Groupe de travail Réseau
Request for Comments : 4512
RFC rendues obsolètes : 2251, 2252, 2256, 3674
Catégorie : Norme
Juin 2006

K. Zeilenga, OpenLDAP Foundation

Traduction Claude Brière de L'Isle
décembre 2006

Protocole léger d'accès à un répertoire (LDAP) : Modèles des informations de répertoire

Statut de ce mémo

Le présent document spécifie un protocole de normalisation Internet pour la communauté de l'Internet, qui appelle à la discussion et à des suggestions pour son amélioration. Prière de se reporter à l'édition en cours des "Normes de protocole officielles de l'Internet" (STD 1) sur l'état de la normalisation et le statut de ce protocole. La distribution du présent mémo n'est soumise à aucune restriction.

Notice de Copyright

Résumé

Le Protocole léger d'accès à un répertoire (LDAP, *Lightweight Directory Access Protocol*) est un protocole Internet pour accéder à des services de répertoire distribuée qui agissent conformément aux modèles de données et de service de la Recommandation UIT-T X.500. Le présent document décrit les modèles d'information de répertoire de la Recommandation UIT-T X.500, tels qu'utilisés dans LDAP.

Table des matières

1 Introduction

Le présent document discute des modèles d'informations de répertoire X.500 [X.501], tels qu'utilisés par le protocole léger d'accès de répertoire (LDAP, *Lightweight Directory Access Protocol*) [RFC4510].

Le répertoire est "une collection de systèmes ouverts coopérant pour fournir des services de répertoire" [X.500]. Les informations détenues dans le répertoire sont connues collectivement comme la base d'informations de répertoire (DIB, *Directory Information Base*). Un utilisateur de répertoire, qui peut être une personne ou une autre entité, accède au répertoire à travers un client (ou agent utilisateur de répertoire (DUA, *Directory User Agent*)). Le client, au nom de l'utilisateur de répertoire, interagit avec un ou plusieurs serveurs (ou agents de système de répertoire (DSA, *Directory System Agents*)). Un serveur détient un fragment du DIB.

Le DIB contient deux classes d'informations :
1) des informations d'utilisateur (par exemple, des informations fournies et administrées par des utilisateurs). La Section 2 décrit le modèle des informations d'utilisateur.

2) des informations administratives et de fonctionnement (par exemple, des informations utilisées pour administrer et/ou faire fonctionner le répertoire). La Section 3 décrit le modèle des informations administratives et de fonctionnement de répertoire.

Ces deux modèles, désignés sous le nom générique de modèles d'informations de répertoire, décrivent comment les informations sont représentées dans le répertoire. Ces modèles génériques fournissent un cadre de travail pour d'autres modèles d'information. La Section 4 discute du modèle d'information de sous-schéma et de la découverte de sous-schéma. La Section 5 discute du modèle d'informations DSA (de serveur).

Les autres modèles d'informations X.500 (tels que les modèles d'informations à connaissance de distribution de contrôle d'accès et connaissance de duplication) peuvent être adaptés pour être utilisés dans LDAP. La spécification de la façon dont ces modèles s'appliquent à LDAP fera l'objet de documents futurs.

1.1 *Relations avec les autres spécifications LDAP*

Le présent document faits partie intégrante de la spécification technique LDAP [RFC4510], qui rend obsolète la spécification technique LDAP précédemment définie, RFC 3377, dans sa totalité.

Le présent document rend obsolète la RFC 2251, paragraphes 3.2 et 3.4, ainsi que des portions des Sections 4 et 6. L'Appendice A.1 résume les changements de ces sections. Le reste de la RFC 2251 est rendu obsolète par les [RFC4511], [RFC4513], et [RFC4510].

Le présent document rend obsolète la RFC 2252, Sections 4, 5, et 7. L'Appendice A.2 résume les changements de ces sections. Le reste de la RFC 2252 est rendu obsolète par la [RFC4517].

Le présent document rend obsolète la RFC 2256, paragraphes 5.1, 5.2, 7.1, et 7.2. L'Appendice A.3 résume les changements de ces sections. Le reste de la RFC 2256 est rendu obsolète par la [RFC4519] et [RFC4517].

Le présent document rend obsolète la RFC 3674 dans sa totalité. L'Appendice A.4 résume les changements depuis la RFC 3674.

1.2 Relations avec X.501

Le présent document inclut des matériaux, avec et sans adaptation, tirés de la Recommandation UIT-T X.501, dans la mesure nécessaire pour décrire le présent protocole. Ces adaptations (et toutes autres différences qu'elles contiennent) s'applique au présent protocole, et seulement à lui.

1.3 Conventions

Les mots clé "DOIT", "NE DOIT PAS", "DEVRAIT", "NE DEVRAIT PAS", "PEUT", et "FACULTATIF" dans le présent document sont à interpréter comme décrit dans le BCP 14 [RFC2119].

Les définitions de schéma sont fournies en utilisant les formats de description LDAP (comme défini au paragraphe 4.1). Les définitions fournies ici sont formatées (avec des sauts de ligne) pour les rendre lisibles. Les règles de correspondance et les syntaxes LDAP référencées dans ces définitions sont spécifiées dans la [RFC4517].

1.4 Productions ABNF communes

Un certain nombre des syntaxes du présent document sont décrites en utilisant la forme Backus-Naur augmentée (ABNF) [RFC4234]. Ces syntaxes (ainsi qu'un certain nombre de syntaxes définies dans d'autres documents) s'appuient sur les productions communes suivantes :

```
keystring       = leadkeychar *keychar
leadkeychar     = ALPHA
```

```
keychar         = ALPHA / DIGIT / HYPHEN
number          = DIGIT / ( LDIGIT 1*DIGIT )
ALPHA           = %x41-5A / %x61-7A   ; "A"-"Z" / "a"-"z"
DIGIT           = %x30 / LDIGIT       ; "0"-"9"
LDIGIT          = %x31-39             ; "1"-"9"
HEX             = DIGIT / %x41-46 / %x61-66 ; "0"-"9" / "A"-"F" / "a"-"f"

SP              = 1*SPACE  ; une ou plusieurs " "
WSP             = 0*SPACE  ; zéro, un ou plusieurs " "
NULL            = %x00 ; nul (0)
SPACE           = %x20 ; espace (" ")
DQUOTE          = %x22 ; guillemets (""")
SHARP           = %x23 ; octothorpe (ou signe dièse) ("#")
DOLLAR          = %x24 ; signe dollard ("$")
SQUOTE          = %x27 ; guillemets simples ("'")
LPAREN          = %x28 ; parenthèse gauche ("(")
RPAREN          = %x29 ; parenthèse droite (")")
PLUS            = %x2B ; signe plus ("+")
COMMA           = %x2C ; virgule (",")
HYPHEN          = %x2D ; trait d'union ("-")
DOT             = %x2E ; point (".")
SEMI            = %x3B ; point-virgule (";")
LANGLE          = %x3C ; signe inférieur à ("<")
EQUALS          = %x3D ; signe égals ("=")
RANGLE          = %x3E ; signe supérieur à (">")
ESC             = %x5C ; barre oblique ("\")
USCORE          = %x5F ; souligné ("_")
LCURLY          = %x7B ; accolade gauche "{"
RCURLY          = %x7D ; accolade droite "}"
; Tout caractère Unicode [Unicode] codé en UTF-8 [RFC3629]
UTF8            = UTF1 / UTFMB
UTFMB           = UTF2 / UTF3 / UTF4
UTF0            = %x80-BF
UTF1            = %x00-7F
UTF2            = %xC2-DF UTF0
UTF3            = %xE0 %xA0-BF UTF0 / %xE1-EC 2(UTF0) / %xED %x80-9F UTF0 / %xEE-EF 2(UTF0)
UTF4            = %xF0 %x90-BF 2(UTF0) / %xF1-F3 3(UTF0) / %xF4 %x80-8F 2(UTF0)
OCTET           = %x00-FF ; tout octet (unité de données de 8 bits)
```

Les identifiants d'objet (OID) [X.680] sont représentés en LDAP en utilisant un format décimal à séparation des octets par des points conforme à l'ABNF :

```
numericoid = number 1*( DOT number )
```

Les noms abrégés, appelés aussi des descripteurs, sont utilisés comme des alias plus lisibles pour les identifiants d'objet. Les noms abrégés sont insensibles à la casse et se conformément à l'ABNF :

```
descr = keystring
```

Lorsqu'un identifiant d'objet ou un nom abrégé peut être spécifié, on utilise la formule suivante :

```
oid = descr / numericoid
```

Alors que la forme <descr> est généralement préférée lorsque l'utilisation est restreinte aux noms abrégés qui se réfèrent aux identifiants d'objet qui identifient des sortes d'objets semblables (par exemple, des descriptions de type d'attribut, des descriptions de règles de correspondance, des descriptions de classe d'objet), la forme <numericoid> devrait être utilisée lorsque les identifiants d'objet peuvent identifier plusieurs sortes d'objets ou lorsqu'un nom abrégé non ambigu (descripteur) n'est pas disponible.

Les mises en œuvre DEVRAIENT traiter les noms abrégés (descripteurs) utilisés de façon ambiguë (comme exposé ci-dessus) comme non reconnus.

Les noms abrégés (descripteurs) sont exposés en détail au paragraphe 6.2.

2 Modèle d'informations d'utilisateur de répertoire

Comme [X.501] déclare :

> L'objet du répertoire est de détenir, et de fournir l'accès à, des informations sur des objets d'intérêt (objets) dans un certain 'monde'. Un objet peut être n'importe quoi d'identifiable (qu'on peut nommer). Une classe d'objets est une famille d'objets identifiés, ou d'objets concevables, qui partagent certaines caractéristiques. Chaque objet appartient au moins à une classe. Une classe d'objets peut être une sous-classe d'autres classes d'objets, auquel cas les membres de la première classe, la sous-classe, sont aussi considérés comme étant membres de la dernière classe, la super classe. Il peut y avoir des sous classes de sous classes, etc., à une profondeur arbitraire.

Une entrée de répertoire, une collection nommée d'informations, est l'unité de base des informations détenues dans le répertoire. Il y a plusieurs sortes d'entrées de répertoire.

Une entrée d'objet représente un objet particulier. Une entrée d'alias fournit une dénomination de remplacement. Une sous-entrée détient des informations administratives et/ou de fonctionnement.

L'ensemble des entrées représentant le DIB est organisé hiérarchiquement dans une structure arborescente appelée l'arbre des informations de répertoire (DIT, *Directory Information Tree*).

Le paragraphe 2.1 décrit l'arbre des informations de répertoire.
Le paragraphe 2.2 expose la structure des entrées.
Le paragraphe 2.3 expose la dénomination des entrées.
Le paragraphe 2.4 expose les classes d'objet.
Le paragraphe 2.5 expose la description des attributs.
Le paragraphe 2.6 expose les pseudonymes (alias) d'entrées.

2.1 L'arbre des informations de répertoire

Comme noté ci-dessus, le DIB se compose d'un ensemble d'entrées organisées hiérarchiquement en une structure arborescente connue sous le nom d'arbre des informations de répertoire (DIT) ; précisément, un arbre où les extrémités sont les entrées.

Les arcs entre les extrémités définissent les relations entre les entrées. Si un arc existe de X à Y, l'entrée à X est alors le supérieur immédiat de Y, et Y est le subordonné immédiat de X. Les supérieurs d'une entrée sont les supérieurs immédiats de l'entrée et ses supérieurs. Les subordonnés d'une entrée sont tous ses subordonnés immédiats et leurs subordonnés.

De même, la relation de supérieur/subordonné entre les entrées d'objet peuvent être utilisées pour déduire une relation entre les objets qu'elle représente. Les règles de la structure DIT peuvent être utilisées pour régler les relations entre les objets.

Note : Le supérieur immédiat d'une entrée est aussi appelé le parent de l'entrée, et un subordonnée immédiat d'une entrée est aussi appelé le fils d'une entrée. Les entrées qui ont le même parent sont appelées de jumeaux.

2.2 Structure d'une entrée

Une entrée consiste en un ensemble d'attributs qui détiennent des informations sur l'objet que représente l'entrée. Certains attributs représentent des informations d'utilisateur et sont appelée attributs d'utilisateur. D'autres attributs représentent des informations opérationnelles et/ou administratives et sont appelées attributs de fonctionnement.

Un attribut est une description d'attribut (un type et zéro, une ou plusieurs options) avec une ou plusieurs valeurs associées. On se réfère souvent à un attribut par sa description d'attribut. Par exemple, l'attribut 'givenName' est l'attribut qui consiste en la description d'attribut 'givenName' (le type d'attribut 'givenName' [RFC4519] et zéro option) et une ou plusieurs valeurs associées.

Le type d'attribut commande si l'attribut peut avoir plusieurs valeurs, la syntaxe et règles de correspondance utilisées pour construire et comparer les valeurs de cet attribut, et d'autres fonctions. Les options indiquent les sous-types et d'autres fonctions.

