Union internationale des télécommunications

UIT-T

SECTEUR DE LA NORMALISATION
DES TÉLÉCOMMUNICATIONS
DE L'UIT

X.518

(08/2005)

SÉRIE X: RÉSEAUX DE DONNÉES, COMMUNICATION ENTRE SYSTÈMES OUVERTS ET SÉCURITÉ

Annuaire

Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: procédures pour le fonctionnement réparti

Recommandation UIT-T X.518

RECOMMANDATIONS UIT-T DE LA SÉRIE X

RÉSEAUX DE DONNÉES, COMMUNICATION ENTRE SYSTÈMES OUVERTS ET SÉCURITÉ

RÉSEAUX PUBLICS DE DONNÉES	
Services et fonctionnalités	X.1–X.19
Interfaces	X.20–X.49
Transmission, signalisation et commutation	X.50–X.89
Aspects réseau	X.90–X.149
Maintenance	X.150–X.179
Dispositions administratives	X.180–X.199
INTERCONNEXION DES SYSTÈMES OUVERTS	
Modèle et notation	X.200–X.209
Définitions des services	X.210–X.219
Spécifications des protocoles en mode connexion	X.220–X.229
Spécifications des protocoles en mode sans connexion	X.230–X.239
Formulaires PICS	X.240–X.259
Identification des protocoles	X.260–X.269
Protocoles de sécurité	X.270–X.279
Objets gérés des couches	X.280–X.289
Tests de conformité	X.290–X.299
INTERFONCTIONNEMENT DES RÉSEAUX	
Généralités	X.300–X.349
Systèmes de transmission de données par satellite	X.350–X.369
Réseaux à protocole Internet	X.370–X.379
SYSTÈMES DE MESSAGERIE	X.400–X.499
ANNUAIRE	**X.500–X.599**
RÉSEAUTAGE OSI ET ASPECTS SYSTÈMES	
Réseautage	X.600–X.629
Efficacité	X.630–X.639
Qualité de service	X.640–X.649
Dénomination, adressage et enregistrement	X.650–X.679
Notation de syntaxe abstraite numéro un (ASN.1)	X.680–X.699
GESTION OSI	
Cadre général et architecture de la gestion-systèmes	X.700–X.709
Service et protocole de communication de gestion	X.710–X.719
Structure de l'information de gestion	X.720–X.729
Fonctions de gestion et fonctions ODMA	X.730–X.799
SÉCURITÉ	X.800–X.849
APPLICATIONS OSI	
Engagement, concomitance et rétablissement	X.850–X.859
Traitement transactionnel	X.860–X.879
Opérations distantes	X.880–X.889
Applications génériques de l'ASN.1	X.890–X.899
TRAITEMENT RÉPARTI OUVERT	X.900–X.999
SÉCURITÉ DES TÉLÉCOMMUNICATIONS	X.1000–

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

NORME INTERNATIONALE ISO/CEI 9594-4
RECOMMANDATION UIT-T X.518

Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: procédures pour le fonctionnement réparti

Résumé

La présente Recommandation | Norme internationale spécifie les procédures grâce auxquelles les composants répartis de l'annuaire interfonctionnent pour fournir un service homogène aux utilisateurs.

Source

La Recommandation UIT-T X.518 a été approuvée le 29 août 2005 par la Commission d'études 17 (2005-2008) de l'UIT-T selon la procédure définie dans la Recommandation UIT-T A.8. Un texte identique est publié comme Norme Internationale ISO/CEI 9594-4.

AVANT-PROPOS

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

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

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

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

NOTE

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

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

DROITS DE PROPRIÉTÉ INTELLECTUELLE

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

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

TABLE DES MATIÈRES

Introduction

La présente Recommandation | partie de Norme internationale a été élaborée, ainsi que d'autres Recommandations | Normes internationales, pour faciliter l'interconnexion des systèmes de traitement de l'information et permettre ainsi d'assurer des services d'annuaire. L'ensemble de tous ces systèmes, avec les informations d'annuaire qu'ils contiennent, peut être considéré comme un tout intégré, appelé *annuaire*. Les informations de l'annuaire, appelées collectivement base d'informations d'annuaire (DIB), sont généralement utilisées pour faciliter la communication entre, avec ou à propos d'objets tels que des entités d'application, des personnes, des terminaux et des listes de distribution.

L'annuaire joue un rôle important dans l'interconnexion des systèmes ouverts, dont le but est de permettre, moyennant un minimum d'accords techniques en dehors des normes d'interconnexion proprement dites, l'interconnexion des systèmes de traitement de l'information:

- provenant de divers fabricants;
- gérés différemment;
- de niveaux de complexité différents;
- de générations différentes.

La présente Recommandation | Norme internationale spécifie les procédures d'interfonctionnement que les composants répartis de l'annuaire mettent en œuvre pour fournir un service cohérent à ses utilisateurs.

La présente Recommandation | Norme internationale fournit les cadres de base permettant la définition de profils industriels par d'autres organismes de normalisation et par des forums industriels. L'utilisation d'un grand nombre des fonctionnalités optionnelles figurant dans ces cadres peut être rendue obligatoire dans certains environnements au moyen de profils. Cette cinquième édition révise techniquement et améliore, mais ne remplace pas, la quatrième édition de la présente Recommandation | Norme internationale. Les implémentations peuvent encore revendiquer la conformité à la quatrième édition mais celle-ci finira par ne plus être prise en compte (c'est-à-dire que les erreurs signalées ne seront plus corrigées). Il est recommandé que les implémentations se conforment, dès que possible, à la présente cinquième édition.

Cette cinquième édition spécifie les versions 1 et 2 des protocoles de l'annuaire.

Les première et deuxième éditions ne spécifiaient que la version 1. La plupart des services et protocoles spécifiés dans la présente édition sont conçus pour fonctionner selon la version 1. Certains services et protocoles améliorés, par exemple les erreurs signées, ne fonctionneront cependant pas avant que toutes les entités d'annuaire mises en jeu dans l'exploitation aient négocié la version 2. Quelle que soit la version négociée, on a traité les différences entre les services et entre les protocoles définis dans les cinq éditions, à l'exception de ceux qui sont spécifiquement définis dans la version 2, en utilisant les règles d'extensibilité définies dans la Rec. UIT-T X.519 | ISO/CEI 9594-5.

L'Annexe A, qui fait partie intégrante de la présente Recommandation | Norme internationale, fournit le module ASN.1 pour les opérations réparties de l'annuaire.

L'Annexe B, qui ne fait pas partie intégrante de la présente Recommandation | Norme internationale, décrit un exemple de résolution répartie du nom.

L'Annexe C, qui ne fait pas partie intégrante de la présente Recommandation | Norme internationale, décrit l'authentification dans un environnement d'opérations réparties.

L'Annexe D, qui fait partie intégrante de la présente Recommandation | Norme internationale, fournit les définitions des classes d'objets d'information en notation ASN.1 qui sont introduites dans la présente Spécification d'annuaire.

L'Annexe E, qui ne fait pas partie intégrante de la présente Recommandation | Norme internationale, décrit la maintenance des connaissances.

L'Annexe F, qui ne fait pas partie intégrante de la présente Recommandation | Norme internationale, énumère les amendements et relevés de défauts qui ont été incorporés pour former la présente édition de cette Recommandation | Norme internationale.

NORME INTERNATIONALE
RECOMMANDATION UIT-T

Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: procédures pour le fonctionnement réparti

SECTION 1 – GÉNÉRALITÉS

1 Domaine d'application

La présente Recommandation | Norme internationale spécifie le comportement des agents de système d'annuaire (DSA, *directory system agent*) qui participent à la mise en œuvre répartie de l'annuaire. Le comportement autorisé a été conçu pour assurer un service cohérent, compte tenu d'une large répartition de la base d'informations annuaire (DIB, *directory information base*) entre de nombreux DSA.

Bien qu'il puisse être construit à partir de tels systèmes, l'annuaire n'est pas destiné à être un système de base de données à usage général. La fréquence des demandes est censée être considérablement plus élevée que celle des mises à jour.

2 Références normatives

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

2.1 Recommandations | Normes internationales identiques

- Recommandation UIT-T X.200 (1994) | ISO/CEI 7498-1:1994, *Technologies de l'information – Interconnexion des systèmes ouverts – Modèle de référence de base: le modèle de référence de base*.
- Recommandation UIT-T X.500 (2005) | ISO/CEI 9594-1:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: aperçu général des concepts, modèles et services*.
- Recommandation UIT-T X.501 (2005) | ISO/CEI 9594-2:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: les modèles*.
- Recommandation UIT-T X.509 (2005) | ISO/CEI 9594-8:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: cadre général des certificats de clé publique et d'attribut*.
- Recommandation UIT-T X.511 (2005) | ISO/CEI 9594-3:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: définition du service abstrait*.
- Recommandation UIT-T X.519 (2005) | ISO/CEI 9594-5:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: spécification des protocoles*.
- Recommandation UIT-T X.520 (2005) | ISO/CEI 9594-6:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: types d'attributs sélectionnés*.
- Recommandation UIT-T X.521 (2005) | ISO/CEI 9594-7:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: classes d'objets sélectionnées*.
- Recommandation UIT-T X.525 (2005) | ISO/CEI 9594-9:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: duplication*.

- Recommandation UIT-T X.530 (2005) | ISO/CEI 9594-10:2005, *Technologies de l'information – Interconnexion des systèmes ouverts – L'annuaire: utilisation de la gestion-systèmes pour l'administration de l'annuaire*.
- Recommandation UIT-T X.680 (2002) | ISO/CEI 8824-1:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification de la notation de base*.
- Recommandation UIT-T X.681 (2002) | ISO/CEI 8824-2:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des objets informationnels*.
- Recommandation UIT-T X.682 (2002) | ISO/CEI 8824-3:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: spécification des contraintes*.
- Recommandation UIT-T X.683 (2002) | ISO/CEI 8824-4:2002, *Technologies de l'information – Notation de syntaxe abstraite numéro un: paramétrage des spécifications de la notation de syntaxe abstraite numéro un*.

2.2 Autres références

- IETF RFC 2251 (1997), *Lightweight Directory Access Protocol (v3)*.
- IETF RFC 3377 (2002), *Lightweight Directory Access Protocol (v3): Technical Specification*.

3 Définitions

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

3.1 Définitions relatives au modèle de communication

Le terme suivant est défini dans la Rec. UIT-T X.519 | ISO/CEI 9594-5:

a) *titre d'entité d'application*.

3.2 Définitions de base relatives à l'annuaire

Les termes suivants sont définis dans la Rec. UIT-T X.500 | ISO/CEI 9594-1:

a) *(l')annuaire*;
b) *base d'informations annuaire*.

3.3 Définitions relatives au modèle de l'annuaire

Les termes suivants sont définis dans la Rec. UIT-T X.501 | ISO/CEI 9594-2:

a) *point d'accès*;
b) *alias; pseudonyme*;
c) *nom distinctif*;
d) *arbre d'informations annuaire*;
e) *agent de système d'annuaire (DSA, directory system agent)*;
f) *agent utilisateur d'annuaire (DUA, directory user agent)*;
g) *nom distinctif relatif*.

3.4 Définitions du modèle d'informations de DSA

Les termes suivants sont définis dans la Rec. UIT-T X.501 | ISO/CEI 9594-2:

a) *catégorie*;
b) *d'utilisation commune*;
c) *préfixe de contexte*;
d) *référence croisée*;
e) *fragment de base DIB*;
f) *arbre des informations de DSA*;
g) *entrée spécifique de DSA (DSE, DSA – specific entry)*;

h) *type d'entrée DSE*;
i) *référence de supérieur immédiat*;
j) *information de connaissance*;
k) *catégorie de référence de connaissance*;
l) *type de référence de connaissance*;
m) *contexte de dénomination*;
n) *connaissance non spécifique*;
o) *référence subordonnée non spécifique*;
p) *attribut opérationnel*;
q) *chemin de référence*;
r) *connaissance spécifique*;
s) *référence subordonnée*;
t) *référence supérieure*.

3.5 Définitions du service abstrait

Le terme suivant est défini dans la Rec. UIT-T X.511 | ISO/CEI 9594-3:

a) *résultat en flux continu.*

3.6 Définitions relatives à la duplication dans l'annuaire

Les termes suivants sont définis dans la Rec. UIT-T X.525 | ISO/CEI 9594-9:

a) *exhaustivité d'attribut*;
b) *rattachement opérationnel de duplication miroir*;
c) *exhaustivité de subordonnés*;
d) *unité de duplication*.

3.7 Définitions relatives au fonctionnement réparti

Les termes suivants sont définis dans la présente Recommandation | Norme internationale:

3.7.1 objet de base: entrée d'objet ou d'alias qui est l'objectif d'une opération, tel qu'il est désigné par l'entité à l'origine de l'opération.

3.7.2 DSA rattaché: l'agent de système d'annuaire auquel l'agent utilisateur d'annuaire (DUA) a été rattaché en ayant effectué une opération Rattachement avec cet agent DSA.

3.7.3 résultats paginés de DSA rattaché: la pagination est exécutée entièrement par le DSA auquel l'agent DUA est rattaché.

NOTE – Il s'agit du seul mode de pagination pris en charge par les systèmes conformes aux éditions antérieures à la cinquième édition.

3.7.4 chaînage: terme générique pouvant désigner l'unichaînage et le multichaînage.

3.7.5 information du préfixe de contexte: information opérationnelle et d'utilisateur fournie par le DSA supérieur au DSA subordonné dans un RHOB concernant les nœuds d'arbre DIT supérieurs au préfixe de contexte subordonné.

3.7.6 résolution répartie du nom: processus par lequel la résolution du nom est effectuée dans plus d'un DSA.

3.7.7 résultats paginés de protocole DSP: dispositions du protocole DSP lorsque le DSA exécutant est différent du DSA rattaché, au moyen desquelles sont obtenus les résultats paginés par l'exécuteur initial.

3.7.8 erreur: information envoyée par l'exécuteur au demandeur et acheminant un résultat négatif concernant une demande précédemment reçue.

3.7.9 erreur bloquante: erreur définie qui indique que l'opération ne peut être accomplie sans intervention extérieure.

3.7.10 rattachement opérationnel hiérarchique (HOB, *hierarchical operational binding*): relation entre deux DSA maîtres détenant des contextes de dénomination, dont l'un est immédiatement subordonné à l'autre, où le DSA supérieur détient une référence subordonnée au DSA subordonné.

3.7.11 exécuteur initial: premier agent DSA à exécuter une opération, c'est-à-dire premier DSA à amorcer la phase d'évaluation de l'opération.

3.7.12 opérations de modification: opérations de type Directory Modify (modification d'annuaire), comme Modify Entry (modifier entrée), Add Entry (ajouter entrée), Remove Entry (supprimer entrée) et Modify DN (modifier nom distinctif).

3.7.13 multichaînage: mode d'interaction dans lequel un DSA traitant une demande lui-même, envoie des demandes multiples, soit parallèlement soit séquentiellement, à une série d'autres DSA.

3.7.14 opérations d'interrogation à entrées multiples: opérations de type Directory Search Operations (recherche dans l'annuaire) comme List (listage) et Search (recherche).

3.7.15 résolution du nom: processus de localisation d'une entrée par adaptation séquentielle, à un nœud du DIT de chaque nom distinctif relatif (RDN) contenu dans un nom prétendu.

3.7.16 rattachement opérationnel hiérarchique non spécifique (NHOB, *non-specific hierarchical operational binding*): relation entre deux DSA maîtres détenant des contextes de dénomination, dont l'un est immédiatement subordonné à l'autre, où le DSA supérieur détient une référence subordonnée non spécifique au DSA subordonné.

3.7.17 décomposition de référence subordonnée non spécifique (NSSR, *non-specific subordinate reference*): décomposition de références non spécifiques de connaissances en sous-demandes de continuation adressées à des DSA; ces sous-demandes peuvent soit être concaténées avec ces DSA par le DSA exécutant la décomposition ou prendre la forme de références de continuation identifiant les DSA: ces références pourront être retournées au demandeur pour lui permettre de continuer; ou bien le DSA décomposeur peut donner suite à certaines des sous-demandes, laissant les autres sans suite, pour continuation par le demandeur.

3.7.18 avancement d'opération: série de valeurs qui dénotent la mesure dans laquelle la résolution de nom a eu lieu.

3.7.19 initiateur: DUA qui a lancé une opération spécifique (répartie).

3.7.20 pagination: envoi discontinu d'un résultat d'opération **search** (recherche) ou **list** (listage) sous la forme d'une ou plusieurs pages comprises dans un nombre limité d'éléments.

3.7.21 exécuteur: DSA recevant une demande (pour accomplir une opération).

NOTE – L'exécuteur est également l'exécuteur initial sauf le cas échéant pour des opérations dont l'évaluation implique plusieurs DSA.

3.7.22 procédure: spécification (informelle) de la manière dont un DSA applique, à un résultat, une série d'arguments d'entrée et son arbre d'informations de DSA.

NOTE – Les arguments d'entrée et les résultats peuvent correspondre à l'information reçue dans une opération demandée et à l'information envoyée dans une réponse; ils peuvent aussi représenter des étapes intermédiaires dans le calcul d'une réponse suite à une opération demandée. Au paragraphe 14.2, le premier type d'arguments d'entrée et de résultats est qualifié d'externe.

3.7.23 rattachement opérationnel hiérarchique approprié (RHOB, *relevant hierarchical operational binding*): rattachement HOB ou NHOB, selon le contexte.

3.7.24 renvoi de référence: résultat qui peut être renvoyé par un DSA qui ne peut effectuer l'opération lui-même; ce résultat identifie un ou plusieurs autres DSA ayant davantage la capacité d'effectuer l'opération.

3.7.25 réponse: résultat ou erreur.

3.7.26 demande: information consistant en un code d'opération et en arguments associés pour acheminer une opération d'annuaire d'un demandeur à un exécuteur.

3.7.27 décomposition de demande: décomposition d'une demande en sous-demandes en vue de la poursuite du traitement par d'autres DSA; ces sous-demandes peuvent être concaténées avec les DSA en question par les DSA qui accomplissent la décomposition, ou bien des références de continuation identifiant les DSA peuvent être renvoyées au demandeur en vue de la poursuite du traitement, ou encore le DSA qui procède à la décomposition peut poursuivre le traitement de certaines des sous-demandes, laissant le soin au demandeur de poursuivre le traitement de celles qu'il ignore.

3.7.28 demandeur: DUA ou DSA envoyant une demande d'exécution (c'est-à-dire invoquant) une opération.

3.7.29 opérations d'interrogation à entrée unique: opérations du type Directory Read Operations, c'est-à-dire Read (lire) et Compare (comparer).

3.7.30 erreur simple: erreur qui peut être passagère ou qui peut dénoter un problème localisé, auquel cas l'utilisation d'une référence de connaissances différente ou d'un point d'accès différent peut permettre l'obtention d'un résultat ou d'une erreur bloquante.

3.7.31 DSA subordonné: de deux DSA partageant un HOB ou un NHOB, celui qui détient le contexte de dénomination subordonné.

3.7.32 sous-demande: demande obtenue par subdivision d'une demande.

3.7.33 DSA supérieur: de deux DSA partageant un HOB ou NHOB, celui qui détient le contexte de dénomination supérieur.

3.7.34 DSA subordonné: de deux DSA maîtres détenant des contextes de dénomination, celui qui est immédiatement subordonné à l'autre; la relation entre les deux DSA est gérée de manière explicite par l'intermédiaire d'un HOB (ou NHOB), ou existe de manière implicite du fait que le DSA supérieur détient une référence subordonnée (ou subordonnée non spécifique) par rapport au DSA subordonné.

3.7.35 nom d'objet cible: nom d'une entrée vers laquelle l'opération doit être dirigée à un stade particulier de la résolution du nom ou qui est impliquée dans l'évaluation de cette opération.

3.7.36 unichaînage: mode d'interaction utilisé facultativement par un DSA qui ne peut effectuer lui-même l'opération demandée. Le DSA *chaîne* par invocation d'une opération d'un autre DSA, puis par renvoi du résultat au demandeur initial.

4 Abréviations

Pour les besoins de la présente Recommandation | Norme internationale, les abréviations suivantes sont utilisées:

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

DISP Protocole de duplication miroir d'informations de l'annuaire (*directory information shadowing protocol*)

DMD Domaine de gestion d'annuaire (*directory management domain*)

DOP Protocole de gestion des rattachements opérationnels pour l'annuaire (*directory operational binding management protocol*)

DSE Entrée spécifique de DSA (*DSA-specific entry*)

HOB Rattachement opérationnel hiérarchique (*hierarchical operational binding*)

NHOB Rattachement opérationnel hiérarchique non spécifique (*non-specific hierarchical operational binding*)

NSSR Référence subordonnée non spécifique (*non-specific subordinate reference*)

RHOB Rattachement opérationnel hiérarchique approprié (*relevant hierarchical operational binding*)

5 Conventions

A quelques exceptions mineures près, la présente Spécification d'annuaire a été élaborée conformément aux "*Règles de représentation des textes communs UIT-T | ISO/CEI*", novembre 2001.

Le terme "Spécification d'annuaire" (comme dans "la présente Spécification d'annuaire") s'entend selon l'acception de la présente Rec. UIT-T X.518 | ISO/CEI 9594-4. Le terme "Spécification d'annuaire" s'entend selon l'acception de toutes les Recommandations de la série X.500 et de toutes les parties de l'ISO/CEI 9594.

La présente Spécification d'annuaire utilise le terme "*systèmes première édition*" pour désigner les systèmes conformes à la première édition des Spécifications d'annuaire, c'est-à-dire à l'édition 1988 des Recommandations CCITT de la série X.500 et de l'ISO/CEI 9594:1990. La présente Spécification d'annuaire utilise le terme "*systèmes deuxième édition*" pour désigner les systèmes conformes à la deuxième édition des Spécifications d'annuaire, c'est-à-dire à l'édition 1993 des Recommandations UIT-T de la série X.500 et de l'ISO/CEI 9594:1995. La présente Spécification d'annuaire utilise le terme "*systèmes troisième édition*" pour désigner les systèmes conformes à la troisième édition des Spécifications d'annuaire, c'est-à-dire à l'édition 1997 des Recommandations UIT-T de la série X.500 et de l'ISO/CEI 9594:1998. La présente Spécification d'annuaire utilise le terme "*systèmes quatrième édition*" pour désigner les systèmes conformes à la quatrième édition des Spécifications d'annuaire, c'est-à-dire aux éditions 2001 des Recommandations UIT-T X.500, X.501, X.511, X.518, X.519, X.520, X.521, X.525 et X.530 et à l'édition 2000 de la Recommandation UIT-T X.509 et aux parties 1 à 10 de l'ISO/CEI 9594:2001.

La présente Spécification d'annuaire utilise le terme "*systèmes cinquième édition*" pour désigner les systèmes conformes à la cinquième édition des Spécifications d'annuaire, c'est-à-dire aux éditions 2005 des Recommandations UIT-T X.500, X.501, X.509, X.511, X.518, X.519, X.520, X.521, X.525 et X.530 et aux parties 1 à 10 de l'ISO/CEI 9594:édition 2005.

Cette Spécification d'annuaire présente la notation ASN.1 en caractères gras de la police Helvetica. Lorsque des types et des valeurs ASN.1 sont cités dans le texte normal, ils en sont différenciés par leur présentation en caractères gras Helvetica. Les noms des procédures, normalement cités lors de la spécification des sémantèmes de traitement, sont différenciés du texte normal par une présentation en caractères gras de la police Times. Les autorisations de contrôle d'accès sont présentées en caractères italiques de la police Times.

Si dans une liste, les points sont numérotés (au lieu d'utiliser des tirets ou des lettres), ils sont considérés comme des étapes d'une procédure.

SECTION 2 – APERÇU GÉNÉRAL

6 Aperçu général

Le service abstrait d'annuaire permet l'interrogation, la recherche et la modification des informations de l'annuaire contenues dans la DIB. Ce service est décrit en termes d'objet abstrait annuaire, tel que spécifié dans la Rec. UIT-T X.511 | ISO/CEI 9594-3. De manière analogue le protocole rapide d'accès à l'annuaire (LDAP, *lightweight directory access protocol*) permet l'interrogation, la recherche et la modification des informations de l'annuaire contenues dans la DIB. Ce protocole et les services dont il autorise l'utilisation sont décrits dans la spécification RFC 3377.

De toute évidence, la spécification de l'objet abstrait annuaire ne saurait concerner la réalisation physique de l'annuaire; en particulier, elle ne concerne pas la spécification des agents de système d'annuaire (DSA) dans lesquels sont enregistrées et gérées les informations de la DIB et par l'intermédiaire desquels le service est assuré. En outre, cette spécification ne tient pas compte de la centralisation éventuelle de la DIB, c'est-à-dire du fait qu'elle est contenue dans un seul DSA, ou de sa répartition entre plusieurs DSA. En conséquence, la description du service ne concerne pas non plus les spécifications des DSA en ce qui concerne leurs connaissances des autres DSA, la navigation vers ces agents et la coopération avec eux en vue de la prise en charge du service abstrait dans un environnement réparti.

La présente Spécification d'annuaire spécifie le raffinement de l'objet abstrait annuaire, ce raffinement étant exprimé en termes d'ensemble d'un ou de plusieurs objets DSA qui constituent collectivement le service réparti d'annuaire.

En outre, la présente Spécification d'annuaire spécifie les modes de répartition autorisés de la base de données DIB entre un ou plusieurs DSA. Dans le cas limite où la base de données DIB est contenue dans un seul DSA, l'annuaire est en fait centralisé; dans le cas où la DIB est répartie entre deux DSA ou plus, des mécanismes de connaissance et de navigation sont spécifiés: ils garantissent que l'ensemble de la DIB est accessible à tous les DSA qui comportent des entrées constitutives de l'annuaire.

Des portions de la DIB peuvent aussi être copiées dans plusieurs DSA. Les protocoles décrits dans la présente Spécification d'annuaire permettent l'emploi de copies d'informations pour l'amélioration de la disponibilité, de la qualité de fonctionnement et de l'efficacité du service d'annuaire réparti. L'emploi d'informations copiées relève dans une certaine mesure de la décision de l'utilisateur, qui peut faire appel à des options de commande de service. Les procédures décrites dans la présente Spécification d'annuaire indiquent également certaines possibilités d'optimisation de conception avec l'emploi d'informations copiées.

De plus, des interactions de traitement des demandes sont spécifiées: elles permettent aux utilisateurs de l'annuaire de garder le contrôle sur des caractéristiques particulières de son fonctionnement. En particulier, l'utilisateur peut décider si un DSA a la possibilité, en réponse à une demande présentée à l'annuaire concernant des informations contenues dans d'autres DSA, d'interroger directement d'autres DSA (chaînage) ou s'il doit fournir en réponse des informations sur d'autres DSA qui pourraient faire progresser le traitement de la demande (renvoi de référence).

En général, la décision d'un DSA d'opérer un chaînage ou un renvoi de référence est déterminée par les commandes de service prévues par l'utilisateur et par les conditions administratives, opérationnelles ou techniques propres au DSA.

Compte tenu du fait qu'en général l'annuaire sera réparti et que les demandes qui lui seront adressées seront satisfaites par un nombre arbitraire de DSA coopérants qui pourront opérer arbitrairement des chaînages ou des renvois de référence selon les critères ci-dessus, la présente Spécification d'annuaire spécifie les procédures appropriées qui doivent être suivies par les DSA en réponse à des demandes adressées à l'annuaire en mode réparti. Ces procédures garantissent que les utilisateurs du service d'annuaire réparti en auront une perception aussi intuitive que cohérente.

SECTION 3 – MODÈLES D'ANNUAIRE RÉPARTI

7 Modèle du système d'annuaire réparti

Le service abstrait d'annuaire, défini dans la Rec. UIT-T X.511 | ISO/CEI 9594-3, modélise l'annuaire comme un objet qui fournit un ensemble de services d'annuaire à ses utilisateurs. Les utilisateurs de l'annuaire accèdent à ses services par un point d'accès. L'annuaire peut avoir un ou plusieurs points d'accès dont chacun est caractérisé par les services qu'il fournit et par le mode d'interaction utilisé pour les fournir.

La Figure 1 représente le modèle d'annuaire réparti qui sera utilisé comme base pour la spécification des aspects répartis de l'annuaire. Elle représente l'objet annuaire comme étant constitué d'un ensemble d'un ou de plusieurs DSA.

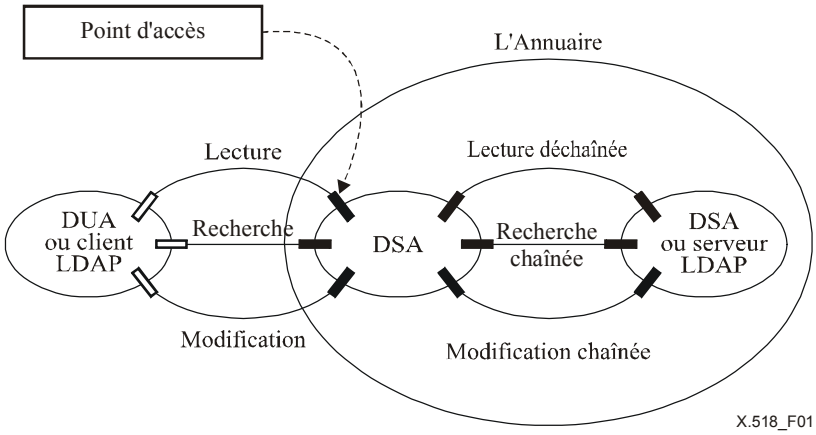


Figure 1 – Objets du modèle d'annuaire réparti

Les DSA sont spécifiés en détail dans les articles suivants de la présente Spécification d'annuaire. Cet article énumère simplement un certain nombre de leurs caractéristiques, à titre d'introduction pour préciser les relations entre la présente Spécification d'annuaire et les autres spécifications de ce type.

Les DSA sont définis en vue de permettre la répartition de la base de données DIB et l'interaction entre plusieurs DSA répartis physiquement, de façon prescrite et coopérative, pour fournir des services d'annuaire aux utilisateurs de celui-ci (DUA ou clients LDAP).

La Figure 1 illustre la relation entre le service abstrait d'annuaire et le service abstrait de DSA. Le service abstrait d'annuaire défini dans la Rec. UIT-T X.511 | ISO/CEI 9594-3 est fourni au moyen d'un certain nombre d'opérations d'annuaire. Pour assurer ce service, les DSA qui ont l'annuaire agissent en interaction. La nature de cette interaction est définie du point de vue du service qu'un DSA peut assurer à un autre DSA, à savoir le service abstrait de DSA. Le service abstrait de DSA est fourni au moyen d'un certain nombre d'opérations appelées opérations chaînées, chacune ayant son pendant dans le service abstrait d'annuaire. Ainsi, une opération donnée dans le service abstrait d'annuaire (par exemple lecture), peut nécessiter de la part du DSA assurant le service une interaction avec un ou plusieurs autres DSA à l'aide d'opérations chaînées (par exemple lecture chaînée).

NOTE – Les DSA qui sont demandeurs LDAP ont aussi la possibilité de chaîner des opérations, par exemple au moyen de mesures de contrôle LDAP ou d'opérations étendues; toutefois les procédures et les protocoles nécessaires à cet effet ne relèvent pas du domaine d'application de la présente Spécification d'annuaire.

