

Remplacée par une version plus récente



UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

UIT-T

X.683

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

(07/94)

**RÉSEAUX POUR DONNÉES ET INTERCONNEXION
DES SYSTÈMES OUVERTS**

**RÉSEAUTAGE OSI ET ASPECTS DES SYSTÈMES –
NOTATION DE SYNTAXE ABSTRAITE NUMÉRO UN**

**TECHNOLOGIES DE L'INFORMATION –
NOTATION DE SYNTAXE ABSTRAITE
NUMÉRO 1: PARAMÉTRAGE DES
SPÉCIFICATIONS DE LA NOTATION
DE SYNTAXE ABSTRAITE NUMÉRO UN**

Recommandation UIT-T X.683

Remplacée par une version plus récente

(Antérieurement «Recommandation du CCITT»)

Remplacée par une version plus récente

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. Au sein de l'UIT-T, qui est l'entité qui établit les normes mondiales (Recommandations) sur les télécommunications, participent quelque 179 pays membres, 84 exploitations de télécommunications reconnues, 145 organisations scientifiques et industrielles et 38 organisations internationales.

L'approbation des Recommandations par les Membres de l'UIT-T s'effectue selon la procédure définie dans la Résolution n° 1 de la Conférence mondiale de normalisation des télécommunications (CMNT), (Helsinki, 1993). De plus, la CMNT, qui se réunit tous les quatre ans, approuve les Recommandations qui lui sont soumises et établit le programme d'études pour la période suivante.

Dans certains secteurs de la technologie 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. Le texte de la Recommandation X.683 de l'UIT-T a été approuvé le 1^{er} juillet 1994. Son texte est publié, sous forme identique, comme Norme internationale ISO/CEI 8824-4.

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.

© UIT 1996

Droits de reproduction réservés. Aucune partie de cette publication ne peut être reproduite ni utilisée sous quelque forme que ce soit et par aucun procédé, électronique ou mécanique, y compris la photocopie et les microfilms, sans l'accord écrit de l'UIT.

Remplacée par une version plus récente

RECOMMANDATIONS UIT-T DE LA SÉRIE X

RÉSEAUX DE COMMUNICATION DE DONNÉES ET COMMUNICATION ENTRE SYSTÈMES OUVERTS

(Février 1994)

ORGANISATION DES RECOMMANDATIONS DE LA SÉRIE X

Domaine	Recommandations
RÉSEAUX PUBLICS POUR DONNÉES	
Services et services complémentaires	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
Définition 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 de couche	X.280-X.289
Test de conformité	X.290-X.299
INTERFONCTIONNEMENT DES RÉSEAUX	
Considérations générales	X.300-X.349
Systèmes mobiles de transmission de données	X.350-X.369
Gestion	X.370-X.399
SYSTÈMES DE MESSAGERIE	X.400-X.499
ANNUAIRE	X.500-X.599
RÉSEAUTAGE OSI ET ASPECTS DES SYSTÈMES	
Réseautage	X.600-X.649
Dénomination, adressage et enregistrement	X.650-X.679
Notation de syntaxe abstraite numéro un (ASN.1)	X.680-X.699
GESTION OSI	X.700-X.799
SÉCURITÉ	X.800-X.849
APPLICATIONS OSI	
Engagement, concomitance et rétablissement	X.850-X.859
Traitement des transactions	X.860-X.879
Opérations distantes	X.880-X.899
TRAITEMENT OUVERT RÉPARTI	X.900-X.999

Remplacée par une version plus récente

TABLE DES MATIÈRES

	<i>Page</i>
1	Domaine d'application..... 1
2	Références normatives 1
2.1	Recommandations Normes internationales identiques..... 1
3	Définitions..... 1
3.1	Spécification de la notation de base 1
3.2	Spécification des objets informationnels 1
3.3	Spécification des contraintes..... 1
3.4	Définitions additionnelles 2
4	Abréviations 2
5	Convention 2
6	Notation..... 2
6.1	Affectations..... 2
6.2	Définitions paramétrées 2
6.3	Symboles..... 3
7	Items ASN.1 3
8	Affectations paramétrées..... 3
9	Référenciation des définitions paramétrées..... 6
10	Paramètres de syntaxe abstraite..... 8
Annexe A – Exemples..... 9	
A.1	Exemple d'utilisation d'une définition de type paramétré 9
A.2	Exemple d'utilisation de définitions paramétrées en même temps qu'une classe d'objets informationnels 9
A.3	Exemple de définition finie de type paramétré 10
A.4	Exemple de définition de valeur paramétrée..... 11
A.5	Exemple de définition d'ensemble de valeurs paramétrées 11
A.6	Exemple de définition de classe paramétrée 11
A.7	Exemple de définition d'ensemble d'objets paramétré 12
A.8	Exemple de définition d'ensemble d'objets paramétré 12
Annexe B – Récapitulation de la notation 13	

Remplacée par une version plus récente

Résumé

La présente Recommandation | Norme internationale définit les dispositions relatives aux noms de référence paramétrés et à l'affectation paramétrée pour des types de données qui sont utiles au concepteur quand il établit des spécifications dont certains aspects, qui ne sont pas encore définis à ce stade, le seront ultérieurement pour aboutir à la définition complète d'une syntaxe abstraite.

Remplacée par une version plus récente

Introduction

Les concepteurs d'applications doivent rédiger des spécifications dont certains aspects ne sont pas définis. Ces aspects seront définis ultérieurement par un ou plusieurs autres groupes (chacun à sa manière), afin de produire une spécification entièrement définie servant à définir une syntaxe abstraite (une pour chaque groupe).

Dans certains cas, certains aspects de la spécification (par exemple, des limites) peuvent ne pas être définis même au moment de la définition de la syntaxe abstraite et complétés par la spécification de profils normalisés au plan international ou de profils fonctionnels fournis par un autre organe.

NOTE 1 – La présente Recommandation | Norme internationale impose la condition suivante: tout aspect ne se rapportant pas uniquement à l'application de contraintes doit être complet avant que la syntaxe abstraite soit définie.

Dans le cas extrême, on peut laisser le responsable de la mise en œuvre achever certains aspects de la spécification, qui sont alors spécifiés comme une partie de la déclaration de conformité d'une instance de protocole (PICS).