Les valeurs d'attribut se conforment à la syntaxe définie du type d'attribut. Deux valeurs d'un attribut ne peuvent pas être équivalentes. Deux valeurs sont considérées équivalentes si et seulement si elle correspondent conformément à la règle de confrontation d'égalité du type d'attribut. Ou, si le type d'attribut est défini sans règle de confrontation d'égalité, deux valeurs sont équivalentes si et seulement si elles sont identiques. (Voir au paragraphe 2.5.1 les autres restrictions.)

Par exemple, un attribut 'givenName' peut avoir plus d'une valeur, elles doivent être des chaînes de répertoire, et elles sont insensibles à la casse. Un attribut 'givenName' ne peut pas détenir "John" et "JOHN", car ce sont des valeurs équivalentes selon la règle de confrontation d'égalité du type d'attribut.

De plus, aucun attribut ne doit avoir une valeur qui ne soit pas équivalente à elle-même. Par exemple, l'attribut 'givenName' ne peut pas avoir comme valeur une chaîne de répertoire qui inclut le codet REPLACEMENT CHARACTER (U+FFFD), car la confrontation impliquant cette chaîne de répertoire est Undefined selon la règle de confrontation d'égalité de cet attribut.

Lorsqu'un attribut est utilisé pour nommer l'entrée, une valeur de l'attribut et une seule est utilisée dans la formation du nom distinctif relatif. Cette valeur est appelée une valeur distinctive.

2.3 Dénomination des entrées

2.3.1 Noms distinctifs relatifs

Chaque entrée est nommée par rapport à son supérieur immédiat. Ce nom relatif, appelé son nom distinctif relatif (RDN, *Relative Distinguished Name*) [X.501], se compose d'un ensemble non ordonné d'une ou plusieurs assertions de valeur d'attribut (AVA, *Attribute Value Assertion*) consistant en une description d'attribut avec zéro option et une valeur d'attribut. Ces AVA sont choisis de façon à correspondre à des valeurs d'attribut (chacune d'une valeur distinctive) de l'entrée.

Un nom distinctif relatif d'une entrée doit être unique parmi tous les subordonnés immédiats du supérieur immédiat de l'entrée (c'est-à-dire, tous les jumeaux).

Ci-après figurent des exemples de représentations de chaînes de RDN [RFC4514] :

```
UID=12345
OU=Engineering
CN=Kurt Zeilenga+L=Redwood Shores
```

Le dernier est un exemple de RDN multi valeurs ; c'est-à-dire, un RDN composé de plusieurs AVA.

2.3.2 Noms distinctifs

C'est le nom pleinement qualifié d'une entrée, appelée son nom distinctif (DN, *Distinguished Name*) [X.501]. C'est la concaténation de son RDN et du DN de son supérieur immédiat. Un nom distinctif se réfère sans ambiguïté à une entrée dans l'arborescence. Ci-après figurent des exemples de représentations de chaînes de DN [RFC4514]:

```
UID=nobody@example.com,DC=example,DC=com
CN=John Smith,OU=Sales,O=ACME Limited,L=Moab,ST=Utah,C=US
```

2.3.3 Pseudonymes

Un alias, ou pseudonyme, est "un nom pour un objet, fourni par l'utilisation des entrées de pseudonymes" [X.501]. Les entrées d'alias sont décrites au paragraphe 2.6.

2.4 Classes d'objet

Une classe d'objet est "une famille identifiée d'objets (ou d'objets concevables) qui partagent certaines caractéristiques" [X.501].

Comme défini dans [X.501] :
Les classes d'objet sont utilisées dans le répertoire pour un certain nombre de propos :
- décrire et catégoriser des objets et les entrées qui correspondent à ces objets,
- où c'este approprié, contrôler le fonctionnement du répertoire,
- réguler, en conjonction avec les spécifications de règle de structure de DIT, la position des entrées dans le DIT,
- réguler, en conjonction avec les spécifications de règle de contenu de DIT, les attributs qui sont contenus dans les entrées,
- identifier les classes d'entrée qui sont à associer à une politique particulière par l'autorité administrative appropriée.

Une classe d'objet (une sous-classe) peut être déduite d'une classe d'objet (sa super classe directe) qui est elle-même déduite d'une classe d'objet encore plus générique. Pour les classes d'objet structurelles, ce processus s'arrête à la classe d'objet la plus générique, 'top' (définie au paragraphe 2.4.1). Un ensemble ordonné de super classes jusqu'à la classe d'objet la plus élevée d'une classe d'objet est sa chaîne de super classe.

Une classe d'objet peut être déduite de deux super classes directes ou plus (les super classes qui ne font pas partie de la même chaîne de super classe). Cette caractéristique de sous classement est appelée héritage multiple.

Chaque classe d'objet identifie l'ensemble d'attributs dont la présence est nécessaire dans les entrées appartenant à la classe et l'ensemble des attributs dont la présence est autorisée dans les entrées appartenant à la classe. Comme une entrée de la classe doit satisfaire aux exigences de chaque classe à laquelle elle appartient, on peut dire qu'une classe d'objet hérite des ensembles des attributs admis et exigés de ses super classes. Une sous classe peut identifier un attribut admis par se super classe s'il en est besoin. Si un attribut est a membre des deux ensembles, il est obligé d'être présent.

Chaque classe d'objet est définie comme étant d'une des trois classes d'objet : Abstraite, Structurelle, ou Auxiliaire.

Chaque classe d'objet est identifiée par un identifiant d'objet (OID) et, facultativement, par un ou plusieurs noms abrégés (descripteurs).

2.4.1 Classes d'objets abstraites

Une classe d'objet abstraite, comme son nom l'indique, fournit une base de caractéristiques d'après lesquelles les autres classes d'objet peuvent être définies pour en hériter. Une entrée ne peut pas appartenir à une classe d'objet abstraite si elle n'appartient pas à une classe structurelle ou auxiliaire qui hérite de cette classe abstraite.

Les classes d'objet abstraites ne peuvent pas dériver de classes d'objet structurelles ou auxiliaires.

Toutes les classes d'objet structurelles dérivent (directement ou indirectement) de la classe d'objet abstraite 'top'. Les classes d'objet auxiliaires ne dérivent pas nécessairement de la classe 'top'.

Ce qui suit est la définition de classe d'objet (voir au paragraphe 4.1.1) pour la classe d'objet 'top' :

(2.5.6.0 NAME 'top' ABSTRACT MUST objectClass)

Toutes les entrées appartiennent à la classe d'objet abstraite 'top'.

2.4.2 Classes d'objets Structurelles

Comme indiqué dans [X.501] :

Une classe d'objet définie pour être utilisée dans la spécification structurelle du DIT est appelée une classe d'objet structurelle. Les classes d'objet structurelles sont utilisées dans la définition de la structure des noms des objets pour les entrées conformes.

Une entrée d'objet ou d'alias est précisément caractérisée par une chaîne de super classe de classe d'objet structurelle avec une seule classe d'objet structurelle comme classe d'objet la plus subordonnée. Cette classe d'objet structurelle est mentionnée comme la classe d'objet structurelle de l'entrée.

Les classes d'objet structurelles se rapportent aux entrées associées :
- une entrée qui se conforme à une classe d'objet structurelle doit représenter l'objet réel avec les contraintes de la classe d'objet,
- les règles de structure de DIT ne se réfèrent qu'aux classes d'objet structurelles ; la classe d'objet structurelle d'une entrée est utilisée pour spécifier la position de l'entrée dans le DIT,
- la classe d'objet structurelle d'une entrée est utilisée, en conjonction avec une règle de contenu DIT associée, pour contrôler le contenu d'une entrée.

La classe d'objet structurelle d'une entrée ne doit pas être changée.

Chaque classe d'objet structurelle est une sous classe (directe ou indirecte) de la classe d'objet abstraite 'top'.

Les classes d'objet structurelles ne peuvent pas comprendre en sous classe des classes d'objet auxiliaires.
Chaque entrée est dite appartenir à sa classe d'objet structurelle aussi bien qu'à toutes les classes dans sa chaîne de super classe de classe d'objet structurelle.

2.4.3 Classes d'objets auxiliaires

Les classes d'objet auxiliaires sont utilisées pour augmenter les caractéristiques des entrées. Elles sont communément utilisées pour augmenter les ensembles d'attributs dont la présence est exigée et permise dans une entrée. Elles peuvent être utilisées pour décrire les entrées ou classes d'entrées.

Les classes d'objet auxiliaires ne peuvent pas comporter de sous classes de classes d'objet structurelles.

Une entrée peut appartenir à tout sous-ensemble de classes d'objet auxiliaires permis par la règle de contenu de DIT associée à la classe d'objet structurelle de l'entrée. Si aucune règle de contenu de DIT n'est associée à la classe d'objet structurelle de l'entrée, l'entrée ne peut pas appartenir à une classe d'objet auxiliaire quelconque.

L'ensemble des classes d'objet auxiliaires auquel peut appartenir une entrée peut changer dans le temps.

2.5 Descriptions d'attribut

Une description d'attribut se compose d'un type d'attribut (voir au paragraphe 2.5.1) et d'un ensemble de zéro, une ou plusieurs options d'attribut (voir au paragraphe 2.5.2).

Une description d'attribut est représentée par l'ABNF :

```
attributedescription = attributetype options
attributetype = oid
options = *( SEMI option )
option = 1*keychar
```

où <attributetype> identifie le type d'attribut et chaque <option> identifie une option d'attribut. Les produits <attributetype> et <option> sont tous deux insensibles à la casse. L'ordre dans lequel <option>s apparaît est sans importance. C'est à dire que tout ensemble de deux <attributedescription> qui consistent en le même ensemble de <attributetype> et le même ensemble de <option> sont équivalents.

Exemples de descriptions d'attribut valides :

2.5.4.0
cn;lang-de;lang-en
owner

Une description d'attribut dont le type d'attribut n'est pas reconnu est à traiter comme non reconnue. Les serveurs DOIVENT traiter une description d'attribut avec une option d'attribut non reconnue comme non reconnue. Les clients PEUVENT traiter une option d'attribut non reconnu comme une option de marquage (voir au paragraphe 2.5.2.1).

Tous les attributs d'une entrée doivent avoir des descriptions d'attribut distinctes.

2.5.1 Types d'attribut

Un type d'attribut commande si l'attribut peut avoir plusieurs valeurs, la syntaxe et les règles de correspondance utilisées pour construire et comparer les valeurs de cet attribut, et d'autres fonctions.

Si aucune confrontation d'égalité n'est spécifiée pour le type d'attribut :
- l'attribut (du type) ne peut pas être utilisé pour nommer,
- lors de l'ajout de l'attribut (ou replacement de toutes les valeurs), deux valeurs ne peuvent être équivalentes (voir au paragraphe 2.2),
- les valeurs individuelles d'un attribut multi-valeurs ne doivent pas être ajoutées ou retranchées indépendamment,
- les assertions de valeur d'attribut (comme la correspondance dans les filtres de recherche et les comparaisons) qui utilisent des valeurs d'un tel type ne peuvent pas être effectuées.

Autrement, la règle de confrontation d'égalité spécifiée doit être utilisée pour évaluer les assertions de valeur d'attribut concernant le type d'attribut. La règle d'égalité spécifiée doit être transitive et commutative.

Le type d'attribut indique si l'attribut est un attribut d'utilisateur ou un attribut de fonctionnement. Si c'est un attribut de fonctionnement, le type d'attribut indique l'utilisation opérationnelle et si l'attribut est modifiable ou non par les utilisateurs. Les attributs de fonctionnement sont exposés au paragraphe 3.4.

Un type d'attribut (un sous-type) peut dériver d'un type d'attribut plus général (un super-type direct). Les restrictions suivantes s'appliquent à la sous-typologie :
- un sous-type doit avoir la même utilisation que son super-type direct,
- une syntaxe de sous-type doit être la même, ou plus raffinée, que la syntaxe de son super-type, et
- un sous-type doit être collectif [RFC3671] si son super-type est collectif.

Une description d'attribut consistant en un sous-type et pas d'option est dit être le sous-type de description directe de la description d'attribut consistant en le super-type direct et pas d'option du sous-type.

Chaque type d'attribut est identifié par un identifiant d'objet (OID) et, facultativement, un ou plusieurs noms abrégés (descripteurs).

2.5.2 Options d'attribut

Il y a plusieurs sortes de descriptions d'options d'attribut. La spécification technique LDAP précise une sorte : les options de marquage.

Toutes les options ne peuvent pas être associées à des attributs détenus dans le répertoire. Les options de marquage peuvent l'être.

Toutes les options ne peuvent pas être utilisées en conjonction avec tous les types d'attribut. Dans de tels cas, la description d'attribut est à traiter comme non reconnue.

Une description d'attribut qui contient des options qui s'excluent mutuellement doit être traitée comme non reconnue. C'est à dire que "cn;x-bar;x-foo", où "x-foo" et "x-bar" s'excluent mutuellement, est à traiter comme non reconnue.

D'autres sortes d'options pourront être spécifiées dans de futurs documents. Ces documents devront préciser comment les nouvelles sortes d'options qu'ils définissent se rapportent aux options de marquage. En particulier, ces documents devront préciser si de nouvelles sortes d'options peuvent être ou non associées avec des attributs détenus dans le

répertoire, comment les nouvelles sortes d'option affectent le transfert des valeurs d'attribut, et comment de nouvelles sortes d'options sont traitées dans les hiérarchies de description d'attribut.