8 Modèle des interactions entre les DSA

Une caractéristique de base de l'annuaire est qu'un utilisateur, pour une certaine DIB répartie, doit pouvoir faire aboutir une quelconque demande de service (compte tenu des politiques de sécurité, de contrôle d'accès et d'administration), quel que soit le point d'accès auquel cette demande est présentée. Pour répondre à cet impératif, il est nécessaire que tous les DSA qui entrent en compte dans la réponse à une demande de service particulière aient une certaine connaissance (spécifiée dans la Rec. UIT-T X.501 | ISO/CEI 9594-2) de l'endroit où l'information demandée est située, qu'ils retournent cette connaissance au demandeur ou qu'ils essayent, pour le compte de celui-ci, de faire aboutir sa demande. (Le demandeur peut être un DUA, un client LDAP ou un autre DSA: dans le dernier cas, les deux DSA doivent supporter le protocole du système d'annuaire (DSP, *directory system protocol*).)

Pour répondre à ces impératifs, trois modes d'interaction des DSA sont définis, à savoir l'"unichaînage", le "multichaînage" et le "renvoi de référence". Dans la suite du texte de la présente Spécification d'annuaire, le terme générique chaînage désignera l'unichaînage ou le multichaînage, selon le contexte. Le "chaînage" désigne la tentative faite par un DSA de répondre à une demande en envoyant une ou plusieurs opérations chaînées à d'autres DSA; le "renvoi de référence" désigne le renvoi d'informations de connaissances au demandeur, qui peut ensuite lui-même agir en interaction avec le DSA ou les DSA identifiés dans les informations de connaissances.

Une interaction par unichaînage ou par renvoi de référence peut faire suite à une demande isolée. En variante, la demande peut être décomposée en plusieurs sous-demandes avant l'interaction. Des interactions par multichaînage ou par renvoi de référence, ou une combinaison des deux modes, peuvent faire suite à une demande décomposée. Deux types de décomposition sont définis: la décomposition de NSSR et la décomposition de demande.

8.1 Décomposition d'une demande

8.1.1 Décomposition NSSR

La décomposition NSSR consiste à préparer des demandes identiques, prêtes au transfert (soit en séquence ou en parallèle) vers plusieurs DSA subordonnés, à la suite de la rencontre d'une NSSR au cours de la résolution du nom. Comme les références subordonnées non spécifiques ne contiennent pas les RDN des contextes de dénomination subordonnés appelés en référence, le DSA attribuant les références n'est pas en mesure de déterminer le DSA subordonné qui contient le ou les contextes de dénomination subordonnés. Au cours de la résolution du nom, un DSA qui rencontre des NSSR doit envoyer une demande identique à chaque DSA subordonné (en l'absence de duplication miroir). Cela peut être fait en séquence ou en parallèle. Généralement, un seul DSA sera en mesure de poursuivre la résolution du nom et les autres renverront l'erreur de service **serviceError** avec la cause **unableToProceed** (traitement impossible). Dans certaines (rares) circonstances, il se peut que plus d'un DSA poursuive la résolution du nom, ce qui donne lieu à des résultats dupliqués.

NOTE – Les NSSR ne peuvent référencer les serveurs LDAP.

8.1.2 Décomposition de demande

La décomposition de demande, autre façon de décomposer une demande, est un processus interne effectué par un DSA avant la communication avec un ou plusieurs autres DSA et/ou serveurs LDAP. Une demande est décomposée en plusieurs sous-demandes, si possible différentes, de telle manière que chacune de ces sous-demandes accomplisse une partie de la tâche initiale. La décomposition de demande ne peut être utilisée qu'au cours de l'évaluation d'une opération de listage (List) ou de recherche (Search). Après décomposition de la demande, chacune des sous-demandes peut être chaînée à d'autres DSA et/ou serveurs LDAP afin de poursuivre la tâche, ou un résultat partiel (renvoi de référence encastré) peut être renvoyé au demandeur. Un exemple de production d'une même sous-demande à destination de DSA et/ou serveurs LDAP différents se présente quand une entrée a des références subordonnées ou des NSSR qui renvoient ensemble à plusieurs DSA ou serveurs LDAP. Un exemple de sous-demandes différentes renvoyées aux mêmes DSA ou à des DSA et/ou serveurs LDAP différents se présente quand deux entrées différentes sont rencontrées au cours d'une recherche (sous-arbre), et que chacune a une référence subordonnée.

8.2 Unichaînage

Ce mode d'interaction (illustré à la Figure 2) peut être utilisé par un DSA pour communiquer une demande à un autre DSA, quand le premier a connaissance des contextes de dénomination détenus par le second. L'unichaînage peut être utilisé pour prendre contact avec un seul DSA désigné dans une référence croisée, une référence subordonnée, une référence supérieure, une référence de fournisseur ou une référence principale.

NOTE – Sur la Figure 2, l'ordre des interactions est défini par les numéros associés aux flèches.

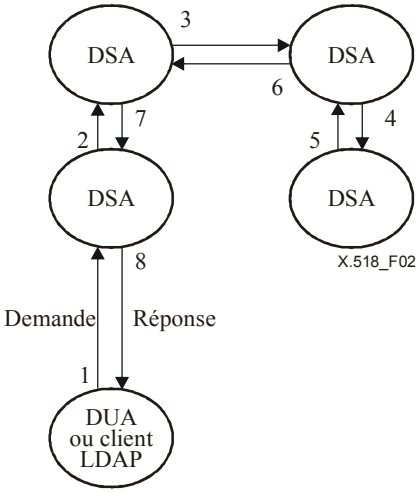


Figure 2 – Mode unichaînage

8.3 Multichaînage

Ce mode d'interaction est utilisé par un DSA pour transférer plusieurs demandes sortantes qui résultent d'une demande entrante, par suite de décomposition de demande ou de NSSR.

8.3.1 Multichaînage parallèle

En multichaînage parallèle, le DSA transfère simultanément plusieurs demandes sortantes (voir Figure 3a). Si le multichaînage parallèle peut améliorer la qualité de fonctionnement, il peut aussi, dans certaines circonstances telles que la duplication miroir, entraîner la réception de résultats en double.

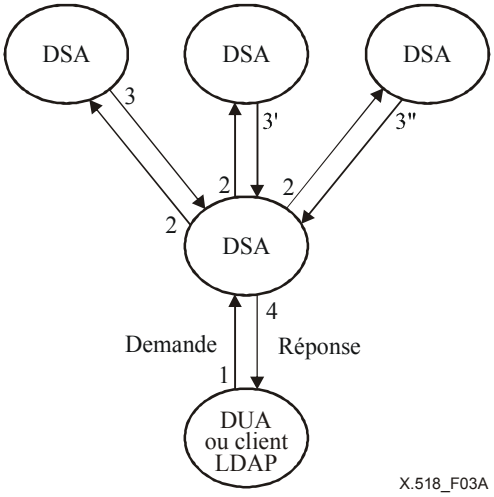
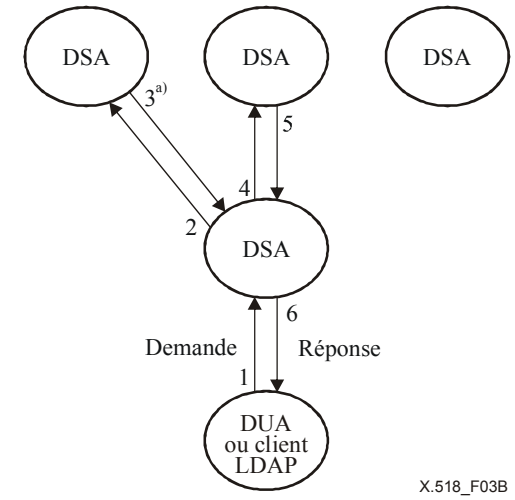


Figure 3a – Multichaînage parallèle

8.3.2 Multichaînage séquentiel

En multichaînage séquentiel, le DSA transfère une seule demande sortante à la fois et attend le résultat (ou l'erreur) correspondant à cette demande avant d'envoyer la suivante (voir Figure 3b). Si le multichaînage séquentiel n'est pas le mode d'interaction le plus rapide, il est cependant peu probable qu'il entraîne la réception de résultats en double.

NOTE – Un DSA peut utiliser une combinaison de multichaînage parallèle et de multichaînage séquentiel.



a) Traitement impossible.

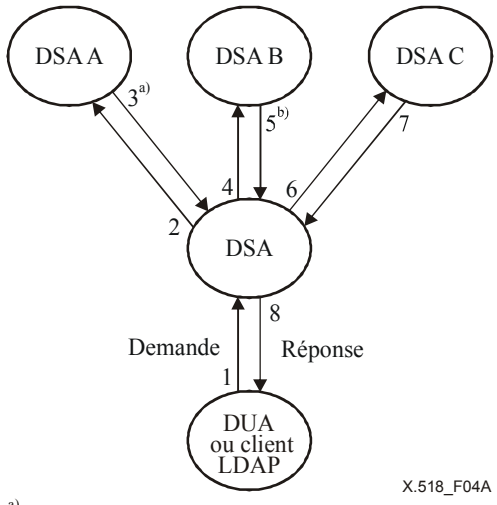
Figure 3b – Multichaînage séquentiel
(à la suite d'une décomposition de NSSR)

8.4 Renvoi de référence

Un renvoi de référence (illustré aux Figures 4a et 4b) est retourné par un DSA dans la réponse à une demande issue d'un DUA, d'un client LDAP ou d'un autre DSA. Le renvoi de référence peut constituer toute la réponse (auquel cas il est considéré comme une erreur) ou seulement une partie de la réponse. Le renvoi de référence contient une référence de connaissance, qui peut être une référence supérieure, subordonnée, croisée, subordonnée non spécifique, de fournisseur ou principale.

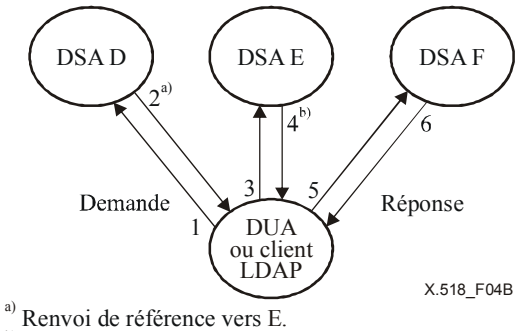
Le DSA (Figure 4a) recevant le renvoi de référence peut utiliser la référence de connaissance qui y est contenue pour ensuite chaîner ou multichaîner (selon le type de référence) la demande initiale vers d'autres DSA. Une autre possibilité serait qu'un DSA recevant un renvoi de référence le retourne à son tour dans sa réponse. Un DUA ou un client LDAP (Figure 4b) recevant un renvoi de référence peut l'utiliser pour prendre contact avec un ou plusieurs autres DSA afin de faire progresser la demande.

NOTE – Sur les Figures 4a et 4b, l'ordre des interactions est défini par les numéros associés aux flèches.



a) Renvoi de référence vers B.
b) Renvoi de référence vers C.

Figure 4a – Mode renvoi de référence
(action de DSA sur les renvois de référence)



a) Renvoi de référence vers E.
b) Renvoi de référence vers F.

Figure 4b – Mode renvoi de référence
(action de DUA sur des renvois de référence)

8.5 Détermination du mode

Si un DSA ne peut pas satisfaire entièrement lui-même une demande, il doit chaîner cette demande (ou une demande résultant de la décomposition de la demande initiale) vers un autre DSA, sauf si:

a) le chaînage est interdit par l'utilisateur via les commandes de service, auquel cas le DSA doit retourner un renvoi de référence ou une erreur de type **serviceError** avec la cause **chainingRequired**;

b) le DSA a des raisons d'ordre administratif, opérationnel ou technique de préférer le non-chaînage, auquel cas le DSA doit retourner un renvoi de référence.

NOTE 1 – Une "raison d'ordre technique" interdisant le chaînage est le fait que le DSA identifié dans la référence de connaissance ne gère pas le protocole DSP.

NOTE 2 – Si la commande de service **localScope** est activée, le DSA (ou le domaine de gestion d'annuaire (DMD)) doit soit résoudre la demande, soit retourner une erreur.

NOTE 3 – Si l'utilisateur préfère les renvois de référence, il active la commande **chainingProhibited**.

SECTION 4 – SERVICE ABSTRAIT DE DSA

9 Aperçu général du service abstrait de DSA

Le service d'annuaire est entièrement décrit dans la Rec. UIT-T X.511 | ISO/CEI 9594-3. Quand un tel service est fourni dans un environnement réparti, selon le modèle décrit au § 7, il peut être considéré comme fourni au moyen d'un ensemble de DSA. Cela est représenté sur la Figure 1.

Pour chaque opération définie dans le service d'annuaire, une opération "chaînée" correspondante est définie dans le service abstrait de DSA en vue d'être utilisée entre les DSA qui contribuent à l'accomplissement de cette opération du service d'annuaire. Ainsi, un DSA recevant une opération lecture de la part d'un DUA pourrait nécessiter l'assistance d'un autre DSA (par exemple un DSA détenant l'entrée cible ou une copie de cette entrée) pour en assurer l'accomplissement et envoyer ainsi à ce DSA une opération lecture chaînée.

Les types d'information échangés dans le service abstrait de DSA sont définis au § 10. Les opérations et les erreurs du service abstrait de DSA sont définies aux § 11 à 13.

10 Types d'information

10.1 Introduction

Le présent paragraphe identifie et définit parfois un certain nombre de types d'information qui seront utilisés par la suite pour la définition de diverses opérations du service abstrait de DSA. Les types d'information concernés sont ceux qui sont communs à plusieurs opérations, qui ont des chances d'être utilisés à l'avenir, ou qui sont suffisamment complexes ou autonomes pour mériter d'être définis séparément des opérations qui les utilisent.

Plusieurs des types d'information utilisés dans la définition du service abstrait de DSA sont en fait définis ailleurs. Le paragraphe 10.2 signale ces types et indique l'origine de leur définition. Chacun des § 10.3 à 10.10 suivants identifie et définit un type d'information.

10.2 Types d'information définis ailleurs

Les types d'information suivants sont définis dans la Rec. UIT-T X.501 | ISO/CEI 9594-2:

- **aliasedEntryName**;
- **DistinguishedName**;
- **Name**;
- **RelativeDistinguishedName**.

Les types d'information suivants sont définis dans la Rec. UIT-T X.511 | ISO/CEI 9594-3:

(Rattachement)

- **DirectoryBind**

(Opérations)

- **Abandon**

(Erreurs)

- **abandoned**;
- **attributeError**;
- **nameError**;
- **securityError**;
- **serviceError**;
- **updateError**.

(Classe d'objets d'information)

- **OPTIONALLY-PROTECTED**

(Type de données)

- **SecurityParameters**

Le type d'information suivant est défini dans la Rec. UIT-T X.520 | ISO/CEI 9594-6:

– **PresentationAddress**.

10.3 Arguments de chaînage

Le composant **ChainingArguments** figure dans chaque opération chaînée pour véhiculer vers un DSA les informations nécessaires au succès de l'exécution de sa partie de la tâche totale:

```
ChainingArguments ::= SET {
        originator                [0]   DistinguishedName OPTIONAL,
        targetObject              [1]   DistinguishedName OPTIONAL,
        operationProgress         [2]   OperationProgress
                                        DEFAULT { nameResolutionPhase notStarted },
        traceInformation          [3]   TraceInformation,
        aliasDereferenced         [4]   BOOLEAN DEFAULT FALSE,
        aliasedRDNs               [5]   INTEGER OPTIONAL,
                        -- présent uniquement dans les systèmes première édition
        returnCrossRefs           [6]   BOOLEAN DEFAULT FALSE,
        referenceType             [7]   ReferenceType DEFAULT superior,
        info                      [8]   DomainInfo OPTIONAL,
        timeLimit                 [9]   Time OPTIONAL,
        securityParameters        [10]  SecurityParameters DEFAULT { },
        entryOnly                 [11]  BOOLEAN DEFAULT FALSE,
        uniqueIdentifier          [12]  UniqueIdentifier OPTIONAL,
        authenticationLevel       [13]  AuthenticationLevel OPTIONAL,
        exclusions                [14]  Exclusions OPTIONAL,
        excludeShadows            [15]  BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster       [16]  BOOLEAN DEFAULT FALSE,
        operationIdentifier       [17]  INTEGER OPTIONAL,
        searchRuleId              [18]  SearchRuleId  OPTIONAL,
        chainedRelaxation         [19]  MRMapping  OPTIONAL,
        relatedEntry              [20]  INTEGER  OPTIONAL,
        dspPaging                 [21]  BOOLEAN DEFAULT FALSE,
        nonDapPdu                 [22]  ENUMERATED { ldap (0) } OPTIONAL,
        streamedResults           [23]  INTEGER OPTIONAL
        excludeWriteableCopies    [24]  BOOLEAN DEFAULT FALSE }
```

Les divers composants ont les significations définies ci-après:

a) le composant **originator** véhicule le nom du (dernier) initiateur de la demande, à moins qu'il ne soit déjà spécifié dans les paramètres de sécurité. Si le composant **requester** est présent dans le composant **CommonArguments**, cet argument peut être omis.

NOTE 1 – Lorsque l'initiateur est doté de plusieurs noms différenciés par le contexte, alors le nom utilisé comme valeur de **originator** doit être le nom distinctif primaire s'il est connu. Si tel n'était pas le cas, les mécanismes de contrôle d'accès et d'authentification fondés sur la valeur d'**originator** risqueraient de ne pas fonctionner comme prévu.

b) le composant **targetObject** véhicule le nom de l'objet dont l'entrée d'annuaire est la cible. Le rôle de cet objet dépend de l'opération concernée: il peut s'agir de l'objet dont l'entrée est la cible ou de l'entrée d'objet qui doit devenir la base pour une demande ou sous-demande impliquant plusieurs objets (par exemple **chainedList** ou **chainedSearch**). Ce composant ne peut être omis que s'il a la même valeur que le paramètre d'objet ou d'entrée d'objet de base dans l'opération chaînée, auquel cas sa valeur implicite sera cette valeur;

lorsque le **targetObject** comporte des RDN qui contiennent des types d'attributs et des couples de valeurs pour lesquels il existe plusieurs valeurs distinctives différenciées par le contexte, alors la résolution des RDN aura obligatoirement donné des RDN primaires;

c) le composant **operationProgress** s'utilise pour informer le DSA du déroulement de l'opération et donc du rôle que ce DSA est supposé jouer dans l'exécution de l'opération complète. Les informations véhiculées dans ce composant sont spécifiées au § 10.5;

d) le composant **traceInformation** est utilisé pour éviter le bouclage entre des DSA lorsque le chaînage est activé. Un DSA ajoute un nouvel élément à l'information de trace avant de chaîner une opération vers un autre DSA. Quand il reçoit une demande d'exécution d'une opération, un DSA vérifie, en examinant l'information de trace, que cette opération n'a pas formé de boucle. Les informations véhiculées dans ce composant sont spécifiées au § 10.6;

e) le composant **aliasDereferenced** est une valeur de type **BOOLEAN** utilisée pour indiquer si une ou plusieurs entrées alias ont déjà été rencontrées ou non et si leur nom a été échangé au cours de la

résolution répartie du nom. La valeur par défaut **FALSE** indique qu'aucune entrée alias n'a été déréférencée;

f) le composant **aliasedRDNs** indique combien de RDN contenus dans le composant **targetObject Name** ont été émis à partir des attributs **aliasedEntryName** d'une (ou de plusieurs) entrées alias. La valeur de l'entier est établie chaque fois qu'une entrée alias est rencontrée et déréférencée. Ce composant ne doit figurer que si et seulement si le composant **aliasDereferenced** a la valeur **TRUE**.

NOTE 2 – Ce paramètre est fourni pour assurer la compatibilité avec les implémentations de l'annuaire conformes à la première édition. Les agents DUA (et DSA) implémentés selon des éditions postérieures des Spécifications d'annuaire ne mentionneront jamais ce paramètre dans l'information **CommonArguments** d'une demande ultérieure. Cela permettra à l'annuaire de ne pas signaler d'erreur lorsque des alias déréférenceront d'autres alias.

g) le composant **returnCrossRefs** est une valeur booléenne qui indique si des références de connaissance, utilisées au cours de l'exécution d'une opération répartie, font ou non l'objet d'une demande de retour au DSA initial en tant que références croisées associées à un résultat ou à un renvoi de référence. La valeur par défaut **FALSE** indique que ces références de connaissance n'ont pas à être retournées;

h) le composant **referenceType** indique au DSA auquel il est demandé d'exécuter l'opération, quel type d'information de connaissance a été utilisé pour acheminer la demande vers lui. Le DSA peut ainsi détecter des erreurs dans les informations de connaissance détenues par l'invocateur. Si une telle erreur est détectée, elle doit être indiquée par un message **serviceError**, avec la cause **invalidReference**. Le composant **ReferenceType** est décrit en détail au § 10.7.

NOTE 3 – Si le composant **referenceType** manque, la valeur **superior** est implicite.

i) le composant **info** est utilisé pour véhiculer des informations spécifiques au domaine de gestion d'annuaire (DMD) entre les DSA qui sont impliqués dans le traitement en commun d'une demande. Ce composant est du type **DomainInfo**, qui est non restreint:

```
DomainInfo ::= ABSTRACT-SYNTAX.&Type
```

j) le composant **timeLimit**, s'il est présent, indique le délai d'exécution de l'opération (voir § 16.1.4.1). Avant qu'une valeur de **Time** ne soit utilisée dans une quelconque opération de comparaison et si la syntaxe de **Time** a été choisie comme étant du type **UTCTime**, la valeur du champ à deux chiffres indiquant l'année doit être convertie comme suit en une valeur à quatre chiffres:

– si la valeur à deux chiffres est comprise entre 00 et 49, inclusivement, on ajoutera 2000 à cette valeur;

– si la valeur à deux chiffres est comprise entre 50 et 99, inclusivement, on ajoutera 1900 à cette valeur.

NOTE 4 – L'emploi de **GeneralizedTime** peut éviter l'interfonctionnement avec des implémentations qui ignorent qu'il est possible de choisir **UTCTime** ou **GeneralizedTime**. Ceux qui spécifient les domaines dans lesquels la présente Spécification d'annuaire sera utilisée, par exemple, les groupes chargés d'établir les profils, auront à déterminer quand **GeneralizedTime** peut être utilisé. En aucun cas, il ne sera fait appel à **UTCTime** pour représenter des dates ultérieures à 2049.

k) le composant **SecurityParameters** est spécifié dans la Rec. UIT-T X.511 | ISO/CEI 9594-3. Son absence équivaut à un ensemble vide de paramètres de sécurité;

l) le composant **entryOnly** est mis à **TRUE** si l'opération initiale était une recherche (l'argument **subset** étant mis à **oneLevel**) et qu'une entrée alias ait été rencontrée en tant que subordonnée immédiate de l'entrée d'objet de base (**baseObject**). Le DSA qui réussit à effectuer la résolution du nom de l'objet cible (**targetObject**) n'effectuera l'évaluation de cet objet que sur l'entrée nommée;

m) le composant **uniqueIdentifier** est fourni facultativement quand il est nécessaire de confirmer le nom de l'initiateur. Il est décrit dans la Rec. UIT-T X.501 | ISO/CEI 9594-2;

n) le composant **authenticationLevel** est fourni facultativement quand il est nécessaire d'indiquer la manière dont l'authentification a été effectuée. Il est décrit dans la Rec. UIT-T X.501 | ISO/CEI 9594-2;

o) le composant **exclusions** n'a de signification que pour les opérations de recherche; s'il est présent, il indique les sous-arbres d'entrées subordonnées à l'objet cible (**targetObject**) qui seront exclues du résultat d'une opération de recherche (voir § 10.9);

p) le composant **excludeShadows** n'a de sens que pour les opérations de recherche et de listage. Il indique que la recherche doit être appliquée aux entrées et non pas aux copies d'entrées. Ce composant facultatif peut être utilisé par un DSA pour éviter la duplication des résultats reçus (voir § 20.1);

q) le composant **nameResolveOnMaster** n'a de sens que pendant la résolution du nom et est uniquement sélectionné si des NSSR ont été rencontrées. Avec la valeur **TRUE**, il indique que la résolution du nom ultérieure, c'est-à-dire l'adaptation des RDN qui restent à partir de **nextRDNToBeResolved**, ne doit pas utiliser l'information de copie d'entrée, y compris les copies programmables d'une implémentation multi-

maîtres; la résolution ultérieure de chaque RDN restant doit être effectuée dans le DSA maître pour l'entrée identifiée par ce RDN (voir § 20.1);

r) le composant **operationIdentifier** rend plus aisée la corrélation des opérations du DAP, tant avec les opérations ultérieures du DSP qui lui sont associées, qu'avec les résultats. Sa valeur est allouée par le premier DSA qui reçoit une demande DAP, ou est copiée sur les arguments de chaînage des demandes DSP qui nécessitent un chaînage subséquent. Il est interdit à un DSA qui alloue une valeur à **operationIdentifier** de réutiliser cet entier pendant une période suffisamment longue. La corrélation entre demandes et résultats correspondants des protocoles DAP et DSP sera plus facile si, pour chaque opération et pour chaque résultat, les DSA journalisent **operationIdentifier** en même temps que le nom du DSA qui lui a affecté sa valeur. C'est le premier DSA figurant dans le paramètre **traceInformation** d'une demande chaînée. Une telle corrélation peut trouver son utilité aux fins de journalisation, d'audit, de taxation et de paiement, etc.;

s) le composant **searchRuleId** véhicule l'identité unique d'une règle de recherche. Elle est incluse par le DSA qui effectue la procédure initiale **Search procedure (I)** au cas où cette procédure débute dans une zone administrative de service et que l'opération de recherche se poursuit vers d'autres DSA soit en progressant vers le bas du DIT en suivant des alias ou des pointeurs de groupes hiérarchiques;

t) le composant **chainedRelaxation** permet d'effectuer l'assouplissement de manière répartie pour des opérations de recherche chaînées. Si un DSA reçoit une opération de recherche chaînée et prend en charge les politiques d'assouplissement, il peut utiliser le composant **chainedRelaxation** fourni à la place de toute autre politique d'assouplissement qu'il pourrait implémenter, permettant ainsi à l'assouplissement d'être coordonné entre les DSA qui peuvent retourner des résultats de recherche;

u) l'élément **relatedEntry** doit être présent chaque fois que le DSA récepteur doit satisfaire des entrées associées. Lorsqu'il est présent, le DSA récepteur doit répondre *uniquement* à l'élément d'entrée associée spécifique spécifié par la valeur de **relatedEntry** dans **joinAttributes** de **SearchArgument**. Donc, une valeur zéro de **relatedEntry** doit sélectionner le premier élément de la séquence **joinAttributes** de **SearchArgument**. La valeur n'excédera jamais un nombre égal au nombre d'éléments moins un dans le composant **joinAttributes** de **SearchArgument**. L'absence de l'élément **relatedEntry** dans **ChainingArguments** d'une opération DSP spécifiant des entrées associées doit indiquer que l'opération répartie sur laquelle le chaînage est en cours est la recherche de base et non la partie entrée associée de la recherche.

NOTE 5 – Si un DSA en cours de chaînage doit traiter tant des résultats de recherches normales que des résultats d'entrées associées, cela se fera par l'envoi au DSA de deux opérations DSP distinctes.

Lorsque l'élément **relatedEntry** est présent, les règles spéciales suivantes s'appliquent:

– dans l'évaluation du sous-composant **infoTypes** du composant **selection** de **SearchArgument, infoTypes** sera considéré comme ayant la valeur **attributeTypesAndValues**, quelle que soit la valeur initialement spécifiée;

– tous les attributs spécifiés dans un composant **joinAtt** quelconque de **JoinAttPair** doivent être inclus dans la sélection, indépendamment du fait qu'il y étaient ou n'y étaient pas précédemment inclus;

– le DSA coordonnant des résultats d'entrées associées doit omettre des valeurs et des arguments non spécifiés de manière à rendre le résultat conforme à la demande initiale de l'utilisateur.

L'argument **relatedEntry** doit être transféré dans des arguments **ChainingArguments** résultants sortants par un DSA qui prend en charge des entrées associées.

v) si le DSA rattaché diffère de l'exécuteur initial (voir § 15.5.5) et supporte les résultats paginés du DSP, il peut alors mettre ce composant à la valeur **TRUE** afin de charger l'exécuteur initial de fournir des résultats paginés DSP. Si ce composant est mis à la valeur **FALSE** (valeur par défaut) l'exécuteur initial ne doit pas fournir de résultat paginé DSP. Un exécuteur initial qui supporte les résultats paginés DSP ne doit pas adresser ce composant aux DSA auxquels il envoie des sous-demandes;

w) le composant **nonDapPdu** sert à indiquer si l'unité PDU encapsulée dans l'argument chaîné provient d'une demande non DAP, par exemple une demande LDAP;

x) le composant **streamedResults** fait office de compteur afin de déterminer si des resultats en flux continu peuvent être chaînés en réponse à cette opération. Chaque DSA impliqué dans une résolution de nom incrémente le compteur d'une unité si et seulement si le compteur est présent, si le DSA admet des résultats en flux continu et s'il est prêt à recevoir des résultats en flux continu pour cette opération chaînée. Ce compteur est ensuite utilisé par le DSA qui termine la résolution de nom afin de déterminer si chacun des précédents DSA est prêt à traiter des résultats en flux continu;

y) le composant **excludeWriteableCopies** a un sens uniquement pour les opérations Search et List; il indique que la recherche doit être appliquée aux copies maîtres primaires des éléments et non aux copies

programmables de ces mêmes éléments. Un DSA peut se servir de ce composant facultatif pour éviter la réception de résultats dupliqués (voir § 20.1).

10.4 Résultats du chaînage

Le composant **ChainingResults** figure dans le résultat de chaque opération et fournit des informations en retour au DSA qui a invoqué cette opération.

```
ChainingResults ::= SET {
        info                [0]  DomainInfo OPTIONAL,
        crossReferences     [1]  SEQUENCE SIZE (1..MAX) OF CrossReference OPTIONAL,
        securityParameters  [2]  SecurityParameters DEFAULT { },
        alreadySearched     [3]  Exclusions OPTIONAL }
```

Les divers composants ont les significations définies ci-après:

a) le composant **info** est utilisé pour véhiculer des informations spécifiques à un DMD entre des DSA qui sont impliqués dans le traitement d'une demande commune. Ce composant est du type **DomainInfo**, qui est non restreint;

b) le composant **crossReferences** ne figure pas dans le composant **ChainingResults**, sauf si le composant **returnCrossRefs** de la demande correspondante a la valeur **TRUE**. Ce composant comprend une séquence d'items **CrossReference** dont chacun contient un **contextPrefix** et un descripteur de point d'accès (**accessPoint**) (voir § 10.8).

```
CrossReference ::= SET {
        contextPrefix       [0]  DistinguishedName,
        accessPoint         [1]  AccessPointInformation }
```

Un composant **CrossReference** peut être ajouté par un DSA quand il trouve une correspondance entre une partie de l'argument **targetObject** d'une opération et un de ses préfixes de contexte. L'autorité administrative d'un DSA peut avoir pour politique de ne pas retourner une telle information de connaissance: dans ce cas, elle n'ajoutera pas d'items à la séquence;

c) le composant **SecurityParameters** est spécifié dans la Rec. UIT-T X.511 | ISO/CEI 9594-3. Lorsque ce paramètre est absent, on considère qu'il existe un ensemble vide de paramètres de sécurité;

d) le composant **alreadySearched**, s'il est présent, indique quels RDN subordonnés au composant **targetObject** ont été traités dans le cadre d'une opération de recherche chaînée. Il doit donc être exclu lors d'une sous-demande subséquente.

