-EMPTY** n'affecte pas la signification du codage. Si l'élément contenant du type séquence possède un attribut d'identification de valeur nulle **false** (ou ne possède pas d'attribut d'identification de valeur nulle), et un contenu vide, alors celui-ci représente une valeur abstraite du type séquence, qui pourrait sinon produire un contenu consistant en la seule valeur "Value" spécifiée d'une seule chaîne **UTF8String** dans la sequence-of **EMBED-VALUES** (la chaîne **UTF8String** est le régulateur pour la valeur "Value").

23.1.6 La valeur "Value" est définie dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 16.7.

NOTE – Cela permet l'emploi d'une référence pour la valeur, définie dans le module ou qui y est importée. La référence pour la valeur peut être définie au moyen de la notation des valeurs XML, mais une telle notation ne peut être employée directement dans l'instruction "DefaultForEmptyInstruction".

23.2 Restrictions

23.2.1 Si les instructions finales de codage pour un type ASN.1 qui est un type susceptible d'être codé au moyen de caractères **NOT UNTAGGED**, contiennent une instruction de codage **DEFAULT-FOR-EMPTY**, alors ce type ne sera pas une composante (d'un type ASN.1 **SEQUENCE** ou **SET**) avec une valeur ASN.1 **DEFAULT**.

NOTE – Cette restriction n'est pas strictement nécessaire, mais elle est imposée pour éviter la confusion entre les mécanismes de repli ASN.1 et les mécanismes de repli EXTENDED-XER.

23.2.2 Cette instruction ne sera attribuée qu'aux types suivants:

a) un type susceptible d'être codé au moyen de caractères sans instruction finale de codage **UNTAGGED**; ou

b) un type séquence **NOT UNTAGGED**, sans instruction finale **EMBED-VALUES** ou **USE-NIL**, un type dont les composantes ont un type susceptible d'être codé au moyen de caractères comportant une instruction finale de codage **UNTAGGED** et toutes les autres composantes (si elles sont présentes) comportent une instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES**; ou

c) un type séquence **NOT UNTAGGED**, sans instruction finale de codage **USE-NIL**, mais avec l'instruction finale de codage **EMBED-VALUES** (voir le § 25.3.1.4); ou

d) un type séquence **NOT UNTAGGED**, sans instruction finale de codage **EMBED-VALUES**, mais avec l'instruction finale de codage **USE-NIL**, dont la composante **OPTIONAL** est un type susceptible d'être codé au moyen de caractères; ou

e) un type séquence **NOT UNTAGGED** avec une instruction finale de codage **EMBED-VALUES** et une instruction finale de codage **USE-NIL**, dont la composante **OPTIONAL** est un type séquence.

23.2.3 Si le § 23.2.2 a) s'applique, et que le vide "empty" est une valeur "ExtendedXMLValue" valable pour l'une des valeurs abstraites (soit V) du type (éventuellement soumis à une contrainte), V différant de la valeur "Value" dans l'instruction "DefaultForEmptyInstruction", alors il y aura au moins une autre variante de codage pour V.

23.2.4 Si le § 23.2.2 b) ou d) s'applique, et que le vide "empty" est une valeur "ExtendedXMLValue" valable pour l'une des valeurs abstraites (soit V) de la composante **UNTAGGED** (cas b)) ou de la composante **OPTIONAL** (cas d)), V différant de la valeur "Value" dans l'instruction "DefaultForEmptyInstruction", alors il y aura au moins une autre variante de codage pour V.

NOTE – Il est admis que certains outils ASN.1 peuvent ne pas être en mesure de vérifier statiquement que les restrictions susmentionnées sont satisfaites pour toutes les valeurs abstraites, mais les codeurs réputés conformes ne peuvent produire des codages dans lesquels la valeur "ExtendedXMLValue" viole cette restriction.

23.2.5 Si le § 23.2.2 c) s'applique, le type **SEQUENCE** sera soumis à une contrainte de manière (sans instruction **DEFAULT-FOR-EMPTY**) qu'il n'y ait pas de valeur abstraite qui puisse produire un contenu vide pour l'élément contenant.

23.2.6 Si un type susceptible d'être codé au moyen de caractères (cas 23.2.2 a)), comportant cette instruction finale de codage, possède un type contenant qui est un type sequence-of ou set-of comportant une instruction finale de codage **LIST**, ou qui est un type choix comportant une instruction de codage **USE-UNION**, alors il ne sera pas tenu compte de cette instruction finale de codage.

23.2.7 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

23.2.8 Un type comportant cette instruction finale de codage n'inclura aucune des instructions finales de codage **ANY-ELEMENT**, **ATTRIBUTE** ou **UNTAGGED**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **USE-TYPE**.

23.2.9 Aucune information restrictive ne figurera dans la liste "TargetList".

23.3 Incidence sur les codages

23.3.1 Cette instruction de codage n'affecte que la valeur "ExtendedXMLValue" du type qui est le régulateur de la valeur "Value" (voir le § 23.1.5).

23.3.2 Le codage "ExtendedXMLValue" de la valeur abstraite spécifiée par la valeur "Value" sera, en fonction du choix du codeur:

a) soit le codage "ExtendedXMLValue" de la valeur qui serait produite si l'instruction **DEFAULT-FOR-EMPTY** n'était pas présente (codage normal);

b) soit le codage vide "empty".

NOTE – Les décodeurs sont tenus d'accepter aussi bien le codage normal que le codage vide "empty" en tant que désignation de la valeur default-for-empty.

23.3.3 Si le § 23.2.2 a) s'applique et que la valeur vide "empty" est une valeur "ExtendedXMLValue" valable pour l'une de valeurs abstraites (soit V) du type, V différant de la valeur "Value" spécifiée dans l'instruction "DefaultForEmptyInstruction", alors une variante de codage quelconque pour V sera employée (en fonction du choix du codeur) au lieu de la valeur vide "empty".

23.3.4 Si le § 23.2.2 b) ou d) s'applique, et que la valeur vide "empty" est une valeur "ExtendedXMLValue" valable pour l'une des valeurs (soit V) de la composante **UNTAGGED** (cas b)) ou de la composante **OPTIONAL** (cas d)), V différant de la valeur "Value" spécifiée dans l'instruction "DefaultForEmptyInstruction", alors une variante de codage quelconque pour V sera employée (en fonction du choix du codeur) au lieu de la valeur vide "empty".

23.3.5 Si le § 23.2.2 c) s'applique, l'effet de cette instruction de codage est spécifiée dans les § 25.3.1.4 et 25.3.1.5.

23.3.6 Si le 23.2.2 e) s'applique, l'effet de cette instruction de codage est spécifiée dans le § 25.3.1.6.

24 Instruction de codage ELEMENT

24.1 Généralités

24.1.1 L'instruction "ElementInstruction" est la suivante:

ElementInstruction ::=
ELEMENT
TargetList

24.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

24.1.3 Cette instruction de codage est synonyme de l'instruction **NOT UNTAGGED**, et n'implique aucune sémantique autre que celle de l'instruction **NOT UNTAGGED**.

24.2 Restrictions

24.2.1 Aucune information restrictive ne figurera dans la liste "TargetList".

24.2.2 Cette instruction de codage ne devrait pas être employée en tant qu'instruction de codage préfixée avec l'une quelconque des instructions de codage préfixées **ANY-ATTRIBUTES**, **ANY-ELEMENT** ou **ATTRIBUTE** pour éviter toute confusion.

24.3 Incidence sur les codages

Cette instruction de codage est la négation d'une instruction de codage **UNTAGGED** et n'affecte pas les codages par ailleurs.

25 Instruction de codage EMBED-VALUES

25.1 Généralités

25.1.1 L'instruction "EmbedValuesInstruction" est la suivante:

EmbedValuesInstruction ::=
EMBED-VALUES
TargetList

25.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

25.1.3 Cette instruction de codage permet à la première composante d'un type séquence (**NOT UNTAGGED**) de fournir une chaîne de caractères, à insérer avant le premier élément XML, après le dernier élément XML et entre les éléments XML, qui forment le codage de la valeur "ExtendedXMLValue" du type séquence.

25.1.4 Si une instruction finale de codage **USE-NIL** est aussi présente, et que la composante **OPTIONAL** prenant en charge l'instruction **USE-NIL** ne figure pas dans une valeur abstraite particulière, il n'y aura pas d'éléments XML pour les composantes du type séquence, et aucune chaîne de caractères ne sera fournie pour cette valeur abstraite. Sinon, pour toutes les valeurs abstraites, le nombre de chaînes de caractères fournies doit dépasser d'une unité le nombre d'éléments dans le codage du type séquence. Certaines ou toutes les chaînes de caractères peuvent être vides.

25.2 Restrictions

25.2.1 Un type ASN.1 ne comportera pas cette instruction finale de codage à moins qu'il ne soit un type séquence. La première composante de la séquence sera une **SEQUENCE OF** de chaîne **UTF8String** et ne sera pas marquée comme étant **OPTIONAL** ou **DEFAULT**.

25.2.2 Aucune instruction finale de codage **UNTAGGED** (voir le § 32) ne s'appliquera ni au type sequence-of ni à la composante du type sequence-of.

25.2.3 Aucune instruction finale de codage **UNTAGGED** ne s'appliquera aux composantes du type séquence, susceptible d'être codé au moyen de caractères.

25.2.4 Si le type séquence comporte aussi une instruction finale de codage **USE-NIL**, la composante **OPTIONAL** prenant en charge l'instruction de codage **USE-NIL** ne sera pas un type susceptible d'être codé au moyen de caractères (voir aussi le § 33.2.4).

25.2.5 Aucune des composantes de la séquence ne sera marquée comme étant **DEFAULT** à moins qu'elle ne comporte une instruction finale de codage **ATTRIBUTE**. S'il existe des composantes de type **SEQUENCE** ou **SET** (à une quelconque profondeur) qui, au moyen de l'emploi de l'indication **UNTAGGED**, peuvent produire des éléments dans la valeur "ExtendedXMLValue" qui sont des éléments descendants immédiats du type séquence, ceux-ci ne seront pas marqués comme étant **DEFAULT**.

25.2.6 Le type séquence sera réglementé de telle façon que:

a) si le type comporte aussi une instruction finale de codage **USE-NIL** et que la composante **OPTIONAL** qui prend en charge l'instruction **USE-NIL** est absente, le nombre de répétitions de la composante sequence-of doit être égal à zéro;

b) sinon, le nombre de répétitions de la composante sequence-of dans chacune des valeurs abstraites dépasse d'une unité le nombre d'éléments XML dans la valeur "ExtendedXMLValue" du type séquence, déterminé après application de toutes les instructions finales de codage aux autres composantes de la séquence, en ne tenant pas compte de la première composante.

NOTE – Il est recommandé que la contrainte sur le type séquence s'exprime comme suit:

```
(CONSTRAINED BY
        {/* Doit être conforme à la Rec. UIT-T X.693 | ISO/CEI 8825-4, § 25 */})
```

25.2.7 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

25.2.8 Un type comportant cette instruction finale de codage ne comportera pas d'instruction finale de codage **UNTAGGED**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **ATTRIBUTE**, **BASE64**, **DECIMAL**, **LIST**, **TEXT**, **USE-NUMBER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

25.2.9 Aucune information restrictive ne figurera dans la liste "TargetList".

25.3 Incidence sur les codages

25.3.1 Un codeur produira d'abord un codage partiel de la valeur "ExtendedXMLValue" du type séquence contenant, en ne tenant pas compte de la première composante. Il modifiera ensuite ce codage comme spécifié dans les paragraphes ci-après.

NOTE – Les valeurs de la chaîne **UTF8String** qui sont insérées peuvent être vides "empty".

25.3.1.1 La première valeur de la chaîne **UTF8String** dans le type sequence-of sera insérée (conformément au § 25.3.1.6) au début du codage partiel, avant l'étiquette de début du premier élément XML (s'il existe).

25.3.1.2 Chacune des valeurs successives de la chaîne **UTF8String** (s'il en existe) sera insérée entre l'étiquette de fin d'un élément XML et l'étiquette de début de l'élément XML suivant, en allant du premier élément au dernier élément.

NOTE – Ce qui précède implique qu'aucune des valeurs de la chaîne **UTF8String** n'est insérée dans l'un quelconque de ces éléments, même s'ils ont des éléments descendants.

25.3.1.3 La dernière valeur de la chaîne **UTF8String** (s'il en existe) sera insérée à la fin du codage partiel, après l'étiquette de fin du dernier élément XML.

25.3.1.4 Si aucun élément XML ne figure dans le codage partiel, qu'une instruction finale de codage **DEFAULT-FOR-EMPTY** (voir le § 23) s'applique aussi au type séquence, et que la valeur de la première (et de la seule) chaîne **UTF8String** dans le type sequence-of est identique à la valeur "Value" spécifiée dans l'instruction de codage **DEFAULT-FOR-EMPTY**, un codeur peut éventuellement coder la chaîne **UTF8String** en tant que chaîne vide (mais voir le § 25.3.1.6).

25.3.1.5 Si aucun élément XML ne figure dans le codage partiel, qu'une instruction finale de codage **DEFAULT-FOR-EMPTY** (voir le § 23) s'applique aussi au type séquence, et que le codage est vide, un décodeur interprétera cela comme le codage de la valeur "Value" spécifiée dans l'instruction de codage **DEFAULT-FOR-EMPTY** et attribuera cette valeur abstraite à la première (et à la seule) chaîne **UTF8String** dans le type sequence-of (mais voir le § 25.3.1.6).