Les options sont représentées comme de courtes chaînes textuelles, insensibles à la casse, se conformant à la production de <option> définie au paragraphe 2.5 du présent document.

Les options de procédures d'enregistrement sont précisées dans le BCP 64, RFC 4520 [RFC4520].

2.5.2.1 Options de marquage

Les attributs détenus dans le répertoire peut avoir des descriptions d'attribut avec un nombre quelconque d'options de marquage. Les options ne marquage ne s'excluent jamais mutuellement.

Une description d'attribut avec N options de marquage est un sous-type direct (description) de toutes les descriptions d'attribut du même type d'attribut et toutes sauf une des N options. Si le type d'attribut a un super-type, la description d'attribut est alors aussi un sous-type direct (description) de la description d'attribut du super-type et des N options de marquage. C'est à dire que 'cn;lang-de;lang-en' est un sous-type direct (description) de 'cn;lang-de', 'cn;lang-en', et 'name;lang-de;lang-en' ('cn' est un sous-type de 'name' ; tous deux sont définis dans la [RFC4519]).

2.5.3 Hiérarchies de description d'attribut

Une description d'attribut peut être le sous-type direct de zéro, une ou plusieurs autres descriptions d'attribut comme indiqué par la subdivision en sous-types de type d'attribut (comme décrit au paragraphe 2.5.1) ou la subdivision en sous-types d'option de marquage d'attribut (comme décrit au paragraphe 2.5.2.1). Ces relations de subdivision en sous-types sont utilisées pour former des hiérarchies de descriptions d'attribut et d'attributs.

D'après [X.501] :
Les hiérarchies d'attributs permettent l'accès au DIB avec divers degrés de granularité. Ceci est obtenu en permettant d'accéder aux composants de valeur des attributs en utilisant leur description d'attribut spécifique (une référence directe à l'attribut) ou une description d'attribut plus générale (une référence indirecte).

Les attributs en relation sémantique peuvent être placés dans une relation hiérarchique, le plus spécialisé étant placé en subordonné du plus général. La recherche ou la restitution d'attributs et de leurs valeurs est rendue plus aisée par la citation des descriptions d'attribut les plus générales ; un élément de filtre ainsi spécifié est évalué pour les descriptions les plus spécialisées ainsi que pour la description citée.

Lorsque des descriptions spécialisées subordonnées sont choisies pour être retournées au titre d'un résultat de recherche, ces descriptions doivent être retournées si elles sont disponibles. Lorsque les descriptions les plus générales sont choisies comme retour au titre du résultat de recherche, les descriptions à la fois générale et spécialisée doivent être retournées, si elles sont disponibles. Une valeur d'attribut doit toujours être retournée comme une valeur de sa propre description d'attribut.

Toutes les descriptions d'attribut dans une hiérarchie d'attributs sont traitées comme des descriptions distinctes et sans relations pour les modifications d'utilisateur du contenu de l'entrée.

Une valeur d'attribut mémorisée dans une entrée d'objet ou d'alias est précisément d'une description d'attribut. La description est indiquée lorsque la valeur est ajoutée à l'entrée dès l'origine.

Pour les besoins de l'administration de sous-schéma de l'entrée, la spécification qu'un attribut est exigé est satisfaite si l'entrée contient une valeur d'une description d'attribut appartenant à une hiérarchie d'attributs où le type d'attribut de cette description est le même que le type de l'attribut exigé. C'est à dire que la spécification "MUST name" est satisfaite par 'name' ou 'name;x-tag-option', mais n'est pas satisfaite par 'CN' ou 'CN;x-tag-option' (même si 'CN' est un sous-type de 'name'). De même, une entrée peut contenir une valeur d'une description d'attribut appartenant à une hiérarchie d'attributs où le type d'attribut de cette description est explicitement inclus dans la définition d'une classe d'objet à laquelle appartient l'entrée ou permise par la règle de contenu de DIT applicable à cette entrée. C'est à dire que 'name' et 'name;x-tag-option' sont permis par "MAY name" (ou par "MUST name"), mais que 'CN' et 'CN;x-tag-option' ne sont pas permis par "MAY name" (ou par "MUST name").

Pour les besoins d'une autre administration de politique, sauf mention contraires dans la spécification du modèle administratif particulier, toutes les descriptions d'attribut dans une hiérarchie d'attributs sont traitées comme des descriptions distinctes et sans relations.

2.6 Entrées d'alias

D'après [X.501] :
Un alias, ou pseudonyme, pour un objet, est un nom de remplacement pour un objet ou une entrée d'objet qui est fourni par l'utilisation des entrées d'alias.

Chaque entrée d'alias contient, au sein de l'attribut 'aliasedObjectName' (appelé l'attribut 'aliasedEntryName' dans in X.500), le nom d'un certain objet. Le nom distinctif de l'entrée d'alias est donc aussi un nom pour cet objet.

NOTE - Le nom au sein de 'aliasedObjectName' est dit pointer par l'alias. Ce n'est pas forcément le nom distinctif d'une entrée quelconque.

La conversion d'un pseudonyme en nom d'objet est appelée déréférencement (d'alias) et comprend le remplacement systématique des pseudonymes, lorsqu'ils sont trouvés dans un nom supposé, par la valeur de l'attribut 'aliasedObjectName' correspondant. Le processus peut exiger l'examen de plus d'une entrée d'alias.

Toute entrée particulière dans le DIT peut avoir zéro, un ou plusieurs pseudonymes. Il s'ensuit donc que plusieurs entrées d'alias peuvent pointer sur la même entrée. Une entrée d'alias peut pointer sur une entrée qui n'est pas une entrée d'extrémité et peut pointer sur une autre entrée d'alias.

Une entrée d'alias ne doit avoir aucun subordonné, de sorte qu'une entrée d'alias est toujours une entrée d'extrémité.

Chaque entrée d'alias doit appartenir à la classe d'objet 'alias'.

Une entrée avec la classe d'objet 'alias' doit aussi appartenir à une classe (ou des classes) d'objet, ou être gouvernée par une règle de contenu de DIT, qui permet la présence d'attributs de dénomination convenables.

Exemple :

```
dn: cn=bar,dc=example,dc=com
objectClass: top
objectClass: alias
objectClass: extensibleObject
cn: bar
aliasedObjectName: cn=foo,dc=example,dc=com
```

2.6.1 Classe d'objet 'alias'

Les entrées d'alias apartiennent à la classe d'objet 'alias'.

```
( 2.5.6.1 NAME 'alias'
SUP top STRUCTURAL
MUST aliasedObjectName )
```

2.6.2 Type d'attribut 'aliasedObjectName'

L'attribut 'aliasedObjectName' détient le nom de l'entrée sur lequel pointe un alias. L'attribut 'aliasedObjectName' est appelé l'attribut 'aliasedEntryName' dans X.500.

```
( 2.5.4.1 NAME 'aliasedObjectName'
EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE )
```

La règle de correspondance 'distinguishedNameMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.12) de DistinguishedName sont définies dans la [RFC4517].

3 Informations administratives et opérationnelles de répertoire

La présente section expose des aspects choisis du modèle d'informations administratives et opérationnelles de répertoire X.500 [X.501]. Les mises en œuvre de LDAP PEUVENT prendre en charge d'autres aspects de ce modèle.

3.1 Sous-arbres

Comme défini dans [X.501] :
Un sous-arbre est une collection d'entrées d'objets et d'alias situées sur les extrémités d'un arbre. Les sous-arbres ne contiennent pas de sous-entrées. Le préfixe sous, dans sous-arbre, souligne que les extrémités de base (ou racines) de cet arbre sont habituellement subordonnées à la racine du DIT.

Un sous-arbre commence à une extrémité et s'étend jusqu'à une limite inférieure identifiable, qui peut s'étendre jusqu'aux feuilles. Un sous-arbre est toujours défini au sein d'un contexte qui borde implicitement le sous-arbre. Par exemple, l'extrémité et les limites inférieures d'un sous-arbre définissant une zone de duplication sont bordées par un contexte de dénomination.

3.2 Sous-entrées

Une sous-entrée est une "sorte d'entrée particulière, connue du répertoire, utilisée pour détenir des informations associées à un sous-arbre ou une ramification de sous-arbre" [X.501]. Les sous-entrées sont utilisées dans un répertoire Directory pour des besoins administratifs et opérationnels comme défini dans [X.501]. Leur utilisation dans LDAP est précisée dans la [RFC3672].

Le terme de "(sous-) entrée" dans la présente spécification indique que les serveurs qui mettent en œuvre les modèles de X.500 (93) vont, conformément à X.500(93) comme décrit dans la [RFC3672], utiliser une sous-entrée et que d'autres serveurs vont utiliser des entrées d'objet appartenant à la classe auxiliaire appropriée normalement utilisée avec la sous-entrée (par exemple, 'subschema' pour les sous-entrées de sous-schéma) pour simuler la sous-entrée. Ce RDN d'entrée d'objet DOIT être formé à partir d'une valeur de l'attribut 'cn' (commonName) [RFC4519] (car toutes les sous-entrées sont nommées avec 'cn').

3.3 Attribut 'objectClass'

Chaque entrée dans le DIT a un attribut 'objectClass'.

```
( 2.5.4.0 NAME 'objectClass'
EQUALITY objectIdentifierMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.38 )
```

La règle de correspondance 'objectIdentifierMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.38) d'OBJECT IDENTIFIER sont définies dans la [RFC4517].

L'attribut 'objectClass' spécifie les classes d'objet d'une entrée, qui (entre autres choses) sont utilisées en conjonction avec le schéma de contrôle pour déterminer les attributs permis pour une entrée. Les valeurs de cet attribut peuvent être modifiées par les clients, mais l'attribut 'objectClass' ne peut pas être retiré.

Les serveurs qui suivent les modèles de X.500(93) DOIVENT restreindre les modifications à cet attribut pour empêcher les changements de la classe structurelle de base de l'entrée. C'est à dire qu'on ne peut pas changer une 'person' en un 'country'.

Lors de la création d'une entrée ou de l'ajout d'une valeur de 'objectClass' à une entrée, toutes les super classes des classes désignées DOIVENT être implicitement ajoutées aussi, si elle ne sont pas déjà présentes. C'est à dire, si la classe auxiliaire 'x-a' est une sous-classe de la classe 'x-b', l'ajout de 'x-a' à 'objectClass' cause l'ajout implicite de 'x-b' (s'il n'est pas déjà présent).

Les serveurs DOIVENT mettre des restrictions à la modification de cet attribut pour empêcher la suppression des super classes des valeurs 'objectClass' restantes. C'est à dire que si la classe auxiliaire 'x-a' est une sous-classe de la classe auxiliaire 'x-b' et que l'attribut 'objectClass' contient 'x-a' et 'x-b', une tentative de suppression de seulement 'x-b' de l'attribut 'objectClass' est une erreur.

3.4 Attributs de fonctionnement

Certains attributs, appelés attributs de fonctionnement, sont utilisés ou entretenus par les serveurs pour des besoins administratifs et opérationnels. Comme indiqué dans [X.501] : "IL y a trois variétés d'attributs de fonctionnement : les attributs de fonctionnement de répertoire, les attributs de fonctionnement partagés DSA, et les attributs de fonctionnement spécifiques de DSA".

Un attribut de fonctionnement de répertoire sert à représenter des informations opérationnelles et/ou administratives dans le modèle d'informations de répertoire. Cela inclut les attributs de fonctionnement maintenus par le serveur (par exemple, 'createTimestamp') ainsi que les attributs de fonctionnement qui détiennent des valeurs administrées par l'utilisateur (par exemple, 'ditContentRules').

Un attribut de fonctionnement partagé DSA sert à représenter des informations du modèle d'informations de DSA qui est partagé entre les DSA.

Un attribut de fonctionnement spécifique de DSA sert à représenter des informations du modèle d'informations de DSA qui sont spécifiques du DSA (bien que dans certains cas, il puisse être déduit des informations partagées entre les DSA; par exemple, 'namingContexts').

L'attribut de fonctionnements du modèle d'informations de DSA est précisé dans [X.501].

Les attributs de fonctionnement ne sont normalement pas visibles. Ils ne sont p s retournés dans les résultats de recherche sauf explicitement exigé parle nom.

Ce ne sont pas tous les attributs de fonctionnement qui sont modifiables par l'utilisateur.

Les entrées peuvent contenir, entre autres, les attributs de fonctionnement suivants :
- creatorsName : nom distinctif de l'utilisateur qui a ajouté cette entrée au répertoire,
- createTimestamp : heure à laquelle cette entrée a été ajoutée au répertoire,
- modifiersName : nom distinctif de l'utilisateur qui a le dernier modifié cette entrée, et
- modifyTimestamp : heure à laquelle cette entrée a été modifiée en dernier.

Les serveurs DEVRAIENT maintenir les attributs 'creatorsName', 'createTimestamp', 'modifiersName', et 'modifyTimestamp' pour toutes les entrées du DIT.