NOTE – Les noms qui apparaissent dans **contextPrefix** et dans **alreadySearched** doivent obligatoirement être des noms distinctifs primaires et ne pas contenir de variantes de noms distinctifs.

10.5 Déroulement de l'opération

Une valeur **OperationProgress** décrit l'état d'avancement de l'exécution d'une opération à laquelle doivent participer plusieurs DSA.

```
OperationProgress ::= SET {
        nameResolutionPhase     [0]     ENUMERATED {
            notStarted          (1),
            proceeding          (2),
            completed           (3) },
        nextRDNToBeResolved     [1]     INTEGER OPTIONAL }
```

Les divers composants ont les significations définies ci-après:

a) le composant **nameResolutionPhase** indique la phase atteinte dans le traitement du nom d'une opération de type **targetObject**. Quand il indique que la résolution du nom a la valeur **notStarted**, cela signale qu'il n'a pas encore été possible d'atteindre un DSA détenant un contexte de dénomination contenant le ou les RDN initiaux du nom. Si la résolution du nom a la valeur **proceeding**, cela indique que la partie initiale du nom a été reconnue, mais que le DSA contenant l'objet cible n'a pas encore été atteint. Le composant **nextRDNToBeResolved** indique quelle fraction du nom a déjà été reconnue [voir § 10.5 b)]. Si la résolution du nom a la valeur **completed**, le DSA contenant le composant targetObject a été atteint et l'exécution de l'opération proprement dite est en cours;

b) le composant **nextRDNToBeResolved** indique au DSA quel est le prochain des RDN contenus dans le composant **targetObject** qu'il faudra résoudre. Il a la forme d'un entier compris entre 1 et le nombre de RDN du nom. Ce composant n'est présent que si le composant **nameResolutionPhase** a la valeur **proceeding**.

10.6 Information de trace

Une valeur **TraceInformation** véhicule un enregistrement des DSA qui ont déjà participé à l'exécution de l'opération. Elle est utilisée pour détecter l'existence de bouclages, ou pour les éviter car ces bouclages peuvent provenir de l'incohérence d'informations de connaissance ou de la présence de bouclages d'alias dans le DIT.

```
TraceInformation ::= SEQUENCE OF TraceItem

TraceItem ::= SET {
        dsa                 [0]  Name,
        targetObject        [1]  Name  OPTIONAL,
        operationProgress   [2]  OperationProgress }
```

Chaque DSA qui renvoie une opération vers un autre agent ajoute un nouvel élément à la fin de la séquence des composants **TraceItem**. Chacun de ces composants **TraceItem** contient:

a) le nom de DSA qui ajoute l'item;

b) le nom du composant **targetObject** que le DSA ajoutant l'item a reçu avec la demande entrante. Ce paramètre est omis si sa demande à chaîner provient d'un DUA (auquel cas sa valeur implicite est le composant **object** ou **baseObject** dans **XOperation**) ou si sa valeur est la même que celle du composant **targetObject** (réelle ou implicite) contenu dans l'argument **ChainingArgument** de la demande sortante;

c) la valeur **operationProgress** que le DSA ajoutant l'item a reçue au sujet de la demande entrante.

dsa doit être le nom distinctif primaire. Il ne doit pas contenir de variantes de noms distinctifs. Tout RDN appartenant au **targetObject** qui a été traité doit être un RDN primaire. Des variantes de valeurs distinctives dépendant du contexte peuvent figurer au sein du composant **valuesWithContext** de **AttributeTypeAndDistinguishedValue** du RDN.

10.7 Type de référence

Une valeur **ReferenceType** indique un des divers types de référence définis dans la Rec. UIT-T X.501 | ISO/CEI 9594-2.

```
ReferenceType ::= ENUMERATED  {
        superior                (1),
        subordinate             (2),
        cross                   (3),
        nonSpecificSubordinate  (4),
        supplier                (5),
        master                  (6),
        immediateSuperior       (7),
        self                    (8),
        ditBridge               (9) }
```

10.8 Information sur le point d'accès

Il y a trois types de points d'accès.

a) une valeur de point d'accès **AccessPoint** identifie un point particulier où peut s'effectuer un accès à l'annuaire, plus précisément à un DSA ou à un serveur LDAP. Pour se référer à un DSA, le point d'accès doit avoir un nom **Name**, celui du DSA concerné et peut avoir une adresse de présentation **PresentationAddress**, à utiliser lors des communications OSI ou IDM vers ce DSA, auquel cas un identificateur labeledURI ne doit pas être présent. Pour se référer à un serveur LDAP, le point d'accès doit avoir un identificateur **labeledURI**, à utiliser lors des communications LDAP vers ce serveur LDAP. Si l'élément **labeledURI** est présent, **Name** et **PresentationAddress** ne doivent pas être pris en compte et **SET OF protocolInformation** ne doit pas être présent.

```
AccessPoint ::= SET {
    ae-title            [0]     Name,
    address             [1]     PresentationAddress,
    protocolInformation [2]     SET SIZE (1..MAX) OF ProtocolInformation OPTIONAL,
    labeledURI          [6]     LabeledURI OPTIONAL }
```

```
LabeledURI ::= DirectoryString{ub-labeledURI}
```

b) une valeur **MasterOrShadowAccessPoint** identifie un point d'accès à l'annuaire. Le composant **category**, (de valeur **master** ou **shadow**) du point d'accès est différent selon que ce point est orienté vers un contexte de dénomination ou vers une zone copiée d'utilisation commune. Le composant **chainingRequired** indique si le chaînage est nécessaire pour ce DSA, c'est-à-dire une orientation ne sera pas renvoyée pour ce DSA.

```
MasterOrShadowAccessPoint ::= SET {
    COMPONENTS OF           AccessPoint,
    category          [3]   ENUMERATED {
        master              (0),
        shadow              (1) } DEFAULT master,
    chainingRequired  [5]   BOOLEAN  DEFAULT FALSE }
```

c) une valeur **MasterAndShadowAccessPoints** identifie une série de points d'accès à l'annuaire, à savoir un groupe de DSA et/ou de serveurs LDAP en relation. Ces points d'accès ont la caractéristique commune que chacun se réfère à un DSA ou un serveur LDAP détenant une information d'entrée provenant d'un contexte de dénomination commun ou d'un tronc commun de contextes de dénomination détenus par un même DSA quand la valeur est celle de l'attribut **nonSpecificKnowledge**. Une valeur **MasterAndShadowAccessPoints** indique le composant **category** de chaque valeur **AccessPoint** qu'elle contient. Le point d'accès du DSA ou de serveur LDAP maître du contexte de dénomination ne doit pas nécessairement faire partie de l'ensemble.

NOTE – Les réalisateurs doivent tenir compte du fait qu'un serveur LDAP, même s'il est identifié comme **shadow**, peut mettre à jour ses éléments d'information en réponse à une opération reçue de mise à jour LDAP.

```
MasterAndShadowAccessPoints ::= SET SIZE (1..MAX) OF MasterOrShadowAccessPoint
```

Une valeur **AccessPointInformation** identifie un ou plusieurs points d'accès à l'annuaire.

```
AccessPointInformation ::= SET {
    COMPONENTS OF           MasterOrShadowAccessPoint,
    additionalPoints  [4]   MasterAndShadowAccessPoints OPTIONAL }
```

Si des DSA première édition produisent une valeur **AccessPointInformation**, la composante facultative de l'ensemble est absente. Lorsque des DSA première édition interprètent une valeur **AccessPointInformation**, toute valeur **MasterAndShadowAccessPoints** présente est ignorée.

Dans le cas des DSA deuxième édition et suivantes, la composante de valeur **MasterOrShadowAccessPoint**, produite pour une valeur **AccessPointInformation**, peut appartenir à la catégorie maître ou à la catégorie duplication miroir, selon ce qui est déterminé par la procédure de sélection de connaissance du DSA produisant la valeur. On peut la considérer comme un point d'accès suggéré fourni par le DSA qui produit la valeur à l'intention du DSA qui la reçoit. Une valeur **MasterAndShadowAccessPoints** peut aussi être (facultativement) produite pour une valeur **AccessPointInformation**. Cela constitue une information supplémentaire qui pourra être utilisée par la procédure de sélection de connaissance du DSA récepteur pour déterminer un point d'accès de rechange.

10.9 Connaissance de routage DIT

Une valeur **ditBridgeKnowledge** identifie un point particulier permettant d'accéder à un autre arbre DIT, notamment à un DSA ou un serveur LDAP. **ditBridgeKnowledge** spécifie un point **accessPoint** au niveau duquel l'accès à ce DSA ou au serveur LDAP est possible.

```
DitBridgeKnowledge ::= SEQUENCE {
    domainLocalID           DirectoryString{ub-domainLocalID}  OPTIONAL,
    accessPoints            MasterAndShadowAccessPoints }
```

domainLocalID contient une description lisible identifiant l'arbre DIT contenu dans la référence.

10.10 Exclusions

Comme cela est défini au § 10.3, le composant **exclusions** de **ChainingArguments** s'utilise pour limiter la portée de l'opération de recherche en identifiant un certain nombre d'entrées subordonnées à l'entrée de l'objet cible qui, avec toutes leurs autres subordonnées, ne doivent pas faire partie du traitement de l'opération de recherche. Le composant **exclusion** est défini comme une valeur du type **Exclusions** de l'ASN.1.

```
Exclusions ::= SET SIZE (1..MAX) OF RDNSequence
```

Chaque valeur **RDNSequence** du groupe **Exclusions** doit normalement identifier le préfixe du contexte d'un subordonné du contexte de dénomination à l'entrée de l'objet cible. Si un DSA reçoit une demande de recherche avec une valeur **RDNSequence** n'obéissant pas à cette contrainte, ce DSA peut ignorer la valeur en question. La valeur **RDNSequence** se rapporte à l'objet cible et n'est pas le nom distinctif du préfixe de contexte.

Les **Exclusions** doivent être des noms distinctifs primaires. Il est permis d'y inclure aussi des variantes de noms distinctifs et des informations contextuelles.

Les **Exclusions** peuvent, non seulement faire partie d'une demande d'utilisateur mais aussi être utilisées par des DSA pour minimiser les informations en double provenance de sous-demandes de recherche émises en présence d'informations miroirs.

La Figure 5 illustre un exemple de l'utilisation du type **Exclusions**. Dans cet exemple, un DSA détient deux zones copiées, l'une en dessous de l'autre. L'une débute par le préfixe de contexte X, l'autre par le préfixe de contexte C. Une copie d'entrée à Y possède trois références subordonnées à trois contextes de dénomination A, B et C.

Si, à titre d'exemple, on effectue pour ce DSA une recherche dans un sous-arbre débutant par une entrée d'objet de base contenue dans un contexte de dénomination X, le DSA peut fournir des informations en provenance des zones copiées X et C. Les informations en provenance des contextes de dénomination A et B peuvent être fournies via les références subordonnées. A la décomposition de la demande, le composant **ContinuationReference**, qu'il faudra utiliser dans des résultats partiels (**partialResults**) ou dans un chaînage, spécifiera Y comme l'objet cible et C comme l'élément unique d'un ensemble de type **Exclusions**.

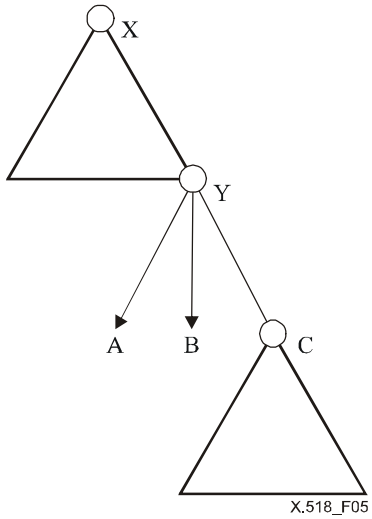


Figure 5 – Exclusions

10.11 Référence de continuation

Le composant **ContinuationReference** décrit comment l'exécution de l'ensemble ou d'une partie d'une opération peut être poursuivie par un ou plusieurs DSA ou serveurs LDAP (ou combinaison de ces éléments) différents. Il est normalement retourné comme renvoi de référence quand le DSA concerné est incapable de faire suivre la demande lui-même ou ne le désire pas.

```
ContinuationReference ::= SET {
        targetObject            [0]  Name,
        aliasedRDNs             [1]  INTEGER OPTIONAL,  -- présent uniquement dans les systèmes
                                                        -- première édition
        operationProgress       [2]  OperationProgress,
        rdnsResolved            [3]  INTEGER OPTIONAL,
        referenceType           [4]  ReferenceType,
        accessPoints            [5]  SET OF AccessPointInformation,
        entryOnly               [6]  BOOLEAN DEFAULT FALSE,
        exclusions              [7]  Exclusions OPTIONAL,
        returnToDUA             [8]  BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster     [9]  BOOLEAN DEFAULT FALSE }
```

Ces divers composants ont les significations définies ci-après:

a) le composant **targetObject** indique le nom qui est proposé pour la suite de l'opération. Il peut être différent du nom **targetObject** reçu dans la demande entrante si, par exemple, un alias a été déréférencé ou si l'entrée d'objet de base d'une recherche a été localisée.

   Les RDN déjà traités de **targetObject** doivent être des RDN primaires. Il peut s'y trouver aussi des variantes de valeurs distinctives avec contexte;

b) le composant **aliasedRDNs** indique combien de RDN éventuellement contenus dans le nom de l'objet cible, résultent de déréférencements d'alias. L'argument n'apparaît que si un nom d'alias a été déréférencé.

   NOTE – Ce paramètre est fourni pour assurer la compatibilité avec les implémentations de l'annuaire selon la première édition. Les agents DUA (et DSA) implémentés selon des éditions postérieures des Spécifications d'annuaire ne mentionneront jamais ce paramètre dans l'information **CommonArguments** d'une demande ultérieure. Cela permettra à l'annuaire de ne pas signaler d'erreur lorsque des alias déréférenceront d'autres alias.

c) le composant **operationProgress** indique l'état d'avancement de la résolution du nom qui a été réalisée et qui déterminera la suite de l'exécution de l'opération par les DSA nommés, à condition que le DSA ou le DUA recevant la référence **ContinuationReference** souhaite lui donner suite;

d) la valeur du composant **rdnsResolved** (qui doit apparaître que si certains des RDN du nom n'ont pas été pleinement résolus, mais ont été considérés comme corrects en vertu d'une référence croisée) indique combien de RDN ont été effectivement résolus, en utilisant uniquement des références internes;

e) le composant **referenceType** indique le type d'informations de connaissance qui a été utilisé pour produire cette référence de continuation;

f) le composant **accessPoints** indique les points d'accès qu'il faut contacter pour réaliser cette continuation. Il ne peut y avoir plus d'un item **AccessPointInformation** qu'en présence de références subordonnées non spécifiques;

g) le composant **entryOnly** est mis à la valeur **TRUE** si l'opération originale était une recherche avec l'argument **subset** mis à **oneLevel** et qu'une entrée alias ait été rencontrée en tant que subordonné immédiat de l'objet **baseObject**. Le DSA qui exécute sans problème la résolution de nom au sujet du nom **targetObject** doit effectuer l'évaluation d'objet sur la seule entrée nommée;

h) le composant **exclusions** identifie une série de contextes de dénomination subordonnés qui ne doivent pas être explorés par le DSA récepteur;

i) l'élément **returnToDUA** est fourni facultativement quand le DSA utilisant la référence de continuation souhaite indiquer qu'il ne veut pas retourner des informations via un DSA intermédiaire (pour des raisons de sécurité par exemple) et qu'il désire faire savoir que ces informations peuvent être directement obtenues par une opération du protocole d'accès à l'annuaire (DAP) ou du protocole rapide d'accès à l'annuaire (LDAP) entre le DUA initiateur ou client LDAP et le DSA. Lorsque l'élément **returnToDUA** est mis à **TRUE**, le composant **referenceType** peut être mis à **self**;

j) sur option, l'élément **nameResolveOnMaster** est fourni lorsque le DSA qui crée la référence de continuation a rencontré des références de type NSSR. Si cet élément est mis à **TRUE**, il signale que la résolution de nom suivante (correspondant aux RDN restants, indiqués par le composant **nextRDNToBeResolved**) ne doit pas employer d'informations de copie d'entrée, y compris de copies programmables dans le cas d'une implémentation multimaîtres; la résolution de nom subséquente doit être faite – pour chaque RDN restant à résoudre – au niveau du DSA maître pour l'entrée identifiée par ce RDN (voir § 20.1).

11 Rattachement et détachement

Les opérations **DSABind** et **DSAUnbind** sont respectivement utilisées par un DSA, au début et à la fin d'une période d'accès à un autre DSA. Le rattachement ou détachement d'une association DSP ne doit pas entraîner, par lui-même, la perte résultats paginés répartis qui ont été demandés au cours de l'association.

11.1 DSA Bind (rattachement de DSA)

L'opération **DSABind** s'utilise pour commencer une période de coopération entre deux DSA assurant le service d'annuaire.

```
DSABind ::= BIND
        ARGUMENT        DirectoryBindArgument
        RESULT          DirectoryBindResult
        BIND-ERROR      DirectoryBindError
```

Les composants de la **DSABind** sont identiques à leurs homologues dans DirectoryBind (voir la Rec. UIT-T X.511 | ISO/CEI 9594-3) sauf sur les points suivants:

- le composant **Credentials** (accréditation) de l'argument **DirectoryBindArgument** permet d'envoyer au DSA répondeur des renseignements identifiant le titre d'AE du DSA initiateur. Ce titre d'entité d'application doit avoir la forme d'un nom distinctif d'annuaire;
- le composant **Credentials** du résultat **DirectoryBindResult** permet d'envoyer au DSA initiateur des renseignements identifiant le titre d'AE du DSA répondeur. Ce titre d'entité d'application doit avoir la forme d'un nom distinctif;
- il est permis au nom du DSA et au titre d'AE d'employer des noms distinctifs auxiliaires et de comporter des informations contextuelles.

NOTE 1 – Lorsque des noms sont utilisés dans des justificatifs d'identité simples ou renforcés, il est possible d'utiliser des variantes de noms distinctifs, s'il en existe. Mais le fonctionnement de l'authentification et du contrôle d'accès par nom risque de ne pas être à la hauteur des espérances si le nom distinctif primaire n'est pas employé. Après qu'une opération de BIND authentifiée a été traitée avec succès, afin de rendre plus aisé le fonctionnement des contrôles d'accès pendant la durée d'action de ce BIND, les entités rattachées se connaîtront mutuellement par leurs noms distinctifs primaires, quel qu'ait été le nom utilisé dans l'argument du BIND.

NOTE 2 – Les justificatifs d'identité nécessaires à l'authentification peuvent être transportés par l'élément de service d'échange de sécurité (voir la Rec. UIT-T X.519 | ISO/CEI 9594-5), auquel cas ils n'apparaissent ni dans les arguments du BIND ni dans les résultats.

11.2 DSA Unbind (détachement de DSA)

Le détachement à la fin d'une période de coopération entre deux DSA assurant le service d'annuaire correspond à l'environnement OSI spécifié aux § 7.6.4 et 7.6.5 de la Rec. UIT-T X.519 | ISO/CEI 9594-5 et à l'environnement TCP/IP spécifié au § 9.3.2 de la Rec. UIT-T X.519 | ISO/CEI 9594-5.

12 Opérations chaînées

Pour chacune des opérations utilisées pour accéder au service d'annuaire, il existe une opération utilisée entre des DSA qui coopèrent au moyen d'une correspondance bilatérale. Les noms des opérations ont été choisis de manière à refléter cette correspondance, c'est-à-dire qu'on a donné le préfixe "chained" aux noms des opérations utilisées entre DSA coopérants.

Les arguments, résultats et erreurs des opérations chaînées sont, à une exception près, formés systématiquement à partir des arguments, résultats et erreurs des opérations correspondantes du service d'annuaire (voir § 12.1). Cette exception est l'opération **ChainedAbandon**, qui est syntaxiquement équivalente à son homologue du service d'annuaire (voir § 12.2).

12.1 Opérations chaînées

Un DSA qui a reçu d'un DUA ou d'un client LDAP une opération peut choisir de construire une forme chaînée de cette opération afin de la communiquer à un autre DSA. Un DSA qui a reçu une forme chaînée d'une opération peut également choisir de la concaténer pour l'envoyer à un autre DSA. Le DSA qui invoque une forme chaînée d'une opération peut, s'il le désire, signer, chiffrer ou signer et chiffrer l'argument de cette opération; le DSA exécutant l'opération peut, sur demande, signer, chiffrer ou signer et chiffrer le résultat ou l'indication d'erreur que renvoie le répondeur de cette opération. Un DSA qui a reçu d'un client LDAP une opération ou qui a reçu d'un autre DSA une opération LDAP, peut choisir de communiquer à un serveur LDAP l'opération initiale fournie par le client LDAP.

La forme chaînée d'une opération est spécifiée au moyen du type paramétré **chained { }**.

```
chained { OPERATION : operation } OPERATION ::= {
        ARGUMENT  OPTIONALLY-PROTECTED {
            SET {
                chainedArgument      ChainingArguments,
                argument         [0] operation.&ArgumentType } }
        RESULT      OPTIONALLY-PROTECTED {
            SET {
                chainedResult        ChainingResults,
                result           [0] operation.&ResultType } }
        ERRORS{ operation.&Errors EXCEPT referral | dsaReferral }
        CODE    operation.&operationCode }
```

NOTE 1 – Les opérations du service abstrait d'annuaire que l'on peut utiliser comme paramètre associé au type **chained { }** comprennent l'erreur **abandoned**. La présence de cette erreur faisant partie de l'ensemble des erreurs possibles d'une opération

chaînée reflète la possibilité examinée au § 12.2 qu'un item **ChainedAbandon** puisse être produit pour une opération **ChainedModify** quand une association liée n'aboutit pas.

NOTE 2 – La spécification définitive du service abstrait de DSA (voir l'Annexe A) applique ce type paramétré pour construire toutes les opérations chaînées du service abstrait.

L'argument de l'opération dérivée contient les composants suivants:

a) le composant **chainedArgument** – C'est une valeur de l'élément **ChainingArguments** contenant les informations (en plus et au-dessus de l'argument initial fourni par le DUA ou le client LDAP) qui sont nécessaires au DSA ou au serveur LDAP exécutant pour effectuer l'opération. Ce type d'information est défini au § 10.3;

b) le composant **argument** – C'est une valeur de l'élément **operation.&Argument** qui contient l'argument initialement fourni par le DUA, tel que spécifié au paragraphe approprié de la Rec. UIT-T X.511 | ISO/CEI 9594-3, l'argument initial fourni par le client LDAP, tel que spécifié dans le paragraphe approprié de la norme RFC 2251.

NOTE 3 – S'il est jugé approprié, l'encapsulage de types d'unités PDU autres que celles provenant du DAP ou du LDAP est également possible. La spécification des mécanismes à prévoir à cet effet doit faire l'objet d'un complément d'étude.

Si la demande aboutit, le résultat sera retourné. Les paramètres du résultat auront le sens suivant:

a) le composant **chainedResult** – C'est une valeur de l'élément **ChainingResults** qui contient les informations, en plus et au-dessus de celles qu'il faut fournir au DUA demandeur, dont peuvent avoir besoin les DSA précédents de la chaîne. Ce type d'information est défini au § 10.4;

b) le composant **result** – C'est une valeur de l'élément **operation.&Result** qui constitue le résultat retourné par le DSA ayant exécuté l'opération et qui est destinée à être communiquée dans le résultat au DUA demandeur. Cette information est telle que spécifiée au paragraphe approprié de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

Si la demande échoue, une des erreurs énumérées dans l'ensemble **operation.&Errors** sera retournée, sauf qu'une erreur **dsaReferral** sera retournée à la place d'une erreur **referral**. L'ensemble des erreurs qui peuvent être signalées est celui qui est décrit pour l'opération correspondante dans la Rec. UIT-T X.511 | ISO/CEI 9594-3. L'erreur de type **dsaReferral** est décrite au § 13.2.

12.2 Opération ChainedAbandon

L'opération **ChainedAbandon** est utilisée par un DSA pour indiquer à un autre DSA qu'il n'est plus intéressé par l'exécution d'une opération répartie qui a été invoquée antérieurement. Plusieurs raisons peuvent motiver cet abandon, comme les suivantes:

– l'opération qui a conduit le DSA à effectuer initialement un chaînage a elle-même été abandonnée ou a été interrompue implicitement par la terminaison d'une association;

– le DSA a obtenu les informations nécessaires d'une autre façon, par exemple de la part d'un DSA impliqué dans le multichaînage parallèle et qui a répondu plus rapidement.

Un DSA n'est jamais obligé d'émettre un composant **ChainedAbandon**, ou d'abandonner une opération, si on lui demande de le faire.

Si le composant **ChainedAbandon** réussit effectivement à interrompre l'exécution d'une opération, un résultat sera retourné et l'opération concernée retournera une erreur **abandoned**. Si **ChainedAbandon** ne réussit pas à interrompre l'opération, il doit retourner lui-même une erreur de type **abandonFailed**.

12.3 Opérations chaînées et version de protocole

Il n'est permis de chaîner des opérations qui, comme **modifyEntry** avec certains arguments, exigent une version du protocole de numéro plus élevé que v1, ou qui, comme **modifyEntry** avec argument signé, renvoient des résultats différents lorsqu'elles sont employées avec une version de protocole de numéro plus élevé que v1, que sur des associations qui utilisent une version de protocole portant un numéro au moins égal à celui de la version utilisée pour passer la demande.

13 Erreurs chaînées

13.1 Introduction

Dans le service abstrait de DSA, les erreurs qui peuvent être retournées sont, pour la plupart, les mêmes que celles du service abstrait d'annuaire. Les exceptions sont que l'"erreur" **dsaReferral** (voir § 13.2) est retournée au lieu de **Referral** et que les problèmes de service suivants ont la même syntaxe abstraite mais une sémantique différente:

a) **invalidReference** – Le DSA qui retourne cette erreur a détecté une erreur dans les informations de connaissance du DSA appelant comme spécifié dans l'argument de chaînage **referenceType**;

b) **loopDetected** – Le DSA qui retourne cette erreur a détecté une boucle dans les informations de connaissance de l'annuaire.

L'ordre de préséance des erreurs qui peuvent se produire est le même que dans le service d'annuaire, tel que spécifié dans la Rec. UIT-T X.511 | ISO/CEI 9594-3.

Si une erreur se produit au cours d'une opération chaînée, le DSA en réponse est autorisé à signer, à chiffrer, ou à signer et chiffrer l'indication d'erreur renvoyée.

13.2 Renvoi de référence de DSA

L'erreur **dsaReferral** est émise par un DSA qui, pour une raison quelconque, ne souhaite pas continuer l'exécution d'une opération en faisant suivre cette opération à un ou à plusieurs autres DSA par chaînage. Les circonstances dans lesquelles il peut retourner un renvoi de référence sont décrites au § 8.3.

```
dsaReferral   ERROR  ::= {
        PARAMETER OPTIONALLY-PROTECTED {
            SET {
                reference           [0]   ContinuationReference,
                contextPrefix       [1]   DistinguishedName OPTIONAL,
                COMPONENTS OF CommonResults } }
        CODE            id-errcode-dsaReferral }
```

Les divers paramètres ont le sens suivant:

a) le composant **ContinuationReference** contient les informations nécessaires à l'invocateur pour donner suite à une demande ultérieure appropriée, éventuellement vers un autre DSA ou un serveur LDAP. Ce type d'information est spécifié au § 10.11;

b) si le composant **returnCrossRefs** de l'élément **ChainingArguments** pour cette opération avait la valeur **TRUE** et que le renvoi de référence soit fondé sur une référence subordonnée ou croisée, le paramètre **contextPrefix** peut être (le cas échéant) inclus. L'autorité administrative d'un DSA quelconque décidera quelles références d'informations de connaissance peuvent, le cas échéant, être retournées de cette façon (les autres pourront être, par exemple, confidentielles pour ce DSA).

contextPrefix et **ContinuationReference** doivent être nécessairement des noms distinctifs primaires. Des variantes de valeurs distinctives dépendant du contexte peuvent figurer au sein du composant **valuesWithContext** de **AttributeTypeAndDistinguishedValue** de n'importe quel RDN.

Les renseignements fournis peuvent facultativement être qualifiés au moyen du composant **notification** de **CommonResults.**

SECTION 5 – PROCÉDURES DE FONCTIONNEMENT RÉPARTI

14 Introduction

14.1 Domaine d'application et limites

Le présent paragraphe spécifie les procédures de fonctionnement réparti de l'annuaire qui sont exécutées par des DSA. Chaque DSA exécute individuellement les procédures décrites ci-après; l'action collective de tous les DSA constitue l'intégralité des services fournis aux utilisateurs de l'annuaire.

14.2 Conformité

La description des procédures de DSA contenues dans la présente section est fondée sur les modèles des paragraphes 8 et 9 de la Rec. UIT-T X.501 | ISO/CEI 9594-2, ainsi que des paragraphes 7 et 8 de la présente Spécification d'annuaire. Les organigrammes et leurs descriptions textuelles correspondantes sont un moyen d'établir le mappage entre une série de données d'entrée de DSA externes (protocole d'accès, protocole rapide d'accès ou protocole de système d'annuaire) et une ou plusieurs sorties externes (à savoir un résultat, une erreur, un renvoi de référence ou des demandes chaînées) produites par le DSA en question, selon l'arbre des informations particulier de DSA détenu par ce DSA.

Il est probable que l'annuaire sera réparti entre des DSA implémentés conformément à différentes éditions des Spécifications d'annuaire, comme à celles dont l'implémentation autorise seulement la prise en charge du protocole LDAP. Le DUA ou le client LDAP lançant la demande n'aura pas connaissance de l'édition selon laquelle le ou les DSA répondant à la demande du DUA ou du client LDAP auront été implémentés. Pour permettre le fonctionnement dans un environnement aussi hétérogène, un DSA doit être implémenté conformément aux règles d'extensibilité définies dans le § 12 de la Rec. UIT-T X.519 | ISO/CEI 9594-5.

NOTE 1 – Les DSA implémentés de façon à ne prendre en charge que le protocole LDAP ne sont pas nécessairement implémentés conformément aux règles d'extensibilité.

L'implémentation d'un DSA doit être l'équivalent, sur le plan fonctionnel, du comportement externe spécifié par les procédures décrites ici. Les algorithmes utilisés par une implémentation de DSA particulière en vue de déterminer la ou les sorties correctes à partir des entrées données et de l'arbre des informations de DSA, ne sont pas normalisés.