Bien que les dispositions de la Rec. UIT-T X.681 | ISO/CEI 8824-2 et de la Rec. UIT-T X.682 | ISO/CEI 8824-3 fournissent un cadre pour l'achèvement ultérieur des parties d'une spécification, elles ne satisfont pas en elles-mêmes aux conditions ci-dessus.

De plus, un seul concepteur doit parfois définir de nombreux types, classes d'objets informationnels, ensembles d'objets informationnels, objets informationnels ou valeurs qui ont la même structure de niveau le plus extérieur, mais dont les types, classes d'objets informationnels, ensembles d'objets informationnels, objets informationnels ou valeurs, utilisés au niveau le plus intérieur, sont différents. Au lieu d'écrire la structure de niveau le plus extérieur pour chaque instance, il est utile de pouvoir l'écrire une fois pour toutes, avec des parties restant à définir ultérieurement, puis de s'y référer et de donner les informations complémentaires.

Toutes ces conditions sont satisfaites par les noms de référence paramétrés et les affectations paramétrées donnés dans la présente Recommandation | Norme internationale.

La forme syntaxique d'un nom de référence paramétré est la même que celle du nom de référence normal correspondant, mais les considérations supplémentaires suivantes s'appliquent:

- lorsqu'elle est affectée dans une déclaration d'affectation paramétrée, elle est suivie par une liste de noms de référence fictifs placés entre accolades, chacun pouvant être accompagné d'un gouvernant; la portée de ces noms de référence s'étend au côté droit de la déclaration d'affectation et à la liste même de paramètres;
NOTE 2 – C'est ce qui permet de la reconnaître comme nom de référence paramétré.
- lorsqu'elle est exportée ou importée, elle est suivie de deux accolades vides pour être distinguée comme nom de référence paramétré;
- lorsqu'elle est utilisée dans une structure, elle est suivie d'une liste de structures syntaxiques, une pour chaque nom de référence fictif, qui donnent une affectation au nom de référence fictif dans le seul cadre de cette utilisation.

Les noms de référence fictifs ont la même forme syntaxique que le nom de référence correspondant normal et peuvent être employés à n'importe quel endroit du côté droit d'une déclaration d'affectation où le nom de référence correspondant normal pourrait figurer. Toutes ces règles doivent être cohérentes.

NORME INTERNATIONALE

RECOMMANDATION UIT-T

**TECHNOLOGIES DE L'INFORMATION –
NOTATION DE SYNTAXE ABSTRAITE NUMÉRO 1:
PARAMÉTRAGE DES SPÉCIFICATIONS DE LA NOTATION
DE SYNTAXE ABSTRAITE NUMÉRO UN**

1 Domaine d'application

La présente Recommandation | Norme internationale, qui fait partie de la notation de syntaxe abstraite numéro un (ASN.1), définit la notation pour le paramétrage des spécifications ASN.1.

2 Références normatives

Les Recommandations et les Normes internationales suivantes contiennent des dispositions qui, par suite de la référence qui y est faite, constituent des dispositions valables pour la présente Recommandation | Norme internationale. Au moment de la publication, les éditions indiquées étaient en vigueur. Toute Recommandation ou Norme internationale est sujette à révision, et les parties prenantes aux accords fondés sur la présente Recommandation | Norme internationale sont invitées à rechercher la possibilité d'appliquer les éditions les plus récentes des Recommandations et Normes internationales indiquées ci-après. Les membres de la CEI et de l'ISO possèdent le registre des Normes internationales en vigueur. Le Bureau de la normalisation des télécommunications de l'UIT tient à jour une liste des Recommandations UIT-T en vigueur.

2.1 Recommandations | Normes internationales identiques

- Recommandation UIT-T X.680 (1994) | ISO/CEI 8824-1: 1995, *Technologies de l'information – Notation de syntaxe abstraite numéro un: Spécification de la notation de base.*
- Recommandation UIT-T X.681 (1994) | ISO/CEI 8824-2: 1995, *Technologies de l'information – Notation de syntaxe abstraite numéro un: Spécification des objets informationnels.*
- Recommandation UIT-T X.682 (1994) | ISO/CEI 8824-3: 1995, *Technologies de l'information – Notation de syntaxe abstraite numéro un: Spécification des contraintes.*

3 Définitions

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

3.1 Spécification de la notation de base

La présente Recommandation | Norme internationale utilise les termes définis dans la Rec. UIT-T X.680 | ISO/CEI 8824-1.

3.2 Spécification des objets informationnels

La présente Recommandation | Norme internationale utilise les termes définis dans la Rec. UIT-T X.681 | ISO/CEI 8824-2.

3.3 Spécification des contraintes

La présente Recommandation | Norme internationale utilise les termes définis dans la Rec. UIT-T X.682 | ISO/CEI 8824-3.

3.4 Définitions additionnelles

3.4.1 nom de référence normal: Nom de référence défini, sans paramètre, à l'aide d'une affectation "Assignment" autre qu'une affectation paramétrée "ParameterizedAssignment". Ce nom fait référence à une définition complète et n'est pas fourni avec des paramètres effectifs lorsqu'il est utilisé.

3.4.2 nom de référence paramétré: Nom de référence défini à l'aide d'une affectation paramétrée, qui fait référence à une définition incomplète et doit donc être fourni avec des paramètres effectifs lorsqu'il est utilisé.

3.4.3 type paramétré: Type défini au moyen d'une affectation type paramétrée, dont les composantes sont donc des définitions incomplètes, qui doit être fourni avec des paramètres effectifs lorsqu'il est utilisé.

3.4.4 valeur paramétrée: Valeur définie au moyen d'une affectation de valeur paramétrée, dont la valeur, incomplètement spécifiée, doit être fournie avec les paramètres effectifs lorsqu'elle est utilisée.

3.4.5 ensemble de valeurs paramétrées: Ensemble de valeurs définies au moyen d'une affectation d'un ensemble de valeurs paramétrées, dont les valeurs composantes sont donc incomplètement spécifiées, et qui doit par conséquent être fourni avec des paramètres effectifs lorsqu'il est utilisé.

3.4.6 classe d'objets paramétrés: Classe d'objets informationnels définie au moyen d'une affectation de classe d'objets paramétrés, dont les composants sont donc incomplètement spécifiés, et qui doit par conséquent être fournie avec des paramètres effectifs lorsqu'elle est utilisée.