NOTE – Cela veut dire qu'une valeur sans élément XML et comportant une seule valeur de chaîne UTF8String vide ne peut être codée. Ce type séquence doit être soumis à des contraintes pour que de telles valeurs soient interdites (voir le § 23.2.5).

25.3.1.6 Si le type comporte aussi une instruction finale de codage **USE-NIL** et que la composante **OPTIONAL** est absente, alors l'instruction de codage **EMBED-VALUES** n'a pas d'incidence. Si le type comporte aussi une instruction finale de codage **USE-NIL** et que la composante **OPTIONAL** est présente, alors le § 25.3.1.4 s'applique. Si un décodeur détermine que la composante **OPTIONAL** est présente, à partir de l'absence d'un attribut d'identification de valeur nulle (ou de sa présence avec la valeur false), alors le § 25.3.1.5 s'applique.

26 Instruction de codage GLOBAL-DEFAULTS

26.1 Généralités

26.1.1 L'instruction "GlobalDefaultsInstruction" est la suivante:

GlobalDefaultsInstruction ::=
GLOBAL-DEFAULTS TargetList DefaultSetting

DefaultSetting ::=
ControlNamespace
| **MODIFIED-ENCODINGS**

ControlNamespace ::=
CONTROL-NAMESPACE
QuotedURI
Prefix ?

26.1.2 La production de la liste "TargetList" est définie dans le § 14.2, et sera vide "empty".

26.1.3 L'identificateur "QuotedURI" et le préfixe "Prefix" sont définis dans le § 29.1.1.

26.1.4 La production de l'espace "ControlNamespace" spécifie le nom de l'espace de commande (l'identificateur "URI" dans l'identificateur "QuotedURI"), et un préfixe recommandé pour cet espace de nom. Si l'instruction de codage **GLOBAL-DEFAULTS** n'est pas présente, l'espace de nom de commande sera celui qui est spécifié dans le § 16.9.

26.1.5 L'emploi des codages **MODIFIED-ENCODINGS** produit des valeurs "ExtendedXMLValues" qui sont modifiées conformément aux § 10.2.7 et 10.2.8.

26.2 Restrictions

26.2.1 L'instruction de codage **GLOBAL-DEFAULTS** ne sera attribuée que dans une section de commande de codage et ne sera pas précédée d'une autre instruction de codage à l'exception des autres instructions de codage **GLOBAL-DEFAULTS**.

26.2.2 Chacune des variantes de l'instruction **GLOBAL-DEFAULTS** sera employée tout au plus une fois dans une section de commande de codage.

26.2.3 Le codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS**, s'il est présent, sera la première instruction de codage dans la section de commande de codage XER d'un module ASN.1.

26.3 Incidence sur les codages

26.3.1 L'application du codage **MODIFIED-ENCODINGS** exige que les codages soient modifiés comme spécifié dans les § 10.2.7 et 10.2.8.

26.3.2 L'espace de nom de commande pour un document XML entier sera celui qui est attribué au type ASN.1 dont le codage forme l'élément de base de ce document XML.

27 Instruction de codage LIST

27.1 Généralités

27.1.1 L'instruction "ListInstruction" est la suivante:

ListInstruction ::=
LIST
TargetList

27.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

27.1.3 Cette instruction de codage exige que la valeur "ExtendedXMLSequenceOfValue" ou "ExtendedXMLSetOfValue" d'un type sequence-of ou set-of (voir le § 17.7) soit la valeur "ExtendedXMLListValue", produisant une liste entrecoupée de blancs des valeurs de la composante du type sequence-of ou set-of.

NOTE – Une attribution courante de cette instruction de codage est l'attribution à la **SEQUENCE OF INTEGER**, à laquelle est aussi attribuée l'instruction de codage **ATTRIBUTE** (voir le § 20).

27.2 Restrictions

27.2.1 Le type auquel cette instruction de codage est attribuée sera un type sequence-of ou set-of.

27.2.2 La composante du type sequence-of ou set-of:

a) sera un type susceptible d'être codé au moyen de caractères;

b) sera tel que, pour toutes ses valeurs abstraites, il y ait au moins un codage de la valeur "ExtendedXMLValue" (en tenant compte de toutes les options pour le codeur) qui ne soit pas vide "empty" et qui ne contienne pas de "blancs avec échappement" (voir le § 8.1.5).

NOTE 1 – Les restrictions ci-dessus impliquent que la composante ne peut être elle-même un type sequence-of ou set-of comportant une instruction de codage **LIST**, ou contenant un type imbriqué sequence-of ou set-of comportant une instruction de codage **LIST** à une profondeur quelconque.

NOTE 2 – Les restrictions ci-dessus seront satisfaites si le type de la composante du type sequence-of ou set-of est le type réel, le type identificateur d'objet, le type identificateur d'objet relatif, ou les types utiles **GeneralizedTime** et **UTCTime**. Elles seront aussi satisfaites si c'est un type chaîne de caractères, contraint à ce que la chaîne comporte toujours au moins un caractère et qu'aucune de ses valeurs ne contiennent des caractères "blancs".

NOTE 3 – Il est admis que certains outils ASN.1 peuvent ne pas être en mesure de vérifier statiquement que les règles susmentionnées sont satisfaites, mais un codeur réputé conforme ne produira pas de codages qui violent l'alinéa b) ci-dessus.

27.2.3 Un type comportant cette instruction finale de codage ne comportera pas d'instruction finale de codage **ANY-ATTRIBUTES**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ELEMENT**, **BASE64**, **DECIMAL**, **EMBED-VALUES**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

27.2.4 Aucune information restrictive ne figurera dans la liste "TargetList".

27.3 Incidence sur les codages

27.3.1 Cette instruction de codage n'affecte que le codage du type auquel elle s'applique.

27.3.2 La production de la valeur "ExtendedXMLSequenceOfValue" ou "ExtendedXMLSetOfValue" (voir le § 17.7) sera la variante de la valeur "ExtendedXMLListValue". La valeur "ExtendedXMLListValue" est la suivante:

ExtendedXMLListValue ::=
empty
| CharacterEncodableValueExtendedXMLListValue

27.3.3 Il y aura des "blancs avec échappement" (voir le § 8.1.5) entre toutes les paires de valeurs adjacentes "CharacterEncodableValue" dans la valeur "ExtendedXMLListValue".

27.3.4 La valeur "CharacterEncodableValue" est définie dans le § 20.3.3. Chacune des valeurs "CharacterEncodableValue" codera une valeur d'une composante du type sequence-of ou set-of.

27.3.5 L'ordre d'apparition des valeurs "CharacterEncodableValue" dans la valeur "ExtendedXMLListValue" sera le même que l'ordre d'apparition des valeurs correspondantes "ExtendedXMLValue" dans la valeur "ExtendedXMLSequenceOfValue" ou "ExtendedXMLSetOfValue" lorsqu'une instruction finale de codage **LIST** n'est pas présente.

27.3.6 Les valeurs "CharacterEncodableValue" dans la valeur "ExtendedXMLListValue" ne seront pas vides "empty" et ne contiendront pas de "blancs avec échappement" (voir le § 8.1.5).

NOTE – Le § 27.2.2 b) assure que cela sera possible, mais le § 27.3.4 peut restreindre les options pour le codeur.

28 Instruction de codage NAME

28.1 Généralités

28.1.1 L'instruction "NameInstruction" est la suivante:

NameInstruction ::=
NAME
TargetList
AS
NewNameOrKeyword

NewNameOrKeyword ::=
NewName
| Keyword

NewName ::=
RestrictedCharacterStringValue

Keyword ::=
CAPITALIZED
| UNCAPITALIZED
| UPPERCASED
| LOWERCASED

28.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

28.1.3 Cette instruction de codage a les cinq objectifs suivants:

a) de modifier le nom d'étiquette associée, le nom d'attribut, ou la valeur d'un éventuel attribut d'identification du type (le nom "NewName" sans information "QualifyingInformation" dans la liste "TargetList") de la cible; ou

b) de modifier la casse entière (ou la casse de la lettre initiale) du nom d'étiquette associée, du nom d'attribut, ou de la valeur d'un éventuel attribut d'identification du type (le mot-clé "Keyword" sans information "QualifyingInformation" dans la liste "TargetList") de la ou des cibles; ou

c) de modifier le nom d'élément employé dans une étiquette d'élément vide, normalement déduit (comme spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1) d'un identificateur spécifié utilisé dans la définition du type (le nom "NewName" avec une information "QualifyingInformation" dans la liste "TargetList" qui n'est pas **ALL**) de la cible; ou

d) de modifier la casse entière (ou la casse de la lettre initiale) du nom d'élément employé dans une étiquette d'élément vide, normalement déduit d'un identificateur spécifié utilisé dans la définition du type (le mot-

clé "Keyword" avec une information "QualifyingInformation" dans la liste "TargetList" qui n'est pas **ALL**) de la ou des cibles; ou

e) de modifier la casse entière (ou la casse de la lettre initiale) des noms d'élément employés dans le codage de la valeur "ExtendedXMLValue" déduit d'un identificateur utilisé dans la définition du type (le mot-clé "Keyword" avec une information "QualifyingInformation" dans la liste "TargetList" qui n'est pas **ALL**) de la ou des cibles.

NOTE 1 – Le nom "NewName" peut être employé pour modifier des noms employés dans un codage EXTENDED-XER déduit des identificateurs ou des références au type, mais il est rarement utile si le nouveau nom peut être employé en premier lieu en tant qu'identificateur ASN.1 ou référence au type. Donc l'emploi normal de l'instruction de codage **NAME** sert à produire des noms d'élément XML ou d'attribut requis, lorsque ceux-ci ne seraient sinon pas admis en raison des règles ASN.1 relatives à la casse de la lettre initiale des identificateurs ou des noms de référence au type, ou lorsque les règles ASN.1 pour les différents identificateurs dans les structures séquence, ensemble et choix empêchent d'appliquer un codage XML souhaité.

NOTE 2 – L'emploi de l'indication **ALL IN ALL AS CAPITALIZED** pour mettre en majuscules tous les identificateurs dans un module peut être particulièrement utile pour obtenir un style commun au moyen des lettres initiales majuscules.

NOTE 3 – Si une instruction de codage **NAME** est attribuée au moyen d'une cible identifiée par un identificateur "identifier" ou une référence "typereference", cela affecte le nom employé dans le codage EXTENDED-XER, mais n'affecte pas le nom qui est employé pour identifier la même cible dans les instructions de codage XER suivantes.

28.1.4 La valeur "RestrictedCharacterStringValue" est définie dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 37.

28.2 Restrictions

28.2.1 Le nom "NewName" ne sera pas employé si l'information "QualifyingInformation" est l'indication **ALL**.

28.2.2 L'instruction de codage **NAME** comportant une information "QualifyingInformation" ne sera attribuée qu'aux définitions suivantes des types:

a) une définition de type booléen; ou

b) une définition de type bitstring avec des bits nommés; ou

c) une définition de type énuméré; ou

d) une définition de type entier avec des nombres nommés.

28.2.3 La valeur "RestrictedCharacterStringValue" dans le nom "NewName", lorsqu'elle est employée dans l'instruction de codage **NAME,** sera soit un nom "NCName" défini dans les espaces de nom de la version XML du Consortium W3C, au § 2, dans la production 4, soit une chaîne de caractères vide. Elle ne sera une chaîne de caractères vide que si l'instruction de codage **NAME** est appliquée à une variante d'un type choix avec une instruction finale de codage **USE-UNION**.

NOTE 1 – Il est prescrit dans les espaces de nom de la version XML du Consortium W3C qu'un nom "NCName" ne doit pas commencer par des caractères qui, en majuscules, sont les caractères "XML".

NOTE 2 – La production du mot "NewNameOrKeyword" (et donc la production du nom "NewName") est aussi employée dans le § 31. Les restrictions ci-dessus relatives à la valeur "RestrictedCharacterStringValue" ne s'appliquent pas à l'emploi dans le § 31.

28.2.4 S'il existe une instruction de codage **GLOBAL-DEFAULTS** comportant un mot-clé **MODIFIED-ENCODINGS**, aucune information "QualifyingInformation" ne figurera dans la liste "TargetList".

NOTE – Ceci est dû au fait que les étiquettes d'élément vide ne sont pas employées dans ce cas. L'instruction de codage **TEXT** peut au lieu de cela être employée pour modifier le codage des valeurs individuelles d'un type.

28.2.5 Cette instruction de codage ne devrait pas être employée en tant qu'instruction de codage préfixée avec l'une quelconque des instructions de codage préfixées **ANY-ATTRIBUTES, ANY-ELEMENT** ou **UNTAGGED,** pour éviter toute confusion.

28.3 Incidence sur les codages

28.3.1 Si le type auquel cette instruction de codage est appliquée comporte une instruction finale de codage **ATTRIBUTE**, le nom "AttributeName" (qui dans ce cas est un identificateur "IdentifierOrModifiedIdentifier") de l'"Attribute" (voir le § 20.3.3) sera la variante du nom "QualifiedOrUnqualifiedName" comme spécifié dans les § 28.3.3 à 28.3.6.

28.3.2 Si le type auquel cette instruction de codage est appliquée ne comporte pas d'instruction finale de codage **ATTRIBUTE**, alors le nom de l'étiquette de l'élément contenant (qui est le nom "TagName" – voir le § 17.5.1) sera la variante du nom "QualifiedOrUnqualifiedName" comme spécifié dans les § 28.3.3 à 28.3.6.