3.4.1 'creatorsName'

Cet attribut apparaît dans les entrées qui ont été ajoutées en utilisant le protocole (par exemple, en utilisant l'opération Add). La valeur est le nom distinctif du créateur.

```
( 2.5.18.3 NAME 'creatorsName'
EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )
```

La règle de correspondance 'distinguishedNameMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.12) de DistinguishedName sont définies dans la [RFC4517].

3.4.2 'createTimestamp'

Cet attribut apparaît dans les entrées qui ont été ajoutées en utilisant le protocole (par exemple, en utilisant l'opération Add). La valeur est l'heure à laquelle l'entrée a été ajoutée.

```
( 2.5.18.1 NAME 'createTimestamp'
EQUALITY generalizedTimeMatch
ORDERING generalizedTimeOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )
```

Les règles de correspondance 'generalizedTimeMatch' et 'generalizedTimeOrderingMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.24) de GeneralizedTime sont définies dans la [RFC4517].

3.4.3 'modifiersName'

Cet attribut apparaît dans les entrées qui ont été ajoutées en utilisant le protocole (par exemple, en utilisant l'opération Modify). La valeur est le nom distinctif de la dernière personne qui l'a modifié.

```
( 2.5.18.4 NAME 'modifiersName'
EQUALITY distinguishedNameMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )
```

La règle de correspondance 'distinguishedNameMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.12) de DistinguishedName sont définies dans la [RFC4517].

3.4.4 'modifyTimestamp'

Cet attribut apparaît dans les entrées qui ont été modifiées en utilisant le protocole (par exemple, en utilisant l'opération Modify). La valeur est l'heure à laquelle l'entrée a été modifiée en dernier.

```
( 2.5.18.2 NAME 'modifyTimestamp'
EQUALITY generalizedTimeMatch
ORDERING generalizedTimeOrderingMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.24
SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )
```

Les règles de correspondance 'generalizedTimeMatch' et 'generalizedTimeOrderingMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.24) de GeneralizedTime sont définies dans la [RFC4517].

3.4.5 'structuralObjectClass'

Cet attribut indique la classe d'objet structurelle de l'entrée.

```
( 2.5.21.9 NAME 'structuralObjectClass'
EQUALITY objectIdentifierMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.38) d'OBJECT IDENTIFIER sont définies dans la [RFC4517].

3.4.6 'governingStructureRule'

Cet attribut indique la règle de structure qui gouverne l'entrée.

```
( 2.5.21.10 NAME 'governingStructureRule'
EQUALITY integerMatch
SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
SINGLE-VALUE NO-USER-MODIFICATION
USAGE directoryOperation )
```

La règle de correspondance 'integerMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.27) de INTEGER sont définies dans la [RFC4517].

4 Schéma de répertoire

Comme défini dans [X.501] :

Le schéma de répertoire est un ensemble de définitions et de contraintes qui concernent la structure du DIT, les différentes façons possibles dont les entrées sont nommées, les informations qui peuvent être détenues dans une entrée, les attributs utilisés pour représenter les informations et leur organisation en hiérarchies pour facilite la recherche et la restitution des informations et les façons dont les valeurs des attributs peut être confrontées aux assertions de valeur d'attribut et de règles de correspondance.

NOTE 1 - Le schéma permet au système de répertoire, par exemple :
- d'empêcher la création d'entrées subordonnées dans la mauvaise clase d'objet (par exemple, un pays comme subordonné d'une personne),
- d'empêcher d'ajouter des types d'attribut à une entrée non appropriée à la classe d'objet (par exemple, un numéro de série à une entrée de personne),
- d'empêcher l'ajout d'une valeur d'attribut d'une syntaxe ne correspondant pas à celle définie pour le type d'attribut (par exemple, une chaîne imprimable dans une chaîne binaire).

Formellement, le schéma de répertoire comporte en ensemble de :
a) définitions de formes de nom qui définissent les relations de dénomination primitives pour les classes d'objet structurelles,
b) définitions de règle de structure de DIT qui définissent les noms que peuvent avoir les entrées et les façons dont les entrées peuvent être en relation les unes avec les autres dans les DIT,
c) définitions de règle de contenu de DIT qui étendent la spécification des attributs admissibles pour les entrées au delà de ceux indiqué par les classes d'objet structurelles des entrées,
d) définitions de classe d'objet qui définissent l'ensemble de base des attributs obligatoires et facultatifs qui doivent être présents, et peuvent être présents, respectivement, dans une entrée d'une classe donnée, et qui indiquent le type de classe d'objet qui est en cours de définition,
e) définitions de type d'attribut qui identifient l'identifiant d'objet par lequel est connu un attribut, sa syntaxe, les règles de correspondance associées, si c'est un attribut de fonctionnement et si tel est son type, si c'est un attribut collectif, si il est permis d'avoir plusieurs valeurs et si il est dérivé ou non d'un autre type d'attribut,
f) définitions de règle de correspondance qui définissent les règles de correspondance.
et dans LDAP :
g) définitions de syntaxe LDAP qui définissent les codages utilisés dans LDAP.

4.1 Définitions de schéma

Dans ce paragraphe, les définitions de schéma sont décrites en utilisant l'ABNF et elles s'appuient sur les produits communs spécifiés au paragraphe 1.2, ainsi que sur les suivants :

```
noidlen = numericoid [ LCURLY len RCURLY ]
len = number

oids = oid / ( LPAREN WSP oidlist WSP RPAREN )
```

```
oidlist = oid *( WSP DOLLAR WSP oid )

extensions = *( SP xstring SP qdstrings )
xstring = "X" HYPHEN 1*( ALPHA / HYPHEN / USCORE )

qdescrs = qdescr / ( LPAREN WSP qdescrlist WSP RPAREN )
qdescrlist = [ qdescr *( SP qdescr ) ]
qdescr = SQUOTE descr SQUOTE

qdstrings = qdstring / ( LPAREN WSP qdstringlist WSP RPAREN )
qdstringlist = [ qdstring *( SP qdstring ) ]
qdstring = SQUOTE dstring SQUOTE
dstring = 1*( QS / QQ / QUTF8 )   ; chaîne UTF-8 formatée

QQ =  ESC %x32 %x37 ; "\27"
QS =  ESC %x35 ( %x43 / %x63 ) ; "\5C" / "\5c"
; Tout caractère Unicode codé en UTF-8 excepté %x27 ("\'") et %x5C ("\")
QUTF8    = QUTF1 / UTFMB
; Tout caractère ASCII excepté %x27 ("\'") et %x5C ("\")
QUTF1    = %x00-26 / %x28-5B / %x5D-7F
```

Les définitions de schéma de ce paragraphe partagent aussi un certain nombre de termes communs.

Le champ NAME fournit un ensemble de noms abrégés (descripteurs) qui sont à utiliser comme alias pour l'OID.

Le champ DESC permet facultativement à l'administrateur et/ou celui qui met en oeuvre le répertoire de fournir une chaîne descriptive. Alors que les spécifications peuvent suggérer une chaîne descriptive, il n'y a aucune exigence d'utiliser la chaîne descriptive suggérée.

Le champ OBSOLETE, s'il est présent, indique que l'élément n'est pas actif.

Les développeurs devraient noter que des versions futures du présent document pourront étendre ces définitions pour y inclure des termes supplémentaires. Les termes dont l'identifiant commence par "X-" sont réservés pour des expériences privées et sont suivis par des jetons <SP> et <qdstrings>.

4.1.1 Définitions de classes d'objet

Les définitions de classe d'objet sont écrites conformément à l'ABNF :

```
ObjectClassDescription = LPAREN WSP
        numericoid                  ; identifiant d'objet
        [ SP "NAME" SP qdescrs ]    ; nom abrégés (descripteurs)
        [ SP "DESC" SP qdstring ]   ; description
        [ SP "OBSOLETE" ]           ; non actif
        [ SP "SUP" SP oids ]        ; classes d'objet supérieures
        [ SP kind ]                 ; sorte de classe
        [ SP "MUST" SP oids ]       ; type d'attributs
        [ SP "MAY" SP oids ]        ; type d'attributs
        extensions WSP RPAREN

kind = "ABSTRACT" / "STRUCTURAL" / "AUXILIARY"
```

où :
<numericoid> est l'identifiant d'objet alloué à cette classe d'objet,
NAME <qdescrs> sont les noms abrégés (descripteurs) qui identifient cette classe d'objet,
DESC <qdstring> est une brève chaîne descriptive,
OBSOLETE indique que cette classe d'objet n'est pas active,
SUP <oids> spécifie les superclasses directes de cette classe d'objet ; la sorte de classe d'objet est indiquée par l'un de ABSTRACT, STRUCTURAL, ou AUXILIARY (STRUCTURALest la valeur par défaut) ;
MUST et MAY spécifient, respectivement, les ensembles de type d'attributs exigés et permis ; et

<extensions> décrit les extensions.

4.1.2 Types d'attribut

Les définitions de type d'attribut sont écrites conformément à l'ABNF :

```
AttributeTypeDescription = LPAREN WSP
        numericoid                          ; identifiant d'objet
        [ SP "NAME" SP qdescrs ]            ; noms abrégés (descripteurs)
        [ SP "DESC" SP qdstring ]           ; description
        [ SP "OBSOLETE" ]                   ; non actif
        [ SP "SUP" SP oid ]                 ; super type
        [ SP "EQUALITY" SP oid ]            ; règle de confrontation d'égalité
        [ SP "ORDERING" SP oid ]            ; règle de correspondance d'ordre
        [ SP "SUBSTR" SP oid ]              ; règle de correspondance de sous-chaînes
        [ SP "SYNTAX" SP noidlen ]           ; syntaxe de valeur
        [ SP "SINGLE-VALUE" ]                ; valeur unique
        [ SP "COLLECTIVE" ]                 ; collectif
        [ SP "NO-USER-MODIFICATION" ]       ; non modifiable par l'utilisateur
        [ SP "USAGE" SP usage ]             ; usage
        extensions WSP RPAREN               ; extensions

usage = "userApplications"     /            ; utilisateur
          "directoryOperation"   /          ; fonctionnement de répertoire
          "distributedOperation" /          ; fonctionnement DSA partagé
          "dSAOperation"                    ; fonctionnement spécifique de DSA
```

où :
<numericoid> est l'identifiant d'objet alloué à ce type d'attribut,
NAME <qdescrs> sont des noms abrégés (descripteurs) qui identifient ce type d'attribut,
DESC <qdstring> est une courte chaîne descriptive,
OBSOLETE indique que ce type d'attribut n'est pas actif,SUP oid spécifie le super type direct de ce type,
EQUALITY, ORDERING, et SUBSTR fournissent l'oid des règles de correspondance de, respectivement equality, ordering, et substrings,
SYNTAX identifie la syntaxe de valeur par identifiant d'objet et peut suggérer une limite supérieure minimum,
SINGLE-VALUE indique que les attributs de ce type sont restreints à une seule valeur,
COLLECTIVE indique que ce type d'attribut est collectif [X.501][RFC3671],
NO-USER-MODIFICATION indique que ce type d'attribut n'est pas modifiable par l'utilisateur,
USAGE indique l'application de ce type d'attribut, et
<extensions> décrit les extensions.

Chaque description de type d'attribut doit contenir au moins un des champs SUP ou SYNTAX. Si le champ SYNTAX n'est pas fourni, la description de type d'attribut tire sa valeur du super type.

Si le champ SUP est fourni, les champs EQUALITY, ORDERING, et SUBSTRING, s'ils ne sont pas spécifiés, tirent leur valeur du super type.

L'utilisation de userApplications, la valeur par défaut, indique que les attributs de ce type représentent des informations d'utilisateur. C'est à dire que ce sont des attributs d'utilisateur.

Une utilisation de directoryOperation, distributedOperation, ou dSAOperation indique que les attributs de ce type représentent des informations opérationnelles et/ou administratives. C'est à dire qu'ils sont des attributs de fonctionnement.

L'utilisation de directoryOperation indique que l'attribut de ce type est un attribut de fonctionnement de répertoire. L'utilisation de distributedOperation indique que l'attribut de ce type est un attribut de fonctionnement à utilisation partagée DSA. L'utilisation de dSAOperation indique que l'attribut de ce type est un attribut de fonctionnement spécifique de DSA.

COLLECTIVE exige l'utilisation de userApplications. L'utilisation de types d'attributs collectifs dans LDAP est discutée dans la [RFC3671].

NO-USER-MODIFICATION exige une utilisation de fonctionnement.

Noter que <AttributeTypeDescription> ne fait pas la liste des règles de correspondance qui peuvent être utilisées avec ce type d'attribut dans un filtre de recherche extensibleMatch [RFC4511]. Ceci est fait en utilisant l'attribut 'matchingRuleUse' décrit au paragraphe 4.1.4.

Le présent document précise la description de schéma de X.501 en exigeant que le champ SYNTAX dans une <AttributeTypeDescription> soit une représentation de chaîne d'un identifiant d'objet pour la définition de syntaxe de chaîne LDAP, avec une indication facultative de la limite suggérée minimum d'une valeur de cet attribut.

Une limite supérieure minimum suggérée du nombre de caractères dans une valeur avec une syntaxe fondée sur la chaîne, ou le nombre d'octets dans une valeur pour toutes les autres syntaxes, peut être indiquée en ajoutant ce compte de limite entre des accolades suivant le OBJECT IDENTIFIER de la syntaxe dans une description de type d'attribut. Cette limite ne fait pas partie du nom de la syntaxe lui-même. Par exemple, "1.3.6.4.1.1466.0{64}" suggère que les mises en œuvre de serveur devraient permettre à une chaîne une longueur de 64 caractères, bien qu'elles puissent permettre des chaînes plus longues. Noter qu'un seul caractère de la syntaxe de chaîne de répertoire peut être codée dans plus d'un octet car UTF-8 [RFC3629] est un codage à longueur variable.

4.1.3 Règles de correspondance

Les règles de correspondance sont utilisées pour effectuer les assertions de valeur d'attribut, comme dans la réalisation de l'opération Compare. Elles sont aussi utilisées pour évaluer les filtres de recherche, déterminer quelles valeurs individuelles sont à ajouter ou retrancher en effectuant une opération Modify, et en comparant les noms distinctifs.

Chaque règle de correspondance est identifiée par un identifiant d'objet (OID) et, facultativement, par un ou plusieurs noms abrégés (descripteurs).

Les définitions de règles de correspondance sont écrites conformément à l'ABNF :