3.4.7 objet paramétré: Objet informationnel défini au moyen d'une affectation d'objet paramétré, dont les composants sont donc incomplètement spécifiés, et qui doit par conséquent être fourni avec des paramètres effectifs lorsqu'il est utilisé.

3.4.8 ensemble d'objets paramétrés: Ensemble d'objets informationnels défini au moyen d'une affectation d'un ensemble d'objets paramétrés, dont les objets sont donc incomplètement spécifiés, et qui doit par conséquent être fourni avec des paramètres effectifs lorsqu'il est utilisé.

3.4.9 contrainte variable: Contrainte employée pour spécifier une syntaxe abstraite paramétrée et dépendant d'un paramètre de cette syntaxe abstraite.

4 Abréviations

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

5 Convention

La présente Recommandation | Norme internationale utilise la convention de notation définie à l'article 5 de la Rec. UIT-T X.680 | ISO/CEI 8824-1.

6 Notation

Cet article récapitule la notation définie dans la présente Recommandation | Norme internationale.

6.1 Affectations

La notation suivante, qui peut être utilisée comme une possibilité d'affectation "Assignment" (voir l'article 10 de la Rec. UIT-T X.680 | ISO/CEI 8824-1), est définie dans cette Recommandation | Norme internationale:

- ParameterizedAssignment (voir 8.1).

6.2 Définitions paramétrées

6.2.1 La présente Recommandation | Norme internationale définit la notation suivante, qui peut être utilisée comme une possibilité de type défini "DefinedType" (voir 11.1 de la Rec. UIT-T X.680 | ISO/CEI 8824-1):

- ParameterizedType (voir 9.2).

6.2.2 La présente Recommandation | Norme internationale définit la notation suivante, qui peut être utilisée comme une possibilité de valeur définie "DefinedValue" (voir 11.1 de la Rec. UIT-T X.680 | ISO/CEI 8824-1):

- ParameterizedValue (voir 9.2).

6.2.3 La présente Recommandation | Norme internationale définit la notation suivante, qui peut être utilisée comme une forme possible de type défini "DefinedType" (voir 11.1 de la Rec. UIT-T X.680 | ISO/CEI 8824-1):

- ParameterizedValueType (voir 9.2).

6.2.4 La présente Recommandation | Norme internationale définit la notation suivante, qui peut être utilisée comme une possibilité de classe d'objets "ObjectClass" (voir 9.2 de la Rec. UIT-T X.681 | ISO/CEI 8824-2):

- ParameterizedObjectClass (voir 9.2).

6.2.5 La présente Recommandation | Norme internationale définit la notation suivante, qui peut être utilisée comme une possibilité d'objets "Object" (voir 11.2 de la Rec. UIT-T X.681 | ISO/CEI 8824-2):

- ParameterizedObject (voir 9.2).

6.2.6 La présente Recommandation | Norme internationale définit la notation suivante, qui peut être utilisée comme une possibilité d'ensemble d'objets "ObjectSet" (voir 12.2 de la Rec. UIT-T X.681 | ISO/CEI 8824-2):

- ParameterizedObjectSet (voir 9.2).

6.3 Symboles

La notation suivante, qui peut être utilisée comme une possibilité de symbole "Symbol" (voir 10.1 de la Rec. UIT-T X.680 | ISO/CEI 8824-1), est définie dans cette Recommandation | Norme internationale:

- ParameterizedReference (voir 9.1).

7 Items ASN.1

La présente Recommandation | Norme internationale utilise les items ASN.1 spécifiés à l'article 9 de la Rec. UIT-T X.680 | ISO/CEI 8824-1.

8 Affectations paramétrées

8.1 Il existe des déclarations d'affectation paramétrées correspondant à chacune des déclarations d'affectation spécifiées dans la Rec. UIT-T X.680 | ISO/CEI 8824-1 et la Rec. UIT-T X.681 | ISO/CEI 8824-2. La structure "ParameterizedAssignment" est la suivante:

```
ParameterizedAssignment ::=
    ParameterizedTypeAssignment      |
    ParameterizedValueAssignment     |
    ParameterizedValueTypeAssignment |
    ParameterizedObjectClassAssignment |
    ParameterizedObjectAssignment   |
    ParameterizedObjectSetAssignment
```

8.2 Chaque structure "Parameterized<X>Assignment" a la même syntaxe que la structure "<X>Assignment", sauf que le premier item est suivi d'une liste de paramètres "ParameterList". Le premier item devient donc un nom de référence paramétré (voir 3.4.2).

NOTE – Dans la Rec. UIT-T X.680 | ISO/CEI 8824-1, la condition suivante est imposée: tous les noms de référence affectés dans un module, paramétrés ou non, doivent être distincts.

```
ParameterizedTypeAssignment ::=
    typereference
    ParameterList
    "::~="
    Type
```

```
ParameterizedValueAssignment ::=
    valuereference
    ParameterList
    Type
    "::~="
    Value
```

ParameterizedValueSetTypeAssignment ::=
 typereference
 ParameterList
 Type
 ::="
 ValueSet

ParameterizedObjectClassAssignment ::=
 objectclassreference
 ParameterList
 ::="
 ObjectClass

ParameterizedObjectAssignment ::=
 objectreference
 ParameterList
 DefinedObjectClass
 ::="
 Object

ParameterizedObjectSetAssignment ::=
 objectsetreference
 ParameterList
 DefinedObjectClass
 ::="
 ObjectSet

8.3 Une liste de paramètres "ParameterList" est une liste de paramètres "Parameter" placés entre accolades.

ParameterList ::= "{" Parameter "," + "}"

Chaque paramètre "Parameter" se compose d'une référence muette "DummyReference" et éventuellement d'un gouvernant "Governor".

Parameter ::= ParamGovernor ":" DummyReference | DummyReference

ParamGovernor ::= Governor | DummyGovernor

Governor ::= Type | DefinedObjectClass

DummyGovernor ::= DummyReference

DummyReference ::= Reference

Une référence muette "DummyReference" dans la notation "Parameter" peut représenter:

- a) un "Type" ou une classe d'objets définis "DefinedObjectClass", auquel cas il n'y aura pas de paramètre gouvernant "ParamGovernor");
- b) une valeur "Value" ou un ensemble de valeurs "ValueSet", auquel cas le paramètre gouvernant "ParamGovernor" est présent; si alors le paramètre gouvernant est un gouvernant "Governor", le gouvernant sera un "Type"; et si le paramètre gouvernant est un gouvernant muet "DummyGovernor", le paramètre effectif de "ParamGovernor" sera un "Type";
- c) un objet "Object" ou un ensemble d'objets "ObjectSet", auquel cas le paramètre gouvernant "ParamGovernor" est présent; si alors le paramètre gouvernant est un gouvernant "Governor", il sera une classe d'objets définie "DefinedObjectClass"; et si le paramètre gouvernant est un gouvernant muet "DummyGovernor", le paramètre effectif de "ParamGovernor" sera une classe d'objets définie "DefinedObjectClass".