28.3.3 Les variantes d'identificateur "IdentifierOrModifiedIdentifier" et de nom "QualifiedOrUnqualifiedName" seront employées. Le nom "UnprefixedName" dans le nom "QualifiedOrUnqualifiedName" sera l'identificateur "identifier" de la composante modifiée conformément au mot "NewNameOrKeyword" comme spécifié ci-après.

28.3.4 Si la variante du nom "NewName" est employée, le nom "UnprefixedName" sera remplacé par le nom "NewName".

28.3.5 Si la variante du mot-clé "Keyword" est employée, le nom "UnprefixedName" sera modifié comme spécifié dans les sous-paragraphes du § 28.3.5.

28.3.5.1 Si le mot-clé "Keyword" est **CAPITALIZED**, alors, si le premier caractère du nom "UnprefixedName" est une lettre minuscule, ce caractère sera remplacé par l'équivalent majuscule, le nom restant autrement inchangé.

28.3.5.2 Si le mot-clé "Keyword" est **UNCAPITALIZED**, alors, si le premier caractère du nom "UnprefixedName" est une lettre majuscule, ce caractère sera remplacé par l'équivalent minuscule, le nom restant autrement inchangé.

28.3.5.3 Si le mot-clé "Keyword" est **UPPERCASED**, tous les caractères du nom "UnprefixedName" qui sont des lettres minuscules seront remplacés par leurs équivalents majuscules. Les autres caractères restent inchangés.

28.3.5.4 Si le mot-clé "Keyword" est **LOWERCASED**, tous les caractères du nom "UnprefixedName" qui sont des lettres majuscules seront remplacés par leurs équivalents minuscules. Les autres caractères restent inchangés.

28.3.6 Le nom "QualifiedOrUnqualifiedName" sera un nom restrictif de l'espace de nom si et seulement si le "Type" comporte une instruction finale de codage **NAMESPACE**.

29 Instruction de codage NAMESPACE

29.1 Généralités

29.1.1 L'instruction "NamespaceInstruction" est la suivante:

NamespaceInstruction ::=
NAMESPACE
TargetList
NamespaceSpecification ?

NamespaceSpecification ::=
AS
QuotedURI
Prefix ?

Prefix ::=
PREFIX
QuotedNCName

QuotedURI ::=
""" & URI & """

QuotedNCName ::=
""" & NCName & """

29.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

NOTE – L'emploi le plus courant de cette instruction de codage est l'instruction **NAMESPACE ALL**.

29.1.3 Cette instruction de codage permet d'attribuer à une ou à plusieurs cibles un nom d'espace de nom et un préfixe d'espace de nom recommandé.

29.1.4 La production de l'identificateur "URI" n'est pas définie dans la présente Recommandation | Norme internationale, mais elle comporte des caractères qui identifient un identificateur uniforme de ressource. La syntaxe (et la sémantique) d'un identificateur URI est définie dans la norme IETF RFC 2396 et commence par le nom d'un schéma URI. Pour les attributions des noms d'espace de nom au moyen de l'instruction de codage **NAMESPACE**, on peut employer le schéma URI.

29.1.5 S'il n'existe aucune spécification "NamespaceSpecification", une valeur par défaut est attribuée, le "Prefix" recommandé étant fixé à la référence "modulereference" et l'identificateur "URI"étant fixé comme suit:

a) le schéma URI (voir la norme IETF RFC 2396) sera donné par **urn**;

b) l'identificateur de l'espace de nom URN (voir la norme IETF RFC 2141) sera donné par **oid**;

c) la chaîne spécifique de l'espace de nom URN (voir la norme IETF RFC 2141) sera donnée par l'identificateur "DefinitiveIdentifier" du module exprimé comme une valeur "XMLObjectIdentifierValue" (voir la norme IETF RFC 3061).

29.1.6 EXEMPLE: Soit une valeur d'identificateur d'objet **{iso standard 1564 modules(0) basic(1)}**. L'identificateur "URI" sera donné par la chaîne de caractères **"urn:oid:1.0.1564.0.1".**

29.1.7 La production du nom "NCName" est définie dans les espaces de nom de la version XML du Consortium W3C, au § 2, dans la production 4, et ne commencera pas par les caractères qui, en majuscules, sont les caractères "**XML**".

NOTE – Ceci est prescrit par les espaces de nom de la version XML du Consortium W3C.

29.2 Restrictions

29.2.1 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

29.3 Incidence sur les codages

29.3.1 Un nom restrictif de l'espace de nom peut être exigé pour un nom d'étiquette associée, pour un nom d'attribut, ou pour la valeur d'un attribut d'identification du type. Un nom restrictif de l'espace de nom est exigé si le type produisant le nom comporte une instruction finale de codage **NAMESPACE**.

29.3.2 Le nom "QualifiedOrUnqualifiedName" est le suivant:

QualifiedOrUnqualifiedName ::=
QualifiedName
| UnqualifiedName

QualifiedName ::=
PrefixedName
| UnprefixedName

UnqualifiedName ::=
UnprefixedName

PrefixedName ::=
DeclaredPrefix & ":" & UnprefixedName

UnprefixedName ::= NCName

DeclaredPrefix ::= NCName

29.3.3 Le codage d'un nom restrictif d'espace de nom exige:

a) soit l'emploi de la variante du nom "PrefixedName" pour le nom "QualifiedName" avec adjonction aux éléments XML d'autres attributs fournissant des déclarations d'espace de nom (comme spécifié dans les espaces de nom de la version XML du Consortium W3C);

b) soit l'emploi de la variante du nom "UnprefixedName" pour le nom "QualifiedName" avec adjonction aux éléments XML d'autres attributs fournissant des déclarations d'espace de nom par défaut (comme spécifié dans les espaces de nom de la version XML du Consortium W3C).

29.3.4 Le choix de ces deux mécanismes et des éléments XML auxquels les attributs de déclaration d'espace de nom sont ajoutés est laissé au codeur.

NOTE 1 – Dans les espaces de nom de la version XML du Consortium W3C, il est spécifié qu'une déclaration d'espace de nom par défaut ne s'applique qu'au nom de l'élément dans lequel elle figure (et de l'élément descendant), mais pas des attributs de cet élément ou des éléments descendants.

NOTE 2 – Il est recommandé, mais il n'est pas exigé, d'employer le préfixe recommandé dans l'instruction de codage **NAMESPACE**.

NOTE 3 – L'emploi du préfixe recommandé peut être inapproprié si des instructions de codage **NAMESPACE** ayant des noms d'espace de nom différents mais le même préfixe recommandé figurent dans le module.

30 Instruction de codage PI-OR-COMMENT

30.1 Généralités

30.1.1 L'instruction "PIOrCommentInstruction" est la suivante:

PIOrCommentInstruction ::=
PI-OR-COMMENT

```
        TargetList
        AS
        RestrictedCharacterStringValue
        Position

    Position ::=
        BEFORE-TAG
        | BEFORE-VALUE
        | AFTER-VALUE
        | AFTER-TAG
```

30.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

30.1.3 Cette instruction de codage a pour effet d'insérer avant ou après la valeur "ExtendedXMLValue" ou avant ou après les étiquettes associées des instructions et/ou des commentaires XML donnés.

NOTE – Le § 10.2.5 permet au codeur (selon son choix) d'insérer des instructions de traitement XML et des commentaires XML supplémentaires.

30.1.4 La valeur "RestrictedCharacterStringValue" est définie dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 37.

30.2 Restrictions

30.2.1 La valeur de la valeur "RestrictedCharacterStringValue" sera la concaténation d'une ou de plusieurs chaînes de caractères, chacune d'elles étant conforme à la syntaxe d'une instruction de traitement spécifiée dans la version XML 1.0 du Consortium W3C, au § 2.6, ou à la syntaxe d'un commentaire spécifié dans la version XML 1.0 du Consortium W3C, au § 2.5, et définissant les instructions de traitement et/ou les commentaires à insérer dans le document XML.

30.2.2 Un type ASN.1 ne comportera pas simultanément une instruction finale de codage **UNTAGGED** et une instruction finale de codage **PI-OR-COMMENT**.

30.2.3 Un type comportant cette instruction finale de codage ne comportera aucune des instructions finales de codage **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **ATTRIBUTE** ou **UNTAGGED**.

30.2.4 Aucune information restrictive ne figurera dans la liste "TargetList".

30.3 Incidence sur les codages

30.3.1 Si la "Position" est **BEFORE-TAG**, les instructions de traitement et/ou les commentaires seront insérés avant l'étiquette de début ou l'étiquette d'élément vide associée. Si cette étiquette de début ou étiquette d'élément vide marque le début d'une certaine valeur "ExtendedXMLValue" contenant, alors toute instruction de traitement et/ou commentaire insérés avant cette valeur "ExtendedXMLvalue" (en appliquant **BEFORE-VALUE** au type correspondant) précédera ces instructions de traitement et/ou commentaires dans le document XML.

30.3.2 Si la "Position" est **BEFORE-VALUE**, les instructions de traitement et/ou les commentaires seront insérés au début de la valeur "ExtendedXMLValue". Si cette valeur "ExtendedXMLValue" commence par une étiquette qui est l'étiquette de début associée d'une certaine valeur "ExtendedXMLValue" imbriquée, alors toute instruction de traitement et/ou commentaire insérés avant cette étiquette de début associée (en appliquant **BEFORE-TAG** au type correspondant) suivra ces instructions de traitement et/ou commentaires dans le document XML.

NOTE – Dans ce cas, le contenu des étiquettes associées n'est jamais vide, et l'étiquette d'élément vide ne peut pas être employée.

30.3.3 Si la "Position" est **AFTER-VALUE**, les instructions de traitement et/ou les commentaires seront insérés à la fin de la valeur "ExtendedXMLValue". Si cette valeur "ExtendedXMLValue" se termine par une étiquette qui est l'étiquette de fin associée d'une certaine valeur "ExtendedXMLValue" imbriquée, alors toute instruction de traitement et/ou commentaire insérés après cette étiquette de fin associée (en appliquant **AFTER-TAG** au type correspondant) précédera ces instructions de traitement et/ou commentaires dans le document XML.

NOTE – Dans ce cas, le contenu des étiquettes associées n'est jamais vide, et l'étiquette d'élément vide ne peut pas être employée.

30.3.4 Si la "Position" est **AFTER-TAG**, les instructions de traitement et/ou les commentaires seront insérés après l'étiquette de fin ou d'élément vide associée. Si cette étiquette de fin ou étiquette d'élément vide marque la fin d'une certaine valeur "ExtendedXMLValue" contenant, alors toute instruction de traitement et/ou commentaire insérés après cette valeur "ExtendedXMLvalue" (en appliquant **AFTER-VALUE** au type correspondant) suivra ces instructions de traitement et/ou commentaires dans le document XML.

31 Instruction de codage TEXT

31.1 Généralités

31.1.1 L'instruction "TextInstruction" est la suivante:

TextInstruction ::=
TEXT
TargetList
TextToBeUsed ?

TextToBeUsed ::=
AS
NewNameOrKeyword

31.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

31.1.3 Cette instruction de codage a pour objet:

a) en l'absence de l'instruction **GLOBAL-DEFAULTS MODIFIED-ENCODINGS**, de permettre que des valeurs des types booléen, énuméré, bitstring avec bits nommés, et entier avec nombres nommés, soient codées comme des chaînes de caractères au lieu d'étiquettes d'élément vide;

b) en présence de l'instruction **GLOBAL-DEFAULTS MODIFIED-ENCODINGS**, de permettre que des chaînes de caractères qui sont employées pour les valeurs des types booléen, énuméré, bitstring avec bits nommés, et entier avec nombres nommés, soient modifiées.

31.1.4 Le mot-clé "NewNameOrKeyword" est défini dans le § 28. Le nom "NewName" dans le mot-clé "NewNameOrKeyword" contiendra au moins un caractère.

31.2 Restrictions

31.2.1 Cette instruction de codage ne sera attribuée qu'aux types suivants, avec des informations restrictives identifiant un ou plusieurs identificateurs employés dans la définition du type (ou les valeurs **true** ou **false** pour le type booléen):

a) une définition de type booléen; ou

b) une définition de type bitstring avec des bits nommés; ou

c) une définition de type énuméré; ou

d) une définition de type entier avec des nombres nommés.

31.2.2 Les chaînes de caractères finales employées pour les valeurs du type auquel cette instruction de codage est attribuée seront distinctes.

31.2.3 Le nom "NewName" dans le mot "NewNameOrKeyword" ne sera pas employé si l'information "QualifyingInformation" est **ALL**. Le § 28.2.3 ne s'applique pas à cet usage du mot "NewNameOrKeyword".

31.2.4 En l'absence d'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS**, l'ensemble des instructions finales de codage **TEXT** pour un type ne produira pas de codage de texte pour certaines valeurs abstraites ni de codage d'élément vide pour d'autres valeurs abstraites.

NOTE – S'il existe une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS**, alors tous les codages sont des codages de texte.

31.2.5 Si l'instruction de codage **TEXT** est appliquée à un type bitstring avec des bits nommés et que le nom "NewName" est employé, celui-ci ne contiendra pas de "blancs avec échappement" (voir le § 8.1.5) et ne commencera pas par un "0" (DIGIT ZERO) ou un "1" (DIGIT ONE).

31.2.6 Un type comportant cette instruction finale de codage ne contiendra pas en même temps l'instruction finale de codage **USE-NUMBER**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **BASE64**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **USE-NIL**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

31.2.7 Les informations "QualifyingInformation" seront toujours présentes.

31.3 Incidence sur les codages

31.3.1 L'un des cinq paragraphes suivants (31.3.2 à 31.3.6) s'applique.

31.3.2 Si le type n'est pas un type bitstring avec des bits nommés et que le texte "TextToBeUsed" est absent, le codage de la valeur "ExtendedXMLValue" de chacune des valeurs auxquelles il est renvoyé par l'information restrictive pour cette instruction sera une chaîne de caractères contenant les caractères de l'identificateur (ou la valeur **true** ou **false** dans le cas de types booléens). Pour les types entiers avec des valeurs nommées, soit les identificateurs soit les nombres correspondants seront employés (en fonction du choix du codeur).