NOTE 2 – Les organigrammes qui accompagnent les procédures sont destinés à servir d'aide à la compréhension de ces procédures. Ils ne doivent pas être considérés comme étant une variante expresse des descriptions textuelles. Lorsqu'il existe une disparité entre la description textuelle et l'organigramme pour une procédure particulière, il est prévu que la description textuelle ait priorité.

14.2.1 Interaction impliquant un DSA première édition

Si les opérations de modification s'étendent au-delà des limites du DSA (par exemple **AddEntry** avec **TargetSystem**, enlever ou renommer un préfixe de contexte), la présente Spécification d'annuaire précise uniquement le comportement de deux DSA deuxième édition ou édition ultérieure. L'interaction entre deux DSA première édition ou entre un DSA première édition et un DSA deuxième édition ou édition ultérieure n'est pas incluse dans le domaine d'application des présentes Spécifications d'annuaire. Lorsque des DSA conformes à des éditions différentes sont en rattachement opérationnel hiérarchique, la connaissance par chacun de l'édition de ses homologues permettra de donner à l'utilisateur un message d'erreur homogène.

14.3 Modèle conceptuel

La complexité du fonctionnement réparti de l'annuaire rend nécessaire une modélisation conceptuelle utilisant des techniques de description aussi bien narratives que graphiques. Toutefois, aucun des diagrammes narratifs ou graphiques ne saurait être interprété comme une description formelle du fonctionnement réparti de l'annuaire.

14.4 Fonctionnement individuel et coopératif des DSA

Le modèle décrit le fonctionnement des DSA de deux points de vue séparés, dont la réunion donne une représentation opérationnelle complète de l'annuaire.

a) **Point de vue DSA** – Dans cette optique, l'ensemble des procédures qui gèrent l'annuaire est décrit du point de vue d'un seul DSA. Cela permet de donner une spécification définitive de chaque procédure et de rendre intégralement compte de leurs relations et de la structure générale de commande. Les procédures des DSA du point de vue DSA sont décrites aux paragraphes 16 à 22.

b) **Point de vue opération** – Le point de vue DSA donne une description détaillée complète mais rend difficile la compréhension de la structure des opérations individuelles qui peuvent impliquer un traitement par plusieurs DSA. En conséquence, le paragraphe 15 adopte un point de vue essentiellement axé sur les opérations pour présenter les phases de traitement applicables à chaque opération.

Afin de pouvoir gérer le fonctionnement réparti de l'annuaire, chaque DSA doit effectuer les actions nécessaires pour réaliser l'objet de chaque opération et les actions supplémentaires nécessaires pour répartir cette réalisation entre plusieurs DSA. Le paragraphe 15 explore les différences entre ces deux catégories d'actions, qui sont spécifiées en détail aux paragraphes 16 à 22.

14.5 Accords de coopération entre DSA

Tous les DSA qui sont en relation de subordonné/supérieur en raison des contextes de dénomination qu'ils détiennent ont entre eux des rattachements opérationnels hiérarchiques ou hiérarchiques non spécifiques, selon les types de référence de connaissance détenus par les DSA subordonnés.

On peut administrer les rattachements opérationnels hiérarchiques et hiérarchiques non spécifiques entre DSA au moyen des procédures des paragraphes 24 et 25, ou par d'autres moyens (comme le téléphone).

Un DSA détenant des entrées qui relèvent de la zone administrative de son DSA supérieur doit administrer le sous-schéma, doit appliquer la règle de recherche pertinente (le cas échéant) et doit contrôler l'accès aux entrées selon les exigences de l'autorité administrative. La gestion des entrées à l'intérieur d'une zone administrative peut être assurée conformément à la Rec. UIT-T X.501 | ISO/CEI 9594-2, ou par des mécanismes locaux.

15 Comportement de l'annuaire réparti

15.1 Exécution coopérative des opérations

Chaque DSA est doté des procédures capables d'exécuter complètement toutes les opérations d'annuaire. Dans le cas où un DSA contient la DIB complète, toutes les opérations sont en fait complètement effectuées dans ce DSA. Dans le cas où la DIB est répartie entre plusieurs DSA, l'exécution d'une opération est normalement fragmentée, seule une partie de cette opération étant effectuée dans chacun des DSA coopérants qui peuvent être nombreux.

En environnement réparti, le DSA voit normalement chaque opération comme un événement transitoire: l'opération est invoquée par un DUA, un client LDAP ou par un autre DSA; le DSA effectue un traitement sur l'objet, puis le dirige vers un autre DSA pour traitement complémentaire.

Une autre manière de voir ce comportement est de considérer le traitement total subi par une opération au cours de son exécution par plusieurs DSA coopérants. Ce point de vue met en évidence les phases de traitement communes à toutes les opérations.

15.2 Phases de traitement d'une opération

Chaque opération d'annuaire peut être considérée comme comprenant trois phases distinctes:

a) la phase de *résolution du nom*, au cours de laquelle le nom de l'objet sur l'entrée duquel une opération particulière doit être effectuée, est utilisé pour localiser le DSA qui contient cette entrée;

b) la *phase d'évaluation*, au cours de laquelle l'opération spécifiée par une demande particulière adressée à l'annuaire (par exemple Read) est réellement effectuée;

c) la *phase de fusionnement des résultats*, au cours de laquelle les résultats d'une opération spécifiée sont retournés au DUA ou client LDAP demandeur. Si un mode de chaînage d'interactions a été choisi, la phase de fusionnement des résultats peut impliquer plusieurs DSA, dont chacun a chaîné la demande ou sous-demande initiale (définie au § 15.3.1: décomposition de demande) vers un autre DSA durant l'une ou l'autre des phases précédentes ou durant les deux.

Dans le cas des opérations Read, Compare, List, Search, Modifiy Entry, Modify DN et Remove Entry (lecture, comparaison, listage, recherche, modification, entrée, modification de nom distinctif et suppression d'entrée), la résolution du nom porte sur le nom d'objet fourni dans l'argument de l'opération. Dans le cas des opérations d'ajout d'entrée, l'entrée cible de la résolution de nom est l'entrée immédiatement supérieure à celle qui a été fournie dans l'argument de l'opération. Elle s'obtient aisément en supprimant le RDN final du nom fourni dans l'argument de l'opération. (Cela sera effectué au moyen de l'argument local **m** de la procédure **FindDSE** (trouver entrée spécifique de DSA) du § 18.3.1.)

Une opération sur une entrée particulière peut initialement être dirigée vers n'importe quel DSA de l'annuaire. Ce DSA utilise ses connaissances, éventuellement en conjonction avec d'autres DSA, pour traiter l'opération suivant les trois phases.

15.2.1 Phase de résolution du nom

La résolution du nom est le processus de mise en correspondance séquentielle de chaque RDN d'un nom prétendu avec un arc (ou nœud) du DIT, en commençant logiquement à la racine et en progressant vers le bas du DIT. Toutefois, comme le DIT est réparti de façon arbitraire entre de nombreux DSA, chaque DSA peut n'être capable d'exécuter qu'une fraction du processus de résolution du nom. Un DSA donné effectue sa part du processus de résolution du nom en parcourant son arbre des informations local de DSA. Ce processus est décrit à l'article 18 dans les diagrammes associés (Figures 9 à 12). Sur la base de son arbre des informations local de DSA, il peut savoir, d'après les informations de connaissances qui y sont contenues, si la résolution peut être reprise en charge par un ou plusieurs autres DSA ou si le nom est erroné.

Si la commande de service comprend l'option **manageDSAIT**, le fonctionnement de la phase de résolution du nom est restreint à l'arbre d'informations d'un DSA.

15.2.2 Phase d'évaluation

Quand la phase de résolution du nom est terminée, l'opération requise proprement dite (par exemple Read ou Search) a lieu.

Les opérations qui impliquent une interrogation à simple entrée – Read et Compare – peuvent être exécutées entièrement à l'intérieur du DSA dans lequel cette entrée est située.

Les opérations qui impliquent une interrogation à entrées multiples – List et Search – doivent localiser des objets subordonnés de la cible, qui peuvent ou non résider dans le même DSA. Si tel n'est pas le cas, les opérations doivent être dirigées vers les DSA spécifiés dans les références subordonnées, subordonnées non spécifiques, fournisseur ou principales (selon le cas) pour exécuter le processus d'évaluation.

Si la commande de service comprend l'option **manageDSAIT**, le fonctionnement de la phase d'évaluation est restreint à l'arbre d'informations d'un DSA. De même, si la phase d'évaluation commence dans une zone administrative de service, l'évaluation est limitée à cette zone.

15.2.3 Phase de fusionnement des résultats

La phase de fusionnement des résultats commence au moment où quelques résultats de la phase d'évaluation sont disponibles.

Si l'opération ne concernait qu'une seule entrée, le résultat de cette opération peut simplement être retourné au DUA ou client LDAP demandeur. Si l'opération concernait plusieurs entrées sur plusieurs DSA, les résultats peuvent être combinés. Les résultats ne doivent pas être combinés si la protection leur est appliquée. Ils devraient être renvoyés au DUA ou au client LDAP sans qu'il soit procédé au fusionnement.

Les réponses permises, retournées à un demandeur après fusionnement des résultats, comprennent:

a) un résultat complet de l'opération;

b) un résultat qui n'est pas complet car certaines parties du DIT restent inexplorées (ne s'applique qu'au listage (List) et à la recherche (Search)). Un tel *résultat partiel* peut incorporer des références de continuation concernant des parties inexplorées du DIT;

c) une erreur (un renvoi de référence étant un cas spécial);

d) et, si le demandeur était un DSA, un composant **ChainingResults**.

15.3 Gestion des opérations réparties

Des informations sont incluses dans l'argument de chaque opération dont l'exécution peut être demandée à un DSA: elles indiquent la progression de chaque opération au cours de sa traversée de divers DSA de l'annuaire. Cela permet à chaque DSA d'exécuter la partie appropriée du traitement requis et d'enregistrer le fait que cette exécution a été effectuée, avant de diriger l'opération vers d'autres DSA.

Des procédures supplémentaires sont incorporées dans le DSA pour répartir physiquement les opérations et répondre à d'autres besoins résultant de leur répartition.

15.3.1 Décomposition de demande

La décomposition de la demande est un processus exécuté à l'intérieur d'un DSA avant qu'il entre en communication avec un ou plusieurs autres DSA et serveurs LDAP. Une demande est décomposée en plusieurs sous-demandes, de

manière telle que chacune de ces dernières accomplisse une partie de la tâche initiale. La décomposition de la demande peut être utilisée, par exemple dans l'opération recherche, après que l'objet de base a été trouvé. Après la décomposition, chacune des sous-demandes peut être transmise par unichaînage ou par multichaînage à d'autres DSA et/ou serveurs LDAP, en vue de la poursuite de l'opération.

L'**argument** d'une demande (voir § 12.1) ou d'une sous-demande chaînée sera l'argument non modifié de l'opération si l'opération a été lancée par un DUA, et restera le LDAPMessage non modifié si l'opération a été lancée par un client LDAP. Un DSA recevant une demande chaînée ne changera pas d'**argument** en effectuant la décomposition de la demande.

NOTE – Les paragraphes suivants mentionnent cette spécification pour les composants individuels de l'**argument.** Cela ne doit pas être interprété comme voulant dire que les composants non mentionnés explicitement peuvent être modifiés.

15.3.2 Réponse à une demande émanant d'un DSA

Un DSA qui reçoit une demande peut vérifier la progression de traitement de cette demande à l'aide du paramètre **operationProgress**, qui indiquera si l'opération est toujours en phase de résolution du nom ou si elle a atteint la phase d'évaluation et quelle partie de l'opération le DSA doit tenter de réaliser. Si le DSA ne peut pas satisfaire entièrement la demande, il doit soit communiquer l'opération par unichaînage ou multichaînage à un ou à plusieurs DSA et/ou serveurs LDAP qui peuvent aider à répondre à la demande ou retourner un renvoi de référence à un autre DSA ou serveur LDAP ou mettre fin à la demande par un message d'erreur.

15.3.3 Exécution des opérations

Chaque DSA qui a lancé une opération ou qui l'a propagée vers un ou plusieurs autres DSA et/ou serveurs LDAP doit garder trace de l'existence de cette opération jusqu'à ce que chacun des autres DSA et/ou serveurs LDAP ait retourné un résultat ou une erreur, ou que le délai maximal de l'opération ait expiré. Cet impératif concerne toutes les opérations, tous les modes de propagation et toutes les phases de traitement. Il garantit la terminaison en bon ordre des opérations réparties qui se sont propagées dans l'annuaire.

15.4 Traitement des boucles

Le DIT peut se trouver dans une situation entraînant un bouclage. Par exemple, un bouclage peut se produire au cours de la résolution d'un nom si le déréférencement d'un ou de plusieurs alias ramène la résolution à la même branche du DIT. Une erreur de configuration des références de connaissance est une autre cause possible de bouclage.

Dans le contexte d'une opération sur l'annuaire particulière, une boucle se forme si, à un moment quelconque, il y a retour à l'*état* précédent, cet état étant défini par les composants suivants:

- le nom du DSA traitant actuellement l'opération;
- le nom de l'objet **targetObject**, tel que contenu dans l'argument de l'opération;
- le paramètre **operationProgress**, tel que contenu dans l'argument de l'opération et défini au § 10.5.

Cela n'implique pas l'impossibilité qu'un même DSA traite plusieurs fois une même opération, mais le fait qu'un DSA ne traitera pas plusieurs fois la même opération dans le même état.

Les bouclages sont contrôlés au moyen de l'argument **traceInformation** défini au § 10.6 qui enregistre la suite d'états qu'une opération donnée a traversés. Deux stratégies ont été définies pour déterminer si des boucles se sont formées ou vont se produire. Il s'agit de la détection des boucles et de la prévention des boucles; ces stratégies sont décrites respectivement aux § 15.4.1 et 15.4.2.

La détection des boucles est obligatoire et la prévention des boucles est optionnelle.

15.4.1 Détection des boucles

A la réception d'une opération d'annuaire, un DSA doit initialement valider l'opération pour faire en sorte qu'elle puisse se poursuivre. Une phase importante de la validation est la recherche des boucles, ce qui se fait en déterminant si l'état actuel de l'opération apparaît dans la suite d'états antérieurs enregistrés dans l'argument **traceInformation** relatif à cette opération. Cette étape du contrôle des bouclages constitue la détection des boucles.

15.4.2 Prévention des boucles

La prévention des boucles nécessite que le DSA détermine, immédiatement avant le transfert d'une opération à un autre DSA, dans le cadre d'une procédure de chaînage, si l'état consécutif de l'opération (qui est l'item **traceItem** que le DSA récepteur va ajouter à l'argument **traceInformation** quand il le recevra) apparaît dans la séquence des états précédemment enregistrés dans l'argument **traceInformation** pour l'opération entrante initiale.

En cas de réception ou de traitement de renvois de référence, on ne peut pas réaliser la détection et la prévention des boucles par simple examen de l'argument **traceInformation**. Si tel est le cas, chaque fois que le DSA agit sur un renvoi

de référence, il doit enregistrer l'état de l'opération qui en résulte (c'est-à-dire le composant **traceItem** que le DSA récepteur va ajouter quand il recevra la demande) avec un enregistrement de la demande entrante. Avant d'agir sur un renvoi de référence ou de le retourner, le DSA doit vérifier dans cette liste s'il n'y a pas eu de demande identique envoyée au cours d'une précédente tentative de réaliser l'opération entrante.

15.5 Autres considérations relatives au fonctionnement réparti

15.5.1 Commandes de service

Certaines commandes de service nécessitent une prise en compte particulière, pour que l'opération soit traitée de la façon demandée.

a) **chainingProhibited** – Un DSA consulte cette commande de service pour déterminer le mode de propagation d'une opération. Si elle est sélectionnée, le DSA utilise toujours le renvoi de référence. Si elle ne l'est pas, le DSA peut choisir d'utiliser le chaînage ou le renvoi de référence, selon ses possibilités.

b) **timeLimit** – Un DSA doit tenir compte de cette commande de service pour garantir que le délai n'est pas dépassé à son niveau. Un DSA auquel un DUA demande d'exécuter une opération note en premier lieu le délai **timeLimit** donné par le DUA, en secondes, pour l'exécution de cette opération. Si un chaînage est nécessaire, le délai **timeLimit** est inclus dans l'argument de chaînage à communiquer à ou aux DSA suivants. Dans ce cas, la même valeur limite est utilisée pour chaque demande chaînée: c'est l'heure UTC à laquelle l'opération doit être achevée pour satisfaire la contrainte spécifiée initialement. Quand un DSA reçoit des arguments **ChainingArguments** où est spécifiée une limite **timeLimit**, il doit respecter cette limite.

c) **sizeLimit** – Un DSA doit tenir compte de cette commande de service pour garantir que la liste des résultats n'excède pas la taille spécifiée. La limite, telle qu'incluse dans l'argument commun de la demande initiale, est véhiculée telle quelle lorsque la demande est chaînée. Si une décomposition de la demande est requise, la même valeur est incluse dans l'argument à communiquer au DSA suivant: c'est-à-dire que la limite intégrale est utilisée pour chacune des sous-demandes. Quand les résultats sont retournés, le DSA demandeur résout les divers résultats et applique la limite à leur ensemble pour garantir que seul le nombre demandé est retourné. Si la limite a été dépassée, cela est indiqué dans la réponse.

d) **priority** – Dans tous les modes de propagation, il incombe au DSA de s'assurer que le traitement des opérations est effectué dans un ordre permettant de se conformer à cette commande de service, si elle est présente.

e) **localScope** – L'opération est limitée à un cadre défini au niveau local et aucun DSA ne peut propager la demande en dehors de ce cadre.

f) **scopeOfReferral** – Si le DSA retourne un renvoi de référence ou un résultat partiel pour une opération List ou Search, les arguments **ContinuationReference** doivent être incorporés dans le cadre du renvoi demandé.

Toutes les autres commandes de service doivent être respectées, mais leur utilisation ne nécessite pas une prise en compte particulière dans l'environnement réparti.

15.5.2 Extensions

Si un DSA rencontre une opération étendue au cours de la phase de résolution du nom et s'il détermine que cette opération doit être chaînée à un ou à plusieurs DSA, il inclut dans l'opération chaînée, sans la modifier, toutes les extensions présentes.

NOTE – Une autorité administrative peut juger approprié d'envoyer un message **serviceError** avec la cause **unwillingToPerform** si elle ne souhaite pas propager une extension.

Si un DSA rencontre une extension qu'il ne prend pas en charge lors de la phase d'évaluation du traitement, il existe deux possibilités. Si l'extension n'est pas critique, le DSA n'en tient pas compte. Si elle l'est, le DSA envoie un message **serviceError** avec la cause **unavailableCriticalExtension**. Une extension critique d'une opération à objets multiples peut se traduire à la fois par des résultats et par des erreurs de ce type. Un DSA qui fusionne de tels résultats et erreurs mettra à l'écart ces erreurs de service et emploiera le composant **unavailableCriticalExtension** du qualifiant **PartialOutcomeQualifier** décrit dans la Rec. UIT-T X.511 | ISO/CEI 9594-3.

15.5.3 Déréférencement d'alias

Le déréférencement d'alias est le processus qui consiste à créer un nouveau nom d'objet cible, en remplaçant la partie relative au nom distinctif de l'entrée d'alias du nom d'objet cible initial par la valeur de l'attribut **AliasedEntryName** de l'entrée alias. Dans l'opération, le nom du composant **object** n'est pas affecté par le déréférencement d'alias.

15.5.4 Résolution des variantes contextuelles de noms

Au fur et à mesure du traitement des RDN durant la phase de résolution du nom, il est créé un nouvel objet cible en assurant que chaque **AttributeTypeAndDistinguishedValue** du RDN utilise comme valeur le nom distinctif primaire de cet attribut. Le nom de l'objet cible s'élabore de cette manière en progressant vers un nom distinctif primaire. Cette manière de procéder a pour but de parvenir à un traitement cohérent du nom, notamment dans le cas où des DSA antérieurs à la troisième édition participeraient à la résolution du nom. Le nom d'objet de cette opération n'est pas affecté par cette substitution.

15.5.5 Résultats paginés

Lorsqu'un agent DUA inclut la demande **PagedResultsRequest** dans la demande **search** ou **list** (voir § 7.9 de la Rec. UIT-T X.511 | ISO/CEI 9594-3), la pagination peut être exécutée par l'agent DSA directement rattaché au DUA, également appelé DSA *rattaché*, ou peut être exécutée par le DSA qui contient l'objet **baseObject**/**object** de la demande **search** ou **list** (éventuellement après un ou plusieurs déréférencements d'un alias), appelé également l'*exécuteur initial*. Si la pagination est effectuée par l'agent DSA rattaché, qui pourrait être également l'exécuteur initial, la pagination est qualifiée de *résultats paginés d'agent DSA rattaché*. Si la pagination est effectuée par l'exécuteur initial et si celui-ci n'est pas l'agent DSA rattaché, alors la pagination est qualifiée de *résultats paginés DSP*.

Un agent DSA qui prend en charge des resultats paginés DSP:

- prend en charge des résultats paginés d'agent DSA rattaché;
- prend en charge des résultats paginés en tant qu'agent DSA;
- prend en charge des résultats paginés en tant qu'exécuteur initial;
- prend en charge le sous-composant **entryCount** du qualificateur **PartialOutcomeQualifier**.

Lorsqu'un agent DSA rattaché reçoit une demande **search** ou **list** incluant la demande **PagedResultsRequest**, et si le DSA rattaché n'est pas l'exécuteur initial de cette demande, alors le DSA ratttaché peut choisir d'inclure le paramètre **dspPaging** dans les arguments **ChainingArguments**. L'exécuteur initial peut choisir de produire des résultats paginés DSP. Cette option est notifiée à l'agent DSA rattaché par l'inclusion d'une référence d'interrogation **queryReference** dans le qualificateur **PartialOutcomeQualifier**. Il s'agit donc de la référence **queryReference** renvoyée à l'agent DUA à utiliser pour rechercher la page suivante.

Si l'exécuteur initial ne prend pas en charge les résultats paginés DSP ou choisit de ne pas les exécuter, alors l'agent DSA rattaché peut effectuer une pagination normale de DSA rattaché.

Un agent DSA qui est un exécuteur, mais non l'exécuteur initial, ignorera un éventuel composant **dspPaging** dans les arguments **chainingArguments**, et se conformera à la commande de service **sizeLimit** éventuellement présente.

15.6 Authentification des opérations réparties

Les utilisateurs de l'annuaire ainsi que les autorités administratives assurant les services d'annuaire peuvent, à leur convenance, exiger l'authentification des opérations. Pour toutes les opérations d'annuaire, la nature de la procédure d'authentification dépend de la politique de sécurité en vigueur.

Il existe deux ensembles de procédures d'authentification qui satisfont toute une gamme de besoins d'authentification. Un de ces ensembles se compose des procédures assurées par l'opération de rattachement Bind, qui facilite l'authentification opérée entre deux entités d'application de l'annuaire afin d'établir une association. Les procédures de rattachement permettent toute une gamme d'échanges à des fins d'authentification, depuis le simple échange d'identités jusqu'à l'authentification renforcée.

En plus de l'authentification de l'entité homologue d'une association, comme celle qui est assurée par Bind, des procédures supplémentaires sont définies dans l'annuaire pour permettre l'authentification d'opérations individuelles. Deux ensembles distincts de procédures d'authentification sont ainsi définis. Le premier facilite les services d'authentification de l'expéditeur et concerne l'identification, par un DSA, de l'initiateur de la demande d'origine. Le second ensemble facilite les services d'authentification des résultats et concerne l'authentification, par un initiateur, de tous résultats retournés.

Deux procédures sont définies pour l'authentification de l'expéditeur, l'une fondée sur un simple échange d'identités, appelée **identity based authentication** et l'autre sur des techniques de signatures numériques, appelée **signature based authentication**. La première de ces procédures est de nature rudimentaire, puisque l'échange d'identités est fondé sur l'échange de noms spécifiques qui sont transmis en clair.

Pour l'authentification de résultats, une procédure unique (**results authentication**) est définie, fondée sur des techniques de signature numérique; du fait de la nature généralement complexe du collationnement des résultats, aucune procédure simple, fondée sur l'identité, n'est définie.

Il est permis à ces procédures de gérer l'authentification des réponses d'erreur.

Les services décrits ci-dessous doivent être considérés comme s'ajoutant à ceux qui sont assurés par le service Bind; les procédures de rattachement sont supposées avoir été exécutées avec succès avant l'authentification des opérations d'annuaire.

Les procédures que doit suivre un DSA pour authentifier l'expéditeur et les résultats d'authentification sont spécifiées au § 22.

16 L'aiguilleur d'opérations

L'aiguilleur d'opérations (**Operation Dispatcher**) est la principale procédure de commande d'un DSA. Il fait passer chaque opération par les trois phases du traitement d'une demande. A cet effet, il dispose d'une série de procédures permettant de traiter complètement la demande, comme illustré à la Figure 6.

16.1 Principes généraux

16.1.1 Procédures

Chacune des procédures utilisées par l'aiguilleur d'opérations est constituée d'une définition de son interface conceptuelle en termes de paramètres propres, c'est-à-dire d'arguments, de résultats et d'erreurs, ainsi que d'une description des étapes de la procédure proprement dite. Le comportement de ces procédures est décrit au moyen d'organigrammes et de commentaires. Les symboles utilisés dans ces organigrammes ont la sémantique suivante (voir Figure 7).

16.1.2 Emploi des structures de données communes

Toutes les procédures font appel à certaines structures de données qui sont disponibles au cours du traitement d'une opération guidée par l'aiguilleur d'opérations (**Operation Distpacher**). Ces structures de données servent à coordonner la circulation des informations au sein de l'aiguilleur. La plupart de ces structures sont directement associées à l'argument de l'opération et au résultat qu'il y a lieu de créer pour l'opération. On se réfère aux composants de l'argument du résultat au moyen de leurs noms, donnés dans la définition ASN.1 correspondante (par exemple le composant **operationProgress** des arguments de chaînage). Si une de ces structures est composite, on peut se référer à un composant de la structure en question au moyen d'un composant composite (**compound.component**, par exemple **operationProgress.nameResolutionPhase**).

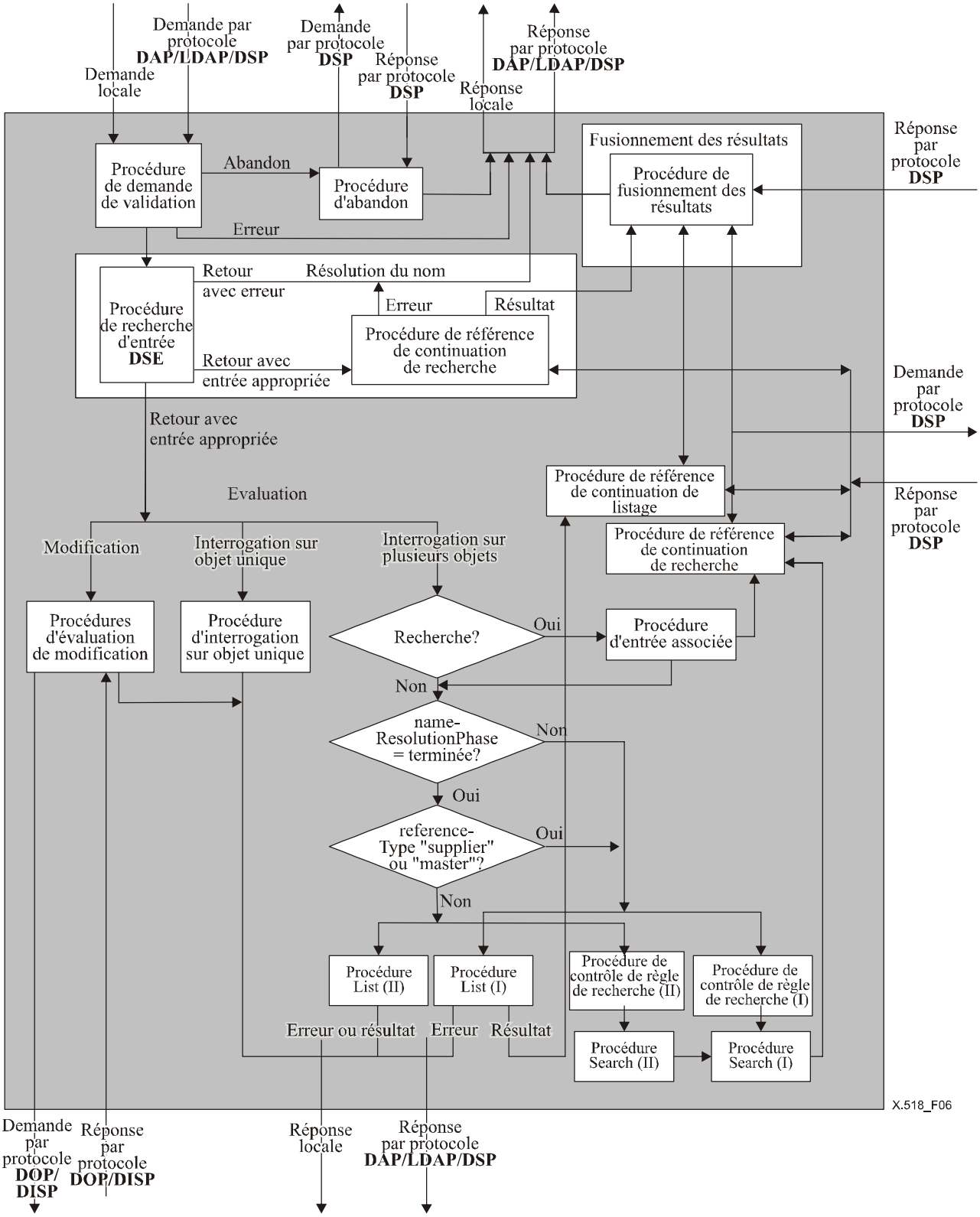


Figure 6 – Aiguilleur d'opérations

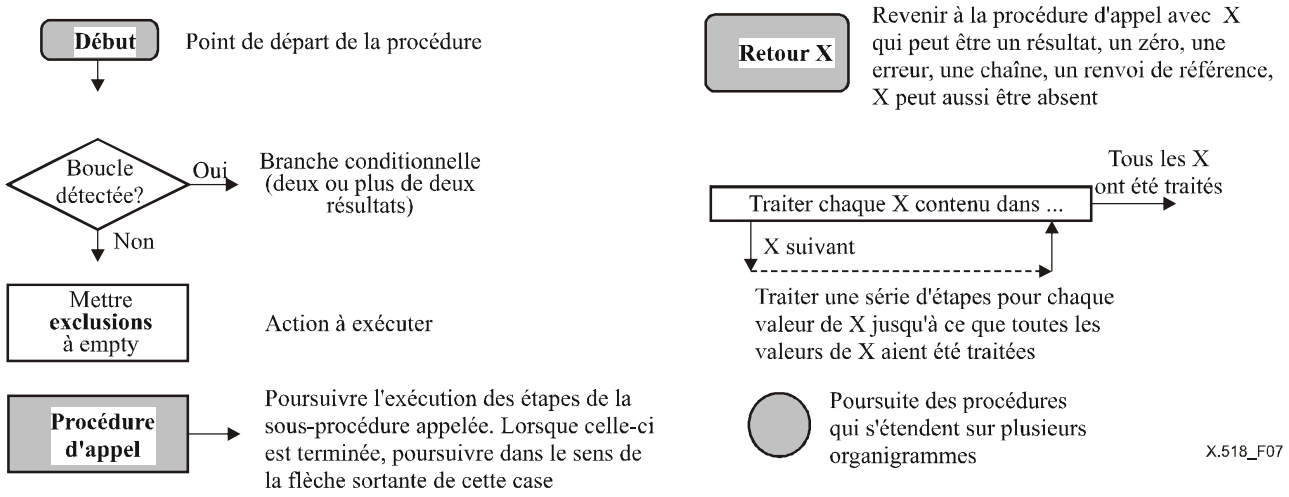


Figure 7 – Symboles utilisés dans les organigrammes

Les structures d'information suivantes sont définies au sein de l'aiguilleur d'opérations:

- **NRcontinuationList** – Liste de références de continuation créée pour être utilisée dans la procédure **Name Resolution Continuation Reference**.
- **SRcontinuationList** – Liste de références de continuation créée pour être utilisée dans la procédure **List** ou **Search Continuation Reference**.
- **admPoints** – Liste de références à des entrées DSE de type "point d'administration" qui est établie au cours de la résolution du nom.
- **referralRequests** – Liste des demandes ou des sous-demandes qui ont été chaînées à la suite de l'exécution de renvois de référence. Chacune de ces demandes ou sous-demandes est résumée sous la forme d'une information de trace **TraceItem**. Cette liste est utilisée par la procédure de prévention des boucles du § 15.4.2.
- **emptyHierarchySelect** – Variable de type booléen qui peut être fixée dans la procédure **Hierarchy Selection**. On suppose qu'elle est remise à zéro lorsque la procédure **Hierarchy Selection** est entamée pour la première fois au cours d'une opération de recherche.
- **streamedResultsOK** – Variable de type booléen qui peut être fixée dans la procédure **Name Resolution** afin d'indiquer que les résultats en flux continu sont susceptibles d'être acceptés pour cette operation. La valeur par défaut de cette variable est False.