```
MatchingRuleDescription = LPAREN WSP
      numericoid                  ; identifiant d'objet
      [ SP "NAME" SP qdescrs ]    ; noms abrégés (descripteurs)
      [ SP "DESC" SP qdstring ]   ; description
      [ SP "OBSOLETE" ]           ; non actif
      SP "SYNTAX" SP numericoid   ; syntaxe d'assertion
      extensions WSP RPAREN       ; extensions
```

où :

<numericoid> est l'identifiant d'objet alloué à cette règle de correspondance,
NAME <qdescrs> sont les noms abrégés (descripteurs) qui identifient cette règle de correspondance,
DESC <qdstring> est une courte chaîne descriptive,
OBSOLETE indique que cette règle de correspondance n'est pas active,
SYNTAX identifie la syntaxe d'assertion (la syntaxe de la valeur de l'assertion) par l'identifiant d'objet, et
<extensions> décrit les extensions.

4.1.4 Utilisations des règles de correspondance

Une utilisation de règle de correspondance fait la liste des types d'attribut qu'il convient d'utiliser avec un filtre de recherche extensibleMatch.

Les descriptions d'utilisation de règle de correspondance sont écrites conformément à l'ABNF suivant :

```
MatchingRuleUseDescription = LPAREN WSP
      numericoid                  ; identifiant d'objet
      [ SP "NAME" SP qdescrs ]    ; noms abrégés (descripteurs)
      [ SP "DESC" SP qdstring ]   ; description
      [ SP "OBSOLETE" ]           ; non actif
```

```
        SP "APPLIES" SP oids          ; types d'attribut
        extensions WSP RPAREN         ; extensions
```

où :
<numericoid> est l'identifiant d'objet de la règle de correspondance associée à la description d'utilisation de la règle de correspondance,
NAME <qdescrs> sont les noms abrégés (descripteurs) qui identifient cette utilisation de règle de correspondance,
DESC <qdstring> est une courte chaîne descriptive,
OBSOLETE indique que cette utilisation de règle de correspondance n'est pas active,
APPLIES donne une liste des types d'attribut qui s'appliquent à la règle de correspondance, et
<extensions> décrit les extensions.

4.1.5 Syntaxes LDAP

Les syntaxes LDAP de valeurs (attribut et assertion) sont décrites en termes d'ASN.1 [X.680] et, facultativement, ont un codage de chaîne d'octet connue sous le nom de codage spécifique LDAP. Habituellement le codage spécifique de LDAP est restreint à une chaîne de caractères Unicode [Unicode] en forme UTF-8 [RFC3629].

Chaque syntaxe LDAP est identifiée par un identifiant d'objet (OID).

Les définitions de syntaxe LDAP son écrites conformément à l'ABNF :

```
SyntaxDescription = LPAREN WSP
        numericoid                ; identifiant d'objet
        [ SP "DESC" SP qdstring ] ; description
        extensions WSP RPAREN     ; extensions
```

où :
<numericoid> est l'identifiant d'objet alloué à cette syntaxe LDAP,
DESC <qdstring> est une courte chaîne descriptive, et
<extensions> décrit les extensions.

4.1.6 Règles de contenu de DIT

Une règle de contenu de DIT est une "règle gouvernant le contenu des entrées d'une classe d'objet structurelle particulière" [X.501].

Pour les entrées de DIT d'une classe d'objet structurelle particulière, une règle de contenu de DIT spécifie à quelles classes d'objet auxiliaires les entrées sont autorisées à appartenir et quels attributs supplémentaires (par type) sont exigés, ou non autorisés à apparaître dans les entrées.

La liste des attributs exclus ne peut pas inclure d'attribut figurant sur la liste de ceux que la règle fixe comme obligatoires, de ceux de la classe d'objet structurelle, ou un de ceux des classes d'objet auxiliaires admises.

Chaque règle de contenu est identifiée par l'identifiant d'objet, ainsi que par tout nom abrégé (descripteur), de la classe d'objet structurelle à laquelle il s'applique.

Une entrée ne peut appartenir qu'aux classes d'objet auxiliaires dont la liste figure dans la règle de contenu qui la gouverne.

Une entrée doit contenir tous les attributs exigés par les classes d'objet auxquelles appartient l'entrée ainsi que tous les attributs requis par la règle de contenu qui la gouverne.

Une entrée peut contenir tous les attributs non exclus admis par les classes d'objet auxquelles appartient l'entrée ainsi que tous les attributs requis par la règle de contenu qui la gouverne.

Une entrée ne peut pas inclure d'attribut exclu par la règle de contenu qui la gouverne.

Une entrée est gouvernée par (si elle est présente et active dans le sous schéma) la règle de contenu de DIT qui s'applique à la classe d'objet structurelle de l'entrée (voir au paragraphe 2.4.2). Si aucune règle active n'est présente

pour la classe d'objet structurelle de l'entrée, le contenu de l'entrée est gouverné par la classe d'objet structurelle (et éventuellement d'autres aspects de schéma d'utilisateur et de système). Les règles de contenu de DIT pour les super classes de la classe d'objet structurelle d'une entrée ne sont pas applicables à cette entrée.

Les descriptions de règle de contenu de DIT sont écrites conformément à l'ABNF :

```
DITContentRuleDescription = LPAREN WSP
        numericoid                  ; identifiant d'objet
        [ SP "NAME" SP qdescrs ]    ; noms abrégés (descripteurs)
        [ SP "DESC" SP qdstring ]   ; description
        [ SP "OBSOLETE" ]           ; non active
        [ SP "AUX" SP oids ]        ; classes d'objet auxiliaires
        [ SP "MUST" SP oids ]       ; type d'attributs
        [ SP "MAY" SP oids ]        ; type d'attributs
        [ SP "NOT" SP oids ]         ; type d'attributs
        extensions WSP RPAREN       ; extensions
```

où :
<numericoid> est l'identifiant d'objet de la classe d'objet structurelle associée à cette règle de contenu de DIT,
NAME <qdescrs> sont les noms abrégés (descripteurs) identifiant cette règle de contenu de DIT,
DESC <qdstring> est une courte chaîne descriptive,
OBSOLETE indique que cette utilisation de règle de contenu de DIT n'est pas active,
AUX spécifie une liste de classes d'objet auxiliaires auxquelles les entrées peuvent appartenir sous réserve de cette règle de contenu de DIT,
MUST, MAY, et NOT spécifient des listes de types d'attributs dont l'apparition dans les entrées soumises à cette règle de contenu de DIT est, respectivement, exigée, admise, ou exclue, et
<extensions> décrit les extensions.

4.1.7 Règles de structure et formes de nom de DIT

Il est parfois souhaitable de réglementer où les entrées d'objet et d'alias peuvent être placées dans le DIT et comment elles peuvent être nommées sur la base de leur classe d'objet structurelle.

4.1.7.1 Règles de structure de DIT

Une règle de structure de DIT est une "règle qui gouverne la structure du DIT en spécifiant à un supérieur permis de subordonner une relation d'entrée. Une règle de structure rapporte une forme de nom, et donc une classe d'objet structurelle, à des règles de structure supérieures. Cela permet aux entrées de la classe d'objet structurelle identifiée par la forme de nom d'exister dans le DIT comme subordonnées aux entrées gouvernées par les règles des structure supérieures indiquées" [X.501].

Les descriptions de règle supérieure de DIT sont écrites conformément à l'ABNF :

```
DITStructureRuleDescription = LPAREN WSP
        ruleid                      ; identifiant de règle
        [ SP "NAME" SP qdescrs ]    ; noms abrégés (descripteurs)[ SP "DESC" SP qdstring ]   ; description
        [ SP "OBSOLETE" ]           ; non active
        SP "FORM" SP oid            ; NameForm
        [ SP "SUP" ruleids ]        ; règles supérieures
        extensions WSP RPAREN       ; extensions

ruleids = ruleid / ( LPAREN WSP ruleidlist WSP RPAREN )
ruleidlist = ruleid *( SP ruleid )
ruleid = nombre
```

où :
<ruleid> est l'identifiant de règle de cette règle de structure de DIT,
NAME <qdescrs> sont les noms abrégés (descripteurs) qui identifient cette règle de structure de DIT,
DESC <qdstring> est une courte chaîne descriptive,
OBSOLETE indique que cette utilisation de règle de structure de DIT n'est pas active,
FORM spécifie la forme de nom associée à cette règle de structure de DIT,

SUP identifie les règles supérieures (par l'identifiant de règle), et<extensions> décrit les extensions.
S'il n'est pas identifié de règles supérieures, la règle de structure de DIT s'applique à un point administratif autonome (par exemple, l'extrémité racine du sous-arbre contrôlé par le sous-schéma) [X.501].

4.1.7.2 Formes de nom

Une forme de nom "spécifie un RDN permis pour les entrées d'une classe d'objet structurelle particulière. Une forme de nom identifie une classe d'objet nommée et un ou plusieurs types d'attribut à utiliser pour les dénominations (c'est-à-dire, pour le RDN). Les formes de nom sont les pièces primitives de spécification utilisées dans la définition de règles de structure de DIT" [X.501].

Chaque forme de nom indique la classe d'objet structurelle à nommer, un ensemble de types d'attribut requis, et un ensemble de types d'attribut admis. Un type d'attribut particulier ne peut pas être dans les deux ensembles.

Les entrées gouvernées par la forme doivent être nommées en utilisant une valeur provenant de chaque type d'attribut requis, et zéro, une ou plusieurs valeurs provenant des types d'attribut admis.

Chaque forme de nom est identifiée par un identifiant d'objet (OID) et, facultativement, un ou plusieurs noms abrégés (descripteurs).

Les descriptions de forme de nom sont écrites conformément à l'ABNF :

```
NameFormDescription = LPAREN WSP
        numericoid                  ; identifiant d'objet
        [ SP "NAME" SP qdescrs ]    ; noms abrégés (descripteurs)
        [ SP "DESC" SP qdstring ]   ; description
        [ SP "OBSOLETE" ]           ; non active
        SP "OC" SP oid              ; classe d'objet structurelle
        SP "MUST" SP oids           ; types d'attribut
        [ SP "MAY" SP oids ]        ; types d'attribut
        extensions WSP RPAREN       ; extensions
```

où :
<numericoid> est l'identifiant d'objet qui identifie cette forme de nom,
NAME <qdescrs> sont les noms abrégés (descripteurs) qui identifient cette forme de nom,
DESC <qdstring> est une courte chaîne descriptive,
OBSOLETE indique que cette forme de nom n'est pas active,
OC identifie la classe d'objet structurelle à laquelle s'applique la règle,
MUST et MAY spécifient respectivement les ensembles d'attributs de dénomination exigés et admis, et
<extensions> décrit les extensions.

Tous les types d'attribut dans les listes exigés ("MUST") et admis ("MAY") doivent être différents.

4.2 Sous-entrées de sous-schéma

Les (sous-) entrées de (sous-) schéma sont utilisées pour administrer les informations sur le schéma de répertoire. Une seule (sous-) entrée de sous-schéma contient toutes les définitions de schéma (voir au paragraphe 4.1) utilisées par les entrées dans une partie particulière de l'arborescence du répertoire.

Les serveurs qui suivent les modèles X.500(93) DEVRAIENT mettre en oeuvre les sous-schémas en utilisant les mécanismes de sous-schéma X.500 (comme précisé à la Section 12 de [X.501]), de sorte que ceux-ci ne sont pas des entrées d'objet ordinaires mais des sous-entrées (voir au paragraphe 3.2). Les clients LDAP NE DEVRAIENT PAS supposer que les serveurs mettent en œuvre d'autres aspects du sous-schéma X.500.

Les serveurs PEUVENT permettre la modification du sous-schéma. Les procédures pour la modification de sous-schéma sont exposées au paragraphe 14.5 de [X.501].

Un serveur qui maîtrise les entrées et permet aux clients de modifier ces entrées DOIT mettre en œuvre et fournir l'accès à ces (sous)-entrées de sous-schéma, et y compris de fournir un attribut 'subschemaSubentry' dans chaque entrée

modifiable. C'est ainsi que les clients peuvent découvrir les attributs et classes d'objet dont la présence est permise. Il est fortement RECOMMANDÉ que tous les autres serveurs mettent cela en œuvre également.

La valeur de l'attribut 'subschemaSubentry' est le nom de la (sous-) entrée de sous-schéma qui détient le sous-schéma qui contrôle l'entrée.