Un gouvernant muet "DummyGovernor" sera une référence muette "DummyReference" qui n'a pas de gouvernant "Governor".

8.4 La portée d'une référence muette "DummyReference" apparaissant dans une liste de paramètres "ParameterList" est la liste de paramètres "ParameterList" elle-même, ainsi que la partie de l'affectation paramétrée "ParameterizedAssignment" placée après les symboles "::=". La référence muette "DummyReference" masque toutes les autres références "Reference" s'écrivant de la même manière et figurant dans ce domaine de visibilité.

8.5 L'emploi d'une référence muette "DummyReference" dans sa portée doit être cohérent avec sa forme syntaxique et, le cas échéant, avec le gouvernant; tous les emplois de la même référence muette "DummyReference" doivent être cohérents entre eux.

NOTE – Lorsque la forme syntaxique d'un nom de référence fictif est ambiguë (par exemple, qu'elle est soit une référence de classe d'objets "objectclassreference", soit une référence de type "typereference"), cette ambiguïté peut normalement être levée lors de la première utilisation du nom de référence fictif du côté droit de la déclaration d'affectation. Par la suite, la nature du nom de référence fictif est connue. La nature de la référence muette n'est toutefois pas déterminée uniquement par le côté droit de la déclaration d'affectation lorsqu'il est à son tour utilisé comme paramètre effectif d'une référence paramétrée; dans ce cas, la nature de la référence muette doit être déterminée en examinant la définition de cette référence paramétrée. Les utilisateurs de la notation sont mis en garde contre le fait qu'une telle pratique peut rendre les spécifications ASN.1 moins claires et il leur est proposé de fournir les commentaires appropriés afin d'expliquer cela aux lecteurs humains.

Exemple

Considérons l'affectation de classe d'objet paramétré suivante:

```
PARAMETERIZED-OBJECT-CLASS { TypeParam, INTEGER:valueParam, INTEGER:ValueSetParam } ::=
  CLASS {
    &valueField1      TypeParam,
    &valueField2      INTEGER DEFAULT valueParam,
    &valueField3      INTEGER (ValueSetParam),
    &ValueSetField    INTEGER DEFAULT { ValueSetParam }
  }
```

Pour déterminer l'emploi exact des références muettes "DummyReference" dans le cadre de l'affectation "ParameterizedAssignment", et uniquement pour cela, on peut envisager de définir ces références muettes de la manière suivante:

```
TypeParam ::= UnspecifiedType
valueParam INTEGER ::= unspecifiedIntegerValue
ValueSetParam INTEGER ::= { UnspecifiedIntegerValueSet }
```

où:

- a) Un paramètre de type "TypeParam" est une référence muette qui tient lieu de "Type". Pour cette raison il peut remplacer systématiquement les références de type "typereference", par exemple en tant que "Type" pour le champ de la valeur "valueField1" de type fixe.
- b) Un paramètre de valeur "valueParam" est une référence muette qui tient lieu de valeur d'un type entier. Pour cette raison il peut remplacer systématiquement les références de valeur "valuereference" d'une valeur entière, par exemple en tant que valeur "valueField2" par défaut pour le champ de la valeur "valueField2" de type fixe.
- c) Le paramètre d'ensemble de valeurs "ValueSetParam" est une référence muette qui tient lieu d'ensemble de valeurs d'un type entier. Pour cette raison il peut remplacer systématiquement les références de type "typereference" d'une valeur entière, par exemple en tant que "Type" dans la notation "ContainedSubtype" pour le champ de valeur "valueField3" et le champ d'ensemble de valeurs "ValueSetField".

8.6 Chaque référence muette "DummyReference" doit être utilisée au moins une fois dans son domaine de visibilité.

NOTE – Si la référence muette "DummyReference" n'apparaît pas de cette manière, le paramètre effectif "ActualParameter" correspondant n'a alors aucun effet sur la définition et devrait simplement être "rejeté", alors que l'utilisateur peut avoir l'impression qu'une spécification se met en place.

Des notations "ParameterizedValueAssignment", "ParameterizedValueSetTypeAssignment", "ParameterizedObjectAssignment" ou "ParameterizedObjectSetAssignment" qui contiendraient des références directes ou indirectes à elles-mêmes sont illicites.

8.7 Dans une définition de type paramétré "ParameterizedType", d'ensemble de valeurs paramétrées "ParameterizedValueSet" ou de classe d'objets paramétrés "ParameterizedObjectClass", une référence muette "DummyReference" ne sera pas passée comme type étiqueté (en tant que paramètre effectif) à une référence récursive à ce type "ParameterizedType", à cet ensemble "ParameterizedValueSet" ou à cette classe "ParameterizedObjectClass" (voir A.3).

8.8 Dans la définition d'un type paramétré "ParameterizedType", d'un ensemble de valeurs paramétrées "ParameterizedValueSet" ou d'une classe d'objets paramétrés "ParameterizedObjectClass", aucune référence circulaire ne sera faite à l'item défini sauf si une telle référence est directement ou indirectement déclarée comme optionnelle OPTIONAL ou, dans le cas de "ParameterizedType" ou de "ParameterizedValueSet", si cette référence est faite par le biais d'un type choix dont l'une au moins des alternatives est par définition non circulaire.

8.9 Le gouvernant d'une référence muette "DummyReference" ne comportera pas de référence à une autre référence muette si cette autre référence muette possède elle-même un gouvernant.

8.10 Dans une affectation paramétrée, le membre droit de l'affectation ne se résumera pas à une simple référence muette "DummyReference".

8.11 Il ne sera pas nécessaire que le gouvernant d'une référence muette "DummyReference" connaisse la référence muette ou le nom de la référence paramétrée en cours de définition.