Par ailleurs, une procédure peut utiliser une série de variables définies sur place.

16.1.3 Erreurs

A chaque stade du traitement, une erreur peut être détectée au cours de l'exécution d'une sous-procédure. L'erreur identifiée dans cette sous-procédure est normalement retournée au demandeur sous forme d'erreur de protocole correspondante. Dans ce cas, l'aiguilleur d'opérations s'arrête immédiatement. En cas de réception de plusieurs erreurs, des procédures locales pourront choisir celle qui sera retournée.

Une procédure peut aussi choisir de traiter les erreurs (par exemple si le message **serviceError** avec la cause **busy** est retourné à une sous-demande de recherche chaînée) à certains points du traitement de l'opération. Dans ce cas, l'exécution de la procédure se poursuit et aucune erreur n'est retournée au demandeur.

Un DSA peut signer, chiffrer ou signer et chiffrer les indications d'erreur renvoyées dans une opération répartie, selon le DirQOP choisi et la protection demandée pour les indications d'erreur.

16.1.4 Evénements asynchrones

Au cours du traitement d'une demande par l'aiguilleur d'opérations, plusieurs événements asynchrones peuvent survenir. Dans les paragraphes qui suivent, il est spécifié comment traiter un dépassement de limite de temps ou de limite de taille ou de limite administrative, une perte d'association et une demande d'abandon pour une opération qui est en cours de traitement. Le traitement de tous les autres événements asynchrones, tels que les décisions qui relèvent de la politique locale, etc., est hors du domaine d'application de la présente Spécification d'annuaire.

16.1.4.1 Limite de temps

Une limite de temps (**timeLimit**), telle que spécifiée dans le composant **CommonArguments**, peut arriver à expiration à n'importe quel moment de l'opération. Si cela se produit, c'est normalement une erreur **serviceError** mentionnant la cause **timeLimitExceeded** qui est retournée au DUA, client LDAP ou DSA demandeur et l'aiguillage des opérations s'interrompt. Une autre solution consiste, pour la procédure, à choisir de traiter cet événement de manière différente (par exemple au cours du traitement d'une demande de recherche **search**).

Si un DSA reçoit une demande d'un autre DSA quand la limite de temps est dépassée, il envoie une erreur **serviceError** avec la cause **timeLimitExceeded** et ne poursuit pas le traitement de la demande.

Si le DSA a des (sous-)demandes en suspens quand la limite de temps (**timeLimit**) vient à expiration alors qu'aucun résultat n'est trouvé, il doit retourner au demandeur une erreur **serviceError** avec la cause **timeLimitExceeded**.

Si le DSA a des sous-demandes en suspens quand la limite de temps (**timeLimit**) expire et que des résultats ont été obtenus, il doit retourner au demandeur un résultat ayant le contenu suivant:

a) tous les résultats recueillis jusqu'à l'expiration de la limite de temps (**timeLimit**);

b) la composante **limitProblem** du paramètre de résultat **partialOutcomeQualifier**, mise à **timeLimitExceeded**;

c) le composant **unexplored** du paramètre de résultat **partialOutcomeQualifier**, contenant une valeur de référence de continuation pour chaque ensemble de DSA auquel les sous-demandes ont été envoyées et dont le résultat n'est pas inclus dans le résultat retourné au demandeur, en plus des références de continuation concernant les DSA auxquels le DSA en question n'a pas tenté d'envoyer de sous-demandes.

16.1.4.2 Perte d'une association

En cas de perte d'association avec le demandeur, il n'y a plus aucune possibilité de renvoi de résultats. Le DSA peut sur option, pour chaque (sous-)demande d'interrogation en suspens, envoyer une demande **chainedAbandon**, sauf si l'association avec le DSA en question a aussi été perdue. Toutes les réponses à de telles demandes **chainedAbandon** et toutes les (sous-)demandes en suspens seront ignorées. Dans le cas de résultats paginés DSP, le DSA rattaché doit annuler tous les résultats paginés en instance en générant une nouvelle demande de résultat paginé, avec le choix de l'option **abandonQuery** dans la demande **PagedResultsRequest**.

En cas de perte de l'association avec une des sous-demandes chaînées en suspens alors que l'association avec le demandeur n'est pas perdue, le DSA peut (uniquement pour les opérations d'interrogation) essayer toute autre référence à un autre DSA qui a la capacité de traiter la demande chaînée (par exemple une référence à un DSA dupliqué, après perte de l'association avec le DSA maître). Si cela n'aboutit pas, le DSA agira de la manière suivante:

1) si la variable **operationProgress.nameResolution** est mise à **notStarted** ou à **proceeding**, retourner au demandeur soit un message **serviceError** indiquant la cause **unavailable**, ou une erreur de renvoi de référence dont la référence de continuation contiendra la série de DSA qui ont la capacité de poursuivre l'opération. Si l'on utilise des références subordonnées non spécifiques au cours de la phase de résolution du nom et que les associations en question ne soient pas toutes perdues, on pourra (le cas échéant) tenter d'opérer la résolution du nom sans les DSA avec lesquels les associations ont été perdues. Si cela échoue, retourner une erreur **serviceError** avec la cause **unavailable** ou une erreur de renvoi de référence contenant l'ensemble complet de NSSR.

   Si le DSA utilisant des connaissances locales, dont il est éventuellement tenu compte dans la valeur **MasterOrShadowAccessPoint** appropriée, sait que le chaînage est nécessaire au DSA qui a perdu une association, il doit choisir d'envoyer un message **serviceError** avec la cause **unavailable**, et le composant **notification** du type de données **CommonResults** doit contenir:

   – un attribut de notification **dSAProblem** avec la valeur **id-pr-targetDsaUnavailable**;

   – un attribut **distinguishedName** ayant comme valeur le nom distinctif du DSA.

2) si la variable **operationProgress.nameResolution** est mise à **completed** et que la demande soit une opération d'interrogation à entrée unique, retourner au demandeur une erreur **serviceError** ayant la cause **unavailable**;

3) si la variable **operationProgress.nameResolution** est mise à **completed** et que la demande soit une interrogation à plusieurs entrées, le DSA ajoute une référence de continuation au composant **partialOutcomeQualifier.unexplored** du résultat de l'opération, avec une information **AccessPointInformation** identifiant l'ensemble des DSA qui sont capables de poursuivre l'opération, y compris tous ceux avec lesquels des associations ont été perdues.

16.1.4.3 Abandon de l'opération

Une demande d'abandon peut arriver pendant le traitement d'une opération. Dans ce cas, la procédure **Abandon** est invoquée au cours du traitement de la demande d'abandon pour l'opération qu'il y a lieu d'abandonner.

16.1.4.4 Limites administratives

L'administrateur de DSA local ou l'implémentation du DSA elle-même peut imposer des limites telles que la durée maximale de traitement d'une demande ou la longueur maximale des données à retourner, etc. Si une de ces limites est dépassée, le DSA doit retourner un message **serviceError** avec la cause **administrativeLimitExceeded** ou un résultat partiel (provenant de l'ensemble des résultats déjà recueillis) avec l'erreur **limitProblem** mise à la cause **administrativeLimitExceeded.**

Des informations supplémentaires doivent être renvoyées comme suit dans un attribut de notification **dSAProblem**:

a) si la limite est imposée par l'administrateur, l'attribut de notification **dSAProblem** doit prendre la valeur **id-pr-administratorImposedLimit**.

NOTE – Cela n'implique pas qu'il faille une implémentation pour que l'administrateur puisse effectuer les adaptations afin de définir les limites administratives.

b) si la limite due à une restriction de l'implémentation et le problème sont de nature permanente, l'attribut de notification **dSAProblem** doit prendre la valeur **id-pr-permanentRestriction**;

c) si la limite due à une restriction de l'implémentation et le problème sont de nature temporaire, par exemple un encombrement temporaire, l'attribut de notification **dSAProblem** doit prendre la valeur **id-pr-temporaryRestriction**.

16.1.4.5 Limite de taille

Une limite de taille, telle que spécifiée dans le composant **CommonArguments**, peut être dépassée à un moment quelconque du traitement d'une opération List ou Search. Dans ce cas, un résultat partiel (choisi dans l'ensemble des résultats déjà recueillis) doit être retourné au demandeur avec l'erreur **limitProblem** mise à la cause **sizeLimitExceeded**. De plus, le composant **unexplored** peut être utilisé pour renvoyer les références de continuation de DSA qui n'ont pas été atteints.

S'il s'agit d'une opération de recherche et que l'option de contrôle de recherche **entryCount** est définie, le DSA doit effectuer une meilleure estimation du nombre potentiel d'entrées qui auraient été retournées s'il n'y avait pas eu de limite de taille en prenant en compte le contrôle d'accès mais non les sélections hiérarchiques puis il retournera ce chiffre dans le composant **entryCount** du qualifiant **PartialOutcomeQualifier** en utilisant le choix **bestEstimate** s'il n'y a pas de DSA (**unaccessed**) qui n'ont pas été atteints. Dans le cas contraire, il fera le choix **lowEstimate**.

L'aiguilleur d'opérations est alors terminé.

16.2 Procédure de l'aiguilleur d'opérations

La procédure utilisée par l'aiguilleur d'opérations **Operation Dispatcher** pour traiter chaque demande reçue (par DAP, LDAP ou DSP) est définie par les étapes ci-après. En raison du déréférencement d'alias, cette procédure peut également s'appeler automatiquement (par demande locale). Dans ce cas, c'est une réponse locale (plutôt qu'une réponse par DAP, LDAP ou DSP) qui sera retournée.

1) Valider les divers aspects des arguments d'opération (procédure **Request Validation**). Si une erreur est rencontrée au cours de la validation, retourner cette erreur localement ou par DAP/LDAP/DSP.

2) Si l'opération reçue était une opération Abandon, appeler la procédure **Abandon** et retourner la réponse ensuite.

3) Résoudre le nom de l'objet cible en exécutant la procédure **Find DSE** (qui inclut les sous-procédures **Target Found** et **Target Not Found**). Si l'entrée demandée a été trouvée et qu'elle convienne (en vertu du choix des commandes de service, des arguments de chaînage et des décisions locales), poursuivre par la procédure **Evaluation Phase** à l'étape 6). Si, au cours de l'opération **Name Resolution**, on détecte une erreur, la retourner. Si l'entrée trouvée n'est pas jugée appropriée, passer à l'étape 4).

4) Appeler la procédure **Name Resolution Continuation Reference** pour traiter la liste des références de continuation telle qu'elle est enregistrée dans la liste **NRcontinuationList**. Pour traiter ces références de continuation, on peut émettre des demandes chaînées vers d'autres DSA (si les commandes de service ou les décisions locales le permettent).

   En cas d'erreur, celle-ci est directement retournée soit localement soit via le DAP/LDAP/DSP. Si la demande chaînée a produit un résultat, passer à l'étape 5).

5) La procédure **Result Merging** est appelée pour fusionner les résultats locaux avec les résultats chaînés reçus. Si les résultats chaînés contiennent des références de continuation incorporées, elles ne pourront être résolues que si les commandes de service et la politique locale le permettent ou l'exigent.

   Cela peut provoquer l'émission de demandes chaînées additionnelles (dont les résultats chaînés pourront également contenir des références de continuation).

   Les résultats fusionnés sont retournés à l'appelant et le traitement de la demande est arrêté.

   Les résultats ne doivent pas être fusionnés si la protection leur est appliquée.

6) Si l'opération concerne une modification, passer à l'étape 7).

   Si l'opération concerne une interrogation à entrée unique (**single entry interrogation operation**), passer à l'étape 8).

   Si l'opération concerne une interrogation à entrées multiples (**multiple entry interrogation operation**), passer à l'étape 9).

7) Dans l'exécution d'une procédure de type modification, il est parfois nécessaire d'établir des rattachements opérationnels, de les modifier ou d'y mettre fin, ou encore de mettre à jour les duplications miroirs par suite de l'exécution de l'opération. Que cela se fasse de manière synchronisée ou non avec l'exécution de l'opération initiale dépend des opérations de modification respectives (et de la politique locale). Un message de résultat local, de résultat par DAP/LDAP/DSP ou d'erreur sera retourné à l'appelant.

8) Le résultat d'une opération de type **single entry interrogation operation** sera directement retourné à l'appelant sous forme d'un résultat local ou d'un résultat par DAP/LDAP/DSP.

9) Si l'opération est de type **multiple entry interrogation operation**, vérifier l'élément **nameResolutionPhase** de l'opération. S'il n'est pas à la valeur **completed**, appeler la procédure **List(I)** ou **Search(I)**, ou la procédure **List(II)** ou **Search(II)**, selon le cas.

10) Le résultat d'un appel à la procédure **List(II)** (résultat ou erreur) et le résultat d'un appel à la procédure **List(I)** (si ce résultat est une erreur) peuvent être directement retournés à l'appelant (en tant que résultat local ou résultat par DAP/LDAP/DSP).

    Si la procédure appelée était la procédure **List(I)**, le résultat peut contenir des références de continuation qui doivent être déréférencées (selon les commandes de service et la politique locale). Cela peut se traduire par l'envoi d'opérations List chaînées aux DSA respectifs. Pour fusionner les résultats, passer à l'étape 5) avec l'appel à la procédure **Result Merging**.

11) Si l'opération était une opération Search, toute référence de continuation est résolue par la procédure **Search Continuation Reference** (si elle est nécessaire et permise). Cela peut entraîner l'envoi de demandes Search chaînées aux DSA respectifs. La procédure **Result Merging** [étape 5)] est appelée pour fusionner les résultats des recherches et pour déréférencer si possible les références de continuation qui y sont éventuellement contenues.

16.3 Aperçu général des procédures

Le présent paragraphe est un aperçu général de la fonctionnalité de base des procédures employées par l'aiguilleur d'opérations **Operation Dispatcher** et qui sont définies aux § 17 à 22.

16.3.1 Procédure de validation de demande (Request Validation procedure)

Cette procédure, décrite au § 17, est appelée afin d'exécuter les vérifications relatives aux bouclages, aux limites et à la sécurité, avant d'effectuer la résolution locale du nom. Elle établit également les réglages par défaut des paramètres de l'argument **ChainingArgument** qui ne sont pas fournis par le DAP ou le LDAP si la demande émane d'un DUA ou d'un client LDAP. Par ailleurs, cette procédure isole toute demande **abandon** et en avise l'aiguilleur d'opérations **Operation Dispatcher**.

16.3.2 Procédure d'abandon (Abandon procedure)

Cette procédure, décrite au § 20.5, cherche l'opération qu'il y a lieu d'abandonner et tente d'y mettre fin. Au cas où il y aurait une sous-demande en suspens, des opérations **Chained Abandon** peuvent être envoyées à leur suite. La procédure retourne à l'appelant soit un **Null Result**, ou une indication d'erreur (par exemple une erreur **abandonError** avec la cause **tooLate**).

16.3.3 Procédure de recherche de DSE (Find DSE procedure)

Cette procédure, décrite aux § 18.2 et 18.3, adapte les composants du nom de l'objet cible aux DSE détenues localement afin de résoudre le nom de l'objet cible. En cas de rencontre d'une DSE alias, cet alias est déréférencé (si cela est permis) et la procédure est relancée pour résoudre le nouveau nom.

Si la cible n'est pas trouvée, le traitement se poursuit par la sous-procédure **Target Not Found**. Si la cible est trouvée, le traitement se poursuit par la sous-procédure **Target Found**.

NOTE – Les sous-procédures **Target Not Found** et **Target Found** sont des continuations de la procédure **Find DSE**.

La procédure peut aboutir à diverses erreurs; dans ce cas, l'erreur de protocole associée est retournée au demandeur et il est mis fin à l'aiguillage des opérations **Operation Dispatcher**.

16.3.3.1 Sous-procédure de cible non trouvée (Target Not Found sub-procedure)

Cette procédure, décrite au § 18.3.2, effectue une évaluation des DSE intermédiaires locales et crée un ensemble de références **continuationReferences** dans la liste **NRcontinuationList**, sur la base de l'ensemble des références de connaissances qui ont été détectées au cours de la procédure **Find DSE**. Cet ensemble de références est ensuite traité par la procédure de référence **Name Resolution Continuation Reference**.

Cette procédure peut aboutir à diverses erreurs; si c'est le cas, l'erreur associée est retournée au demandeur et il est mis fin à l'aiguillage des opérations **Operation Dispatcher**.

16.3.3.2 Sous-procédure de cible trouvée (Target Found sub-procedure)

Cette procédure, définie au § 18.3.3, vérifie si la DSE qui a été trouvée convient à l'opération demandée, c'est-à-dire s'il s'agit d'une information dupliquée. Cela peut inclure le contrôle de l'utilité de l'ensemble du sous-arbre des informations miroirs au-dessous de l'objet cible dans le cas d'une opération sur plusieurs objets (par exemple une recherche sous-arborescente).

Si l'entrée qui a été localisée convient, on invoque la procédure d'évaluation de l'opération appropriée. Dans le cas contraire, une référence **ContinuationReference** pointant vers le fournisseur (ou maître) de l'information est créée dans la liste **NRcontinuationList** et c'est la procédure de référence **Name Resolution Continuation Reference** qui est alors invoquée.

16.3.4 Procédure d'interrogation à entrée unique (Single entry interrogation procedure)

Cette procédure, décrite au § 19.2, est invoquée pour exécuter finalement les opérations qui n'interviennent que sur une entrée unique, à savoir Read et Compare. Lorsqu'elle est terminée, une réponse (résultat ou erreur) créée par la procédure est retournée au DSA/DUA/client LDAP demandeur.

16.3.5 Procédures de modification (Modification procedures)

Ces procédures, décrites au § 19.1, sont exécutées pour traiter les opérations de modification, à savoir Add Entry, Remove Entry, Modify Entry et Modify DN. Cela se fait par l'exécution d'une sous-procédure spécifique, définie pour chacune de ces opérations. Au cours de ces procédures (ou après), des demandes par DOP et DISP peuvent être émises vers d'autres DSA. Après exécution normale, un résultat (créé par les sous-procédures) est retourné au DSA/DUA/client LDAP demandeur.

16.3.6 Procédures d'interrogation à entrées multiples (Multiple entry interrogation procedures)

Ces procédures, décrites au § 19.3, sont exécutées pour traiter des opérations qui interviennent sur plusieurs entrées pouvant (le cas échéant) être situées dans le même DSA. Cela se fait en exécutant des sous-procédures spécifiques, définies pour chacune des opérations Search et List afin de réaliser la décomposition de la demande. Ces procédures créent un résultat local de l'évaluation de l'opération et établissent (le cas échéant) un ensemble de références de continuation dans la liste **SRcontinuationList**. Si celle-ci est vide à la fin de la procédure, le résultat créé est directement retourné au DSA/DUA/client LDAP demandeur. S'il s'agit d'une opération de recherche, si le résultat est vide et si la variable **emptyHierarchySelect** est fixée, renvoyer dans le composant **notification** du paramètre **PartialOutcomeQualifier**:

- un attribut de notification **searchServiceProblem** dont la valeur est **id-pr-emptyHierarchySelection**.

Si le résultat **SRcontinuationList** n'est pas vide, ces références de continuation sont traitées par invocation des procédures **List** ou **Search Continuation Reference** selon le type de l'opération.

16.3.7 Procédure de référence de continuation de résolution de nom (Name Resolution Continuation Reference procedure)

Cette procédure, décrite au § 20.4.1, traite les références de continuation contenues dans la liste **NRcontinuationList** créée au cours de la phase de résolution du nom. Ces références de continuation sont utilisées pour émettre des sous-demandes chaînées ou sont retournées dans une erreur de renvoi de référence. En cas de chaînage, les résultats ou erreurs émanant de la demande chaînée sont retournés pour traitement ultérieur par la procédure de fusionnement des résultats (**Result Merging**).

16.3.8 Procédure de référence de continuation des opérations de listage et de recherche (List and Search Continuation Reference procedure)

Ces procédures, décrites aux § 20.4.2 et 20.4.3, traitent les références de continuation contenues dans la liste **SRcontinuationList** créée par les procédures d'interrogation à entrées multiples, soit pour les résoudre en émettant des sous-demandes chaînées ou en créant une ou plusieurs références de continuation au sein de la variable **partialOutcomeQualifier.unexplored**. Quand les résultats ou erreurs pour toutes les sous-demandes en suspens ont été obtenus, ils sont retournés pour suite de traitement par la procédure de fusionnement des résultats (**Result Merging**).

16.3.9 Procédure de fusionnement des résultats (Result Merging procedure)

Cette procédure, décrite au § 21, a pour effet d'examiner le résultat d'une demande chaînée ou de combiner les résultats des opérations locales avec les résultats reçus des sous-demandes chaînées. Si une sous-demande a retourné une erreur, cette procédure détermine la manière dont cette erreur doit être traitée.

Si d'éventuelles références de continuation sont laissées dans le résultat, elles seront déréférencées (si cela est permis par la politique locale et si les commandes de service le requièrent) par la procédure **Name Resolution**, **List** ou **Search Continuation Reference**, selon le cas. Les valeurs en double seront retirées du résultat si celui-ci est non signé.

Le résultat fusionné (c'est-à-dire tous les résultats fusionnés et toutes les références de continuation non résolues) est retourné au DUA/DSA/client LDAP demandeur.

Les résultats ne doivent pas être fusionnés si la protection leur est appliquée.

17 Validation de demande

17.1 Introduction

Il s'agit d'une procédure (**Request validation**) qui représente le point d'entrée de l'aiguilleur d'opérations pour les entrées issues des DUA, clients LDAP et DSA et qui prépare de telles entrées en vue du traitement de résolution du nom. La fonction de cette procédure est de détecter les opérations d'abandon, d'effectuer les contrôles de sécurité, d'ajuster les entrées reçues des DUA ou des clients LDAP afin qu'elles puissent être traitées de la même manière que les entrées reçues des DSA, de vérifier les arguments de la demande de validité syntaxique et sémantique, de détecter les boucles et d'effectuer divers autres contrôles. Le déroulement de la validation de la demande (**Request Validation**) est illustré à la Figure 8.

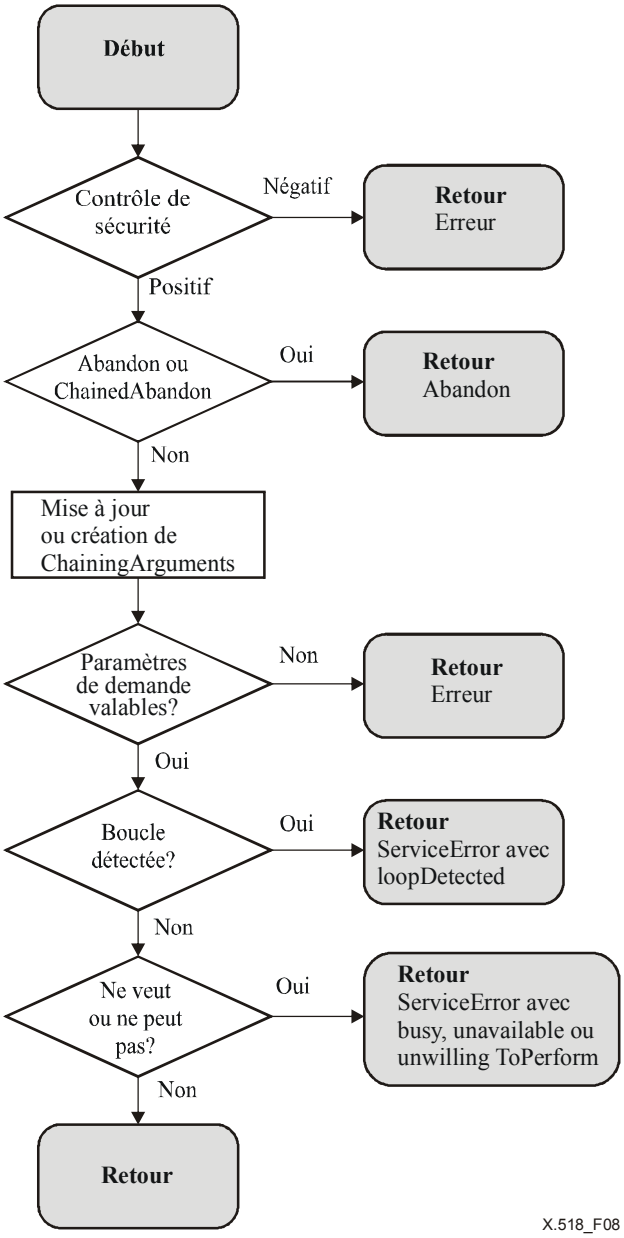


Figure 8 – Procédure de validation de demande

17.2 Paramètres de la procédure

17.2.1 Arguments

Les arguments d'entrée à la validation de demande (**Request Validation**) sont les composants **ChainingArguments** (sauf dans le cas des opérations de type **chainedAbandon**) si la demande émane d'un DSA et si l'argument est émis par l'expéditeur de la demande.

17.2.2 Résultats

Les cinq résultats possibles restitués par la procédure de validation de demande (**Request Validation**) sont les suivants:

a) si le contrôle de sécurité est négatif, une erreur est retournée au demandeur;

b) si l'entrée est une opération **abandon** ou **chainedAbandon**, le résultat est l'argument de l'opération;

c) si les arguments de la demande sont non valides, une erreur est retournée au demandeur. Compte tenu de la politique locale, le DSA peut toutefois retourner un message **serviceError** ou **securityError**;

d) en cas de détection d'une boucle, c'est une erreur de type **serviceError** avec la cause **loopDetected** qui sera retournée au demandeur.

e) si, pour des problèmes de ressources ou des considérations de politique, le DSA ne peut ou ne souhaite pas effectuer l'opération, une erreur de type **serviceError** (avec la cause **busy**, **unavailable** ou **unwillingToPerform**) est retournée au demandeur. Le cas échéant, un message **serviceError** avec la cause **dataSourceUnavailable** peut être retourné;

f) dans tous les autres cas, l'entrée validée, transformée par adjonction des arguments **ChainingArguments** si elle émane d'un DUA ou d'un client LDAP ou par la mise à jour de l'information **ChainingArguments.traceInformation** si elle émane d'un DSA, constitue le résultat de la procédure et devient ensuite l'entrée pour la résolution du nom (**Name Resolution**).

17.3 Définition de la procédure

Le contrôle de sécurité décrit au § 17.3.2 est effectué. Cela peut se traduire par le retour d'une erreur et par l'arrêt de l'aiguillage des opérations.

Si l'entrée est une opération **abandon** ou **chainedAbandon**, seules les étapes du § 17.3.1 sont ensuite effectuées; à défaut, ce sont celles des § 17.3.3 à 17.3.5. Le paragraphe 17.3.5 décrit la procédure de détection des boucles, qui peut aboutir au retour d'une erreur et à l'arrêt de l'aiguillage des opérations.

Ensuite les contrôles du § 17.3.6 sont effectués. Ceux-ci peuvent aboutir au retour d'une erreur et à l'arrêt de l'aiguillage des opérations.

Si les contrôles des § 17.3.2 à 17.3.6 n'aboutissent pas à l'arrêt de l'aiguillage des opérations, ce sont les opérations du § 17.3.7 qui sont effectuées et la procédure se termine par le transfert de son résultat à la procédure de résolution du nom (**Name Resolution**).

17.3.1 Traitement d'un abandon

L'argument d'une opération **abandon** ou **chainedAbandon** est transmis à la procédure **Abandon** (voir § 20.5), en vue du traitement de cette demande.

17.3.2 Contrôles de sécurité

Si l'argument de l'opération est signé, chiffré ou signé et chiffré, il est permis de vérifier le sceau. S'il n'est pas valide ou manque dans un cas où il est attendu, ou si le déchiffrement a échoué, il est permis de renvoyer au demandeur une indication d'erreur. Une autre possibilité consiste pour le DSA à effectuer toute autre action localement définie.

17.3.3 Préparation à l'entrée

17.3.3.1 Demande émanant d'un DUA ou d'un client LDAP

Si l'opération émane d'un DUA ou d'un client LDAP, une valeur **ChainingArguments** est créée de la manière suivante:

a) le composant **ChainingArguments.originator** est défini comme indiqué au § 10.3;

b) le composant **ChainingArguments.operationProgress** est mis à la valeur du composant **CommonArguments.operationProgress**;

c) le composant **ChainingArguments.traceInformation** est mis à une séquence contenant une valeur de **TraceItem** unique. Cette valeur est construite de la manière suivante: le composant **TraceItem.dsa** prend le nom du DSA exécutant la validation de la demande (**Request Validation**), le composant **TraceItem.targetObject** est omis. Le paramètre **TraceItem.operationProgress** est mis à la valeur entrante;

d) si la commande de service de l'opération spécifie une limite de temps (temps disponible, en secondes, pour l'exécution de l'opération), le composant **ChainingArguments.timeLimit** est mis à l'heure UTC à laquelle l'opération devra être terminée pour répondre à la limite de temps spécifiée par l'utilisateur;

e) les composants **ChainingArguments.AuthenticationLevel** et **ChainingArguments.UniqueIdentifier** sont sélectionnés conformément à la politique de sécurité locale;

f) le composant **ChainingArguments.nameResolveOnMaster** est recopié du composant **CommonArguments.nameResolveOnMaster**;

g) les composants **ChainingArguments.exclusions, ChainingArguments.entryOnly** et **ChainingArguments.referenceType** sont recopiés des composants **CommonArguments.exclusions, CommonArguments.entryOnly** et **CommonArguments.referenceType** lorsque ceux-ci sont présents. Si tel n'est pas le cas, ils sont omis;

h) si l'option **manageDSAIT** de **ServiceControls** est activée, alors:
   - le composant **nameResolutionPhase** de **operationProgress** doit prendre la valeur **completed**;
   - le composant **nextRDNToBeResolved** de **operationProgress** doit être omis;
   - **referenceType** doit prendre la valeur **self**;
   - **entryOnly** doit prendre la valeur **FALSE**;
   - **nameResolveOnMaster** doit prendre la valeur **FALSE**;
   - l'option **chainingProhibited** des **ServiceControls** doit être activée;
   - les autres éléments facultatifs de **ChainingArguments** sont omis, les valeurs par défaut étant choisies lorsqu'elles sont définies,

i) si l'option **manageDSAIT** de **ServiceControls** n'est pas activée, alors les autres éléments facultatifs de **ChainingArguments** sont omis, les valeurs par défaut étant choisies lorsqu'elles sont définies;

j) **ChainingArguments.SecurityParameters.ProtectionRequest** est employé pour indiquer le niveau de protection (signé, chiffré ou signé et chiffré) qu'il faut appliquer aux résultats.