```
( 2.5.18.10 NAME 'subschemaSubentry'
  EQUALITY distinguishedNameMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  SINGLE-VALUE NO-USER-MODIFICATION
  USAGE directoryOperation )
```

La règle de correspondance 'distinguishedNameMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.12) de DistinguishedName sont définies dans la [RFC4517].

Le sous-schéma est détenu dans les (sous-)entrées qui appartiennent à la classe d'objet auxiliaire du sous-schéma.

```
( 2.5.20.1 NAME 'subschema' AUXILIARY
  MAY ( dITStructureRules $ nameForms $ ditContentRules $
  objectClasses $ attributeTypes $ matchingRules $
  matchingRuleUse ) )
```

L'attribut de fonctionnement 'ldapSyntaxes' peut aussi être présent dans les entrées de sous-schéma.
Les serveurs PEUVENT fournir des attributs supplémentaires (décrits dans d'autres documents) dans les (sous-)entrées de sous-schéma.

Les serveurs DEVRAIENT fournir les attributs 'createTimestamp' et 'modifyTimestamp' dans les (sous-)entrées de sous-schéma, afin de permettre aux clients de maintenir leurs mémoires cache d'informations de schéma.

Les paragraphes qui suivent donnent les définitions de type d'attribut pour chacun des types d'attribut de définition de schéma.

4.2.1 'objectClasses'

Cet attribut détient les définitions des classes d'objet.

```
( 2.5.21.6 NAME 'objectClasses'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.37
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.37) de ObjectClassDescription sont définies dans la [RFC4517].

4.2.2 'attributeTypes'

Cet attribut détient les définitions of type d'attributs.

```
( 2.5.21.5 NAME 'attributeTypes'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.3
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.3) de AttributeTypeDescription sont définies dans la [RFC4517].

4.2.3 'matchingRules'

Cet attribut détient les définitions des règles de correspondance.

```
( 2.5.21.4 NAME 'matchingRules'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.30
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.30)de MatchingRuleDescription sont définies dans la [RFC4517].

4.2.4 'matchingRuleUse'

Cet attribut détient les définitions des utilisations des règles de correspondance.

```
( 2.5.21.8 NAME 'matchingRuleUse'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.31
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.31) de MatchingRuleUseDescription sont définies dans la [RFC4517].

4.2.5 'ldapSyntaxes'

Cet attribut détient les définitions des syntaxes LDAP.

```
( 1.3.6.1.4.1.1466.101.120.16 NAME 'ldapSyntaxes'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.54
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.54) de SyntaxDescription sont définies dans la [RFC4517].

4.2.6 'dITContentRules'

Cet attribut fait la liste des règles de contenu de DIT qui sont présentes dans le sous-schéma.

```
( 2.5.21.2 NAME 'dITContentRules'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.16
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.16) de DITContentRuleDescription sont définies dans la [RFC4517].

4.2.7 'dITStructureRules'

Cet attribut fait la liste des règles de structure de DIT qui sont présentes dans le sous-schéma.

```
( 2.5.21.1 NAME 'dITStructureRules'
  EQUALITY integerFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.17
  USAGE directoryOperation )
```

La règle de correspondance 'integerFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.17) de DITStructureRuleDescription sont définies dans la [RFC4517].

4.2.8 'nameForms'

Cet attribut fait la liste des règles de formes de noms qui sont en vigueur.

```
( 2.5.21.7 NAME 'nameForms'
  EQUALITY objectIdentifierFirstComponentMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.35
  USAGE directoryOperation )
```

La règle de correspondance 'objectIdentifierFirstComponentMatch' et la syntaxe (1.3.6.1.4.1.1466.115.121.1.35) de NameFormDescription sont définies dans la [RFC4517].

4.3 Classe d'objet 'extensibleObject'

La classe d'objet auxiliaire 'extensibleObject' permet aux entrées qui lui appartiennent de détenir tout attribut d'utilisateur. L'ensemble des types d'attribut admis de cette classe d'objet est implicitement l'ensemble de tous les types d'attribut d'utilisation de userApplications.

```
( 1.3.6.1.4.1.1466.101.120.111 NAME 'extensibleObject'
  SUP top AUXILIARY )
```

La présence des attributs obligatoires des autres classes d'objet de cette entrée est toujours requise, et aucun attribut exclu n'est autorisé à être présent.

4.4 Découverte de sous-schéma

Pour découvrir le DN de la (sous-)entrée du sous-schéma qui détient le sous-schéma contrôlant une entrée particulière, un client lit l'attribut de fonctionnement 'subschemaSubentry' de cette entrée. Pour lire les attributs de schéma d'après la (sous-)entrée de sous-schéma, les clients DOIVENT produire une opération Search [RFC4511] où baseObject est le DN de la (sous-)entrée de sous-schéma, le domaine est baseObject, le filtre est "(objectClass=subschema)" [RFC4515], et le champ attributes fait la liste des noms des attributs de schéma désirés (qui sont opérationnels).
Note : le filtre "(objectClass=subschema)" permet aux serveurs LDAP qui font passerelle pour X.500 de détecter que des informations de sous-entrée sont demandées.

Les clients NE DEVRAIENT PAS supposer qu'un sous schéma publié est achevé, que le serveur prend en charge tous les éléments de schéma qu'il publie, ou que le serveur ne prend pas en charge un élément non publié.

5 Modèle informationnel DSA (serveur)

Le protocole LDAP suppose qu'il y a un ou plusieurs serveurs qui fournissent conjointement l'accès à un arbre d'informations de répertoire (DIT, *Directory Information Tree*). Le serveur qui détient les informations d'origine s'appelle le "maître" (pour ces informations). Les serveurs qui détiennent les copies des informations originelles sont appelés serveurs "miroir" ou "cache".

Comme défini dans [X.501] :
Préfixe de contexte : c'est la séquence des RDN qui conduit de la racine du DIT à l'extrémité initiale d'un contexte de dénomination ; il correspond au nom distinctif de cette extrémité.

Contexte de dénomination : C'est un sous-arbre des entrées détenant un seul DSA maître.
C'est à dire qu'un contexte de dénomination est la plus grande collection d'entrées, commençant à une entrée qui est sous la maîtrise d'un serveur particulier, et incluant tous ses subordonnés et leurs subordonnés, jusqu'aux entrées qui sont sous la maîtrise de serveurs différents. Le préfixe de contexte est le nom de l'entrée initiale.

La racine du DIT est une entrée spécifique de DSA (DSE, *DSA-specific Entry*) et ne fait pas partie d'un contexte de dénomination (ou d'un sous-arbre) ; chaque serveur a des valeurs d'attribut différentes dans le DSE racine.

5.1 Exigences pour les données spécifiques du serveur

Un serveur LDAP DOIT fournir des informations sur lui-même et d'autres informations qui sont spécifiques de chaque serveur. Ceci est représenté comme un groupe des attributs localisés dans le DSE racine, qui est nommé par le DN avec zéro RDN (dont la représentation [RFC4514] est la chaîne de longueur zéro).

Ces attributs sont restituables, sous réserve des contrôles d'accès et autres restrictions, si un client effectue une opération Search [RFC4511] avec un baseObject vide, domaine de baseObject, filtre "(objectClass=*)" [RFC4515], et le champ attributes qui fait la liste des noms des attributs désirés. Il sera noté que les attributs DSE racine sont opérationnels et, comme les autres attributs de fonctionnement, ne sont pas retournés dans les demandes de recherche s'ils ne sont pas demandés par leur nom.

Le DSE racine NE DOIT PAS être inclus si le client effectue une recherche de sous-arbre commençant de puis la racine.

Les serveurs peuvent permettre aux clients de modifier les attributs du DSE racine, quand c'est approprié.

Les attributs suivants du DSE racine sont définis ci-après. Des attributs supplémentaires peuvent être définis dans d'autres documents.
- altServer : serveurs de remplacement ;
- namingContexts : contextes de dénomination ;
- supportedControl : contrôles LDAP reconnus ;
- supportedExtension : opérations d'extension LDAP reconnues ;
- supportedFeatures : caractéristiques LDAP reconnues ;
- supportedLDAPVersion : versions LDAP prises en charge ;
- supportedSASLMechanisms : mécanismes reconnus d'authentification simple et couches de sécurité (SASL) [RFC4422].

Les valeurs fournies pour ces attributs peuvent dépendre de facteurs spécifiques de la session et autres. Par exemple, un serveur qui prend en charge le mécanisme SASL EXTERNAL pourrait seulement afficher "EXTERNAL" lorsque l'identité du client a été établie par un niveau inférieur. Voir la [RFC4513].

Le DSE racine peut aussi inclure un attribut 'subschemaSubentry'. S'il ne le fait pas, l'attribut se réfère à la (sous-)entrée de sous-schéma qui détient le schéma contrôlant le DSE racine. Les clients NE DEVRAIENT PAS supposer que cette (sous-)entrée de sous-schéma contrôle d'autres entrées détenues par le serveur. Les procédures générales de découverte de sous-schéma sont données au paragraphe 4.4.

5.1.1 'altServer'

L'attribut 'altServer' fait la liste des URI qui se réfèrent aux serveurs de remplacement qui peuvent être contés lorsque ce serveur devient indisponible. Les URI pour les serveurs qui mettent en œuvre LDAP sont écrits conformément à la [RFC4516]. D'autres sortes d'URI peuvent être fournis. Si le serveur ne connaît pas d'autre serveur pouvant être utilisé, cet attribut sera absent. Les clients peuvent mettre ces informations en mémoire cache au cas où leur serveur préféré devienne ultérieurement indisponible.

```
( 1.3.6.1.4.1.1466.101.120.6 NAME 'altServer'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.26
  USAGE dSAOperation )
```

La syntaxe (1.3.6.1.4.1.1466.115.121.1.26) de IA5String est définie dans la [RFC4517].

5.1.2 'namingContexts'

L'attribut 'namingContexts' fait la liste des préfixes de contexte des contextes de dénomination que le serveur maîtrise ou duplique en miroir (en partie ou en totalité). Si le serveur est un DSA de premier niveau [X.501], il devrait mettre sur la liste (en plus) une chaîne vide (indiquant la racine du DIT). Si le serveur ne maîtrise ou ne duplique en miroir aucune information (par exemple, c'est une passerelle LDAP vers un répertoire X.500 public) cet attribut sera absent. Si le serveur croit qu'il maîtrise ou duplique en miroir le répertoire tout entier, l'attribut aura une seule valeur, et cette valeur sera la chaîne vide (indiquant la racine du DIT).

Cet attribut peut être utilisé, par exemple, pour choisir un nom d'entrée convenable pour des opérations ultérieures avec ce serveur.

```
( 1.3.6.1.4.1.1466.101.120.5 NAME 'namingContexts'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.12
  USAGE dSAOperation )
```

La syntaxe (1.3.6.1.4.1.1466.115.121.1.12) de DistinguishedName est définie dans la [RFC4517].

5.1.3 'supportedControl'

L'attribut 'supportedControl' fait la liste des identifiants d'objet qui identifient les contrôles de demande [RFC4511] que prend en charge le serveur. Si le serveur ne prend pas en charge de contrôles de demande, cet attribut sera absent. Il n'est pas nécessaire de faire la liste des identifiants d'objet qui identifient les contrôles de réponse.

Les procédures d'enregistrement des identifiants d'objet utilisés pour découvrir les mécanismes de protocole sont précisées dans le BCP 64, RFC 4520 [RFC4520].

```
( 1.3.6.1.4.1.1466.101.120.13 NAME 'supportedControl'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
  USAGE dSAOperation )
```

La syntaxe (1.3.6.1.4.1.1466.115.121.1.38) de OBJECT IDENTIFIER est définie dans la [RFC4517].

5.1.4 'supportedExtension'

L'attribut 'supportedExtension' fait la liste des identifiants d'objet qui identifient les opérations d'extension [RFC4511] que le serveur prend en charge. Si le serveur ne prend pas en charge d'opérations d'extension, cet attribut sera absent.

Une opération d'extension consiste généralement en une demande étendue et une réponse d'extension mais peut aussi inclure d'autres unités de données de protocole (telles que des réponses intermédiaires). L'identifiant d'objet alloué à la demande étendue est utilisée pour identifier l'opération d'extension. Les autres identifiants d'objet utilisés dans l'opération d'extension n'ont pas besoin de figurer sur la liste comme valeurs de cet attribut.

Les procédures d'enregistrement des identifiants d'objet utilisés pour découvrir les mécanismes de protocole sont précisées dans le BCP 64, RFC 4520 [RFC4520].