9 Référenciation des définitions paramétrées

9.1 Lorsqu'elle se trouve dans une liste de symboles "SymbolList" (dans "Exports" ou "Imports"), une définition paramétrée est donnée en référence par une référence paramétrée "ParameterizedReference":

ParameterizedReference ::= Reference | Reference "{" "}"

où "Reference" est le premier item de l'affectation paramétrée "ParameterizedAssignment" spécifiée au 8.2 ci-dessus.

NOTE – la première forme possible de "ParameterizedReference" est donnée uniquement pour en faciliter la compréhension, mais les deux formes ont la même signification.

9.2 Lorsqu'elle ne se trouve pas dans "Exports" ou "Imports", une définition paramétrée est donnée en référence par une structure "Parameterized<X>", qui peut être utilisée comme une possibilité de la structure correspondante "<X>".

ParameterizedType ::=
SimpleDefinedType
ActualParameterList

SimpleDefinedType ::=
Externaltypereference |
typereference

ParameterizedValue ::=
SimpleDefinedValue
ActualParameterList

SimpleDefinedValue ::=
Externalvaluereference |
valuereference

ParameterizedValueSetType ::=
SimpleDefinedType
ActualParameterList

ParameterizedObjectClass ::=
DefinedObjectClass
ActualParameterList

ParameterizedObjectSet ::=
DefinedObjectSet
ActualParameterList

ParameterizedObject ::=
DefinedObject
ActualParameterList

9.3 Le nom de référence figurant dans la structure "Defined<X>" doit être un nom de référence affecté dans une affectation paramétrée "ParameterizedAssignment".

9.4 Les restrictions appliquées à la possibilité "Defined<X>" à utiliser, spécifiées dans la Rec. UIT-T X.680 | ISO/CEI 8824-1 et la Rec. UIT-T X.681 | ISO/CEI 8824-2 en tant que noms de référence normaux, s'appliquent aussi aux noms de référence paramétrés correspondants.

NOTE – Par nature, les restrictions sont les suivantes: chaque structure "Defined<X>" a deux formes possibles, "<x>reference" et "External<x>Reference". La première est utilisée dans le module de définition ou si la définition a été importée et qu'il n'y a pas de conflit de noms; la seconde est utilisée lorsqu'il n'y a pas d'importation énumérée (cas déconseillé) ou s'il y a un conflit de noms entre le nom importé et une définition locale (cas aussi déconseillé) ou entre des importations.

9.5 La liste de paramètres effectifs "ActualParameterList" s'écrit comme suit:

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

ActualParameter ::=
Type |
Value |

```

ValueSet          |
DefinedObjectClass |
Object            |
ObjectSet

```

9.6 Il doit y avoir exactement un paramètre effectif "ActualParameter" pour chaque paramètre "Parameter" de l'affectation paramétrée "ParameterizedAssignment" correspondante et les paramètres effectifs doivent apparaître dans le même ordre que les paramètres. Le choix particulier d'un paramètre effectif "ActualParameter" et du gouvernant (le cas échéant) sera déterminé par l'examen de la forme syntaxique du paramètre "Parameter" et de l'environnement dans lequel il survient dans l'affectation paramétrée "ParameterizedAssignment". La forme du paramètre effectif "ActualParameter" doit être celle requise pour remplacer la référence muette "DummyReference" dans tout son domaine de visibilité (voir 8.4).

Exemple

La définition de classe d'objet paramétré donnée dans l'exemple précédent (voir 8.5) peut être donnée en référence de la manière suivante, par exemple:

```
MY-OBJECT-CLASS ::= PARAMETERIZED-OBJECT-CLASS { BIT STRING, 123, {4 | 5 | 6} }
```

9.7 Le paramètre effectif remplace le nom de la référence muette pour déterminer le type, la valeur, l'ensemble de valeurs, la classe d'objets, l'objet ou l'ensemble d'objets, effectivement donné en référence par cette utilisation du nom de référence paramétrée.

9.8 La signification des références apparaissant dans le paramètre effectif "ActualParameter" et l'étiquette par défaut applicable aux étiquettes apparaissant ainsi sont déterminées conformément à l'environnement d'étiquetage de ce paramètre effectif "ActualParameter" plutôt qu'à celui de la référence muette "DummyReference" correspondante.

NOTE – Le paramétrage, comme la référencement, les types "selection" et les structures "COMPONENTS OF" (composants de), entre autres, n'est donc pas exactement une substitution littérale.

Exemple

Considérons les modules suivants:

```

M1 DEFINITIONS AUTOMATIC TAGS ::= BEGIN
  EXPORTS T1;

  T1 ::= SET {
    f1    INTEGER,
    f2    BOOLEAN
  }
END

M2 DEFINITIONS EXPLICIT TAGS ::= BEGIN
  IMPORTS T1 FROM M1;

  T3 ::= T2{T1}

  T2{X} ::= SEQUENCE {
    a    INTEGER,
    b    X
  }
END

```

L'application du 9.8 comporte que l'étiquette pour la composante f1 de T3 (à savoir @T3.b.f1) sera implicitement étiqueté étant donné que l'environnement d'étiquetage du paramètre muet X, à savoir l'étiquetage explicite, n'intervient pas sur l'étiquetage des composantes du paramètre effectif T1.

Considérons le module M3:

```

M3 DEFINITIONS AUTOMATIC TAGS ::= BEGIN
  IMPORTS T1 FROM M1;

  T5 ::= T4{T1}

  T4{Y} ::= SEQUENCE {
    a    INTEGER,
    b    Y
  }
END

```

L'application du 28.6 de la Rec. UIT-T X.680 | ISO/CEI 8824-1 implique que l'étiquette pour la composante b de T5 (à savoir @T5.b) sera explicitement étiquetée, car le paramètre muet (Y) est toujours explicitement étiqueté; par conséquent, @T5 est équivalent à:

```
T5 ::= SEQUENCE {
  a  [0] IMPLICIT INTEGER,
  b  [1] EXPLICIT SET {
    f1 [0] INTEGER,
    f2 [1] BOOLEAN
  }
}
```

alors que @T3 est équivalent à:

```
T3 ::= SEQUENCE {
  a  INTEGER,
  b  SET {
    f1 [0] IMPLICIT INTEGER,
    f2 [1] IMPLICIT BOOLEAN
  }
}
```

10 Paramètres de syntaxe abstraite

10.1 On trouvera à l'Annexe B de la Rec. UIT-T X.681 | ISO/CEI 8824-2 la classe d'objets informationnels ABSTRACT-SYNTAX (syntaxe abstraite) dont l'utilisation est recommandée pour définir des syntaxes abstraites. Dans cette annexe, on prend pour exemple une syntaxe abstraite définie comme l'ensemble de valeurs d'un seul type ASN.1 non paramétré au niveau le plus extérieur.