17.3.3.2 Demande émanent d'un client LDAP

Si l'opération émane d'un client LDAP, une valeur **ChainingArguments** est créée conformément au § 17.3.3.1, hormis le fait qu'il faut mettre le composant **ChainingArguments.operationProgress** à la valeur **nameResolutionPhase notStarted**, et omettre les valeurs à attribuer aux composants **ChainingArguments.exclusions**, **ChainingArguments.entryOnly**, et **ChainingArguments.referenceType.**

17.3.3.3 Demande émanant d'un DSA

Si la demande émane d'un DSA, le composant **ChainingArguments.traceInformation** est mis à jour par l'adjonction d'une valeur à la fin de la séquence **TraceItem**. Cette valeur est construite de la manière suivante:

a) **TraceItem.dsa** est mis au nom du DSA exécutant la validation de la demande.

b) **TraceItem.targetObject** est mis à la valeur de **ChainingArguments.targetObject** à moins que **object** (ou **baseObject** dans le cas d'une recherche) de l'argument de recherche ne soit identique à **ChainingArguments.targetObject**, en quel cas **TraceItem.targetObject** est omis.

c) **TraceItem.operationProgress** est mis à la valeur de **ChainingArguments.operationProgress**.

Si l'opération émane d'un DSA, et si le composant **ChainingArguments.streamedResults** contient une valeur supérieure ou égale à 1, alors, si et seulement si le DSA admet des résultat en flux continu et s'il est prêt à en recevoir pour cette opération, il incrémente d'une unité la valeur du composant **ChainingArguments.streamedResults.**

17.3.4 Assertion de validité

Cette opération est contrôlée au plan de la validité syntaxique et sémantique de ses arguments conformément aux règles contenues dans les articles définissant chaque opération (il convient par exemple de vérifier que le numéro **nextRDNToBeResolved** ne donne pas un nombre dépassant le nombre de RDN du **targetObject**). S'il apparaît que la demande contient des arguments non valides, il est mis fin à l'opération et une erreur est retournée à l'utilisateur en fonction du type de non-validité constatée.

17.3.5 Détection de boucle

Si deux valeurs quelconques de **TraceItem** de **ChainingArguments.traceInformation** (comme préparé au § 17.3.3) sont identiques, le traitement de l'opération est revenu à un état précédent, c'est-à-dire qu'une boucle a été détectée. Dans ce cas, une erreur **serviceError** (avec la cause **loopDetected**) est retournée au demandeur et il est mis fin à l'aiguillage des opérations (**Operation Dispatcher**).

17.3.6 Incapacité ou refus d'exécution

La procédure de validation de demande (**Request Validation**) peut évaluer les ressources disponibles et déterminer que l'opération ne peut être effectuée. Elle peut également déterminer, sur la base de considérations politiques, qu'il n'y a pas lieu d'effectuer l'opération. Dans ces cas, une erreur **serviceError** (avec la cause **busy**, **unavailable** ou **unwillingToPerform**) peut être retournée au demandeur et on arrête l'aiguillage des opérations.

Si un DSA peut déterminer par des moyens locaux qu'un problème est lié à l'indisponibilité de ressources DIB locales, il enverra un message **ServiceError** avec la cause **unavailable**, et le composant **notification** du type de données **CommonResults** contiendra:

- un attribut de notification **dSAProblem** ayant comme valeur **id-pr-dataSourceUnavailable**;
- un attribut **distinguishedName** ayant comme valeur le nom distinctif du DSA.

17.3.7 Traitement à la sortie

Dans la phase finale de la procédure de validation de demande (**Request Validation**), l'entrée validée, transformée par adjonction de **ChainingArguments** si elle émane d'un DUA ou d'un client LDAP ou par la mise à jour de **ChainingArguments.traceInformation** si elle émane d'un DSA, est renvoyée et utilisée comme entrée pour la procédure de résolution du nom (**Name Resolution**).

18 Résolution du nom

18.1 Introduction

Le présent paragraphe contient la description de la procédure de résolution du nom (**Name Resolution**), de ses arguments, de ses résultats et de ses éventuelles conditions d'erreur. Ainsi qu'il est montré à la Figure 6 (Aiguilleur d'opérations), la procédure de résolution du nom (**Name Resolution**) comprend deux parties:

- la procédure **Find DSE**;
- la procédure **Name Resolution Continuation Reference**.

La procédure **Find DSE** est décrite au moyen de trois organigrammes, à savoir **Find DSE**, **Target Found** et **Target Not Found**. La procédure **Find DSE** établit la correspondance entre le nom d'entrée cible et les DSE enregistrées sur place, composant par composant. Si l'entrée cible est trouvée sur place, la procédure **Find DSE** est suivie de la sous-procédure **Target Found**, qui à son tour appelle la procédure **Check Suitability** afin d'évaluer la DSE venant d'être trouvée. Si l'entrée cible n'est pas trouvée sur place, la procédure **Find DSE** est suivie de la sous-procédure **Target Not Found** qui prépare une ou plusieurs références de continuation à ajouter à la liste **NRcontinuationList** en vue de l'aiguillage par la procédure **Name Resolution Continuation Reference**.

NOTE 1 – Lors de la recherche de concordance, la procédure de résolution du nom (**Name Resolution**) doit effectuer la recherche par rapport aux différentes valeurs distinctives différenciées par leur contexte, décrites au § 9.4 de la Rec. UIT-T X.501 | ISO/CEI 9594-2.

NOTE 2 – La procédure de résolution du nom (**Name Resolution**) risque d'échouer si un DSA supérieur, conforme à une édition antérieure à la troisième édition, contient une référence subordonnée à une entrée contenue dans un DSA d'édition ultérieure dont le RDN inclut des contextes. La résolution du nom (**Name Resolution**) échouera si elle est exécutée par référence à une copie miroir d'une entrée lorsqu'une variante de nom est utilisée comme nom prétendu et si l'entrée miroir est contenue dans un DSA de première ou de seconde édition.

18.2 Paramètres de la procédure Find DSE

18.2.1 Arguments

La procédure utilise les arguments suivants:

a) **ChainingArguments.traceInformation**;
b) **ChainingArguments.aliasDereferenced**;
c) **ChainingArguments.aliasedRDNs**;
d) **ChainingArguments.excludeShadows**;
e) **ChainingArguments.nameResolveOnMaster**;
f) **ChainingArguments.operationProgress (nameResolutionPhase, nextRDNToBeResolved)**;
g) **ChainingArguments.referenceType**;
h) **ChainingArguments.targetObject**;
i) **ChainingArguments.relatedEntry**;
j) **ChainingArguments.streamedResults**;
k) le type d'opération;
l) l'argument de l'opération.

NOTE – Si aucune valeur précise n'existe, on utilisera les valeurs par défaut ou implicites, comme spécifié au § 10.3.

18.2.2 Résultats

Deux cas de réussite de la procédure **Find DSE** sont possibles (désignés respectivement **entry suitable** et **entry unsuitable**).

Le premier retourne (en provenance de la sous-procédure **Target Not Found**) une ou plusieurs références de continuation contenues dans **NRcontinuationList** qui est ou sont ensuite transmises à la procédure **Name Resolution Continuation Reference** afin de poursuivre la phase de résolution du nom.

Le second retourne (en provenance de la sous-procédure **Target Found**) une (référence à une) DSE qui est transmise à l'une des procédures d'évaluation.

18.2.3 Erreurs

Les erreurs suivantes peuvent être retournées:

a) **serviceError: unableToProceed, invalidReference, unavailableCriticalExtension, requestedServiceNotAvailable**;

b) **nameError: noSuchObject, aliasDereferencingProblem, contextProblem**.

18.2.4 Variables globales

La procédure fait appel aux variables globales suivantes:

– **NRcontinuationList**: liste d'enregistrement de la ou des références de continuation nécessaires à la poursuite de la résolution du nom dans la procédure **Name Resolution Continuation Reference**.

– **StreamedResultsOK** afin d'enregistrer la possibilité pour ce DSA de chaîner des résultats en flux continu en réponse à cette opération.

18.2.5 Variables locales et partagées

La procédure utilise les variables locales suivantes:

a) **i** indice utilisé pour identifier le composant du nom cible sur lequel s'effectue le traitement;

b) **m** longueur du nom d'objet cible à utiliser pour la résolution du nom. Pour les opérations qui résolvent le nom par rapport à l'entrée parente (c'est-à-dire Add Entry), la valeur **m** est mise au (nombre de RDN contenus dans l'objet cible) – 1. Pour toutes les autres opérations, la valeur **m** est mise au nombre de RDN contenus dans l'objet cible;

c) **lastEntryFound** indice tel que la DSE (lastEntryFound) soit la dernière entrée DSE de type **entry**;

d) **lastCP** indice tel que la DSE (lastCP) soit le dernier préfixe de contexte dupliqué en miroir qui a été rencontré;

e) **candidateRefs** ensemble de références de continuation.

La variable partagée **admPoints** (définie dans l'aiguilleur d'opérations) est également utilisée. Pour des raisons de commodité, la composante **i** du nom de l'objet cible est représentée par **N(i)**.

18.3 Procédures

NOTE – Certains des textes de l'organigramme ne relèvent que d'opérations particulières. L'organigramme ne le montre pas, mais le texte d'accompagnement le décrit.

18.3.1 Procédure Find DSE

Voir Figure 9.

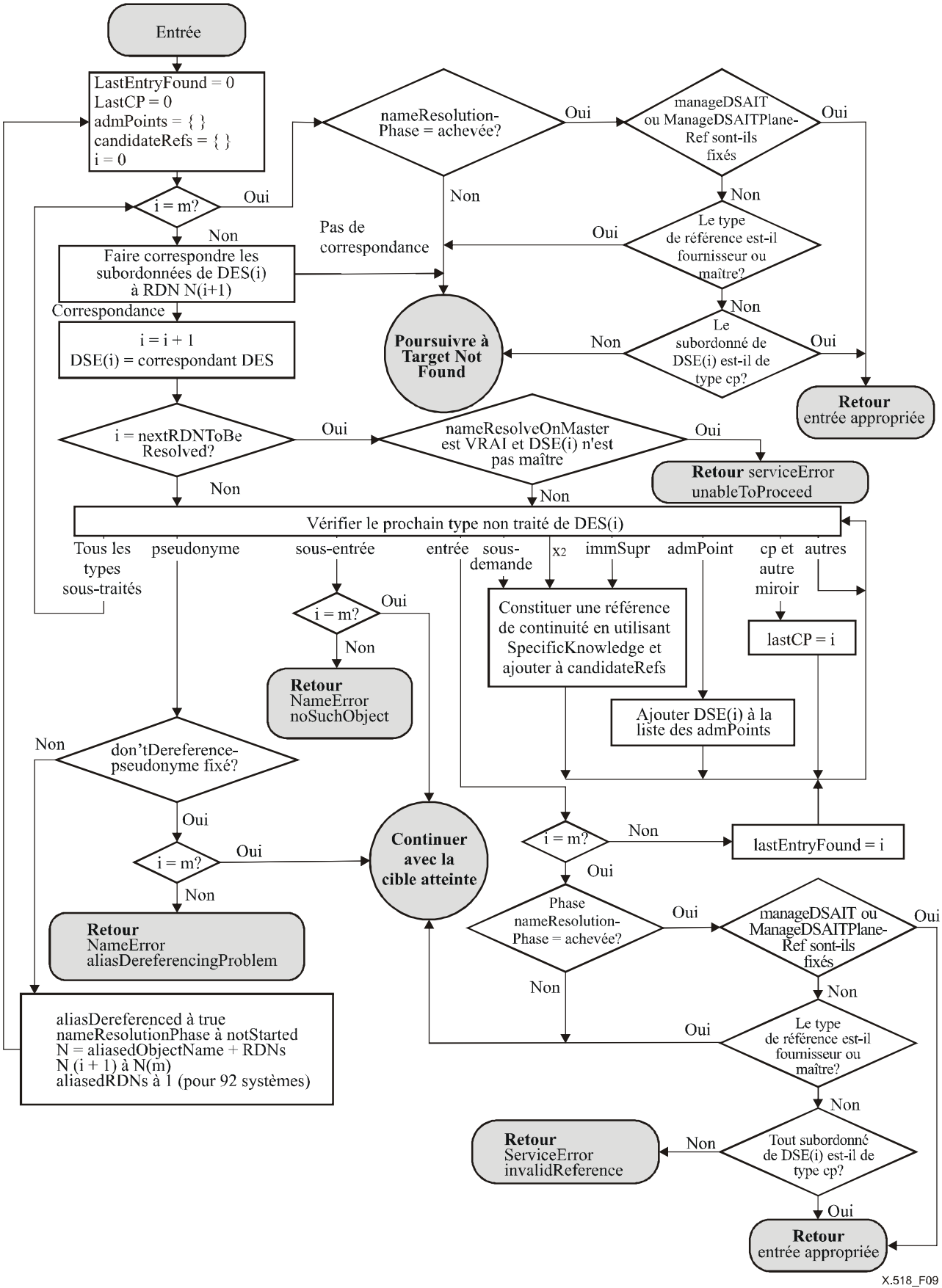


Figure 9 – Procédure Find DSE

Le nom d'objet cible est déterminé de la manière suivante:

a) si **targetObject** est présent dans **ChainingArguments**, la valeur de ce composant est utilisée;

b) si **relatedEntry**, mais pas **targetObject**, est présent dans **ChainingArguments**, on utilise le composant **baseObject** de **JoinArgument** identifié par **relatedEntry**.

NOTE 1 – Ceci ne s'applique qu'à une demande **search** protégée.

c) si ni **relatedEntry** ni **targetObject** est présent dans **ChainingArguments**, on utilise le composant **base** (**baseObject**) de l'argument de l'opération.

Cette procédure cherche à résoudre le nom d'objet localement.

1) Initialiser les variables locales **lastEntryFound** et **lastCP** à **0**; **admPoints** et **candidateRefs** à un ensemble vide et **i** à **0**.

2) Comparer **i** et **m**. S'ils ne sont pas égaux, passer à l'étape 5).

3) S'ils sont égaux, vérifier si la phase **nameResolutionPhase** est à **completed**. Si elle n'est pas à la valeur **completed**, passer à la sous-procédure **Target Not Found**.

Si la phase **nameResolutionPhase** est déjà à la valeur **completed** et si l'extension critique **manageDSAIT** est active, alors retourner avec **entry suitable**.

4) Si la phase **nameResolutionPhase** est à la valeur **completed**, vérifier si une quelconque subordonnée immédiate de la DSE(i) est un préfixe de contexte (de type **cp**).

– Si une ou plusieurs subordonnées immédiates le sont, retourner avec **entry suitable**.

NOTE 2 – Ce cas s'applique à des sous-demandes **List(II)** et **Search(II)**.

– Si aucune ne l'est, poursuivre avec la sous-procédure **Target Not Found**.

5) Tenter de trouver une concordance entre le **(i + 1)**e composant du nom de l'objet cible et le nom d'une subordonnée de la dernière DSE avec laquelle il y avait concordance. Dans le cas de i = 0, tenter d'établir une correspondance entre l'une des DSE immédiatement subordonnées et la DSE racine. Si l'on ne peut trouver de concordance, passer à la sous-procédure **Target Not Found**. Si l'on trouve une concordance unique, incrémenter **i** et enregistrer la DSE, pour laquelle il y a concordance, comme étant le **i**e élément dans le vecteur des DSE trouvées.

NOTE 3 – La recherche de concordance inclut le traitement de plusieurs valeurs distinctives différenciées par le contexte, s'il en est connu, selon la description donnée au § 9.4 de la Rec. UIT-T X. 501 | ISO/CEI 9594-2.

Si l'on trouve plus d'une concordance, alors retourner avec **nameError** et la cause **contextProblem**.

NOTE 4 – Ce cas pourrait se produire par exemple quand un **AttributeTypeAndDistinguishedValue** d'un nom prétendu contient différentes valeurs d'attribut distinctives différenciées par le contexte et que plusieurs d'entre elles concordent avec des valeurs appartenant à plusieurs noms cibles.

6) Si **i** égale **nextRDNToBeResolved**, vérifier si les deux conditions suivantes sont remplies:

– la variable **ChainingArgument.nameResolveOnMaster** est à **TRUE**;

– la DSE(**i**) n'est pas une entrée principale.

Si les deux conditions sont remplies, retourner une erreur de service avec **unableToProceed**.

NOTE 5 – Cela indique l'utilisation de **nameResolveOnMaster** pour éviter les trajets multiples vers le même objet cible.

7) Vérifier tous les bits de type DSE de DSE(**i**). Pour chaque bit correspondant à un type, un certain traitement est sans doute requis. L'action à entreprendre pour chaque type trouvé est donnée ci-dessous:

– si les bits **cp** et **shadow** sont tous deux sélectionnés, mémoriser l'indice **i** dans le préfixe **lastCP**;

– si le bit **admPoint** est sélectionné, vérifier l'attribut opérationnel **administrativeRole**. Si c'est le début d'une zone administrative autonome, vider la liste des points **admPoints**. Si c'est le début d'une ou de plusieurs zones administratives spécifiques, vérifier la liste **admPoints** et éliminer tous les points existants qui ne sont plus applicables (à savoir que leur rôle a été remplacé par celui des nouveaux points administratifs). Enregistrer dans la liste les DSE(**i**);

– si l'un des bits **subr**, **xr, immSupr** ou **ditBridge** est sélectionné, produire une référence de continuation utilisant l'attribut **specificKnowledge** avec **operationProgress.nameResolutionPhase** mis à **proceeding**, **nextRDNToBeResolved** mis à **i**, **targetObject** construit par concaténation des composants résolus en utilisant les RDN primaires (les RDN peuvent contenir des variantes de noms distinctifs), avec les composants restants non encore résolus et **accessPoints** et **referenceType** mis comme il convient. Ajouter la référence de continuation à la liste des références de continuation dans **candidateRefs**;

– si le bit **entry** est sélectionné, essayer **i** égal à **m** (et pour cette raison le nom de l'objet cible aura une concordance complète). Si **i** n'est pas égal à **m**, mémoriser l'entrée trouvée en mettant **lastEntryFound** à **i** et continuer le traitement des bits de type de la DSE(**i**). Si **i** et **m** sont égaux, passer à l'étape 8);

– si le bit **subentry** est sélectionné, faire l'essai de **i** égal à **m** (dans ce cas le nom de l'objet cible aura une concordance complète). S'ils sont égaux, poursuivre avec la procédure **Target Found**; s'ils ne le sont pas, retourner une erreur **nameError** avec la cause **noSuchObject**;

– si le bit **alias** est sélectionné, vérifier si le composant **dontDereferenceAliases** est sélectionné;

s'il ne l'est pas, l'alias peut être déréférencé. Pour cela, mettre **chainingArguments.aliasDereferenced** à **TRUE**, **nameResolutionPhase** à **notStarted**, le nom de l'objet cible à **aliasedEntryName** tel que fourni dans l'entrée alias concaténée avec les composants non concordants qui restent du nom de l'objet cible précédent (c'est-à-dire concaténer avec les $(\mathbf{i} + \mathbf{1})^{e}$ au $\mathbf{m}^{e}$ composants du nom de l'objet cible précédent). Les agents DSA deuxième édition et suivantes ne mettront pas à 1 les noms **aliasedRDN** (alors que les agents DSA première édition mettront les noms **aliasedRDN** au nombre de noms RDN indiqué dans le composant **aliasedEntryName**). Relancer la résolution du nom **Name Resolution** en passant à l'étape 1);

si **dontDereferenceAliases** est sélectionné, l'alias ne peut être déréférencé. Vérifier si le nom de l'objet cible a été complètement traité en comparant **i** et **m** en ce qui concerne leur égalité. S'ils sont égaux (le nom aura alors une concordance complète), poursuivre au moyen de la sous-procédure **Target Found**. S'ils ne sont pas égaux (et que le nom n'ait pas de concordance complète), retourner **nameError** avec la cause **aliasDereferencingProblem**;

– pour tous les autres types de DSE possibles, aucune action n'est requise. Marquer de façon interne que ce type de DSE est traité et poursuivre le traitement des bits de type de DSE(**i**) qui ne sont toujours pas traités;

– si tous les bits de type de DSE(**i**) sont traités, passer à l'étape 2).

8) Vérifier si la phase **nameResolutionPhase** est à la valeur **completed**. Si elle ne l'est pas, passer à la sous-procédure **Target Found**.

9) Si la phase **nameResolutionPhase** est déjà à la valeur **completed** et si l'extension critique **manageDSAIT** est active, alors retourner avec **entry suitable**.

10) Si tel n'est pas le cas, vérifier si l'une quelconque des DSE subordonnées immédiates à la DSE(**i**) est un préfixe de contexte (et, pour cette raison, est de type **cp**). Dans l'affirmative (une ou plusieurs subordonnées immédiates), retourner **entry suitable**. Si aucune des entrées subordonnées immédiates n'est du type préfixe de contexte, retourner une erreur de type **serviceError** avec la cause **invalidReference**.

NOTE 6 – Ce cas s'applique à des sous-demandes **List (II)** et **Search (II)**.

18.3.2 Sous-procédure Target Not Found

Voir Figure 10.

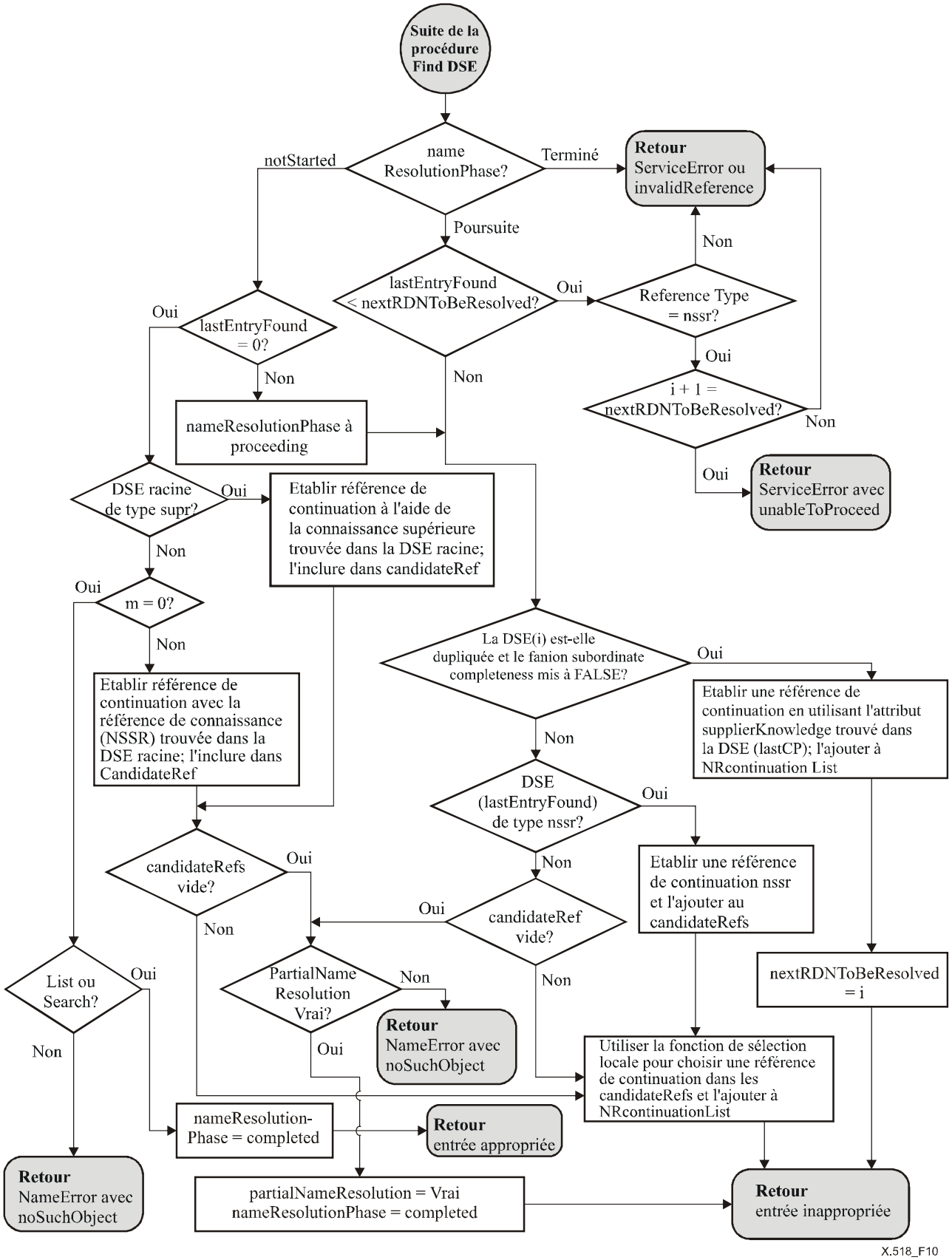


Figure 10 – Sous-procédure Target not Found

Cette sous-procédure est appelée quand le nom de l'objet cible n'est pas trouvé dans le DSA local. La sous-procédure détermine le meilleur type de référence de connaissance qu'il convient d'utiliser pour continuer la résolution du nom à moins qu'une erreur ne soit détectée; dans ce cas une erreur est retournée.

1) En poursuivant depuis la procédure **Find DSE**, il convient de faire la distinction entre les trois possibilités de la phase de résolution du nom.
   - Si **nameResolutionPhase** est **notStarted**, passer à l'étape 2).
   - Si **nameResolutionPhase** est **proceeding**, passer à l'étape 8).
   - Si **nameResolutionPhase** est **completed**, passer à l'étape 12).
2) Si une entrée a été trouvée (**lastEntryFound** de valeur différente de **0**), mettre la phase **nameResolutionPhase** à **proceeding** et passer à l'étape 9).
3) Si ce n'est pas le cas (**lastEntryFound=0**), vérifier si le DSA est du premier niveau.

   Si c'est le cas, la DSE racine ne contient pas de référence supérieure et dès lors elle n'est pas du type **supr**. Dans ce cas, passer à l'étape 4).

   Si le DSA n'est pas du premier niveau, la DSE racine contient une référence supérieure et est donc du type **supr**. Dans ce cas, émettre une référence de continuation en utilisant la connaissance supérieure telle qu'elle est trouvée dans la DSE racine. Mettre:
   - **targetObject** au nom de l'objet cible construit par concaténation des composants résolus en utilisant les RDN primaires (les RDN peuvent contenir des variantes de noms distinctifs), avec les composants restants non encore résolus;
   - **operationProgress.nameResolutionPhase** à **notStarted**;
   - **referenceType** à **superior**; et sélectionner
   - **accessPoints** selon les besoins.

   Ajouter la référence de continuation à la liste des références de continuation dans **candidateRefs**. Passer à l'étape 6).
4) Vérifier si l'opération est dirigée vers l'entrée racine (**m = 0?**). Si c'est le cas, passer à l'étape 5). Si ce n'est pas le cas, émettre une référence de continuation en utilisant toute connaissance de NSSR trouvée dans la DSE racine. Mettre:
   - **targetObject** au nom de l'objet cible construit par concaténation des composants résolus en utilisant les RDN primaires (les RDN peuvent contenir des variantes de noms distinctifs), avec les composants restants non encore résolus;
   - **operationProgress.nameResolutionPhase** à **proceeding**;
   - **operationProgress.nextRDNToBeResolved** à **1**;
   - **referenceType** à **nonSpecificSubordinate**; et sélectionner
   - **accessPoints** selon les besoins.

   Ajouter la référence de continuation à la liste des références de continuation de **candidateRefs**. Passer à l'étape 6).
5) A une DSA de premier niveau, seule l'opération List ou Search peut être effectuée avec l'entrée racine comme objet de base. Pour cette raison et si l'opération n'était pas une List ou Search, retourner une **nameError** avec la cause **noSuchObject**. Si c'était une opération List ou Search, mettre **nameResolutionPhase** à **completed** et retourner le message **entry suitable**.
6) Vérifier s'il y a une quelconque référence de continuation dans **candidateRefs**. Si **candidateRefs** est vide, et si **partialNameResolution** a la valeur **FALSE**, retourner avec **nameError** et le problème **noSuchObject.** Si **candidateRefs** est vide et si **partialNameResolution** a la valeur **TRUE**, alors mettre dans le résultat **partialName** à **TRUE**, **nameResolutionPhase** à **completed** et retourner avec **entry suitable**. Sinon, passer à l'étape 7).
7) Utiliser une fonction de sélection locale pour choisir une référence de continuation dans la liste des références de continuation contenue dans **candidateRefs**, l'ajouter à la liste des références de continuation contenue dans **NRcontinuationList** et retourner le message **entry unsuitable**.
8) Si le DSA ne peut poursuivre la résolution du nom (auquel cas la valeur de **lastEntryFound** sera inférieure à celle de **nextRDNToBeResolved**), passer à l'étape 11). Sinon passer à l'étape suivante.

9) Si la DSE(**i**) est une DSE dupliquée avec une connaissance subordonnée incomplète (**subordinateCompletenessFlag** à **FALSE**), émettre une référence de continuation à partir de l'attribut **supplierKnowledge** trouvé dans la **DSE(lastCP)**. Mettre:

   – **targetObject** au nom de l'objet cible construit par concaténation des composants résolus, en utilisant les RDN primaires (les RDN peuvent contenir des variantes de noms distinctifs), avec les composants restants non encore résolus;

   – **operationProgress.nameResolutionPhase** à **proceeding**;

   – **operationProgress.nextRDNToBeResolved** à **lastEntryFound**;

   – **referenceType** à **supplier**; et sélectionner

   – **accessPoints** selon les besoins.

   Ajouter la référence de continuation à la liste des références de continuation contenue dans **NRcontinuationList** et retourner le message **entry unsuitable**.