```
( 1.3.6.1.4.1.1466.101.120.7 NAME 'supportedExtension'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
  USAGE dSAOperation )
```

La syntaxe (1.3.6.1.4.1.1466.115.121.1.38) de OBJECT IDENTIFIER est définie dans la [RFC4517].

5.1.5 'supportedFeatures'

L'attribut 'supportedFeatures' fait la liste des identifiants d'objet qui identifient des caractéristiques sélectives que le serveur prend en charge. Si le serveur ne prend pas en charge de caractéristiques sélectives trouvables, cet attribut sera absent.

```
( 1.3.6.1.4.1.4203.1.3.5 NAME 'supportedFeatures'
  EQUALITY objectIdentifierMatch
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.38
  USAGE dSAOperation )
```

Les procédures d'enregistrement des identifiants d'objet utilisées pour découvrir les mécanismes de protocole sont précisées dans le BCP 64, RFC 4520 [RFC4520].

La syntaxe (1.3.6.1.4.1.1466.115.121.1.38) de OBJECT IDENTIFIER et la règle de correspondance objectIdentifierMatch sont définies dans la [RFC4517].

5.1.6 'supportedLDAPVersion'

L'attribut 'supportedLDAPVersion' fait la liste des versions de LDAP que le serveur prend en charge.

```
( 1.3.6.1.4.1.1466.101.120.15 NAME 'supportedLDAPVersion'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.27
  USAGE dSAOperation )
```

La syntaxe (1.3.6.1.4.1.1466.115.121.1.27) de INTEGER est définie dans la [RFC4517].

5.1.7 'supportedSASLMechanisms'

L'attribut 'supportedSASLMechanisms' fait la liste des mécanismes SASL [RFC4422] que le serveur reconnaît et/ou prend en charge [RFC4513]. Le contenu de cet attribut peut dépendre de l'état en cours de la session. Si le serveur ne prend pas en charge de mécanisme SASL, cet attribut ne sera pas présent.