10.2 Lorsque le type ASN.1 utilisé pour définir la syntaxe abstraite *est* paramétré, certains paramètres peuvent être indiqués comme paramètres effectifs lorsque la syntaxe abstraite est définie, les autres pouvant rester des paramètres de cette syntaxe abstraite elle-même.

Exemple

Si un type paramétré, appelé YYY-PDU, a été défini avec deux références muettes (la première étant un ensemble d'objets d'une classe d'objets définie, la seconde une valeur entière d'une limite, par exemple), la notation:

```
yyy-Abstract-Syntax { INTEGER:bound } ABSTRACT-SYNTAX ::=
  { YYY-PDU { ValidObjects }, bound } IDENTIFIED BY {yyy 5 }
```

définit alors une syntaxe abstraite paramétrée dans laquelle l'ensemble d'objets a été déterminé, mais où la limite "bound" reste un paramètre de cette syntaxe abstraite.

Un paramètre de syntaxe abstraite sera utilisé:

- directement ou indirectement dans le contexte d'une contrainte;
- directement ou indirectement comme paramètre effectif éventuellement utilisé dans le contexte d'une contrainte.

NOTE – Voir l'exemple au A.2, ainsi que l'exemple au G.5 de la Rec. UIT-T X.680 | ISO/CEI 8824-1.

10.3 Une contrainte dont l'ensemble de valeurs dépend d'un ou de plusieurs paramètres de la syntaxe abstraite est une contrainte variable. De telles contraintes sont déterminées après la définition de cette syntaxe abstraite (éventuellement par des profils normalisés au plan international ou dans des déclarations de conformité d'une instance de protocole).

NOTE – Si, en un point quelconque de la chaîne de définitions intervenant dans la spécification des valeurs de contraintes, apparaît un paramètre de la syntaxe abstraite, la contrainte est variable, même si l'ensemble des valeurs de la contrainte qui en résulte est indépendant de la valeur effective du paramètre de la syntaxe abstraite.

Exemple – La valeur de (((1..3)EXCEPT a) UNION (1 .. 3)) est toujours 1..3 quelle que soit la valeur de "a", mais il s'agit néanmoins d'une contrainte variable si "a" est un paramètre de la syntaxe abstraite.

10.4 Formellement, une contrainte variable ne s'applique pas à l'ensemble de valeurs de la syntaxe abstraite.

NOTE – Il est vivement recommandé que les contraintes prévues pour rester des contraintes variables dans une syntaxe abstraite comportent une spécification d'erreurs utilisant la notation donnée au 43.4 de la Rec. UIT-T X.680 | ISO/CEI 8824-1.

Annexe A

Exemples

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

A.1 Exemple d'utilisation d'une définition de type paramétré

Supposons qu'un concepteur de protocole doive souvent acheminer un authentificateur avec un ou plusieurs champs du protocole. Celui-ci sera acheminé comme une chaîne binaire "BIT STRING" dans le champ. S'il n'y a pas de paramétrage, l'authentificateur "Authenticator" devrait être défini comme une chaîne binaire "BIT STRING", puis être ajouté chaque fois qu'il doit apparaître, avec un texte permettant d'identifier à quoi il s'applique. Sinon, le concepteur pourrait adopter une méthode transformant les champs ayant un authentificateur en séquences "SEQUENCE" de ce champ et en authentificateurs "Authenticator". Le mécanisme de paramétrage permet d'effectuer cette tâche d'une manière rapide et pratique.

Tout d'abord, il faut définir le type paramétré "SIGNED{ }":

```
SIGNED { ToBeSigned } ::= SEQUENCE
{
    authenticated-data  ToBeSigned,
    authenticator       BIT STRING
}
```

puis, dans le corps du protocole, la notation (par exemple)

```
SIGNED { OrderInformation }
```

est une notation de type représentant

```
SEQUENCE
{
    authenticated-data  OrderInformation,
    authenticator       BIT STRING
}
```

On suppose ensuite que, pour certains champs, l'émetteur doit pouvoir choisir d'ajouter l'authentificateur, par exemple en rendant la chaîne binaire "BIT STRING" optionnelle ou en ayant recours à une solution plus élégante (moins de bits se trouvant sur la ligne) consistant à définir un autre type paramétré:

```
OPTIONALLY-SIGNED {ToBeSigned} ::= CHOICE
{
    unsigned-data      [0] ToBeSigned,
    signed-data        [1] SIGNED { ToBeSigned }
}
```

NOTE – L'étiquetage n'est pas nécessaire dans la possibilité "CHOICE" si l'auteur s'assure qu'aucune des utilisations du type paramétré ne produit un argument effectif qui soit une chaîne binaire "BIT STRING" (le type de "SIGNED"), mais il est utile pour éviter des erreurs en d'autres endroits de la spécification dans son ensemble.

A.2 Exemple d'utilisation de définitions paramétrées en même temps qu'une classe d'objets informationnels

Utiliser des classes d'objets informationnels pour rassembler tous les paramètres d'une syntaxe abstraite. De cette manière, le nombre de paramètres d'une syntaxe abstraite peut être ramené à un seul, qui est une instance de la classe de rassemblement. La production "InformationFromObject" peut servir à extraire les informations de l'objet paramétré.