31.3.3 Si le type est un type bitstring avec des bits nommés et que le texte "TextToBeUsed" est absent, une chaîne de caractères identique à l'identificateur du bit représentera celui-ci lorsqu'il est fixé. Chacune des valeurs abstraites sera codée comme la concaténation (éventuellement vide) de ces chaînes de caractères pour tous les bits fixés, séparées par des "blancs avec échappement" (voir le § 8.1.5).

31.3.4 Si le type n'est pas un type bitstring avec des bits nommés et que le texte "TextToBeUsed" est présent, alors les sous-paragraphes suivants s'appliquent (mais voir le § 31.3.5).

31.3.4.1 Si la variante du nom "NewName" est employée, la chaîne de caractères employée pour coder la valeur identifiée par l'information "QualifyingInformation" est le nom "NewName". Chaque occurrence des caractères "**<**", "**>**", et "**&**" dans le nom "NewName" sera remplacée soit par une des séquences d'échappement "**<**", "**>**", et "**&**" respectivement, soit par une séquence d'échappement de la forme "**&#n;**" ou "**&#xn;**", comme spécifié dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, au § 11.15.8.

31.3.4.2 Si la variante du mot-clé "Keyword" est employée, la chaîne de caractères servant à coder les valeurs du type est le nom de l'identificateur, modifié comme spécifié ci-après.

31.3.4.3 Si le mot-clé "Keyword" est **CAPITALIZED**, alors le premier caractère du nom est remplacé par l'équivalent majuscule, le nom restant autrement inchangé.

31.3.4.4 Si le mot-clé "Keyword" est **UNCAPITALIZED**, alors le nom est inchangé.

31.3.4.5 Si le mot-clé "Keyword" est **UPPERCASED**, alors tous les caractères du nom qui sont des lettres minuscules sont remplacés par leurs équivalents majuscules. Les autres caractères restent inchangés.

31.3.4.6 Si le mot-clé "Keyword" est **LOWERCASED**, alors tous les caractères du nom qui sont des lettres majuscules sont remplacés par leurs équivalents minuscules. Les autres caractères restent inchangés.

31.3.5 Si le type est un type entier avec des valeurs nommées, les chaînes de caractères produites en application des § 31.3.4.1 à 31.3.4.6 seront employées au lieu des identificateurs. Soit les chaînes des caractères, soit les nombres correspondants seront employés (en fonction du choix du codeur).

31.3.6 Si le type est un type bitstring avec des bits nommés et que le texte "TextToBeUsed" est présent, les § 31.3.4.1 à 31.3.4.6 s'appliqueront à chaque identificateur binaire pour produire la chaîne de caractères qui représente le bit lorsque celui-ci est fixé. La valeur bitstring sera ensuite codée comme la concaténation (éventuellement vide) de ces chaînes de caractères pour tous les bits fixés, séparées par des "blancs avec échappement".

32 Instruction de codage UNTAGGED

32.1 Généralités

32.1.1 L'instruction "UntaggedInstruction" est la suivante:

UntaggedInstruction ::=
UNTAGGED
TargetList

32.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

32.1.3 (A des fins didactiques) Une description informelle de l'incidence de l'instruction **UNTAGGED** pour les constructeurs en notation ASN.1 figure à l'Annexe B. Il y est donné une introduction didactique illustrant certains des effets de l'emploi de l'instruction **UNTAGGED**.

32.1.4 Employée (éventuellement de façon répétée et imbriquée) avec les types séquence, ensemble, choix, sequence-of, et set-of, cette instruction permet de spécifier une configuration presque arbitraire des éléments XML. Elle a pour effet d'enlever l'étiquette de début XML qui précède la valeur "ExtendedXMLValue" du "Type" auquel elle s'applique et l'étiquette de fin XML qui la suit, de sorte que les éléments XML normalement compris entre ces étiquettes constituent un contenu XML partiel.

32.1.5 Appliquée au type choix en tant que composante d'une séquence ou d'un ensemble, cette instruction spécifie l'insertion en ce point dans la séquence (ou l'ensemble) d'une variante précise du type choix (ou d'aucune variante si le type choix est une composante **OPTIONAL**). L'identificateur du type choix ne figure pas dans le codage. Certaines

variantes du type choix peuvent être des éléments XML, tandis que d'autres peuvent posséder, après l'emploi de l'instruction **UNTAGGED** dans la définition de ces variantes, un contenu XML partiel qui consiste en presque toute la configuration arbitraire des multiples éléments.

32.1.6 Appliquée au type sequence-of en tant que composante d'une séquence ou d'un ensemble, cette instruction spécifie l'insertion en ce point dans la séquence (ou l'ensemble) d'un nombre donné ou arbitraire de répétitions de la composante sequence-of (qui peut produire un unique élément XML, ou un contenu XML partiel si elle est elle-même soumise à l'instruction **UNTAGGED**).

32.1.7 Appliquée à un type séquence (ou ensemble) ou à un type sequence-of (ou set-of) en tant que variante d'un type choix, cette instruction permet à cette variante de posséder un contenu XML partiel qui consiste en la valeur "ExtendedXMLValue" de la séquence, de l'ensemble, de sequence-of ou de set-of.

32.1.8 Une autre fonction de l'instruction **UNTAGGED,** lorsqu'elle est appliquée à un type susceptible d'être codé au moyen de caractères, est de permettre au contenu en caractères de figurer dans le codage d'une séquence, sans que des étiquettes n'entourent le contenu. Cet emploi est limité à une composante d'une séquence qui ne possède elle-même pas d'étiquettes.

NOTE – Cette restriction vise à simplifier les règles nécessaires qui assurent un codage facile et non ambigu.

32.2 Restrictions

32.2.1 Lors de toutes les utilisations, le type contenant sera un type séquence, ensemble, choix, sequence-of, ou set-of.

32.2.2 Si le type est susceptible d'être codé au moyen de caractères, le type contenant sera un type séquence sans instruction finale de codage **UNTAGGED**. Le type ne sera pas marqué comme étant **OPTIONAL** ou **DEFAULT**. Toutes les autres composantes du type séquence contenant (si elles existent) comporteront une instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES**.

32.2.3 Si le type n'est pas un type susceptible d'être codé au moyen de caractères, il sera un type séquence, ensemble, choix, sequence-of, set-of, octetstring ou bitstring avec un "Type" contenu sans indication **ENCODED BY**, ou un type ouvert.

NOTE – A l'Annexe B sont données des directives qui garantissent qu'aucune ambiguïté ne découle de l'emploi de cette instruction de codage.

32.2.4 Cette instruction de codage ne sera pas appliquée à un type dont le codage de la valeur "ExtendedXMLValue" est vide pour l'une de ses valeurs abstraites, si le type est employé comme:

a) composante d'un type séquence ou ensemble avec l'indication **OPTIONAL** ou **DEFAULT**; ou

b) composante d'un type sequence-of ou set-of; ou

c) variante d'un type choix, si pour une autre variante du même type choix le codage de la valeur "ExtendedXMLValue" est vide pour l'une de ses valeurs abstraites et l'instruction finale de codage **UNTAGGED** est présente.

EXEMPLE: Un type, qui est un type séquence dont toutes les composantes sont indiquées comme étant **OPTIONAL,** a une valeur abstraite avec un codage vide de la valeur "ExtendedXMLValue", comme le type sequence-of où aucune répétition n'est autorisée.

32.2.5 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

32.2.6 Un type comportant cette instruction finale de codage ne comportera aucune des instructions finales de codage **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **ATTRIBUTE**, **DEFAULT-FOR-EMPTY**, **EMBED-VALUES**, **PI-OR-COMMENT**, **USE-NIL**, **USE-ORDER** ou **USE-TYPE**.

32.2.7 Aucune information restrictive ne figurera dans la liste "TargetList".

32.3 Incidence sur les codages

32.3.1 Si le type est codé comme un type de haut niveau, il ne sera pas tenu compte de cette instruction de codage.

32.3.2 Si le type contenant est un type choix, la valeur "ExtendedXMLChoiceValue" (voir le § 17.5.1) pour cette variante du type contenu sera la valeur "ExtendedXMLValue" de la variante (la deuxième variante de la production de la valeur "ExtendedXMLChoiceValue").

NOTE – Cette valeur "ExtendedXMLValue" pour la variante peut correspondre à un seul élément XML ou à un contenu partiel XML comportant plusieurs éléments XML.

32.3.3 Si le type contenant est un type séquence ou ensemble, la valeur "ExtendedXMLNamedValue" (voir le § 17.6) pour cette composante du type contenu sera remplacée par la valeur "ExtendedXMLValue" de la composante (la deuxième variante de la production de la valeur "ExtendedXMLNamedValue").

NOTE – Cette valeur "ExtendedXMLValue" peut correspondre à un seul élément XML ou à un contenu partiel XML comportant plusieurs éléments XML.

32.3.4 Si le type contenant est un type sequence-of ou set-of, l'unité lexicale "ExtendedXMLDelimitedItem" (si elle est employée – voir le § 17.7) de chacune des répétitions sera remplacée par la valeur "ExtendedXMLValue" contenue dans l'unité lexicale "ExtendedXMLDelimitedItem".

NOTE 1 – Il n'est pas possible d'employer l'instruction **UNTAGGED** à moins que l'instruction **GLOBAL-DEFAULTS** du codage **MODIFIED-ENCODINGS** n'ait été introduite dans la section de commande de codage, auquel cas la liste "ExtendedXMLValueList" n'est pas autorisée (voir le § 17.7.2).

NOTE 2 – Cette valeur "ExtendedXMLValue" peut correspondre à un seul élément XML ou à un contenu partiel XML comportant plusieurs éléments XML.

32.3.5 Si le type est un type octetstring ou bitstring avec un "Type" contenu sans indication **ENCODED BY**, ou un type ouvert, la valeur "ExtendedXMLValue" sera une valeur "ExtendedXMLTypedValue" (pas une chaîne "xmlhstring" ou "XMLBase64String").

NOTE – De tels types ne satisfont pas à la définition du type susceptible d'être codé au moyen de caractères (voir le § 3.2.2 *ter*). Le § 32.3.5 implique que lorsqu'ils comportent une instruction finale de codage **UNTAGGED,** ils sont toujours codés comme des éléments XML.

33 Instruction de codage USE-NIL

33.1 Généralités

33.1.1 L'instruction "UseNilInstruction" est la suivante:

UseNilInstruction ::=
USE-NIL
TargetList

33.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

33.1.3 Cette instruction de codage assure un codage EXTENDED-XER optimisé pour une séquence avec une seule composante **OPTIONAL** dont les autres composantes (s'il en existe) comportent toutes une instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES**, éventuellement précédée d'un type initial sequence-of prenant en charge l'instruction **USE-ORDER** (voir le § 35).

33.1.4 En l'absence de cette instruction de codage, la composante en option sera codée comme suit:

a) (cas "non manquant mais vide") si la composante figure dans la valeur abstraite, dont le codage de la valeur "ExtendedXMLValue" est vide, une valeur "ExtendedXMLNamedValue" pour la composante figure dans le document XML, habituellement sous la forme d'une étiquette d'élément vide (ou avec des étiquettes adjacentes de début et de fin);

b) (cas "manquant") si la composante ne figure pas dans la valeur abstraite, la valeur "ExtendedXMLNamedValue" n'est pas présente;

c) (cas "non manquant et non vide") si la composante figure dans la valeur abstraite, dont le codage n'est pas vide, une valeur "ExtendedXMLNamedValue" de contenu non vide pour la composante est présente.

33.1.5 L'emploi de l'instruction **USE-NIL** exige que l'absence de la composante en option (cas b) ci-dessus) soit signalée par l'introduction d'un attribut d'identification de valeur nulle, de nom "**nil**" et de valeur "**true**" ou "**1**".

33.1.6 Dans les cas a) et c) du § 33.1.4, l'attribut d'identification de valeur nulle peut être soit omis (en fonction du choix du codeur), soit présent avec une valeur "**false**" ou "**0**". La composante en option sera codée en omettant les étiquettes associées.

33.2 Restrictions

33.2.1 L'instruction de codage **USE-NIL** ne sera attribuée qu'à un type séquence qui comporte une composante **OPTIONAL** sans instruction finale de codage **ATTRIBUTE**. Toutes les autres composantes du type séquence, si elles existent, comportent une instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES** ou seront les composantes sequence-of prenant en charge une instruction de codage **USE-ORDER** ou **EMBED-VALUES,** qui sont aussi des instructions finales de codage appliquées au type séquence.

33.2.2 Le type séquence ne comportera pas d'instruction finale de codage **UNTAGGED**.

33.2.3 La composante **OPTIONAL** ne contiendra aucune des instructions finales de codage **ANY-ELEMENT**, **ANY-ATTRIBUTES**, **DEFAULT-FOR-EMPTY**, **EMBED-VALUES**, **PI-OR-COMMENT**, **UNTAGGED**, **USE-NIL**, **USE-ORDER** ou **USE-TYPE**.

NOTE – A l'exception de l'instruction **UNTAGGED**, les instructions de codage énumérées ci-dessus ne peuvent s'appliquer à un type qui comporte une instruction finale de codage **UNTAGGED**.