```
( 1.3.6.1.4.1.1466.101.120.14 NAME 'supportedSASLMechanisms'
  SYNTAX 1.3.6.1.4.1.1466.115.121.1.15
  USAGE dSAOperation )
```

La syntaxe (1.3.6.1.4.1.1466.115.121.1.15) de Directory String est définie dans la [RFC4517].

6 Autres considérations

6.1 Préservation des informations d'utilisateur

Des syntaxes peuvent être définies avec des exigences spécifiques de préservation de valeurs et/ou forme (représentation) de valeur. Par exemple, une syntaxe contenant des données à signature numérique peut rendre obligatoire que le serveur préserve à la fois la valeur et la forme de valeur présentées pour assurer que la signature n'est pas invalidée.

Lorsque de telles exigences n'ont pas été explicitement déclarées, les serveurs DEVRAIENT préserver la valeur des informations d'utilisateur mais PEUVENT retourner la valeur sous une forme différente. Et lorsqu'un serveur n'est pas capable (ou ne veut pas) préserver la valeur les informations d'utilisateur, le serveur DOIT s'assurer qu'une valeur équivalente (selon la Section 2.3) est retournée.

6.2 Noms abrégés

Les noms abrégés, aussi appelés descripteurs, sont utilisés comme alias plus lisibles pour les identifiants d'objet et sont utilisés pour identifier divers éléments de schéma. Cependant, il n'est pas prévu que les mises en œuvre de LDAP avec interface d'utilisateur humain affichent ces noms abrégés (ou les identifiants d'objet auxquels il se réfèrent) à l'utilisateur. A la place, elles vont plus vraisemblablement effectuer des traductions (comme d'exprimer le nom abrégé dans une des langues nationales). Par exemple, le nom abrégé "st" (stateOrProvinceName) peut être affiché à un locuteur germanique comme "Land".
Le même nom abrégé pourrait avoir des significations différentes dans des sous-schémas différents, et, au sein d'un sous-schéma particulier, le même nom abrégé pourrait se référer à des identifiants d'objet différents identifiant chacun une différents sorte d'élément de schéma.

Les mises en œuvre DOIVENT être prêtes à ce que le même nom abrégé soit utilisé dans un sous-schéma pour se référer aux différentes sortes d'éléments de schéma. C'est à dire qu'il peut y avoir une classe d'objet 'x-fubar' et un type d'attribut 'x-fubar' dans un sous-schéma.

Les mises en œuvre DOIVENT être prêtes à ce que le même nom abrégé soit utilisé dans les différents sous-schémas pour se référer aux différents éléments de schéma. C'est à dire qu'il peut y avoir deux règles de correspondance 'x-fubar', chacune dans un sous-schéma différent.

Les procédures d'enregistrement des noms abrégés (descripteurs) sont précisées dans le BCP 64, [RFC4520].

6.3 Mise en mémoire cache et duplication en miroir

Certains serveurs peuvent détenir des copies d'entrées en mémoire cache ou dupliquées en miroir, qui peuvent être utilisées pour répondre aux demandes de recherche et aux interrogations de comparaison, mais vont retourner des renvois de références ou vont contacter d'autres serveurs si des opérations de modification sont demandées. Les serveurs qui effectuent la duplication en miroir ou la mise en mémoire cache DOIVENT s'assurer qu'ils ne violent aucune des contraintes de contrôle d'accès imposées aux données par le serveur d'origine.

7 Lignes directrices pour les mises en œuvre

7.1 Lignes directrices pour les serveurs

Les serveurs DOIVENT reconnaître tous les noms des types d'attribut et classes d'objet définies dans le présent document mais, sauf mention contraire, n'ont pas besoin de prendre en charge les fonctionnalités associées. Les serveurs DEVRAIENT reconnaître tous les noms des types d'attribut et classes d'objet définies, respectivement aux Sections 3 et 4 de la [RFC4519].

Les serveurs DOIVENT s'assurer que les entrées se conforment aux règles de schéma d'utilisateur et système ou autres contraintes de modèle de données.

Les serveurs PEUVENT prendre en charge les règles de contenu de DIT. Les serveurs PEUVENT prendre en charge les règles de structure et formes de nom de DIT.

Les serveurs PEUVENT prendre en charge les entrées d'alias.

Les serveurs PEUVENT prendre en charge la classe d'objet 'extensibleObject'.

Les serveurs PEUVENT prendre en charge les sous-entrées. S'il en est ainsi, ils DOIVENT le faire conformément à la [RFC3672]. Les serveurs qui ne prennent pas en charge les sous-entrées DEVRAIENT utiliser des entrées d'objet pour imiter les sous-entrées comme précisé au paragraphe 3.2.

Les serveurs PEUVENT mettre en œuvre des éléments de schéma additionnels. Les serveurs DEVRAIENT fournir des définitions de tout élément de schéma qu'ils prennent en charge dans les (sous-)entrées de sous-schéma.

7.2 Lignes directrices pour les clients

En l'absence d'accord préalable avec les serveurs, les clients NE DEVRAIENT PAS supposer que les serveurs vont prendre en charge des éléments de schéma particuliers au delà de ceux référencés au paragraphe 7.1. Le client peut restituer des informations de sous-schéma comme décrit au paragraphe 4.4.

Les clients NE DOIVENT PAS afficher une valeur en ASN.1 ou essayer de la décoder si la syntaxe de la valeur n'est pas connue. Les clients NE DOIVENT PAS supposer qu'un codage de chaîne spécifique de LDAP est restreint à une chaîne de caractères Unicode codés en UTF-8 ou à un sous-ensemble particulier d'Unicode (comme le sous-ensemble des caractères imprimables) si de telles restrictions ne sont pas déclarées explicitement. Les clients NE DEVRAIENT PAS envoyer de valeurs d'attribut dans une demande qui n'est pas valide conformément à la syntaxe définie pour les attributs.

8 Considérations sur la sécurité

Les attributs des entrées de répertoire sont utilisés pour fournir des informations descriptives sur les objets réels qu'ils représentent, qui peuvent être des personnes, des organisations, ou des appareils. La plupart des pays ont des lois sur la confidentialité au sujet de la publication d'informations sur les personnes.

Les considérations générales de sécurité pour l'accès à des informations de répertoire avec LDAP sont exposées dans les [RFC4511] et [RFC4513].

9 Considérations relatives à l'IANA

L'Autorité d'allocation des numéros Internet (IANA, *Internet Assigned Numbers Authority*) a mis à jour le registre des descripteurs LDAP comme indiqué dans le modèle suivant :

Sujet : Demande de mise à jour pour l'enregistrement de descripteur LDAP
Descripteur (nom abrégé) : voir le commentaire
Identifiant d'objet : voir le commentaire
Adresse de la personne & adresse de messagerie électronique de la personne à contacter pour des précisions :
Kurt Zeilenga <kurt@OpenLDAP.org>
Utilisation : voir le commentaire
Spécification : RFC 4512
Auteur/Controleur de modifications : IESG
Commentaires :
Les descripteurs (noms abrégés) ont été mis à jour pour ajouter au registre.

NOM	Type d'OID
governingStructureRule	A 2.5.21.10
structuralObjectClass	A 2.5.21.9

Les descripteurs (noms abrégés) ont été mis à jour pour se référer à la présente RFC.

NOM	Type d'OID
alias	O 2.5.6.1
aliasedObjectName	A 2.5.4.1
altServer	A 1.3.6.1.4.1.1466.101.120.6
attributeTypes	A 2.5.21.5
createTimestamp	A 2.5.18.1
creatorsName	A 2.5.18.3
dITContentRules	A 2.5.21.2
dITStructureRules	A 2.5.21.1
extensibleObject	O 1.3.6.1.4.1.1466.101.120.111
ldapSyntaxes	A 1.3.6.1.4.1.1466.101.120.16
matchingRuleUse	A 2.5.21.8
matchingRules	A 2.5.21.4
modifiersName	A 2.5.18.4
modifyTimestamp	A 2.5.18.2
nameForms	A 2.5.21.7
namingContexts	A 1.3.6.1.4.1.1466.101.120.5
objectClass	A 2.5.4.0
objectClasses	A 2.5.21.6
subschema	O 2.5.20.1
subschemaSubentry	A 2.5.18.10
supportedControl	A 1.3.6.1.4.1.1466.101.120.13
supportedExtension	A 1.3.6.1.4.1.1466.101.120.7
supportedFeatures	A 1.3.6.1.4.1.4203.1.3.5
supportedLDAPVersion	A 1.3.6.1.4.1.1466.101.120.15
supportedSASLMechanisms	A 1.3.6.1.4.1.1466.101.120.14
top	O 2.5.6.0

10 Remerciements

Le présent document est fondé en partie sur la RFC 2251 par M. Wahl, T. Howes, et S. Kille, sur la RFC 2252 par M. Wahl, A. Coulbeck, T. Howes, S. Kille, et sur la RFC 2556 par M. Wahl, toutes produites par le groupe de travail Accès, recherche et indexation des répertoires (ASID) de l'IETF. Le présent document se fonde aussi en partie sur "The Directory: Models" [X.501], produit de l'Union Internationale des télécommunications (UIT). Des textes supplémentaires ont été empruntés à la RFC 2253 par M. Wahl, T. Howes, et S. Kille.

Le présent document a été produit par le groupe de travail Révision LDAP (LDAPBIS) de l'IETF.

11 Références normatives

[RFC2119] Bradner, S., "Mots clé à utiliser dans les RFC pour indiquer les niveaux d'exigences", BCP 14, RFC 2119, mars 1997.

[RFC3629] Yergeau, F., "UTF-8, un format de transformation de ISO 10646", STD 63, RFC 3629, novembre 2003.

[RFC3671] Zeilenga, K., "Attributs collectifs dans le protocole léger d'accès de répertoire (LDAP)", RFC 3671, décembre 2003.

[RFC3672] Zeilenga, K., "Sous-entrées dans le protocole léger d'accès de répertoire (LDAP)", RFC 3672, décembre 2003.

[RFC4234] Crocker, D. et P. Overell, "BNF Augmenté pour les spécifications de syntaxe : ABNF", RFC 4234, octobre 2005.

[RFC4422] Melnikov, A., Ed. et K. Zeilenga, Ed., "Authentification simple et couche de sécurité (SASL)", RFC 4422, juin 2006.

[RFC4510] Zeilenga, K., Ed., "Protocole léger d'accès de répertoire (LDAP) : Descriptif des spécifications techniques", RFC 4510, juin 2006.

[RFC4511] Sermersheim, J., Ed., "Protocole léger d'accès de répertoire (LDAP) Le protocole", RFC 4511, juin 2006.

[RFC4513] Harrison, R., Ed., "Protocole léger d'accès de répertoire (LDAP) : Méthodes d'authentification et mécanismes de sécurité", RFC 4513, juin 2006.

[RFC4514] Zeilenga, K., Ed., "Protocole léger d'accès de répertoire (LDAP) : Représentation de chaîne des noms distinctifs", RFC 4514, juin 2006.

[RFC4515] Smith, M., Ed. et T. Howes, "Protocole léger d'accès de répertoire (LDAP) : Représentation de chaîne des filtres de recherche", RFC 4515, juin 2006.

[RFC4516] Smith, M., Ed. et T. Howes, "Protocole léger d'accès de répertoire (LDAP) : Localisateur universel de ressource", RFC 4516, juin 2006.

[RFC4517] Legg, S., Ed., "Protocole léger d'accès de répertoire (LDAP) : Syntaxes et règles de correspondance", RFC 4517, juin 2006.

[RFC4519] Sciberras, A., Ed., "Protocole léger d'accès de répertoire (LDAP) : Schéma pour les applications d'utilisateur", RFC 4519, juin 2006.

[RFC4520] Zeilenga, K., "Autorité d'allocation des numéros de l'Internet (IANA) : Considérations sur le Protocole léger d'accès de répertoire (LDAP)", BCP 64, RFC 4520, juin 2006.

[Unicode] The Unicode Consortium, "Norme Unicode, Version 3.2.0" est définie par "The Unicode Standard, Version 3.0" (Reading, MA, Addison-Wesley, 2000. ISBN 0-201-61633-5), tel qu'amendé par le "Unicode Standard Annex #27: Unicode 3.1" (http://www.unicode.org/reports/tr27/) et par le "Unicode Standard Annex #28: Unicode 3.2" (http://www.unicode.org/reports/tr28/).

[X.500] Union Internationale des Télécommunication – Secteur de la normalisation des télécommunications, "L'annuaire – Vue générales des concepts, modèles et services," X.500(1993) (aussi ISO/IEC 9594-1:1994).

[X.501] Union Internationale des Télécommunication – Secteur de la normalisation des télécommunications, "L'annuaire -- Modèles," X.501(1993) (aussi ISO/IEC 9594-2:1994).

[X.680] Union Internationale des Télécommunication – Secteur de la normalisation des télécommunications, "Notation de syntaxe abstraite n° 1 (ASN.1) - Spécification de la notation de base", X.680(2002) (aussi ISO/IEC 8824-1:2002).

Appendice A Changements

Le présent Appendice n'est pas normatif.

Le présent document aboutit à une réécriture presque complète de portions des RFC 2251, RFC 2252, et RFC 2256. Cette réécriture a été entreprise pour améliorer la clarté générale de la spécification technique. Le présent appendice donne un résumé des changements de substance faits à des portions de ces documents incorporés dans le présent document. Les lecteurs devraient consulter les [RFC4510], [RFC4511], [RFC4517], et [RFC4519] pour un résumé des changements aux autres portions de ces documents.

A.1 Changements à la RFC 2251

Le présent document incorpore les paragraphes 3.2 et 3.4, et des portions des Sections 4 et 6 de la RFC 2251, comme résumé ci-dessous.

A.1.1 Paragraphe 3.2 de la RFC 2251

Le paragraphe 3.2 de la RFC 2251 faisait une brève introduction au modèle de données de X.500, tel qu'utilisé par LDAP. La spécification précédente s'appuyait sur [X.501] mais manquait de clarté sur la façon dont les modèles X.500 sont adaptés pour leur utilisation par LDAP. Le présent document décrit les modèles de données X.500, tels qu'utilisés par LDAP, en plus grand détail, spécialement dans les domaines où une adaptation est nécessaire.

Le paragraphe 3.2.1 de la RFC 2251 décrit un attribut comme "un type avec une ou plusieurs valeurs associées". Dans LDAP, un attribut est mieux décrit comme une description d'attribut, un type avec zéro, une ou plusieurs options, et une ou plusieurs valeurs associées.

Le paragraphe 3.2.2 de la RFC 2251 rend obligatoire que les sous-entrées de sous-schéma contiennent des attributs objectClasses et attributeTypes, alors que X.500(93) traite ces attributs comme facultatifs. Alors que généralement toutes les mises en oeuvre qui prennent en charge les mécanismes de sous-schéma de X.500(93) vont fournir ces deux attributs, il n'est absolument pas exigé pour l'interopérabilité que tous les serveurs le fassent. L'obligation a été retirée pour la cohérence avec X.500(93). Le mécanisme de découverte de sous-schéma a été lui aussi précisé pour indiquer que le sous-schéma de contrôle d'une entrée est obtenu en lisant la (sous-)entrée à laquelle se réfère l'attribut 'subschemaSubentry' de cette entrée.

A.1.2 Paragraphe 3.4 de la RFC 2251

Le paragraphe 3.4 de la RFC 2251 fournit des "exigences de données spécifiques du serveur". Ces éléments ont été incorporés avec des changements au paragraphe 5.1 du présent document.

Changements :
- Précision que les attributs du DSE racine sont soumis aux "autres restrictions" en plus des contrôles d'accès.
- Précision que seules les demandes étendues reconnues doivent être énumérées dans 'supportedExtension'.
- Précision que seuls les contrôles de demande reconnus doivent être énumérés dans 'supportedControl'.
- Précision que les attributs de DSE racine sont de fonctionnement et, comme les autres attributs de fonctionnement, ne sont pas retournés dans les demandes de recherche sauf s'ils sont demandés par leur nom.

- Précision que tous les attributs de DSE racine ne sont pas modifiables par l'utilisateur.
- Retrait du texte incohérent sur le traitement de l'attribut 'subschemaSubentry' dans le DSE racine. La spécification précédente déclarait que l'attribut 'subschemaSubentry' détenu dans le DSE racine se référait aux "entrées de sous-schéma (ou sous-entrées) connues par ce serveur". Ceci n'est pas cohérent avec l'utilisation prévue de l'attribut ainsi qu'à sa définition formelle comme une seule valeur d'attribut [X.501]. Il est aussi noté qu'une simple liste (éventuellement incomplète) de (sous-)entrée de sous-schéma n'est pas très utile. Le présent document (au paragraphe 5.1) spécifie que l'attribut 'subschemaSubentry' de DSE racine se réfère au sous-schéma qui contrôle le DSE racine. Il est noté que le mécanisme général de découverte de sous-schéma reste disponible (voir au paragraphe 4.4 du présent document).

A.1.3 Section 4 de la RFC 2251

Des portions de la Section 4 de la RFC 2251 qui détaillent des aspects du modèle d'information utilisé par LDAP ont été incorporées ai présent document, comprenant :
- Restriction des valeurs distinctives aux attributs dont les descriptions n'ont pas d'option (d'après le paragraphe 4.1.3),
- Aspects sur les modèles de données des types d'attribut (d'après le paragraphe 4.1.4), descriptions d'attribut (d'après le paragraphe 4.1.5), Attribute (d'après 4.1.8), Identifiant de règle de correspondance (d'après le 4.1.9), et
- Exigences de schéma d'utilisateur (d'après les paragraphes 4.1.6, 4.5.1, et 4.7).

Les précisions à ces portions comportent :
- Subtyping et AttributeDescriptions avec des options.

A.1.4 Paragraphe 6 de la RFC 2251

Le paragraphe 6.1 et le second alinéa du paragraphe 6.2 de la RFC 2251 ont été incorporés dans le présent document.

A.2 Changements à la RFC 2252

Le présent document incorpore les Sections 4, 5, et 7 de la RFC 2252.

A.2.1 Section 4 de la RFC 2252

La spécification a été mise à jour pour utiliser le BNF augmenté [RFC4234]. La représentation de chaîne d'un OBJECT IDENTIFIER a été resserrée pour interdire les éros en tête comme décrit dans la RFC 2252.

La syntaxe <descr> a été changée pour interdire le caractère point-virgule (U+003B) afin d'être cohérent avec sa spécification de langage naturel "descr est la représentation syntaxique d'un descripteur d'objet, qui consiste en lettres et chiffres, commençant par une lettre". Dans une modification qui s'y rapporte, la déclaration que "une AttributeDescription peut être utilisée comme valeur dans la partie NAME d'une AttributeTypeDescription" a été supprimée. La RFC 2252 ne spécifiait pas la sémantique des options d'attribut apparaissant dans les champs NAME.

La RFC 2252 déclarait que la forme <descr> de <oid> DEVRAIT être préférée à la forme <numericoid>. Cependant, la forme<descr> peut être ambiguë. Pour régler ce problème, l'impératif a été remplace par une déclaration (au paragraphe 1.4) que alors que la forme <descr> est généralement préférée, <numericoid> devrait être utilisé lorsque un <descr> non ambigu n'est pas disponible. De plus, il y a un exposé développé sur la question des descripteurs au paragraphe 6.2 ("Noms abrégés").

L'ABNF pour une chaîne entre guillemets (qdstring) a été mis à jour pour refléter la prise en charge du mécanisme d'échappement (codage en pourcentage) décrit au paragraphe 4.3 de la RFC 2252.

A.2.2 Section 5 de la RFC 2252

Les définitions des attributs de fonctionnement fournies au paragraphe 5 de la RFC 2252 ont été incorporées dans le présent document.

La description de 'namingContexts' a été précisée. Un DSA de premier niveau devrait publier, en plus des autres valeurs, "" indiquant la racine du DIT.

La description de 'altServer' a été précisée. Elle peut détenir tout URI.

La description de 'supportedExtension' a été précisée. Un serveur a seulement besoin de faire la liste des OBJECT IDENTIFIER associés aux demandes étendues d'opérations d'extension qu'il reconnaît.

La description de 'supportedControl' a été précisée. Un serveur a seulement besoin de faire la liste des OBJECT IDENTIFIER associés aux contrôles de demandes qu'il reconnaît.

Les descriptions des types d'attribut de fonctionnement 'structuralObjectClass' et 'governingStructureRule' ont été ajoutées.

La définition d'attribut de 'subschemaSubentry' a été corrigée pour faire la liste des termes SINGLE-VALUE et NO-USER-MODIFICATION dans le bon ordre.

A.2.3 Section 7 de la RFC 2252

La Section 7 de la RFC 2252 donne les définitions des classes d'objet 'subschema' et 'extensibleObject'. Ces définitions ont été intégrées dans les paragraphes 4.2 et 4.3 du présent document, respectivement. La Section 7 de la RFC 2252 contenait aussi l'exigence de mise en œuvre de la classe d'objet. Cela a été incorporé dans la Section 7 du présent document.

La spécification de 'extensibleObject' a été précisée en ce qui concerne la façon dont il interagit avec les attributs exclus.

A.3 Changements à la RFC 2256

Le présent document incorpore les paragraphes 5.1, 5.2, 7.1, et 7.2 de la RFC 2256.

Le paragraphe 5.1 de la RFC 2256 fournit la définition du type d'attribut 'objectClass'. Cela a été intégré dans le paragraphe 2.4.1 du présent document. La déclaration "Une des valeurs est soit 'top' soit 'alias'" a été remplacée par la déclaration qu'une des valeurs est 'top' comme entrée appartenant à 'alias' appartenant aussi à 'top'.

Le paragraphe 5.2 de la RFC 2256 fournit la définition du type d'attribut 'aliasedObjectName'. Cela a été intégré dans le paragraphe 2.6.2 du présent document.

Le paragraphe 7.1 de la RFC 2256 fournit la définition de la classe d'objet 'top'. Cela a été intégré dans le paragraphe 2.4.1 du présent document.

Le paragraphe 7.2 de la RFC 2256 fournit la définition de la classe d'objet 'alias'. Cela a été intégré dans le paragraphe 2.6.1 du présent document.

A.4 Changes to RFC 3674

Le présent document n'a pas fait de modification de substance aux spécifications techniques 'supportedFeatures' fournies dans la RFC 3674.

Adresse de l'éditeur

Kurt D. Zeilenga
OpenLDAP Foundation
EMail: Kurt@OpenLDAP.org

Déclaration de copyright

GARANTIES, EXPRIMÉES OU IMPLICITES, Y COMPRIS MAIS NON LIMITÉES À TOUTE GARANTIE QUE L'UTILISATION DES INFORMATIONS CI-ENCLOSES NE VIOLENT AUCUN DROIT OU AUCUNE GARANTIE IMPLICITE DE COMMERCIALISATION OU D'APTITUDE À UN OBJET PARTICULIER.

Propriété intellectuelle

L'IETF ne prend pas position sur la validité et la portée de tout droit de propriété intellectuelle ou autres droits qui pourrait être revendiqués au titre de la mise en œuvre ou l'utilisation de la technologie décrite dans le présent document ou sur la mesure dans laquelle toute licence sur de tels droits pourrait être ou n'être pas disponible o pas plus qu'elle ne prétend avoir accompli aucun effort pour identifier de tels droits. Les informations sur les procédures de l'ISOC au sujet des droits dans les documents de l'ISOC figurent dans les BCP 78 et BCP 79.

Des copies des dépôts d'IPR faites au secrétariat de l'IETF et toutes assurances de disponibilité de licences, ou le résultat de tentatives faites pour obtenir une licence ou permission générale d'utilisation de tels droits de propriété par ceux qui mettent en œuvre ou utilisent la présente spécification peuvent être obtenues sur répertoire en ligne des IPR de l'IETF à http://www.ietf.org/ipr.

L'IETF invite toute partie intéressée à porter son attention sur tous copyrights, licences ou applications de licence, ou autres droits de propriété qui pourraient couvrir les technologies qui peuvent être nécessaires pour mettre en œuvre la présente norme. Prière d'adresser les informations à l'IETF à ietf- ipr@ietf.org.

Remerciement

Le financement de la fonction d'édition des RFC est actuellement fourni par l'activité de soutien administratif de l'IETF (IASA, Administrative Support Activity).