Exemple:

```
-- Une instance de cette classe contient tous les paramètres nécessaires
-- à la syntaxe abstraite Message-PDU.
```

```
MESSAGE-PARAMETERS ::= CLASS {
    &maximum-priority-level    INTEGER,
    &maximum-message-buffer-size  INTEGER,
    &maximum-reference-buffer-size  INTEGER
}
```

```

WITH SYNTAX {
  THE MAXIMUM PRIORITY LEVEL IS          &maximum-priority-level
  THE MAXIMUM MESSAGE BUFFER SIZE IS     &maximum-message-buffer-size
  THE MAXIMUM REFERENCE BUFFER SIZE IS   &maximum-reference-buffer-size
}

```

-- La production "ValueFromObject" sert à extraire des valeurs du paramètre de syntaxe
 -- abstraite "param". Ces valeurs ne peuvent être utilisées que dans des contraintes.
 -- De plus, le paramètre est transmis à un autre type paramétré.

```

Message-PDU { MESSAGE-PARAMETERS : param } ::= SEQUENCE {
  priority-level  INTEGER (0..param.&maximum-priority-level),
  message        BMPString (SIZE (0..param.&maximum-message-buffer-size)),
  reference       Reference { param }
}

```

```

Reference { MESSAGE-PARAMETERS : param } ::=
  SEQUENCE OF
    IA5String (SIZE (0..param.&maximum-reference-buffer-size))

```

-- Définition d'un objet informationnel de syntaxe abstraite paramétrée.
 -- Le paramètre de syntaxe abstraite n'est utilisé que dans des contraintes.

```

message-Abstract-Syntax { MESSAGE-PARAMETERS : param }
ABSTRACT-SYNTAX ::=
{
  Message-PDU { param }
  IDENTIFIED BY { joint-iso-ccitt asn1(1) examples(123) 0 }
}

```

La classe "MESSAGE-PARAMETERS" et l'objet de syntaxe abstraite paramétrée "message-Abstract-Syntax" sont utilisés de la manière suivante:

-- Cette instance de la classe "MESSAGE-PARAMETERS" définit les valeurs paramétriques
 -- de la syntaxe abstraite.

```

my-message-parameters MESSAGE-PARAMETERS ::= {
  THE MAXIMUM PRIORITY LEVEL IS 10
  THE MAXIMUM MESSAGE BUFFER SIZE IS 2000
  THE MAXIMUM REFERENCE BUFFER SIZE IS 100
}

```

-- La syntaxe abstraite peut maintenant être définie avec toutes les contraintes variables spécifiées.

```

my-message-Abstract-Syntax ABSTRACT-SYNTAX ::=
  message-Abstract-Syntax { my-message-parameters }

```

A.3 Exemple de définition finie de type paramétré

Lorsqu'on spécifie un type paramétré qui représente une liste générique, spécifier le type de façon à ce que la notation ASN.1 résultante reste finie. On peut par exemple spécifier:

```

List1 { ElementTypeParam } ::= SEQUENCE {
  elem    ElementTypeParam,
  next    List1 { ElementTypeParam } OPTIONAL
}

```

qui est finie, car lorsqu'on écrit:

```

IntegerList1 ::= List1 { INTEGER }

```

la notation ASN.1 résultante est celle qu'on aurait normalement sous la forme:

```

IntegerList1 ::= SEQUENCE {
  elem    INTEGER,
  next    IntegerList1 OPTIONAL
}

```

alors qu'en écrivant:

```
List2 { ElementTypeParam } ::= SEQUENCE {
    elem    ElementTypeParam,
    next    List2 { [0] ElementTypeParam } OPTIONAL
}

IntegerList2 ::= List2 { INTEGER }
```

la notation ASN.1 résultante est infinie:

```
IntegerList2 ::= SEQUENCE {
    elem    INTEGER,
    next    SEQUENCE {
        elem    [0] INTEGER,
        next    SEQUENCE {
            elem    [0][0] INTEGER,
            next    SEQUENCE {
                elem    [0][0][0] INTEGER,
                next    SEQUENCE {
                    ... -- et ainsi de suite
                } OPTIONAL
            } OPTIONAL
        } OPTIONAL
    } OPTIONAL
}
```

A.4 Exemple de définition de valeur paramétrée

Si une valeur de chaîne paramétrée est définie de la manière suivante:

```
genericBirthdayGreeting { IA5String : name } IA5String ::= { "Happy birthday, ", name, "!!" }
```

alors les deux valeurs de chaîne suivantes sont identiques:

```
greeting1 IA5String ::= genericBirthdayGreeting { "John" }
greeting2 IA5String ::= "Happy birthday, John!!"
```

A.5 Exemple de définition d'ensemble de valeurs paramétrées

Si deux ensembles de valeurs paramétrées sont comme suit:

```
QuestList1 { IA5String : extraQuest } IA5String ::= { "Jack" | "John" | extraQuest }
QuestList2 { IA5String : ExtraQuests } IA5String ::= { "Jack" | "John" | ExtraQuests }
```

alors les ensembles de valeurs suivants sont identiques:

```
SetOfQuests1 IA5String ::= { QuestList1 { "Jill" } }
SetOfQuests2 IA5String ::= { QuestList2 { {"Jill"} } }
SetOfQuests3 IA5String ::= { "Jack" | "John" | "Jill" }
```

et les ensembles de valeurs suivants sont identiques:

```
SetOfQuests4 IA5String ::= { QuestList2 { {"Jill" | "Mary"} } }
SetOfQuests5 IA5String ::= { "Jack" | "John" | "Jill" | "Mary" }
```

A noter qu'un ensemble de valeurs est *toujours* spécifié entre accolades, même s'il s'agit d'une référence d'ensemble de valeurs paramétrées. Si on omet les accolades d'une référence à un identificateur "identifiant" antérieurement créé dans une affectation d'ensemble de valeurs, ou d'une référence à un type d'ensemble de valeurs paramétrées "ParameterizedValueSetType", alors la notation est celle d'un type et non celle d'un ensemble de valeurs.