33.2.4 Si la composante **OPTIONAL** n'est pas un type susceptible d'être codé au moyen de caractères, alors elle sera un type séquence, ensemble, choix, sequence-of, set-of, un type ouvert, ou un type octetstring ou bitstring avec un "Type" contenu et sans instruction **ENCODED BY**.

33.2.5 Si la composante **OPTIONAL** est un type séquence, aucune de ses composantes ne comportera d'instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES**.

33.2.6 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction finale de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

33.2.7 Un type comportant cette instruction finale de codage ne contiendra pas en même temps l'instruction finale de codage **UNTAGGED** ou **USE-QNAME**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **ATTRIBUTE**, **BASE64**, **DECIMAL**, **LIST**, **TEXT**, **USE-NUMBER**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

33.2.8 Aucune information restrictive ne figurera dans la liste "TargetList".

33.3 Incidence sur les codages

33.3.1 Si la composante **OPTIONAL** est absente (cas b) du § 33.1.4), alors un attribut d'identification de valeur nulle dénommé "**nil**" et ayant une valeur "**true**" ou "**1**" sera ajouté à la liste "AttributeList" de l'élément contenant.

33.3.2 Si la composante **OPTIONAL** est présente (cas a) et c) du § 33.1.4), l'attribut d'identification de valeur nulle peut être soit omis (en fonction du choix du codeur), soit ajouté à la liste "AttributeList" de l'élément contenant avec une valeur "**false**" ou "**0**". La composante en option sera codée en omettant les étiquettes associées.

34 Instruction de codage USE-NUMBER

34.1 Généralités

34.1.1 L'instruction "UseNumberInstruction" est la suivante:

UseNumberInstruction ::=
USE-NUMBER
TargetList

34.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

34.1.3 La présente instruction de codage vise à modifier le codage d'un type énuméré de manière que les nombres dans les énumérations "NamedNumber" soient employés au lieu des noms.

34.2 Restrictions

34.2.1 Il ne sera pas tenu compte de cette instruction de codage à moins qu'elle ne soit appliquée à un type énuméré.

34.2.2 Un type comportant cette instruction finale de codage ne contiendra pas en même temps une instruction finale de codage **TEXT**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **BASE64**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **USE-NIL**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

34.2.3 Aucune information restrictive ne figurera dans la liste "TargetList".

34.3 Incidence sur les codages

34.3.1 La valeur "ExtendedXMLEnumeratedValue" est la suivante:

ExtendedXMLEnumeratedValue ::=
EmptyElementEnumerated
| TextEnumerated
| XMLSignedNumber

34.3.2 Les entités énumérées "EmptyElementEnumerated" et "TextEnumerated" sont définies dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 19.8 et 19.9.

34.3.3 Le nombre "XMLSignedNumber" défini dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, § 18.9 et 18.12, sera le nombre dans l'entité "NamedNumber" de l'énumération.

34.3.4 La variante du nombre XMLSignedNumber sera employée si et seulement si le type énuméré comporte cette instruction finale de codage.

NOTE – Si une instruction **GLOBAL-DEFAULTS** du codage **MODIFIED-ENCODINGS** figure dans la section de commande de codage XER mais que le type énuméré ne comporte pas cette instruction de codage, la deuxième variante est employée. Si cette instruction ne figure pas dans la section de commande de codage XER, la première variante est employée.

35 Instruction de codage USE-ORDER

35.1 Généralités

35.1.1 L'instruction "UseOrderInstruction" est la suivante:

UseOrderInstruction ::=
USE-ORDER
TargetList

35.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

35.1.3 Cette instruction de codage vise à assurer un codage EXTENDED-XER optimisé d'un type séquence comportant une composante sequence-of qui détermine l'ordre sémantique des valeurs des composantes suivantes du type séquence, codées comme des éléments. Elle peut aussi être employée, lorsqu'une instruction finale de codage **USE-NIL** est aussi présente (voir le § 33), et que la seule composante **OPTIONAL** exigée par l'emploi de l'instruction **USE-NIL** est une séquence, pour déterminer l'ordre sémantique des composantes de cette séquence **OPTIONAL**.

35.1.4 La composante sequence-of qui permet de déterminer l'ordre sémantique est la première composante de la séquence, à moins qu'il n'y ait aussi une composante sequence-of prenant en charge une instruction finale de codage **EMBED-VALUES** appliquée au type séquence. Dans ce cas, la composante sequence-of prenant en charge l'instruction de codage **EMBED-VALUES** précède la composante sequence-of prenant en charge l'instruction de codage **USE-ORDER**.

35.1.5 La composante permettant de déterminer l'ordre doit être un type sequence-of avec une composante qui est un type énuméré. Ce type sequence-of et sa sémantique dépendent de la présence ou de l'absence d'une instruction de codage **USE-NIL** appliquée au type séquence, comme décrit dans les sous-paragraphes suivants.

35.1.5.1 En l'absence d'une instruction finale de codage **USE-NIL**, les noms des énumérations sont identiques aux identificateurs ASN.1 des composantes du type séquence. L'ordre des énumérations dans chacune des valeurs abstraites détermine l'ordre sémantique des valeurs des composantes suivantes du type séquence, qui sont présentes dans le codage.

35.1.5.2 Lorsqu'une instruction finale de codage **USE-NIL** est aussi présente, la composante **OPTIONAL** exigée par l'emploi de l'instruction **USE-NIL** doit être un type séquence (par exemple le type B), et les noms des énumérations sont identiques aux identificateurs ASN.1 des composantes du type séquence B. L'ordre des énumérations dans chacune des valeurs abstraites détermine l'ordre sémantique des valeurs des composantes du type séquence B, qui sont présentes dans le codage.

35.2 Restrictions

35.2.1 Cette instruction de codage ne sera attribuée qu'au type séquence. Ce type séquence contiendra une composante qui est un type sequence-of (par exemple le type A) avec une composante qui est un type énuméré. Si le type séquence ne comporte pas non plus d'instruction finale de codage **EMBED-VALUES**, alors le type A sera la première composante, sinon il sera la deuxième composante. S'il n'existe pas d'instruction finale de codage **USE-NIL**, le type séquence aura au moins aussi une autre composante sans instruction finale de codage **ATTRIBUTE** ou **ANY-ATTRIBUTES** (une composante non attribut). S'il existe une instruction finale de codage **USE-NIL**, la composante **OPTIONAL** prenant en charge cette instruction **USE-NIL** sera un type séquence, et elle aura au moins une composante.

35.2.2 Le type énuméré comportera des identificateurs qui dépendent de la présence ou de l'absence d'une instruction finale de codage **USE-NIL** appliquée au type séquence avec une instruction de codage **USE-ORDER**, comme spécifié dans les sous-paragraphes suivants.

35.2.2.1 S'il n'existe pas d'instruction finale de codage **USE-NIL**, alors le type énuméré possédera des identificateurs pour les énumérations, qui sont en correspondance bi-univoque (et sont dans le même ordre textuel) avec les identificateurs des composantes non-attributs suivantes (voir le § 35.2.1) de la séquence. Le type sequence-of sera

soumis à des contraintes de manière que chaque valeur abstraite contienne exactement un identificateur pour chaque composante non-attribut de la séquence, qui figure dans la valeur abstraite.

35.2.2.2 S'il existe une instruction finale de codage **USE-NIL**, alors le type énuméré possédera des identificateurs pour les énumérations, qui sont en correspondance bi-univoque avec les identificateurs des composantes de la composante **OPTIONAL** dans le type séquence. Le type sequence-of sera soumis à des contraintes de manière que chaque valeur abstraite contienne exactement un identificateur pour chaque composante de la séquence **OPTIONAL,** qui figure dans la valeur abstraite.

NOTE – Il est recommandé que la contrainte sur le type séquence s'exprime comme suit:

(CONSTRAINED BY {/* *Doit être conforme à la Rec. UIT-T X.693 \| ISO/CEI 8825-4, § 35* ***/})**

35.2.2.3 Les unités lexicales "EnumerationItem" dans les énumérations seront les identificateurs "identifier" ou les nombres "NamedNumber" de valeur 0 pour la première unité lexicale "EnumerationItem", 1 pour la deuxième, et ainsi de suite jusqu'à la dernière unité lexicale "EnumerationItem".

35.2.3 Le type sequence-of ne sera pas marqué comme étant **OPTIONAL** ou **DEFAULT**.

35.2.4 Les composantes suivantes de la séquence (s'il n'existe pas d'instruction finale de codage **USE-NIL**), et les composantes de la séquence **OPTIONAL** (s'il existe une instruction finale de codage **USE-NIL**) ne seront pas marquées comme étant **DEFAULT** à moins qu'elles ne comportent une instruction finale de codage **ATTRIBUTE**.

35.2.5 Quelle que soit la séquence, aucune de leurs composantes comportant cette instruction finale de codage ou la séquence **OPTIONAL** (lorsqu'une instruction finale de codage **USE-NIL** est présente) n'inclura d'instruction finale de codage **UNTAGGED**, que le type de cette composante soit susceptible d'être codé au moyen de caractères ou non.

35.2.6 Aucune composante de la séquence comportant cette instruction finale de codage n'inclura d'instruction finale de codage **ANY-ELEMENT**.

35.2.7 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** figure dans la section de commande de codage.

35.2.8 Un type comportant cette instruction finale de codage n'inclura pas non plus une instruction finale de codage **UNTAGGED**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **ATTRIBUTE**, **BASE64**, **DECIMAL**, **LIST**, **TEXT**, **USE-NUMBER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

35.2.9 Aucune information restrictive ne figurera dans la liste "TargetList".

35.3 Incidence sur les codages

35.3.1 Le type sequence-of possédant une composante énumérée ne sera pas codé directement.

35.3.2 Un codeur codera la sémantique de ce type (l'ordre sémantique des composantes de la séquence ou des composantes de la séquence **OPTIONAL**) en effectuant le codage des composantes qui sont codées comme des éléments dans l'ordre spécifié par le type sequence-of comportant la composante énumérée. Un décodeur retrouvera la valeur de la composante sequence-of en employant l'ordre des éléments codés.

36 Instruction de codage USE-QNAME

36.1 Généralités

36.1.1 L'instruction "UseQNameInstruction" est la suivante:

UseQNameInstruction ::=
USE-QNAME
TargetList

36.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

36.1.3 La présente instruction de codage vise à modifier le codage d'un type séquence, chacune de ses valeurs spécifiant un nom d'espace de nom en option (un identificateur URI) et un nom non préfixé, de manière à coder un nom restrictif et non restrictif d'espace de nom XML.

NOTE – Cette spécification a été faite parce qu'elle est prévue dans d'autres notations de schéma. A titre d'exemple, un type séquence auquel elle pourrait s'appliquer est le type **QName** défini dans la Rec. UIT-T X.694 | ISO/CEI 8825-5.

36.1.4 Si la composante en option figure dans la valeur abstraite du type séquence, alors cette valeur abstraite correspond au nom restrictif d'espace de nom. Si la composante n'y figure pas, le type séquence correspond à un nom non restrictif.

36.2 Restrictions

36.2.1 Cette instruction de codage ne sera attribuée qu'à une séquence comportant exactement deux composantes, toutes deux du type **UTF8String**. La première composante sera indiquée comme étant **OPTIONAL**.

36.2.2 La première composante sera contrainte de représenter un identificateur URI (voir la norme IETF RFC 2396). La deuxième composante sera contrainte de contenir un nom "NCName" comme spécifié dans les espaces de nom XML du Consortium W3C, au § 2, dans la production 4, et ne commencera pas par des caractères qui, en majuscules, sont les caractères "**XML**".

36.2.3 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

36.2.4 Un type comportant cette instruction finale de codage ne contiendra pas en même temps une instruction finale de codage **USE-NIL**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **BASE64**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **TEXT**, **USE-NUMBER**, **USE-ORDER**, **USE-TYPE**, **USE-UNION**, **WHITESPACE**.

36.2.5 Aucune information restrictive ne figurera dans la liste "TargetList".

36.3 Incidence sur les codages

36.3.1 L'incidence de la présence de cette instruction de codage sur un type, si la composante en option est présente, exige qu'une déclaration d'espace de nom (ou déclaration d'espace de nom par défaut) s'applique à la valeur de l'attribut ou au contenu de l'élément qui code la valeur de ce type, conformément au § 29. La valeur de l'attribut ou le contenu de l'élément est ensuite codé comme spécifié dans le § 29 pour un nom restrictif d'espace de nom.

36.3.2 Si la composante en option est absente, une déclaration d'espace de nom par défaut ne s'appliquera pas à la valeur de l'attribut ou au contenu de l'élément qui code la valeur de ce type.

37 Instruction de codage USE-TYPE

37.1 Généralités

37.1.1 L'instruction de codage "UseTypeInstruction" est la suivante:

UseTypeInstruction ::=
USE-TYPE
TargetList

37.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

37.1.3 Cette instruction de codage optimise le codage EXTENDED-XER d'un type choix. Elle exige qu'un attribut d'identification du type soit codé dans un élément contenant pour identifier la variante qui a été codée (à moins que ce ne soit la première variante) et l'enlèvement des étiquettes de début et de fin entourant le codage des variantes.

37.1.4 L'attribut d'identification du type identifie le type d'un élément XML. Le nom de l'attribut doit être le nom "**type**" provenant de l'espace de nom de commande (voir le § 16.9) et sa valeur identifie une variante du type choix à laquelle cette instruction de codage s'applique (elle fournit une variante pour la détermination du type choix).