10) Si la dernière entrée trouvée contient une NSSR (**DSE(lastEntryFound)** est de type **nssr**), émettre une référence de continuation à partir de la connaissance NSSR trouvée dans la **DSE(lastEntryFound)**. Mettre:

   – **targetObject** au nom de l'objet cible construit par concaténation des composants résolus en utilisant les RDN primaires (les RDN peuvent contenir des variantes de noms distinctifs), avec les composants restants non encore résolus;

   – **operationProgress.nameResolutionPhase** à **proceeding**;

   – **operationProgress.nextRDNToBeResolved** à **lastEntryFound+1**;

   – **referenceType** à **nonSpecificSubordinate**; et sélectionner

   – **accessPoints** selon les besoins.

   Ajouter la référence de continuation à la liste des références de continuation de **candidateRefs**. Passer à l'étape 7).

   Si la **DSE(lastEntryFound)** n'est pas de type **nssr**, passer à l'étape 6).

11) Si **chainingArguments.referenceType** est de type **nssr**, passer à l'étape 13), sinon à l'étape 12).

12) Retourner une erreur **serviceError** avec la cause **invalidReference**.

13) Si **i + 1** est égal à **nextRDNToBeResolved**, la demande a été acheminée à cet endroit en raison d'une NSSR et le DSA n'a pas la capacité de poursuivre la résolution du nom; dans ce cas, retourner une erreur **serviceError** avec la cause **unableToProceed**; sinon, passer à l'étape 12).

18.3.3 Sous-procédure Target Found

Cette sous-procédure est introduite quand le nom de l'objet cible correspond localement à une DSE; elle vérifie si l'entrée trouvée convient pour le traitement local de la demande et sa représentation est donnée à la Figure 11.

1) Appeler la procédure Check Suitability.

2) Si l'entrée est conforme (**entry suitable**), procéder alors comme suit:

   – mettre **nameResolutionPhase** à **completed**;

   – comparer la valeur (présente éventuellement) dans **ChainingArguments.streamedResults** au nombre d'éléments contenus dans **ChainingArguments.traceInformation**; si elle est égale à ce nombre, mettre **StreamedResultsOK** à la valeur TRUE;

   – retourner le message **entry suitable**.

3) Si l'entrée n'est pas conforme (**entry unsuitable**), émettre une référence de continuation en utilisant l'attribut **supplierKnowledge** trouvé dans la **DSE(lastCP)**. Mettre:

   – **targetObject** au nom de l'objet cible construit par concaténation des composants résolus, en utilisant les RDN primaires (les RDN peuvent contenir des variantes de noms distinctifs), avec les composants restants non encore résolus;

   – **operationProgress.nameResolutionPhase** à **proceeding**;

   – **operationProgress.nextRDNToBeResolved** à **m**;

   – **referenceType** à **supplier**; et sélectionner

   – **accessPoints** selon les besoins.

   Ajouter la référence de continuation à la liste des références de continuation de **NRcontinuationList**. Retourner le message **entry unsuitable**.

NOTE – Si la commande de service **localScope** est sélectionnée, le DSA peut, compte tenu des pratiques locales, décider de considérer cette entrée comme étant conforme et de procéder comme indiqué à l'étape 2).

4) Si une extension critique n'est pas supportée (**unsupported critical extension**), retourner une erreur de type **serviceError** avec la cause **unavailableCriticalExtension**.

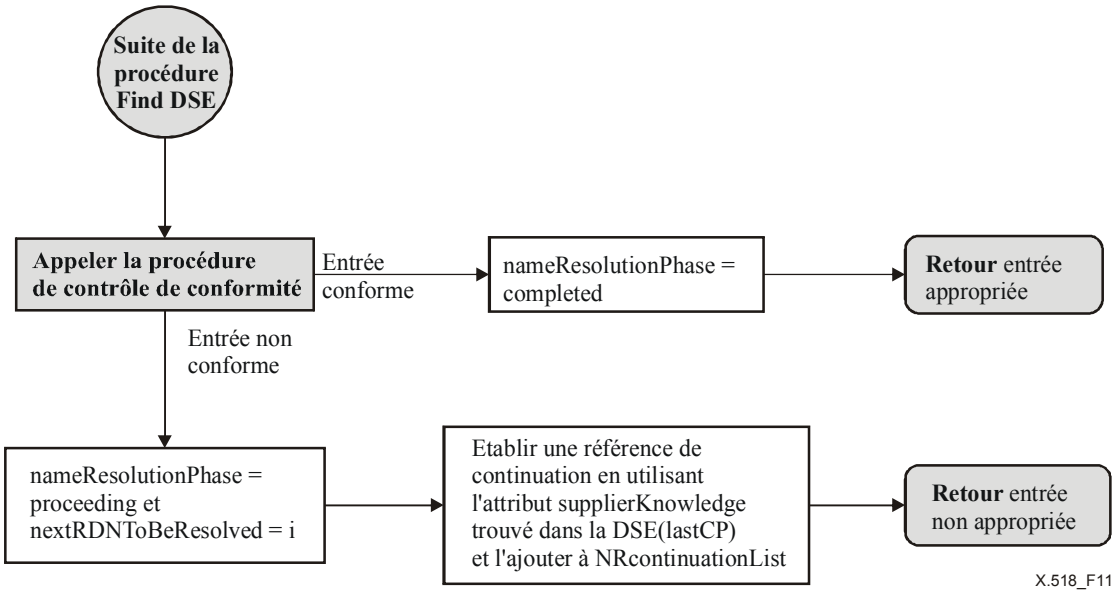


Figure 11 – Sous-procédure Target Found

18.3.4 Procédure Check suitability

Cette procédure est appelée pour décider si une DSE qui a été trouvée convient pour l'exécution de l'opération demandée (voir Figure 12). Elle tient compte des composants **ChainingArguments**, **ServiceControls**, des arguments fournis par l'utilisateur, du type d'opération et des caractéristiques de la DSE (duplication miroir, connaissance subordonnée, présence d'attributs, etc.).

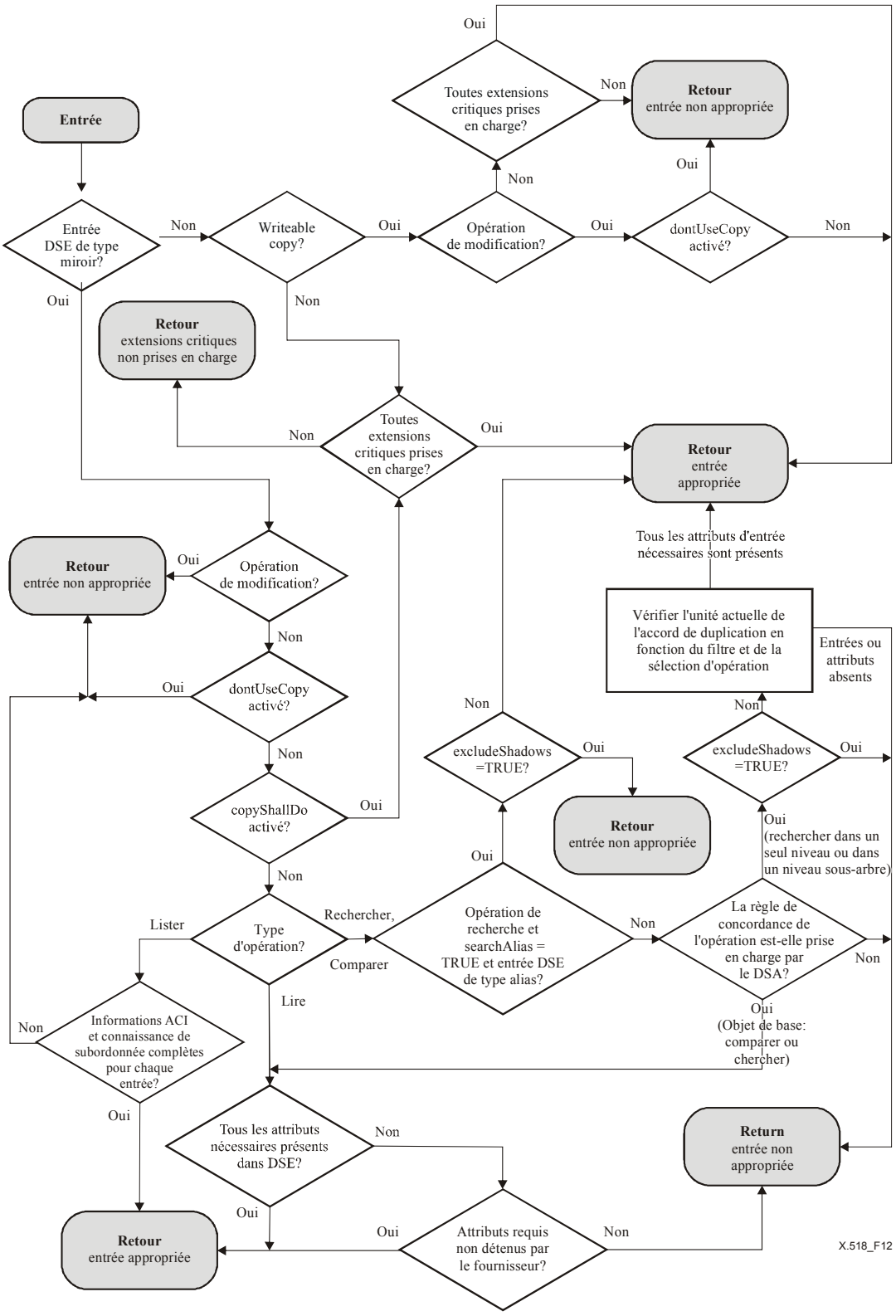


Figure 12 – Procédure de vérification de concordance

18.3.4.1 Paramètres de la procédure

Les arguments d'entrée de cette procédure sont les suivants:

- une référence à une DSE;
- le type d'opération pour lequel la conformité de la DSE doit être vérifiée;
- le composant **ChainingArguments**;
- l'argument de l'opération.

Le résultat est soit: **entry suitable**, **entry unsuitable** ou **unsupported critical extension**.

1) Si la DSE n'est pas du type **shadow** ou **writeableCopy**, vérifier si toutes les **criticalExtensions** (extensions critiques) sont supportées. Si c'est le cas, retourner entry suitable. Sinon, retourner unsupported critical extension.
2) Si la DSE est du type **shadow**, retourner **entry unsuitable** si l'une des conditions suivantes est vérifiée:
   - l'opération demandée est du type modification;
   - la commande de service **dontUseCopy** est sélectionnée.

   Dans le cas contraire, passer à l'étape suivante.
3) Si la DSE est du type **writeableCopy**, retourner entry unsuitable si l'une des conditions suivantes est vérifiée:
   - l'opération demandée est du type modification et la commande de service **dontUseCopy** est sélectionnée;
   - l'opération demandée est du type interrogation et la commande de service **allowWriteableCopy** n'est pas sélectionnée.

   Dans le cas contraire, retourner entry suitable.
4) Si la commande de service **copyShallDo** est sélectionnée, vérifier si toutes les **criticalExtensions** (extensions critiques) sont supportées. Si c'est le cas, retourner **entry suitable**. Sinon, retourner **unsupported critical extension**.
5) Si la commande de service **copyShallDo** n'est pas sélectionnée, vérifier si toutes les **criticalExtensions** (extensions critiques) sont supportées. Si c'est le cas, passer à l'étape 5); sinon, retourner **entry unsuitable**.
6) Faire la distinction entre les types d'opération suivants:

   en cas d'opération List, passer à l'étape 6);

   en cas d'opération Read, passer à l'étape 7);

   s'il s'agit d'une opération Search ou Compare, passer à l'étape 8).
7) Si l'entrée bénéficie d'une connaissance subordonnée complète, l'opération List peut être effectuée. Dans ce cas, retourner **entry suitable** et, dans le cas contraire, **entry unsuitable**.
8) Si tous les attributs demandés sont présents dans la DSE, retourner **entry suitable**. Si certains attributs manquent, déterminer par des moyens locaux si la copie dupliquée détient tous les attributs détenus par le maître (par exemple en consultant l'accord de duplication miroir). Si c'est le cas, l'entrée est appropriée (retourner **entry suitable**). A défaut, le fournisseur peut détenir les attributs demandés qui ne sont pas présents dans la duplication miroir. Dans ce cas, la demande doit être chaînée (retourner **entry unsuitable**).
9) Si l'opération est de type **search** avec l'attribut **searchAliases** mis à la valeur **TRUE**, que l'entrée DSE soit de type **alias** et que l'argument **chainingArguments.excludeShadows** ait la valeur **FALSE**, le résultat renvoyé est **entry suitable**; si cet argument a la valeur **TRUE**, le résultat renvoyé est **entry unsuitable**.
10) Si le DSA prend en compte la règle de concordance pour la comparaison ou la recherche, et que l'opération soit **compare** ou **search** avec **subset** de l'entrée **baseObject**, passer à l'étape 7). Si le DSA prend en compte la règle de concordance et qu'il s'agisse d'une opération **search** avec le sous-ensemble **oneLevel** ou **subtree**, passer à l'étape 10). Dans le cas contraire, retourner **entry unsuitable**.
11) Si le composant composite **chainingArguments.excludeShadows** est à **TRUE**, retourner **entry unsuitable**. Dans le cas contraire, vérifier la compréhension locale de la spécification de l'information dupliquée par rapport au filtre et au choix de l'opération. En présence de toutes les entrées et de tous les attributs nécessaires, retourner **entry suitable**. Si une entrée ou un attribut manque, retourner un message **entry unsuitable**.

19 Evaluation de l'opération

Le présent paragraphe définit la procédure que doit suivre un DSA si l'entrée cible d'une opération a été trouvée sur place (au cours de la résolution du nom). Suivant le type d'opération, l'une des procédures suivantes est invoquée:

- pour une opération **addEntry**, **chainedAddEntry**, **removeEntry**, **chainedRemoveEntry**, **modifyEntry**, **chainedModifyEntry**, **modifyDN** ou **chainedModifyDN**, suivre les procédures du § 19.1;
- pour une opération **read**, **chainedRead**, **compare** ou **chainedCompare**, suivre les procédures du § 19.2;
- pour une opération **search**, **chainedSearch**, **list** et **chainedList**, suivre les procédures du § 19.3.

19.1 Procédure de modification

Selon le type d'opération de modification, suivre les procédures correspondantes définies aux § 19.1.1 à 19.1.4.

19.1.1 Opération Add Entry

1) Le DSA s'assure que l'initiateur a suffisamment de droits d'accès, comme défini au § 11.1.5 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si ce n'est pas le cas, l'erreur appropriée est retournée.

2) Le DSA s'assure qu'une entrée ayant le nom de l'entrée à ajouter n'existe pas encore; si elle existe, il doit retourner une erreur **updateError** avec la cause **entryAlreadyExists**. Si la DSE est de type additionnel de **nssr**, le DSA doit suivre la procédure définie au § 19.1.5 (Opérations de modification et NSSR) pour s'assurer que le nom de la nouvelle entrée n'est pas ambigu. Si le nom de l'entrée à ajouter contient, pour l'un des attributs du RDN final, plusieurs valeurs distinctives différenciées par le contexte, le DSA doit s'assurer qu'aucune des variantes de RDN qu'il est possible de construire ne formera, indépendamment du contexte, de nom pour une entrée qui existe déjà.

3) Si le composant **targetSystem** est présent et que le point **AccessPoint** ne soit pas celui du DSA actuel, passer à l'étape 4). Si le composant **targetSystem** n'est pas présent ou s'il est présent et que **AccessPoint** soit celui du DSA actuel, passer à l'étape 5).

4) Si l'entrée est une sous-entrée, le DSA doit retourner l'erreur **updateError** avec la cause **affectsMultipleDSA**. Si l'entrée n'est pas une sous-entrée, le DSA peut choisir localement si oui ou non il souhaite établir un HOB avec le DSA spécifié. Si ce n'est pas le cas, le DSA doit retourner une erreur de type **serviceError** avec la cause **unwillingToPerform**; sinon il doit établir un rattachement opérationnel hiérarchique avec le DSA subordonné spécifié. Si le DOP est supporté, la procédure du § 24.3.1.1 doit être suivie; sinon, des moyens locaux sont utilisés pour établir le HOB. Si le DSA subordonné ne souhaite pas l'établissement du rattachement opérationnel, une erreur de type **serviceError** avec la cause **unwillingToPerform** est retournée pour l'opération **addEntry**. Si le HOB est normalement établi, passer à l'étape 7).

   NOTE 1 – Cette étape de la procédure n'est pas applicable à la création de zones administratives autonomes chez un DSA subordonné.

5) Le DSA doit s'assurer que la nouvelle entrée à ajouter est conforme au sous-schéma, ou que la nouvelle sous-entrée ou la DSE appartenant à d'autres types sont en conformité avec le schéma du système (que, par exemple, la DSE immédiatement supérieure à une sous-entrée est du type **admPoint**), auquel cas il doit ajouter la DSE. Sinon, il doit renvoyer une **updateError** ou **attributeError** qui convient. S'il s'agit d'une entrée, continuer à l'étape 7), à l'étape 6) s'il s'agit d'une sous-entrée. Sinon, il y a exécution des procédures de gestion des connaissances appropriées aux autres types de DSE. Voir la Section 6.

6) Le DSA doit expédier, à un moment approprié, un rattachement opérationnel de modification à tous les DSA subordonnés avec lesquels il possède des rattachements opérationnels hiérarchiques ou hiérarchiques non spécifiques applicables. Les rattachements applicables sont ceux qui sont associés à des contextes de dénomination subordonnés à la DSE supérieure. Les contextes de dénomination dont les préfixes correspondent à des points administratifs autonomes ne sont pas applicables. Si le DOP est supporté, les procédures indiquées aux § 24.3.2.1 et 25.3.2 doivent être suivies. Si le DOP n'est pas supporté, des moyens locaux doivent être utilisés pour modifier les RHOB.

   NOTE 2 – Un moment approprié est spécifié par l'administrateur des DSA. Il peut s'insérer immédiatement après (ou même un peu avant) le retour du résultat d'opération selon une stratégie de récurrence (par exemple à une heure de consigne). Le moment peut varier selon le motif de la modification, par exemple des mises à jour d'informations de contrôle d'accès (ACI) prenant effet immédiatement et des modifications périodiques du schéma.

7) Si l'entrée ou sous-entrée ajoutée fait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir, les consommateurs d'informations miroirs doivent être mis à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

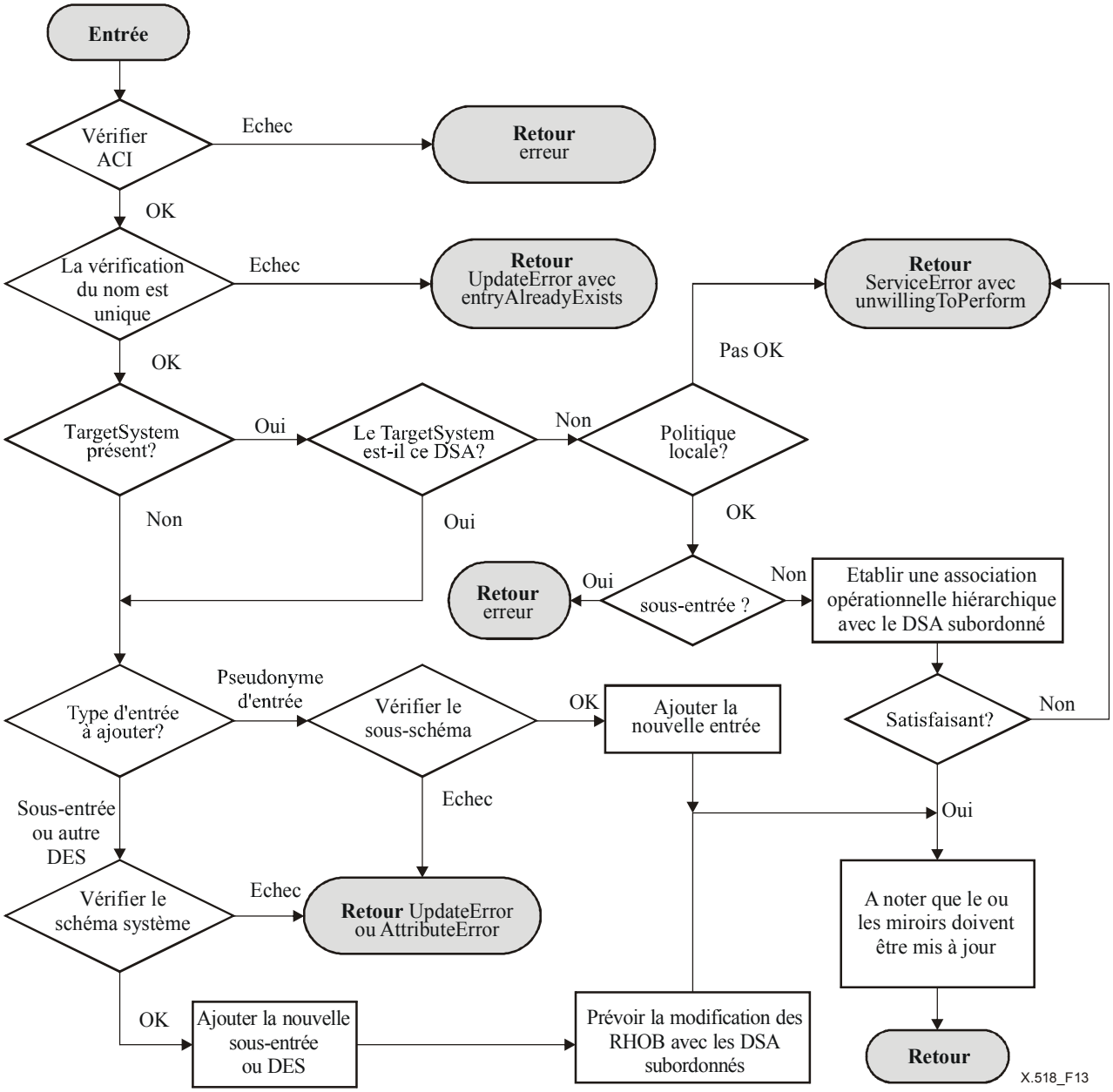


Figure 13 – Procédure Add Entry

19.1.2 Opération Remove Entry (suppression d'entrée)

1) Le DSA doit s'assurer que l'initiateur a suffisamment de droits d'accès, par exemple ceux qui sont définis au § 11.2.5 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si ce n'est pas le cas, une erreur appropriée est retournée.

2) Le DSA doit s'assurer que l'entrée à supprimer est une entrée feuille. Si ce n'est pas le cas, le DSA retourne une **updateError** avec la cause **notAllowedOnNonLeaf**.

3) Le type de DSE est contrôlé sur l'entrée à supprimer. S'il s'agit du type **subentry**, passer à l'étape 5). S'il s'agit du type **cp**, passer à l'étape 6). S'il s'agit du type **entry** ou **alias**, passer à l'étape 4). Sinon, il y a exécution des procédures de gestion des connaissances appropriées aux autres types de DSE. Voir la Section 6.

4) Supprimer l'entrée ou l'alias et passer à l'étape 7).

5) Supprimer la sous-entrée. A un moment approprié, modifier les rattachements opérationnels hiérarchiques de tous les DSA subordonnés appropriés avec lesquels le DSA actuel possède des rattachements opérationnels hiérarchiques ou hiérarchiques non spécifiques. Les rattachements pertinents sont ceux qui sont associés aux contextes de dénomination subordonnés à la DSE supérieure.

Les contextes de dénomination dont les préfixes correspondent à des points administratifs autonomes ne sont pas pertinents. Si le DOP est pris en charge, les procédures des § 24.3.2.1 et 25.3.2 doivent être suivies. A défaut, des moyens locaux doivent être utilisés. Passer à l'étape 7).

6) Supprimer le contexte de dénomination. Si le DSA a un rattachement opérationnel hiérarchique pour ce contexte de dénomination, il met fin à ce rattachement avec son DSA immédiatement supérieur. Si le DSA a un rattachement opérationnel hiérarchique non spécifique pour ce contexte de dénomination et qu'il s'agisse du dernier contexte de dénomination du rattachement en question, il met fin à ce rattachement avec son DSA immédiatement supérieur. Si le DOP est pris en charge, les procédures des § 24.3.3.2 et 25.3.3.2 doivent être suivies. Sinon, des moyens locaux sont utilisés pour mettre fin au RHOB.

7) Si la suppression du contexte de dénomination, de l'entrée, de l'entrée alias ou de la sous-entrée faisait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir, la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

   Si la suppression de la référence subordonnée ou subordonnée non spécifique détenue par le DSA immédiatement supérieur (dont le RHOB a été arrêté) faisait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir, la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

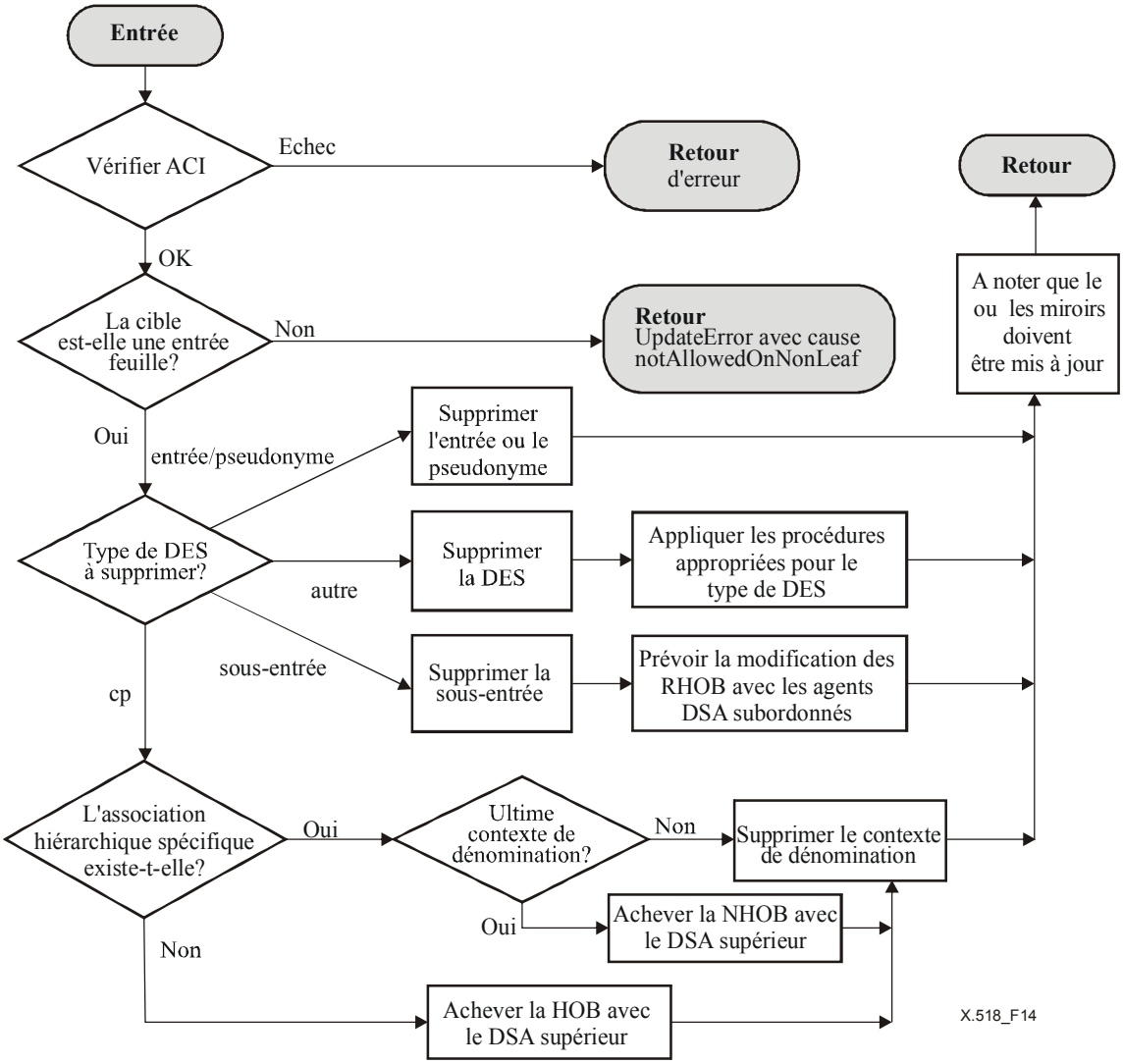


Figure 14 – Procédure Remove Entry

19.1.3 Opération Modify Entry (modification d'entrée)

1) Le DSA doit s'assurer que l'initiateur dispose de droits d'accès comme ceux qui ont été définis au § 11.3.5 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si ce n'est pas le cas, une erreur appropriée est retournée.
2) Les modifications à l'entrée ou à l'alias doivent se conformer au sous-schéma. Toute modification d'une DSE appartenant à d'autres types, sous-entrées comprises, doit se conformer au schéma du système; à défaut, le DSA doit retourner l'erreur appropriée, **updateError** ou **attributeError**. Après avoir exécuté les modifications, si la DSE cible est de type **subentry**, passer à l'étape 3); si elle est de type **entry** ou **alias**, passer à l'étape 4); sinon, il y a exécution des procédures de gestion des connaissances appropriées aux autres types de DSE. Voir la Section 6.
3) Le DSA doit, à un moment approprié, modifier les rattachements opérationnels à tous les DSA subordonnés pertinents avec lesquels il entretient des rattachements opérationnels hiérarchiques ou hiérarchiques non spécifiques. Les rattachements pertinents sont ceux qui sont établis avec des contextes de dénomination subordonnés au point administratif dont dépend la sous-entrée modifiée. Les contextes de dénomination dont les préfixes correspondent à des points administratifs autonomes ne sont pas pertinents. Si le DOP est pris en charge, suivre la procédure des § 24.3.2.1 et 25.3.2. Sinon, utiliser des moyens locaux.
4) Si la modification d'entrée, d'entrée alias ou de sous-entrée faisait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir, la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

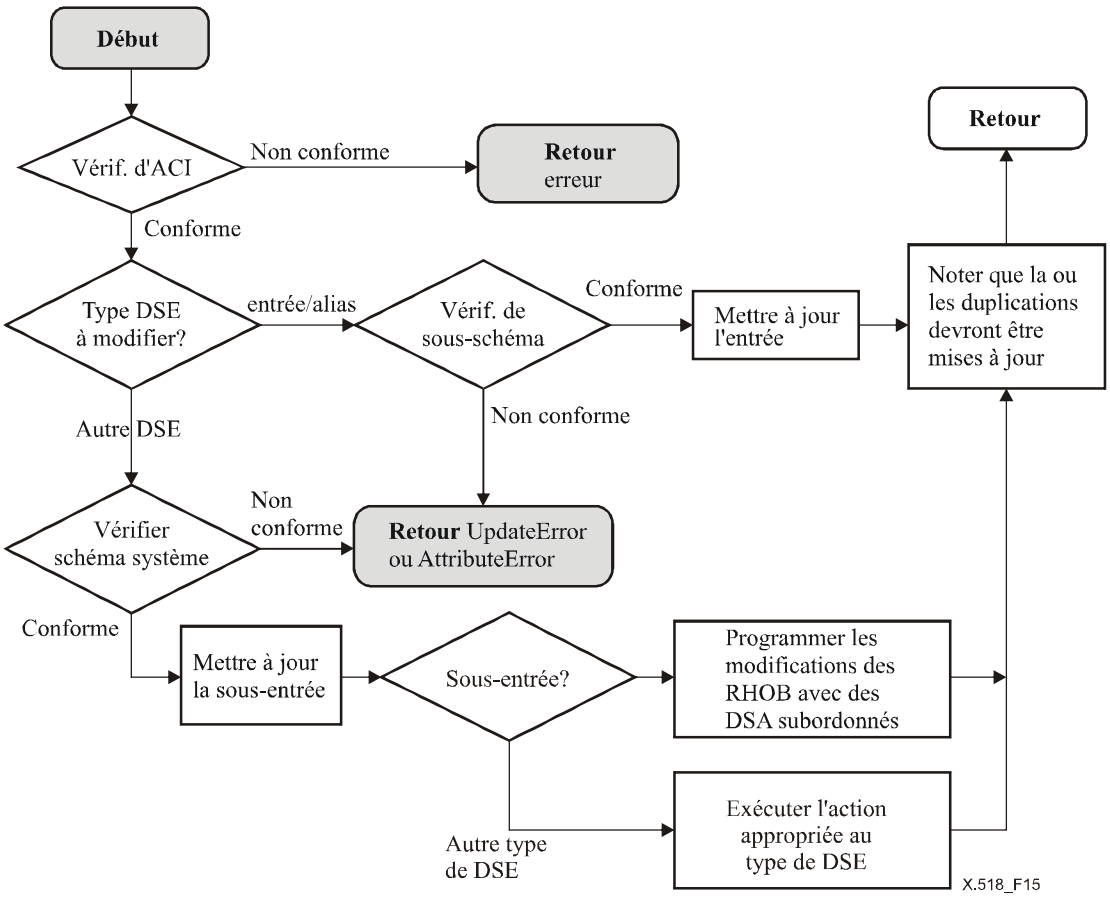


Figure 15 – Procédure Modify Entry

19.1.4 Opération Modify DN (modification de nom distinctif)

1) Le DSA doit s'assurer que l'initiateur dispose de suffisamment de droits d'accès comme ceux qui sont définis au § 11.4.5 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si ce n'est pas le cas, l'erreur appropriée est retournée.