A.6 Exemple de définition de classe paramétrée

La classe paramétrée suivante peut servir à définir des classes d'erreurs contenant les codes d'erreurs de différents types. A noter que le paramètre de type de code d'erreur "ErrorCodeType" n'est utilisé que comme gouvernant muet "DummyGovernor" du paramètre des codes d'erreurs valides "ValidErrorCodes".

```
GENERIC-ERROR { ErrorCodeType, ErrorCodeType : ValidErrorCodes } ::= CLASS {
    &errorCode    ValidErrorCodes
}
```

```

WITH SYNTAX {
  CODE &errorCode
}

```

La définition de classe paramétrée peut être utilisée de la manière suivante pour définir différentes classes partageant certaines caractéristiques, une même syntaxe définie par exemple.

```

ERROR-1 ::= GENERIC-ERROR { INTEGER, { 1 | 2 | 3 } }
ERROR-2 ::= GENERIC-ERROR { ErrorCodeString, { StringErrorCodes } }
ERROR-3 ::= GENERIC-ERROR { EnumeratedErrorCode, { fatal | error } }
ErrorCodeString ::= IA5String (SIZE (4))
StringErrorCodes ErrorCodeString ::= { "E001" | "E002" | "E003" }
EnumeratedErrorCode ::= ENUMERATED { fatal, error, warning }

```

Les classes définies peuvent alors être utilisées de la manière suivante:

```

My-Errors ERROR-2 ::= { { CODE "E001" } | { CODE "E002" } }
fatalError ERROR-3 ::= { CODE fatal }

```

A.7 Exemple de définition d'ensemble d'objets paramétré

La définition d'ensemble d'objets paramétré "AllTypes" forme un ensemble d'objets contenant un ensemble de base d'objets "BaseTypes" et un ensemble d'objets supplémentaires fournis sous la forme du paramètre "AdditionalTypes".

```

AllTypes { TYPE-IDENTIFIER : AdditionalTypes } TYPE-IDENTIFIER ::= { BaseTypes | AdditionalTypes }
BaseTypes TYPE-IDENTIFIER ::= {
  { BaseType-1 IDENTIFIED BY basic-type-obj-id-value-1 } |
  { BaseType-2 IDENTIFIED BY basic-type-obj-id-value-2 } |
  { BaseType-3 IDENTIFIED BY basic-type-obj-id-value-3 }
}

```

La définition de l'ensemble d'objets paramétré "AllTypes" peut être utilisée de la manière suivante:

```

My-All-Types TYPE-IDENTIFIER ::= { AllTypes {
  { My-Type-1 IDENTIFIED BY my-obj-id-value-1 } |
  { My-Type-2 IDENTIFIED BY my-obj-id-value-2 } |
  { My-Type-3 IDENTIFIED BY my-obj-id-value-3 }
} }

```

A.8 Exemple de définition d'ensemble d'objets paramétré

Le type défini au A.4 de la Rec. UIT-T X.682 | ISO/CEI 8824-3 peut être utilisé de la manière suivante dans une définition de syntaxe abstraite paramétrée:

```

-- "PossibleBodyTypes" (types de corps possibles) est un paramètre de syntaxe abstraite.
message-abstract-syntax { MHS-BODY-CLASS : PossibleBodyTypes } ABSTRACT-SYNTAX ::= {
  INSTANCE OF MHS-BODY-CLASS ({PossibleBodyTypes})
  IDENTIFIED BY { joint-iso-itu asn1(1) examples(1) 123 }
}

-- Cet ensemble d'objets énumère tous les couples possibles de valeurs et d'identificateurs
-- de types du type instance-de. Cet ensemble d'objets est utilisé comme paramètre effectif
-- de la définition de syntaxe abstraite paramétrée.
My-Body-Types MHS-BODY-CLASS ::= {
  { My-First-Type IDENTIFIED BY my-first-obj-id } |
  { My-Second-Type IDENTIFIED BY my-second-obj-id }
}

my-message-abstract-syntax ABSTRACT-SYNTAX ::=
  message-abstract-syntax { { My-Body-Types } }

```

Annexe B**Récapitulation de la notation**

(Cette annexe ne fait pas partie intégrante de la présente Recommandation | Norme internationale)

Les items suivants sont définis dans la Rec. UIT-T X.680 | ISO/CEI 8824-1 et utilisés dans la présente Recommandation | Norme internationale:

```

typereference
valuereference
"::="
"{"
"}"
","

```

Les items suivants sont définis dans la Rec. UIT-T X.681 | ISO/CEI 8824-2 et utilisés dans la présente Recommandation | Norme internationale:

```

objectclassreference
objectreference
objectsetreference

```

Les productions suivantes sont définies dans la Rec. UIT-T X.680 | ISO/CEI 8824-1 et utilisées dans la présente Recommandation | Norme internationale:

```

DefinedType
DefinedValue
Reference
Type
Value
ValueSet

```

Les productions suivantes sont définies dans la Rec. UIT-T X.681 | ISO/CEI 8824-2 et utilisées dans la présente Recommandation | Norme internationale:

```

DefinedObjectClass
DefinedObject
DefinedObjectSet
ObjectClass
Object
ObjectSet

```

Les productions suivantes sont définies dans la présente Recommandation | Norme internationale:

```

ParameterizedAssignment ::=
  ParameterizedTypeAssignment |
  ParameterizedValueAssignment |
  ParameterizedValueSetTypeAssignment |
  ParameterizedObjectClassAssignment |
  ParameterizedObjectAssignment |
  ParameterizedObjectSetAssignment

ParameterizedTypeAssignment ::=
  typereference ParameterList "::<=" Type

ParameterizedValueAssignment ::=
  valuereference ParameterList Type "::<=" Value

ParameterizedValueSetTypeAssignment ::=
  typereference ParameterList Type "::<=" ValueSet

ParameterizedObjectClassAssignment ::=
  objectclassreference ParameterList "::<=" ObjectClass

ParameterizedObjectAssignment ::=
  objectreference ParameterList DefinedObjectClass "::<=" Object

ParameterizedObjectSetAssignment ::=
  objectsetreference ParameterList DefinedObjectClass "::<=" ObjectSet

ParameterList ::= "{" Parameter "," + "}"

```

Parameter ::= ParamGovernor ":" DummyReference | DummyReference
ParamGovernor ::= Governor | DummyGovernor
Governor ::= Type | DefinedObjectClass
DummyGovernor ::= DummyReference
DummyReference ::= Reference
ParameterizedReference ::=
 Reference | Reference "{" "}"
SimpleDefinedType ::= Externaltypereference | typereference
SimpleDefinedValue ::= Externalvaluereference | valuereference
ParameterizedType ::= SimpleDefinedType ActualParameterList
ParameterizedValue ::= SimpleDefinedValue ActualParameterList
ParameterizedValueSetType ::= SimpleDefinedType ActualParameterList
ParameterizedObjectClass ::= DefinedObjectClass ActualParameterList
ParameterizedObjectSet ::= DefinedObjectSet ActualParameterList
ParameterizedObject ::= DefinedObject ActualParameterList
ActualParameterList ::= "{" ActualParameter "," + "}"
ActualParameter ::= Type | Value | ValueSet | DefinedObjectClass | Object | ObjectSet