37.2 Restrictions

37.2.1 Le type auquel l'instruction **USE-TYPE** est attribuée sera un type choix sans instruction finale de codage **UNTAGGED**.

37.2.2 Aucune des variantes du type choix ne comportera d'instruction finale de codage **UNTAGGED**.

37.2.3 Aucune des variantes du type choix sera elle-même un type choix comportant une instruction finale de codage **USE-TYPE**.

NOTE – Une ou plusieurs variantes du type choix peuvent être des types choix comportant une instruction finale de codage **USE-UNION**.

37.2.4 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

37.2.5 Un type comportant cette instruction finale de codage ne contiendra pas en même temps les instructions finales de codage **UNTAGGED** ou **USE-UNION**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **ATTRIBUTE**, **BASE64**, **DECIMAL**, **DEFAULT-FOR-EMPTY**, **EMBED-VALUES**, **LIST**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **WHITESPACE**.

37.2.6 Aucune information restrictive ne figurera dans la liste "TargetList".

37.3 Incidence sur les codages

37.3.1 Si la variante du choix codé n'est pas la première variante, alors un attribut d'identification du type (voir les § 37.3.3 et 37.3.4) sera ajouté à la liste "AttributeList" de l'élément contenant, à moins que le § 37.3.8 ne s'applique.

37.3.2 Si la variante du choix codé est la première variante, l'attribut d'identification du type peut être ajouté ou omis, en fonction du choix du codeur, à moins que le § 37.3.8 ne s'applique.

37.3.3 L'attribut d'identification du type sera une instance de la production de l'"Attribute" (voir le § 20.3.3) avec un nom "ControlAttributeName" restrictif de l'espace de nom (voir le § 20.3.5) d'un "**type**" provenant de l'espace de nom de commande (voir le § 16.9).

37.3.4 La valeur de l'attribut d'identification du type sera l'identificateur de la variante choisie, éventuellement modifiée conformément aux instructions finales de codage **NAME** et **NAMESPACE**.

37.3.5 Si aucun attribut d'identification du type ne figure dans le codage d'un type comportant cette instruction finale de codage, le décodeur supposera que la première variante du choix est présente.

37.3.6 La présence d'un attribut d'identification du type avec une valeur imprévue ne doit pas impliquer d'erreur de décodage. Lorsqu'un décodeur observe un tel attribut au cours du décodage, il supposera que la première variante du choix a été identifiée, et peut négliger l'attribut d'identification du type (ou le transmettre à l'application). En outre, dans ces cas, le décodeur peut négliger (ou transmettre à l'application) tout autre attribut imprévu et tout élément descendant imprévu rencontré après tous les éléments descendants prévus dans la valeur "ExtendedXMLValue" de la variante.

37.3.7 Tous les "Attribute" qui figureraient autrement dans la liste "AttributeList" des valeurs "ExtendedXMLChoiceValue" seront ajoutés à la liste "AttributeList" de l'élément contenant et la valeur "ExtendedXMLChoiceValue" du type choix sera remplacée par la valeur "ExtendedXMLValue" dans la valeur "ExtendedXMLChoiceValue".

37.3.8 Si une ou plusieurs variantes du type choix comportant l'instruction finale de codage **USE-TYPE** sont des types choix comportant une instruction finale de codage **USE-UNION,** l'attribut d'identification du type peut, en fonction du choix du codeur, identifier l'une des variantes du type choix comportant l'instruction finale **USE-UNION** au lieu d'une variante du type choix comportant l'instruction finale de codage **USE-TYPE**.

38 Instruction de codage USE-UNION

38.1 Généralités

38.1.1 L'instruction "UseUnionInstruction" est la suivante:

UseUnionInstruction ::=
USE-UNION
TargetList

38.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

38.1.3 Cette instruction de codage optimise le codage d'un type choix dans les cas où le codage des valeurs abstraites de chaque variante diffère suffisamment du codage des valeurs abstraites des autres variantes pour que le décodeur puisse déterminer la valeur abstraite indiquée par l'analyse du codage.

38.1.4 Si le type choix comportant une instruction finale de codage **USE-UNION** ne comporte pas en même temps une instruction finale de codage **ATTRIBUTE** ou **UNTAGGED**, alors cette instruction de codage peut impliquer l'insertion d'un attribut d'identification du type dans l'élément contenant pour identifier la variante qui a été codée. Si le type choix comporte une instruction finale de codage **ATTRIBUTE** ou **UNTAGGED**, ou est la composante d'un type sequence-of ou set-of comportant une instruction de codage **LIST**, l'insertion de l'attribut d'identification du type n'est pas possible.

38.1.5 Cette instruction de codage implique l'enlèvement des étiquettes de début et de fin entourant le codage de la variante.

38.2 Restrictions

38.2.1 Un type comportant une instruction finale de codage **USE-UNION** sera un type choix.

38.2.2 Toutes les variantes du type choix seront des types susceptibles d'être codés au moyen de caractères, mais ne seront pas des types choix comportant une instruction finale de codage **USE-UNION**.

38.2.3 Si le type choix inclut une instruction finale de codage **ATTRIBUTE** ou **UNTAGGED** ou est employé dans une définition du type comme une composante d'un type sequence-of ou set-of comportant une instruction finale de codage **LIST**, les variantes du type choix seront limitées de manière que, pour chacune d'elles, toutes les valeurs abstraites comportent au moins un codage (sa valeur "ExtendedXMLValue") qui diffère de l'ensemble des codages admis de toutes les variantes contextuellement antérieures.

NOTE – Cette prescription est imposée parce qu'il est impossible d'insérer un attribut d'identification du type déterminant la variante qui a été choisie. Sans cette prescription, le codage serait ambigu.

38.2.4 Dans les deux paragraphes suivants, le terme identificateur "identifier" veut dire: identificateur (éventuellement modifié conformément à une quelconque instruction finale de codage **NAME** ou **NAMESPACE**) d'une variante (du type choix).

38.2.5 Si le type choix (par exemple, le type U) est codé comme une variante d'un type choix contenant (par exemple, le type E) qui comporte une instruction finale de codage **USE-TYPE** et que l'identificateur de l'une des variantes du type E est identique à l'identificateur de l'une des variantes du type U, alors chacune des valeurs abstraites de cette variante du type U possédera au moins un codage qui diffère de tous les codages des variantes contextuellement antérieures du type U.

NOTE – Cette prescription est imposée parce que, dans ce cas, il n'est pas possible d'identifier la variante du type U, l'identificateur dans un attribut d'identification du type E identifiant simplement l'ensemble du type U.

38.2.6 Si le type choix (par exemple, le type U1) est codé comme une variante d'un type choix contenant (par exemple, le type E) qui comporte une instruction finale de codage **USE-TYPE** et que le type E contient un autre type choix (par exemple, le type U2) comportant une instruction de codage **USE-UNION** qui suit contextuellement le type U1 dans le type E, et que l'identificateur de chacune des variantes du type U2 est identique à l'un des identificateurs dans le type U1, alors chaque valeur abstraite de cette variante du type U2 possédera au moins un codage qui diffère de l'ensemble des codages de toutes les variantes du type U1.

NOTE – Cette prescription est imposée parce que, dans ce cas, il n'est pas possible d'identifier la variante du type U2, l'identificateur dans un attribut d'identification du type E identifiant la variante dans le type U1.

38.2.7 Cette instruction de codage ne sera pas attribuée à moins qu'une instruction de codage **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** ne figure dans la section de commande de codage.

38.2.8 Un type comportant cette instruction finale de codage ne contiendra pas en même temps une instruction finale de codage **USE-TYPE**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **ANY-ELEMENT**, **BASE64**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **WHITESPACE**.

38.2.9 Aucune information restrictive ne figurera dans la liste "TargetList".

38.3 Incidence sur les codages

38.3.1 Si le type choix ne comporte pas d'instruction finale de codage **ATTRIBUTE** ou **UNTAGGED** et n'est pas codé comme la composante d'un type sequence-of ou set-of comportant une instruction finale de codage **LIST**, alors un attribut d'identification du type peut être ajouté, en fonction du choix du codeur, à la liste "AttributeList" de l'élément contenant (mais voir le § 38.3.8).

NOTE – Si le type choix est codé comme une variante d'un choix comportant une instruction de codage **USE-TYPE,** l'attribut d'identification du type spécifié par l'instruction de codage **USE-UNION** peut être employé au lieu de l'attribut d'identification du type spécifié par l'instruction de codage **USE-TYPE** (voir le § 37.3.8).

38.3.2 Si chacun des codages éventuels de la valeur abstraite codée est identique à l'un des codages d'une valeur abstraite d'une variante contextuellement antérieure, alors un attribut d'identification du type sera ajouté.

NOTE – Ce § 38.3.2 supprime l'option pour le codeur prévue dans le § 38.3.1 et rend obligatoire l'adjonction de l'attribut d'identification du type. Les restrictions spécifiées dans les § 38.2.4 à 38.2.6 assurent que cela ne peut se produire que lorsque le type choix est codé comme un élément et qu'aucune ambiguïté due aux identificateurs identiques n'est possible.

38.3.3 Si le type choix comporte une instruction finale de codage **ATTRIBUTE** ou **UNTAGGED** ou que son type contenant est un type sequence-of ou set-of comportant une instruction finale de codage **LIST**, aucun attribut d'identification du type ne peut être inséré dans un quelconque élément. Dans le cas des scénarios décrits dans les § 38.2.4 à 38.2.6, un attribut d'identification du type ne peut être inséré pour identifier précisément certaines des variantes des types U ou U2. Les décodeurs se fonderont donc sur les conditions des § 38.2.4 à 38.2.6 pour déterminer la valeur abstraite qui a été codée.

NOTE – Ces règles impliquent qu'un décodeur est nécessaire, en l'absence d'une identification du type (ou en présence d'une identification ambiguë), pour tenter le décodage de la variante contextuellement première, puis de la suivante, et ainsi de suite, en acceptant le premier décodage réussi rencontré (ou en diagnostiquant une erreur si aucun décodage n'est réussi).

38.3.4 L'attribut d'identification du type sera une instance de la production de l'"Attribute" (voir le § 20.3.3) portant un nom restrictif "ControlAttributeName" d'espace de nom (voir le § 20.3.5) du "**type**" provenant de l'espace de nom de commande (voir le § 16.9).

38.3.5 La valeur de l'attribut d'identification du type sera l'identificateur de la variante choisie, éventuellement modifié conformément aux instructions finales de codage **NAME** et **NAMESPACE**.

38.3.6 Tous les "Attribute" qui figureraient sinon dans la liste "AttributeList" de la valeur "ExtendedXMLChoiceValue" seront ajoutés à la liste "AttributeList" de l'élément contenant et la valeur "ExtendedXMLChoiceValue" du type choix sera remplacée par la valeur "ExtendedXMLValue" dans la valeur "ExtendedXMLChoiceValue".

38.3.7 La valeur "ExtendedXMLValue" du type susceptible d'être codé au moyen de caractères sera l'un des codages qui ne contiennent pas d'étiquettes XML.

NOTE – Cela peut restreindre les choix du codeur.

38.3.8 Si une variante du type choix comporte une instruction finale de codage **NAME AS ""**, aucun attribut d'identification du type ne sera ajouté pour cette variante.

39 Instruction de codage WHITESPACE

39.1 Généralités

39.1.1 L'instruction "WhiteSpaceInstruction" est la suivante:

WhiteSpaceInstruction ::=
WHITESPACE
TargetList
WhiteSpaceAction

WhiteSpaceAction ::=
REPLACE
| **COLLAPSE**

39.1.2 La production de la liste "TargetList" est définie dans le § 14.2.

39.1.3 Cette instruction de codage exige que les décodeurs acceptent des options supplémentaires pour le codage du caractère SPACE (32) et, lors de l'emploi des "blancs avec échappement" avant et arrière (voir le § 8.1.5), pour le codage des chaînes de caractères.

39.2 Restrictions

39.2.1 Cette instruction de codage ne peut être attribuée qu'à un type chaîne de caractères limités qui ne contient pas les caractères suivants, ou est contraint de ne pas les contenir:

a) HORIZONTAL TABULATION (9);

b) LINE FEED (10);

c) CARRIAGE RETURN (13).

39.2.2 Si cette instruction de codage possède l'option **COLLAPSE,** elle ne s'appliquera pas à un type chaîne de caractères limités à moins que ce type ne soit contraint de ne pas avoir de blancs avant ou après ou de contenir des blancs adjacents multiples pour toute valeur abstraite.

NOTE – Il est admis que certains outils ASN.1 peuvent ne pas être en mesure de vérifier statiquement que les restrictions susmentionnées seront satisfaites pour toutes les valeurs abstraites, mais les codeurs réputés conformes ne peuvent produire des codages dans lesquels la valeur "ExtendedXMLValue" viole cette restriction.

39.2.3 Un type comportant cette instruction finale de codage n'inclura pas en même temps une instruction finale de codage **ANY-ELEMENT** ou **BASE64**.

NOTE – Les instructions finales de codage suivantes ne peuvent jamais être présentes en même temps que cette instruction finale de codage parce que leur application au type est interdite: **ANY-ATTRIBUTES**, **DECIMAL**, **EMBED-VALUES**, **LIST**, **TEXT**, **USE-NIL**, **USE-NUMBER**, **USE-ORDER**, **USE-QNAME**, **USE-TYPE**, **USE-UNION**.

39.2.4 Aucune information restrictive ne figurera dans la liste "TargetList".

39.3 Incidence sur les codages

39.3.1 Si le mot-clé **REPLACE** est employé, tout caractère SPACE (32) dans la valeur abstraite peut être remplacé, en fonction du choix du codeur, par un unique caractère "blancs avec échappement" (voir le § 8.1.5).

39.3.2 Si le mot-clé **COLLAPSE** est employé, tout caractère SPACE (32) peut être remplacé, en fonction du choix du codeur, par un nombre quelconque de caractères "blancs avec échappement". Par ailleurs, un ou plusieurs de ces caractères peuvent être ajoutés, en fonction du choix du codeur, au début ou à la fin du codage de la valeur "ExtendedXMLValue".

Remplacer le § 10 existant et ses sous-paragraphes comme suit (en observant qu'il lui a été attribué le nouveau numéro 40):

40 Valeurs des identificateurs d'objet faisant référence aux règles de codage

40.1 Les règles de codage définies dans la présente Recommandation | Norme internationale peuvent être citées en référence et appliquées lorsqu'il est nécessaire de spécifier une représentation de chaîne de caractères non ambiguë pour les valeurs d'un type ASN.1 identifié unique.

40.2 Les valeurs suivantes d'identificateur d'objet et de descripteur d'objet sont attribuées pour identifier les règles de codage définies dans la présente Recommandation | Norme internationale:

Pour les règles BASIC-XER:

```
{joint-iso-itu-t asn1 (1) xml-encoding (5) basic (0)}
"Basic XML encoding of a single ASN.1 type"
```

Pour les règles CXER:

```
{joint-iso-itu-t asn1 (1) xml-encoding (5) canonical (1)}
"Canonical XML encoding of a single ASN.1 type"
```

Pour les règles EXTENDED-XER:

```
{joint-iso-itu-t asn1 (1) xml-encoding (5) extended (2)}
"Extended XML encoding of a single ASN.1 type"
```

Ajouter un nouveau § 40.3 comme suit:

40.3 Les valeurs suivantes d'identificateur d'objet et de descripteur d'objet sont attribuées pour identifier l'espace de nom ASN.1 (voir le § 16.9):

```
asn1Namespace OBJECT IDENTIFIER ::=
            {joint-iso-itu-t asn1 (1) xml-encoding (5) extended (2)
                                        modules (0) support (1) }
            "ASN.1 namespace for EXTENDED-XER support"
```

Remplacer l'Annexe A par le texte suivant:

Annexe A

Exemples de codage selon les règles en langage de balisage extensible de base et canoniques

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

La présente annexe donne un aperçu de l'utilisation des règles de codage XML de base et canoniques définies dans la présente Recommandation | Norme internationale au moyen de représentations de balisage XML d'un dossier personnel (fictif) défini en ASN.1.

A.1 Description en notation ASN.1 de la structure d'un dossier

La structure du dossier personnel (fictif) est décrite de manière formelle au moyen de la notation ASN.1 de la Rec. UIT-T X.680 | ISO/CEI 8824-1. Il s'agit du même exemple que celui de l'Annexe A de la Rec. UIT-T X.690 | ISO/CEI 8825-1.

```
PersonnelRecord ::= [APPLICATION 0] IMPLICIT SET {
            name                    Name,
            title                   [0] VisibleString,
            number                  EmployeeNumber,
            dateOfHire              [1] Date,
            nameOfSpouse            [2] Name,
            children                [3] IMPLICIT
                  SEQUENCE OF ChildInformation DEFAULT {} }
ChildInformation ::= SET
            { name                  Name,
              dateOfBirth           [0] Date}
Name ::= [APPLICATION 1] IMPLICIT SEQUENCE
      {givenName                    VisibleString,
        initial                     VisibleString,
        familyName                  VisibleString}
EmployeeNumber ::= [APPLICATION 2] IMPLICIT INTEGER
Date ::= [APPLICATION 3] IMPLICIT VisibleString  -- YYYYMMDD
```

NOTE – Les étiquettes utilisées dans cet exemple le sont uniquement parce qu'il a été jugé opportun d'utiliser le même exemple que celui qui figurait dans la plus ancienne version de la Rec. UIT-T X.680 | ISO/CEI 8824-1. Elles n'ont pas d'effet sur les codages BASIC-XER, CXER et EXTENDED-XER.

A.2 Description en notation ASN.1 de la valeur d'un dossier

La valeur du dossier personnel de John Smith est décrite ci-dessous de manière formelle en notation de valeur ASN.1:

```
{ 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"}}}
```

A.3 Représentation de cette valeur de dossier en langage XML de base

La représentation de la valeur du dossier ci-dessus (après application des règles de codage XML de base définies dans la présente Recommandation | Norme internationale) est proposée ci-dessous (le prologue est supposé vide).

La longueur du codage BASIC-XER est de 653 octets, en faisant abstraction des "blancs". A titre de comparaison, la même valeur codée selon la variante "UNALIGNED" des règles PER (voir la Rec. UIT-T X.690 | ISO/CEI 8825-1) est de 84 octets; selon la variante "ALIGNED" des règles PER, elle est de 94 octets, selon les règles BER (voir la Rec. UIT-T X.691 | ISO/CEI 8825-2) avec la forme de longueur définie, elle est d'au moins 136 octets et selon les règles BER avec la forme de longueur indéfinie, d'au moins 161 octets.

```
<PersonnelRecord>
      <name>
            <givenName>John</givenName>
            <initial>P</initial>
            <familyName>Smith</familyName>
      </name>
      <title>Director</title>
      <number>51</number>
      <dateOfHire>19710917</dateOfHire>
      <nameOfSpouse>
            <givenName>Mary</givenName>
            <initial>T</initial>
            <familyName>Smith</familyName>
      </nameOfSpouse>
      <children>
            <ChildInformation>
                  <name>
                        <givenName>Ralph</givenName>
                        <initial>T</initial>
                        <familyName>Smith</familyName>
                  </name>
                  <dateOfBirth>19571111</dateOfBirth>
            </ChildInformation>
            <ChildInformation>
                  <name>
                        <givenName>Susan</givenName>
                        <initial>B</initial>
                        <familyName>Jones</familyName>
                  </name>
                  <dateOfBirth>19590717</dateOfBirth>
            </ChildInformation>
      </children>
</PersonnelRecord>
```

A.4 Représentation de cette valeur de dossier en langage XML canonique

La représentation des valeurs ci-dessus (après application des règles canoniques de codage XML définies dans la présente Recommandation | Norme internationale) est la suivante:

```
<PersonnelRecord><name><givenName>John</givenName><initial>P</initial><familyNam
e>Smith</familyName></name><number>51</number><title>Director</title><dateOfHire
>19710917</dateOfHire><nameOfSpouse><givenName>Mary</givenName><initial>T</initi
al><familyName>Smith</familyName></nameOfSpouse><children><ChildInformation><nam
e><givenName>Ralph</givenName><initial>T</initial><familyName>Smith</familyName>
</name><dateOfBirth>19571111</dateOfBirth></ChildInformation><ChildInformation><
name><givenName>Susan</givenName><initial>B</initial><familyName>Jones</familyNa
me></name><dateOfBirth>19590717</dateOfBirth></ChildInformation></children></Per
sonnelRecord>
```

Ajouter la nouvelle Annexe B comme suit:

Annexe B

Contenu partiel en langage de balisage extensible et codage déterministe

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

B.1 Contenu partiel en langage XML

NOTE – La présente annexe décrit la validité lorsque l'instruction **`MODIFIED-ENCODINGS`** est employée.

B.1.1 Les paragraphes suivants décrivent la construction d'un contenu partiel d'élément XML. La section dans son ensemble décrit le contenu partiel d'élément XML, qui est produit en tant que codage, tandis que le § B.2 identifie les restrictions sur le contenu d'élément XML partiel, qui sont nécessaires pour satisfaire aux prescriptions du § 10.2.11. Si une spécification ASN.1 avec des instructions de codage XER ne viole pas ces restrictions, c'est une spécification licite, et les outils peuvent facilement vérifier sa régularité. Si les restrictions **sont** violées, la spécification peut encore ne pas violer les prescriptions normatives du § 10.2.11, mais les outils ne le vérifieront que difficilement dans ce cas.

NOTE – Les restrictions sont conçues pour assurer qu'un décodeur peut facilement et sans ambiguïté retrouver les valeurs abstraites qui ont été employées par le codeur dans la production d'un codage.

B.1.2 Un contenu partiel d'élément XML est constitué d'une combinaison d'éléments XML uniques, fournis par les types **[ELEMENT] SEQUENCE**, **SET**, **SEQUENCE OF**, **SET OF** ou **CHOICE**, et d'un autre contenu partiel d'élément XML fourni par les types **[UNTAGGED] SEQUENCE**, **SET**, **SEQUENCE OF**, **SET OF** ou **CHOICE**.

NOTE – La frontière entre un contenu partiel d'élément XML et un contenu partiel d'élément XML plus grand n'est pas visible dans le codage, mais peut être déterminée à partir du schéma ASN.1 et des restrictions sur les noms des éléments.

B.1.3 Un contenu partiel d'élément XML comporte:

a) soit un seul élément XML;

b) soit un groupe de concaténation, comportant une concaténation ordonnée de zéro, un ou plusieurs contenus partiels d'élément XML, certains d'entre eux pouvant ne pas figurer dans une occurrence de codage (ce qui correspond à l'absence d'une valeur abstraite en option);

NOTE – Le codage d'un type **[UNTAGGED] SEQUENCE** ou **SET** produira en général un groupe d'éléments concaténés.

c) soit un groupe de répétition, comportant la répétition (illimitée et avec contrainte) des contenus partiels d'élément XML (nommés composantes répétées) produits à partir de la composante d'un type **SEQUENCE OF** ou **SET OF**;

NOTE – Le codage d'un type **[UNTAGGED] SEQUENCE OF** ou **SET OF** produira en général un groupe d'éléments répétés.

d) soit un groupe de variantes, consistant en la présence d'un seul contenu partiel d'élément XML choisi dans un ensemble de variantes de contenus partiels d'élément XML (dont une variante précisément figure dans le codage).

NOTE – Le codage d'un type **CHOICE** produit un groupe de variantes. Chacune des variantes du type **CHOICE** produit l'une des variantes de contenus partiels d'élément XML pour ce type **CHOICE**.

B.2 Restrictions recommandées en ce qui concerne les codages produisant des contenus partiels d'élément en langage XML

B.2.1 Aux fins du présent paragraphe seulement, tout groupe de répétition est considéré comme s'il était en option, c'est-à-dire il peut y avoir zéro répétition.

NOTE – La restriction imposant que le groupe de répétition est considéré comme s'il était en option n'est pas strictement nécessaire si des contraintes existent, qui exigent au moins une répétition du type ASN.1 correspondant. Elle est néanmoins introduite pour des raisons de simplicité.

B.2.2 Aux fins du présent paragraphe seulement, la prescription imposant que les noms d'élément soient différents doit être interprétée comme suit:

a) toutes les comparaisons sont faites après l'application d'une quelconque instruction finale de codage **NAME** et **NAMESPACE** sur le type qui a produit le nom;

b) les noms qui sont des noms restrictifs d'espace de nom diffèrent des noms non restrictifs;

c) les noms restrictifs d'espace de nom sont différents si, et seulement si, ils diffèrent par leur nom non préfixé ou leur nom d'espace de nom, ou les deux.

B.2.3 Pour tout contenu partiel d'élément XML résultant, il y a une éventuelle ambiguïté (et donc une violation possible du § 10.2.11) si les conditions spécifiées dans le présent § B.2 ne sont pas satisfaites pour tous les choix possibles de variantes dans un groupe de variantes, pour toutes les options possibles dans un groupe de concaténation, pour toutes les répétitions possibles dans un groupe de répétition, et pour tous les ordres possibles des codages des composantes d'un ensemble.

NOTE – A la lecture et à l'implémentation des paragraphes suivants, le texte ci-dessus mentionnant toutes les possibilités est très important. Ceux qui implémentent les outils permettant de déterminer les spécifications non ambiguës et celles qui ne le sont pas devront analyser toutes les combinaisons possibles de choix, d'options, de répétitions et d'ordre.

B.2.4 (Prescription relative à la délimitation) Deux contenus partiels d'élément XML ne devraient pas être adjacents lorsque le premier élément du deuxième contenu partiel d'élément XML a le même nom que le dernier élément du premier contenu partiel d'élément XML, à moins que le premier contenu partiel d'élément XML ne soit autodélimitant.

EXEMPLE 1: Le contenu partiel d'élément XML produit par un type **[UNTAGGED] SEQUENCE** est autodélimitant s'il ne se termine pas par un élément **OPTIONAL**.

EXEMPLE 2: Le contenu partiel d'élément XML produit par un type **[UNTAGGED] SEQUENCE OF** est autodélimitant s'il comporte un nombre fixe d'itérations, qui sont elles-mêmes auto-délimitantes. Cela veut dire, entre autres, que le type **SEQUENCE OF** [**UNTAGGED**] **SEQUENCE OF INTEGER** est ambigu et viole le § 10.2.11 à moins que le nombre de répétitions du deuxième type **SEQUENCE OF** ne soit fixé.

EXEMPLE 3: Le contenu partiel d'élément XML produit par un type [**UNTAGGED**] **SET** n'est jamais autodélimitant s'il comporte des éléments en option.

B.2.5 (Prescription relative à la détermination de la variante) Les premiers éléments XML de la variante du contenu partiel d'élément XML dans un groupe de variantes devraient avoir des noms d'élément distincts.

NOTE – Dans le texte ci-dessus, il n'est pas tenu compte de l'emploi éventuel des types **USE-TYPE** et **USE-UNION**, qui sort du cadre de la présente annexe.

EXEMPLE 4: Le codage suivant:

```
BadExample1 ::= CHOICE {
      -- First alternative partial XML element content
      alt1  [UNTAGGED] SEQUENCE {
            name       UTF8String,
            zip-code   UTF8String },
      alt2 [UNTAGGED] SEQUENCE {
            name       UTF8String,
            post-code  UTF8String } }
```

n'est en fait pas un codage ambigu EXTENDED-XER (pour un décodeur humain), mais il viole la prescription ci-dessus ainsi que le § 10.2.11. Il s'agit d'un emploi illicite des instructions de codage.

B.2.6 (Prescription relative à la détermination du choix) Les noms du premier élément XML de tous les contenus partiels d'élément XML consécutifs en option ainsi que ceux du contenu partiel d'élément XML obligatoire suivant devraient être distincts.

NOTE – Cela veut dire, entre autres, que tout contenu partiel d'élément XML en option à la fin d'un groupe qui est répété et que tout contenu partiel d'élément XML en option à son début doivent avoir des noms d'élément XML distincts, à moins que le nombre de répétitions est limité à un maximum de 1. Si le contenu partiel entier d'élément XML du groupe qui est répété est en option, leurs noms d'élément XML devraient être distincts.

EXEMPLE 5: Le codage suivant:

```
BadExample2 ::= SEQUENCE OF {
            [UNTAGGED] SEQUENCE {
                  first       [UNTAGGED] CommonInitialParms,
                  second      MainInformation,
                  third       [UNTAGGED] CommonEndParms } }
```

où

```
CommonInitialParms ::= SEQUENCE {  date  GeneralizedTime OPTIONAL,
                                   married BOOLEAN}
CommonEndParms ::=      SEQUENCE { name  UTF8String,
                                   date GeneralizedTime OPTIONAL}
```

viole la prescription relative à la détermination du choix ainsi que le § 10.2.11. Il s'agit d'un emploi illicite des instructions de codage.

B.2.7 (Prescription relative à la détermination du comptage des répétitions) Tous les groupes de répétition possédant un nombre de répétitions qui n'est pas fixé devraient être suivis d'un contenu partiel d'élément XML dont le premier élément XML porte un nom qui est distinct du nom du premier élément XML du contenu partiel d'élément XML répété.

EXEMPLE 6: Le codage suivant:

```
BadExample3 ::=   SEQUENCE {
      required-items    [UNTAGGED] SEQUENCE OF Book,
      optional-items    [UNTAGGED] SEQUENCE OF Book }
```

viole la prescription relative à la détermination du comptage des répétitions ainsi que le § 10.2.11. Il s'agit d'un emploi illicite des instructions de codage. Par contre, le codage suivant:

```
GoodExample1      ::=   SEQUENCE {
      required-items    [UNTAGGED] SEQUENCE OF required-books Book ,
      optional-items    [UNTAGGED] SEQUENCE OF optional-books Book }
```

correspond à un emploi licite des instructions de codage.

B.2.8 (Prescription relative à la détermination de l'ensemble de composantes) Le premier élément XML dans le contenu partiel XML des composantes d'un groupe de concaténation qui est le codage d'un type ensemble devrait porter un nom d'élément XML qui est distinct du nom du premier élément XML dans le contenu partiel XML de toutes les autres composantes.

EXEMPLE 7: Le codage suivant:

```
BadExample4 ::=   SET {
      uk-mailing  UNTAGGED] SEQUENCE {name UTF8String, post-code UTF8String}
      us-mailing  [UNTAGGED] SEQUENCE {name UTF8String, zip-code UTF8String}}
```

viole la prescription relative à la détermination des composantes ainsi que le § 10.2.11. Il s'agit d'un emploi illicite des instructions de codage. Par contre le codage suivant:

```
GoodExample2     ::=   SET {
      uk-mailing  UNTAGGED] SEQUENCE {uk-name UTF8String, post-code UTF8String}
      us-mailing  [UNTAGGED] SEQUENCE {us-name UTF8String, zip-code UTF8String}}
```

correspond à un emploi licite des instructions de codage.

Ajouter une nouvelle Annexe C comme suit:

Annexe C

Exemples de codage selon les règles étendues en langage de balisage extensible employant des instructions de codage

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

C.1 Introduction

C.1.1 Dans la présente annexe sont données des informations à titre didactique et des exemples d'application des instructions de codage XER.

NOTE – On suppose dans tous les exemples ASN.1 de la présente annexe qu'on évolue dans un environnement **AUTOMATIC TAGS**.

C.1.2 Les instructions de codage ne doivent normalement être attribuées à une spécification ASN.1 que si le concepteur dispose d'une prescription relative à la forme en vigueur du codage XML pour qu'elle concorde avec celles qui sont définies dans les spécifications d'autres schémas, ou sont prévues par d'autres outils. Sinon, la notation ASN.1 seule (avec les codages BASIC-XER ou CXER) peut être employée.

C.1.3 Si la notation ASN.1 est employée pour la définition du schéma, alors l'utilisation en outre des instructions de codage fournira en général des codages XML plus compacts que ceux qui sont obtenus au moyen de la notation ASN.1 seule, mais de loin plus prolixes que ceux qui sont obtenus en combinant la notation ASN.1 avec les règles PER.

NOTE – Les exemples (et les identificateurs et les noms de type employés) sont conçus pour illustrer des propriétés des règles EXTENDED-XER, et ne représentent en général pas les spécifications réellement employées.

C.1.4 Les instructions de codage XER sont grossièrement subdivisées en deux catégories.

C.1.5 La première catégorie concerne les instructions de codage qui sont habituellement censées être utiles lors de la conception de la forme d'un document XML. Elles sont généralement admises même si les instructions **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** font défaut. Les plus utiles sont les instructions **ATTRIBUTE** et **LIST**, et le § C.2 donne des exemples simples de leur emploi.

C.1.6 La deuxième catégorie concerne les instructions de codage qui sont conçues pour prendre en charge la projection du schéma XML du Consortium W3C spécifié dans la Rec. UIT-T X.694 | ISO/CEI 8825-5. Elles nécessitent généralement la présence des instructions **GLOBAL-DEFAULTS MODIFIED-ENCODINGS** dans une section de commande de codage, mais ceci n'a pas été inclus dans les exemples. Dans ces exemples, toute référence au type commençant par "XSD." est supposée provenir de l'Annexe A de la Rec. UIT-T X.694 | ISO/CEI 8825-5. Le § C.3 donne des exemples de cette utilisation. **Ces exemples ne constituent pas des modules ASN.1 complets, ni des documents XML complets**: les en-têtes de modules sont généralement omis; et tout attribut XML commençant par "**asn1:**" est supposé être un attribut de commande employant un espace de nom asn1, où le préfixe "**asn1**" est supposé avoir déjà été déclaré. (En pratique, si le codage est déduit du schéma XML du Consortium W3C, le préfixe "**xsi**" a des chances d'être employé, avec l'espace de nom XSI.)

C.1.7 Dans presque tous les cas, les instructions de codage préfixées sont employées pour plus de clarté, même si dans une spécification réelle une façon plus succincte (et une séparation plus claire de la définition de syntaxe abstraite par rapport aux questions de codage) peuvent être obtenues en employant une section de commande de codage.

C.2 Exemples simples

C.2.1 Carte de base-ball

```
BBCard ::= SEQUENCE {
      name   [ATTRIBUTE] IA5String,
      team   [ATTRIBUTE] IA5String,
      age    INTEGER,
      position    IA5String,
      handedness ENUMERATED {
      left-handed,
      right-handed,
      ambidextrous },
      batting-average REAL }
```

En ne tenant pas compte des instructions de codage (BASIC-XER), nous pourrions obtenir ce qui suit:

```
<BBCard>
      <name>Jorge Posada</name>
      <team>New York Yankees</team>
      <age>29</age>
      <position>C</position>
      <handedness><right-handed/></handedness>
      <batting-average>0.277</batting-average>
</BBCard>
```

Le codage EXTENDED-XER (avec les codages MODIFIED-ENCODINGS) de la même valeur est le suivant:

```
<BBCard name = "Jorge Posada" team = "New York Yankees" >
      <age>29</age>
      <position>C</position>
      <handedness>right-handed</handedness>
      <batting-average>0.277</batting-average>
</BBCard>
```

C.2.2 Employé

```
Employee ::= [NAME AS UNCAPITALIZED] SEQUENCE {
      id          [ATTRIBUTE] INTEGER(0..MAX),
      recruited   XSD.Date,
      salaries    [LIST] SEQUENCE
                      OF salary REAL }
```

En ne tenant pas compte des instructions de codage (BASIC-XER), nous pourrions obtenir ce qui suit:

```
<Employee>
      <id>239</id>
      <recruited>27-11-2002</recruited>
      <salaries>
           <salary>29876</salary>
           <salary>54375</salary>
           <salary>98435</salary>
      </salaries>
</Employee>
```

Le codage EXTENDED-XER de la même valeur est le suivant:

```
<employee id = "239">
      <recruited>27-11-2002</recruited>
      <salaries>29876 54375 98435</salaries>
</employee>
```

En employant une section de commande de codage XER nous obtenons ce qui suit:

```
      Employee ::= SEQUENCE {
            id          INTEGER(0..MAX),
            recruited   Date,
            salaries    SEQUENCE
                   OF salary REAL }

      ENCODING-CONTROL XER
             NAME Employee AS UNCAPITALIZED
```

```
                    ATTRIBUTE Employee.id
                    LIST Employee.salaries
```

C.3 Exemples plus complexes

C.3.1 Emploi du regroupement de deux types simples

```
Int-or-boolean ::= [USE-UNION] CHOICE {
      int         INTEGER,
      boolean     BOOLEAN }
```

Les codages pourraient être les suivants:

```
<Int-or-boolean><int>39</int></Int-or-boolean>                -- BASIC-XER
<Int-or-boolean><boolean><true/></boolean></Int-or-boolean>   -- BASIC-XER
<Int-or-boolean>39</Int-or-boolean>                           -- EXTENDED-XER
<Int-or-boolean>true</Int-or-boolean>                         -- EXTENDED-XER
```

C.3.2 Emploi d'un attribut d'identification de type

```
Int-or-boolean ::= [USE-TYPE] CHOICE {
      int         INTEGER,
      boolean     BOOLEAN }
```

Les codages pourraient être les suivants:

```
<Int-or-boolean><int>39</int></Int-or-boolean>                -- BASIC-XER
<Int-or-boolean><boolean><true/></boolean></Int-or-boolean>   -- BASIC-XER
<Int-or-boolean asn1:type="int">39</Int-or-boolean>           -- EXTENDED-XER
<Int-or-boolean asn1:type="boolean">true</Int-or-boolean>     -- EXTENDED-XER
```

C.3.3 Emploi de valeurs d'énumération

```
PrimesUnder30 ::= [USE-NUMBER] ENUMERATED {
                  int2(2), int3(3), int5(5), int7(7), int11(11), int13(13),
                  int17(17), int19(19), int23(23), int29(29)}
InputValues ::= [ATTRIBUTE] [LIST] SEQUENCE OF PrimesUnder30
PrimeProducts ::= SEQUENCE {
            input  InputValues,
            output [ATTRIBUTE] [DECIMAL] REAL}
```

Les codages pourraient être les suivants:

```
<PrimeProducts>
      <input><int2/><int7/><int17/><int23/><int29/><int3/></input>
      <output>476338.00</output>
</PrimeProducts>                                  -- BASIC-XER
<PrimeProducts input="2  7  17  23  29  3" output="476338.00"/>
                                                  -- EXTENDED-XER
```

C.3.4 Emploi d'un codage blanc pour une valeur par défaut

```
Responses ::= ENUMERATED {ringing, engaged, number-not-known }
CallDetails ::= [DEFAULT-FOR-EMPTY number-not-known] SEQUENCE {
      number      [ATTRIBUTE] NumericString,
      response    Response }
```

Les codages pourraient être les suivants:

```
<CallDetails>
      <number>0164593746</number>
      <response><number-not-known/></response>
</CallDetails>                                    -- BASIC-XER
<CallDetails number="0164593746"/>       -- EXTENDED-XER
```

C.3.5 Emploi de valeurs imbriquées pour la notification d'un paiement dû

```
Notification ::= SEQUENCE {
      text [EMBED-VALUES] SEQUENCE OF UTF8String,
      account      INTEGER,
```

```
        amount-due  INTEGER,
        payable-by  XSD:Date } (CONSTRAINED BY {/* Doit être conforme à la Rec. UIT-T
        X.693 | ISO/CEI 8825-4, § 25.2 */})
```

Une valeur en notation ASN.1 de base pourrait être la suivante:

```
firstNotification Notification ::= {
                text {"Please note the following details:",
                      "(your business account)",
                       "This is in excess of your normal monthly allowance",
                       "or earlier"},
                account 568903,
                amount-due  536,
                payable-by  "27-08-2003" }
```

Les codages EXTENDED-XER seraient les suivants:

```
        <Notification>
              Please note the following details:
              <account>568903</account>
              (your business account)
              <amount-due>536</amount-due>
              This is in excess of your normal monthly allowance
              <payable-by>27-08-2003</payable-by>
              or earlier
        </Notification>
```

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série B	Moyens d'expression: définitions, symboles, classification
Série C	Statistiques générales des télécommunications
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	RGT et maintenance des réseaux: systèmes de transmission, circuits téléphoniques, télégraphie, télécopie et circuits loués internationaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	**Réseaux de données et communication entre systèmes ouverts**
Série Y	Infrastructure mondiale de l'information, protocole Internet et réseaux de nouvelle génération
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Genève, 2005