2) Si l'opération consiste soit à déplacer une entrée soit à la fois à déplacer une entrée et à modifier son nom distinctif relatif, passer à l'étape 3). Si l'opération consiste seulement à modifier le nom distinctif relatif d'une entrée, passer à l'étape 4).

3) L'opération doit être exécutée conformément à la définition donnée au § 11.4.1 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si soit l'ancien supérieur, le nouveau supérieur, l'entrée ou une de ses subordonnées ne se trouve pas dans cet agent DSA, ou si le nouveau supérieur possède des références NSSR, l'opération doit être rejetée avec la cause **updateError** avec la cause **affectsMultipleDSA**. L'agent DSA doit s'assurer qu'aucune autre entrée ayant le nouveau nom n'existe déjà; sinon, il doit renvoyer une erreur **updateError** avec la cause **entryAlreadyExists**. L'agent DSA doit s'assurer que le nouveau nom de l'entrée est conforme au sous-schéma; sinon, il doit renvoyer une erreur **attributeError** ou **updateError**, selon le cas. Si aucun de ces problèmes ne se pose, déplacer l'entrée (en modifiant le nom RDN au besoin) et passer à l'étape 9).

4) Le texte qui suit s'applique aux changements du nom distinctif relatif d'entrée/sous-entrée/alias, qui peut ou ne peut pas être une entrée feuille et qui peut ou non avoir une ou plusieurs subordonnées dans un ou plusieurs DSA. Le type de DSE est vérifié pour l'entrée à renommer. S'il s'agit du type **subentry**, passer à l'étape 7). S'il s'agit du type **cp**, passer à l'étape 6). S'il s'agit du type **entry** ou **alias**, passer à l'étape 5).

5) Le DSA doit s'assurer qu'il n'existe pas d'autres entrées portant le nouveau nom; si une telle entrée existe, il doit retourner une erreur de type **updateError** avec la cause **entryAlreadyExists**. Si la DSE supérieure de l'entrée à renommer est de type additionnel **nssr**, le DSA doit suivre la procédure définie au § 19.1.5 (Opérations de modification et NSSR) pour s'assurer que le nouveau nom de l'entrée n'est pas ambigu. Si le nom de l'entrée à ajouter contient, pour l'un des attributs du RDN final, plusieurs valeurs distinctives différenciées par le contexte, le DSA doit s'assurer qu'aucune des variantes de RDN qu'il est possible de construire ne formera, indépendamment du contexte, de nom pour une entrée qui existe déjà. Le DSA doit s'assurer que le nouveau nom de l'entrée ou de la sous-entrée est conforme au schéma ou sous-schéma du système (selon le cas), à défaut de quoi il doit retourner une erreur **attributeError** ou **updateError**, selon le cas. Renommer l'entrée ou l'entrée alias. Si l'entrée n'est pas une entrée feuille et possède des subordonnées dans d'autres DSA, passer à l'étape 8). Sinon, passer à l'étape 9).

6) Le DSA doit s'assurer que le nouveau nom du contexte de dénomination est conforme au sous-schéma. Sinon, il doit retourner une erreur appropriée, de type **attributeError** ou **updateError.**

   Si le DSA possède un HOB avec le DSA supérieur, ce DSA subordonné doit tenter de modifier ce HOB avant de répondre à l'opération de modification du nom distinctif. Le DSA supérieur doit veiller, avant d'accepter la modification, à ce qu'aucune autre entrée portant le nouveau nom n'existe déjà. Si le DOP est pris en charge, la procédure du § 24.3.2.2 doit être suivie. S'il n'est pas pris en charge, la façon de modifier le HOB et la vérification de l'unicité du nouveau nom relèveront de moyens locaux. Si le HOB est bien modifié et que le contexte de dénomination possède des contextes subordonnés dans d'autres DSA, passer à l'étape 8). Sinon, passer à l'étape 9). Si le HOB ne peut pas être modifié, retourner une erreur du type **updateError** avec la cause **affectsMultipleDSA.**

   Si le DSA possède un NHOB avec le DSA supérieur pour ce contexte de dénomination, la façon dont les entrées dupliquées sont détectées n'est pas dans le domaine d'application de la présente Spécification d'annuaire. Renommer l'entrée. Si le contexte de dénomination a des contextes subordonnés dans d'autres DSA, passer à l'étape 8). Sinon, passer à l'étape 9).

7) Le DSA doit s'assurer que le nouveau nom de la sous-entrée est conforme au schéma du système. Sinon, il doit retourner une erreur appropriée: **attributeError** ou **updateError**. Le DSA doit s'assurer qu'aucune autre sous-entrée portant le nouveau nom n'existe déjà. Sinon, il doit retourner une erreur de type **updateError** avec la cause **entryAlreadyExists**.

8) Le DSA doit, à un moment approprié, modifier les rattachements opérationnels à tous les DSA subordonnés avec lesquels il a des rattachements opérationnels hiérarchiques ou hiérarchiques non spécifiques. Les rattachements pertinents sont ceux qui sont associés à tous les contextes de dénomination subordonnés à la sous-entrée modifiée ou aux contextes de dénomination pertinents qui sont subordonnés au point administratif dont la sous-entrée a été renommée. Les contextes de dénomination dont les préfixes correspondent à des points administratifs autonomes ne sont pas pertinents. Si le DOP est pris en charge, suivre la procédure des § 24.3.2.1 et 25.3.2. Sinon, utiliser des moyens locaux pour mettre à jour les RHOB.

9) Si le renommage du contexte de dénomination, de l'entrée ou d'une de ses subordonnées, de l'entrée alias ou de la sous-entrée fait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir détenus par le DSA, la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

Si l'entrée, l'entrée alias ou la sous-entrée faisait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir détenus par le DSA, et que l'entrée supérieure de l'entrée, de l'entrée alias ou de la sous-entrée renommée n'est pas présente dans ce composant **UnitOfReplication,** la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9; dans ce cas l'entrée ayant fait l'objet d'une duplication miroir et toutes les entrées qui lui sont subordonnées seront supprimées.

Si l'entrée, l'entrée alias ou la sous-entrée ne faisait pas partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir détenus par le DSA, et que l'entrée, l'entrée alias ou la sous-entrée renommée est maintenant présente dans ce composant **UnitOfReplication,** la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9; dans ce cas l'entrée ayant fait l'objet d'une duplication miroir et toutes les entrées qui lui sont subordonnées feront l'objet d'une duplication miroir.

Si la référence subordonnée renommée dans le DSA immédiatement supérieur (dont le HOB a été modifié à l'étape 6) ci-dessus) fait partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir détenus par le DSA, la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

Si les composants d'un RHOB à un DSA subordonné (tel que modifié à l'étape 8) ci-dessus) font partie du composant **UnitOfReplication** d'un ou de plusieurs accords de duplication miroir détenus par le DSA, la liste des consommateurs d'informations miroirs doit être mise à jour au moyen des procédures du service de duplication miroir d'informations d'annuaire spécifiées dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

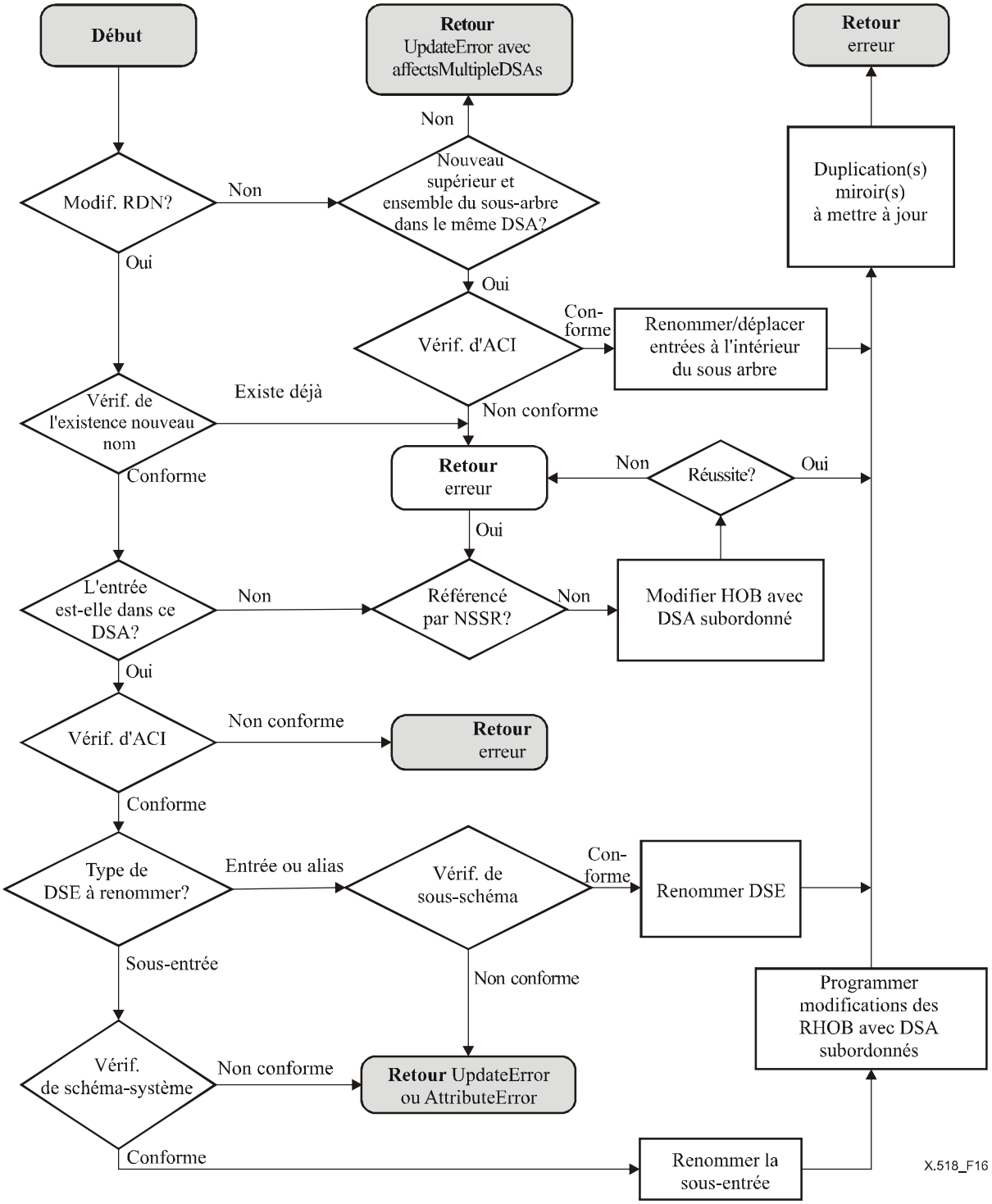


Figure 16 – Procédure Modify DN

19.1.5 Opérations de modification et références subordonnées non spécifiques (NSSR)

Si un DSA a des NSSR mais qu'il ne connaisse pas l'ensemble complet des noms des subordonnés d'une entrée vers laquelle:

a) soit une opération **addEntry** a été envoyée;

b) soit une opération **modifyDN** a été envoyée;

le DSA en question peut exécuter la série suivante de procédures avant d'exécuter l'opération.

1) Si la commande de service **chainingProhibited** est mise à l'opération **addEntry** ou **modifyDN**, le DSA retourne une erreur de type **updateError** avec la cause **affectsMultipleDSA**.

2) Si le DSA n'est ni disposé ni en mesure de multichaîner les demandes sortantes, il doit retourner une erreur de type **serviceError** avec, selon le cas, la cause **unwillingToPerform** ou **unavailable**.

3) Le DSA doit multichaîner une opération **chainedReadEntry** vers chacun des DSA de l'ensemble **accessPointInformation** de la NSSR. (Le DSA utilise uniquement le DSA maître de chaque variable **MasterAndShadowAccessPoints** en raison de l'incohérence passagère résultant de la duplication miroir.) Les paramètres de la variable **ReadArgument** seront réglés comme suit:

   object mis soit au nom de l'entrée à ajouter (dans le cas de l'opération **addEntry**) soit au nom proposé d'une entrée existante (dans le cas de l'opération **modifyDN**).

   selection l'attribut de la classe d'objets.

   Les paramètres de la variable **CommonArguments** seront réglés comme suit:

   – **ServiceControls.option** mis à **dontDereferenceAliases**;

   – **OperationProgress.nameResolutionPhase** mis à **completed**.

   Les paramètres de la variable **ChainingArguments** seront réglés comme suit:

   – **originator** mis au nom de l'expéditeur;

   – **targetObject** omis;

   – **OperationProgress.nameResolutionPhase** mis à **proceeding** et **nextRDNToBeResolved** mis à (nombre de RDN contenus dans le nom objet) – 1;

   – **traceInformation** mis à une séquence vide;

   – **referenceType** mis à **nonSpecificSubordinate**;

   – **timeLimit**, selon le contenu de la demande entrante.

   Les autres paramètres, comme **SecurityParameters**, pourront être réglés selon les besoins, par exemple selon une politique locale.

4) Le DSA attend la série complète de réponses. Si l'une d'elles est de type **ReadResult**, une erreur est retournée comme à l'étape 6) ci-dessous.

5) Si toutes les réponses sont du type **serviceError** avec la cause **unableToProceed**, l'évaluation de l'opération peut se poursuivre.

6) En cas de retour d'un résultat **ReadResult**, une erreur **updateError** avec la cause **entryAlreadyExists** sera retournée pour l'opération initiale.

7) Si une quelconque autre erreur est retournée à la demande de **readEntry**, une erreur **serviceError** avec la cause **unwillingToPerform** est retournée.

Le DSA recevant une demande **chainedRead** donnera une réponse selon la présence ou non de l'entrée et compte tenu de sa politique de contrôle d'accès.

19.2 Procédure d'interrogation à entrée unique

Les opérations **read**, **chainedRead**, **compare** et **chainedCompare** relèvent du groupe de procédures d'interrogation à entrée unique. Ces procédures contiennent uniquement les trois étapes ci-dessous:

1) vérification de contrôle d'accès, comme décrit au § 9 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si l'opération est désapprouvée, retourner l'erreur de sécurité appropriée;

2) effectuer l'opération sur la DSE trouvée comme indiqué au § 9 de la Rec. UIT-T X.511 | ISO/CEI 9594-3;

3) préparer la réponse et la retourner.

19.3 Procédure d'interrogation à entrées multiples

Selon le type d'opération d'interrogation (**list** ou **search**), les procédures correspondantes, définies aux § 19.3.1 et 19.3.2, s'appliquent.

19.3.1 Procédure List

Le présent paragraphe contient la spécification de la procédure d'évaluation relative aux opérations **list** et **chainedList**.

La procédure **List (I)** doit être suivie quand le composant **operationProgress.nameResolutionPhase** de la demande est mis à **notStarted** ou à **proceeding** ou quand le DSA, après avoir effectué la résolution du nom, constate qu'il contient l'entrée d'objet de base. La procédure **List (II)** doit être suivie quand le composant **nameResolutionPhase** de la demande de listage a été mis à **completed**.

19.3.1.1 Paramètres de la procédure

19.3.1.1.1 Arguments

Les arguments utilisés par cette procédure sont les suivants:

- **ListArgument**;
- l'entrée cible DSE **e**;
- le paramètre **operationProgress** du composant **chainingArgument**.

19.3.1.1.2 Résultats

Si la procédure aboutit, elle retourne:

- un ensemble de subordonnés de **e** dans **listInfo.subordinates**;
- la cause **limitProblem** indiquée dans **partialOutcomeQualifier**;
- un ensemble de références de continuation dans **SRcontinuationList**.

19.3.1.2 Définition de la procédure

19.3.1.2.1 List Procedure (I)

La procédure **List (I)** est constituée des étapes suivantes, illustrées à la Figure 17:

1) Si la commande de service **subentry** est sélectionnée, passer à l'étape 5); sinon, passer à l'étape 2).
2) Si la DSE **e** est de type **nssr**, ajouter une **continuationReference** à **SRcontinuationList** avec les composants suivants:
   - **targetObject** mis au nom distinctif primaire de la DSE **e** (les RDN peuvent contenir des variantes de valeurs distinctives);
   - **aliasedRDNs** absent;
   - **operationProgress** avec **nameResolutionPhase** mis à **completed** et **nextRDNtoBeResolved** absent;
   - **rdnsResolved** absent;
   - **referenceType** mis à **nonSpecificSubordinate**;
   - **accessPoints** mis à un ensemble d'items **accessPointInformation** tirés chacun d'une valeur de l'attribut **nonSpecificKnowledge** de la DSE **e**.
3) Pour chaque entrée DSE **e'** immédiatement subordonnée à la DSE **e**, exécuter les opérations suivantes:
   a) vérifier dans la sous-entrée **e'** si l'information ACI est disponible. Si l'information ACI interdit le listage des RDN de **e'**, omettre cette DSE. Si l'information ACI n'est pas disponible (par exemple en cas de référence subordonnée et d'entrée interstitielle), poursuivre ou non compte tenu de la politique locale;
   b) vérifier tous les types de DSE de **e'**.
      i) Si **e'** est du type **subr**, deux cas sont possibles. Dans le premier, l'information ACI de l'entrée subordonnée et la classe d'objets sont disponibles sur place; dans ce cas, conformément à la politique locale et aux informations ACI, ajouter le RDN de la sous-entrée **e'** à **listInfo.subordinates** avec **aliasEntry** mis à **TRUE** si **e'** est de type **sa** et **fromEntry** mis à **FALSE**. L'autre cas est celui où l'entrée de l'information ACI n'est pas disponible dans **e'**; dans ce cas, ajouter une référence **continuationReference** à la liste **SRcontinuationList** avec les composants suivants:
         - **targetObject** mis au nom distinctif primaire de la DSE **e** (les RDN peuvent contenir des variantes de valeurs distinctives);
         - **aliasedRDNs** absents;
         - **operationProgress** avec **nameResolutionPhase** mis à **completed** et **nextRDNtoBeResolved** absent;
         - **rdnsResolved** absent;

– **referenceType** mis à **subordinate**;

– **accessPoints** mis à la valeur contenue dans l'attribut **specificKnowledge** de la sous-entrée DSE **e'**.

ii) Si la sous-entrée DSE **e'** est de type **entry** ou **glue** (interstitielle), ajouter le RDN de **e'** à **listInfo.subordinates**, avec **aliasEntry** mis à **FALSE** et **fromEntry** sélectionné selon que **e'** est ou non une copie.

NOTE – Si l'entrée **e'** est interstitielle (**glue**), elle doit avoir un ou plusieurs subordonnés, ce qui implique qu'elle ne peut être un alias dans le DSA maître. Par ailleurs, toute information ACI applicable au listage (**List**) est enregistrée dans cette DSE, fournie via le protocole de duplication miroir.

iii) Si la sous-entrée DSE **e'** est de type **alias**, ajouter le RDN de l'entrée **e'** à **listInfo.subordinates** avec **aliasEntry** mis à **TRUE**, et **fromEntry** sélectionné si **e'** est une copie;

c) vérifier si l'heure, la taille ou la limite administrative est dépassée. Si c'est le cas, sélectionner **limitProblem** en conséquence dans **partialOutcomeQualifier** et faire le retour;

d) poursuivre à partir du point 3) a) jusqu'à ce que toutes les DSE subordonnées aient été traitées.

4) Quand toutes les DSE subordonnées ont été traitées, retourner à l'aiguilleur d'opérations.

5) Pour chaque sous-entrée **e'** immédiatement subordonnée à l'entrée DSE **e**, exécuter les sous-étapes suivantes:

a) vérifier l'information ACI dans la sous-entrée **e'**. Si l'information ACI interdit le listage des noms RDN de **e'**, omettre cette entrée DSE. Sinon, ajouter les noms RDN de **e'** au composite **listInfo.subordinates** avec le composant **aliasEntry** mis à **FALSE** et le composant **fromEntry** sélectionné selon que **e'** est ou non une copie;

b) vérifier si la limite de temps, de taille ou d'administration est dépassée. Si c'est le cas, sélectionner le composant **limitProblem** en conséquence dans le qualificateur **partialOutcomeQualifier** et faire le retour.

6) Revenir à l'aiguilleur d'opérations.

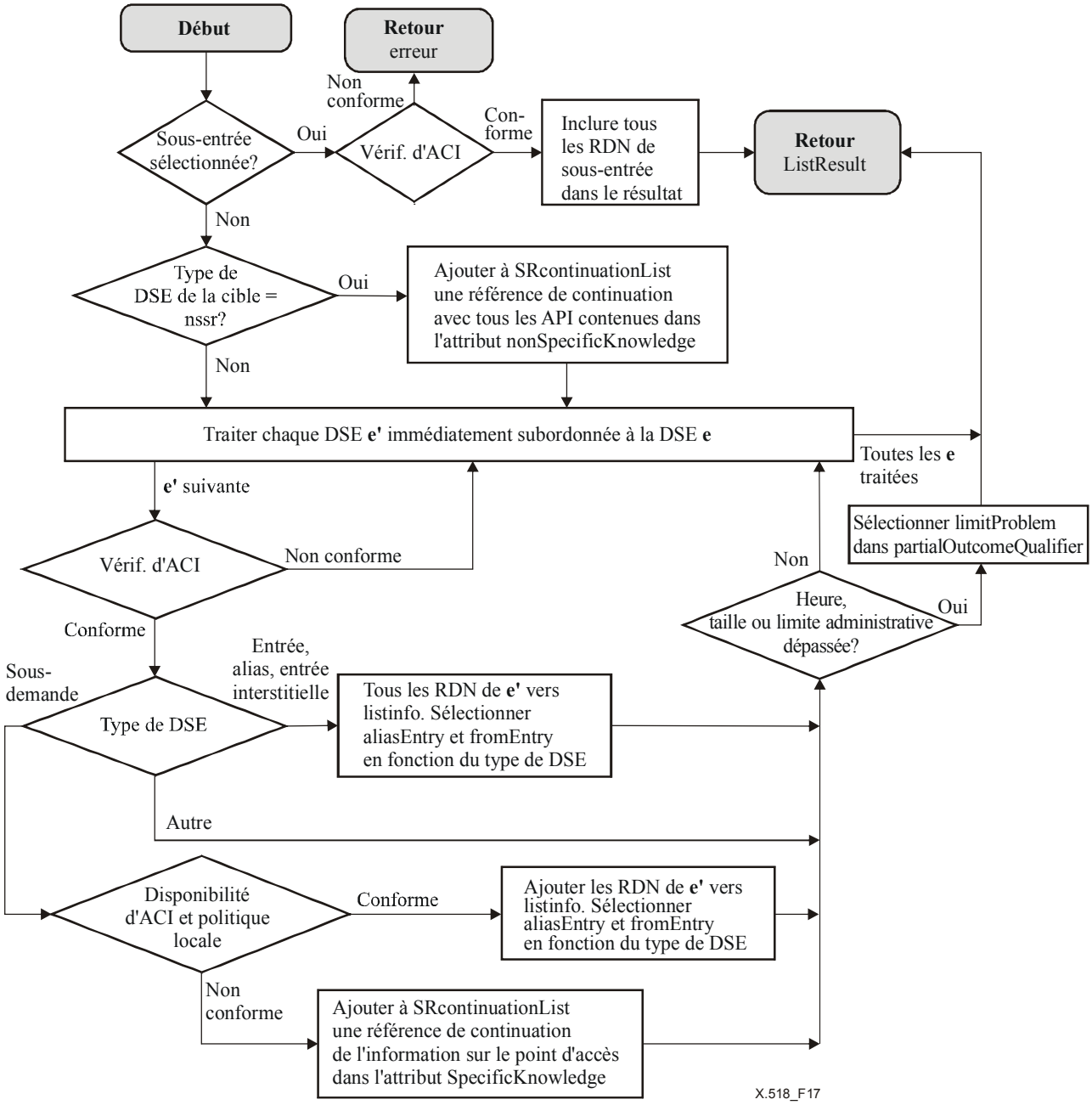


Figure 17 – Procédure List (I)

19.3.1.2.2 Procédure List (II)

La procédure **List (II)** est constituée des étapes suivantes, illustrées à la Figure 18:

1) pour chaque sous-entrée DSE **e'** immédiatement subordonnée à la DSE **e**, exécuter les opérations 1), a) à 1) d) suivantes:
   a) si **e'** n'est pas une entrée ou un alias, poursuivre avec la subordonnée immédiatement suivante;
   b) vérifier l'information ACI dans **e'**. Si l'opération est interdite par l'information ACI, poursuivre avec la subordonnée immédiatement suivante de **e**;
   c) ajouter le RDN de la sous-entrée DSE **e'** au composant composite **listInfo.subordinates**, le composant **aliasEntry** de **listInfo.subordinates** étant sélectionné si **e'** est un alias et le composant **fromEntry** étant sélectionné selon que **e′** est ou non une copie. Si le composant **excludeShadows** a la valeur **TRUE**, ne pas tenir compte des DSE de type **shadow** ou **writableCopy**;
   d) vérifier si l'heure, la taille ou la limite administrative est dépassée. Si oui, sélectionner **limitProblem** de **partialOutcomeQualifier** en conséquence et retourner le résultat;
   e) poursuivre l'étape 1) a) jusqu'à ce que toutes les DSE subordonnées aient été traitées;

2) quand toutes les DSE subordonnées ont été traitées, vérifier si cette sous-demande a été acheminée par DAP ou DSP. Si cette sous-demande est soumise via le DAP et que **ListResult** soit vide, retourner une **serviceError** avec la cause **invalidReference** à l'aiguilleur d'opérations. Sinon, retourner **ListResult**.

NOTE – La référence **invalidReference** s'utilise à titre de sécurité si l'utilisateur n'a pas accès à l'entrée supérieure. Si l'information de contrôle d'accès (ACI) de l'entrée supérieure est disponible (fournie par le rattachement opérationnel pertinent RHOB), un résultat nul peut être retourné si cela est permis.

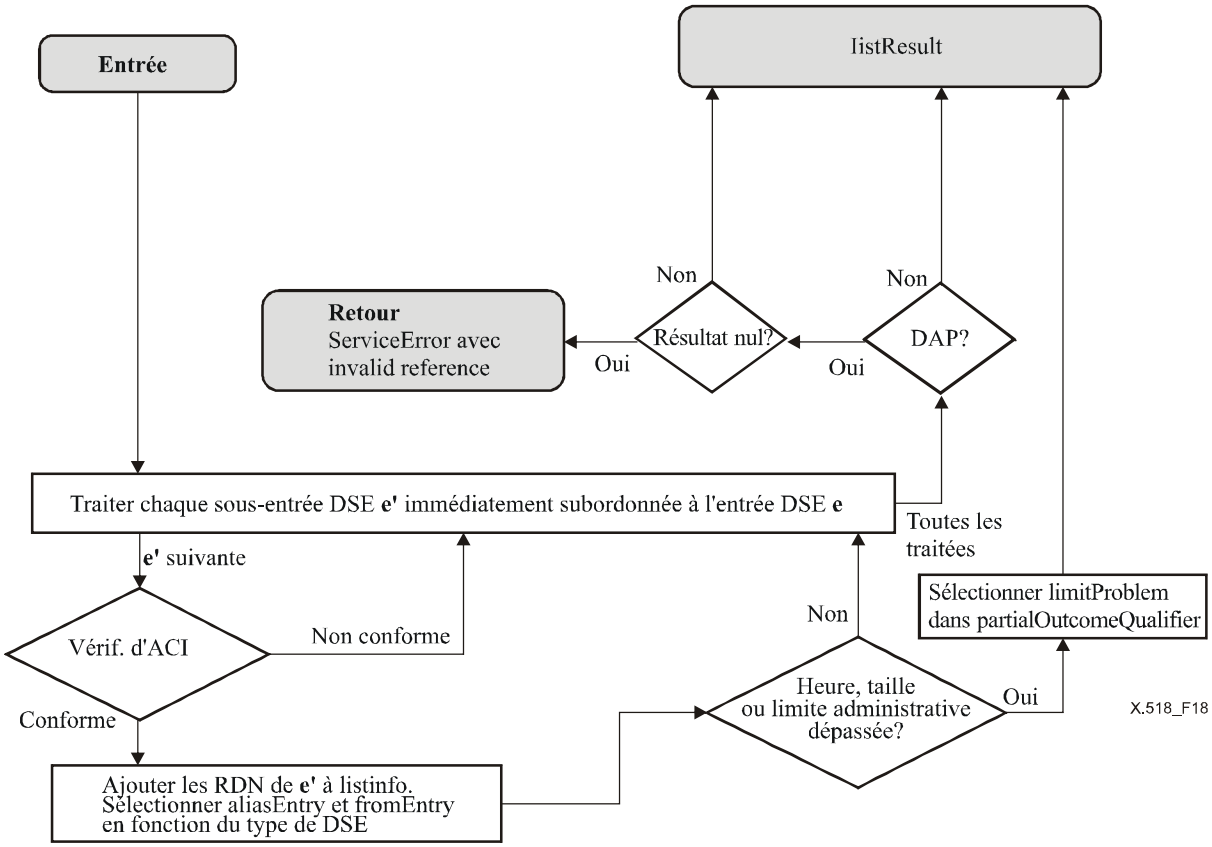


Figure 18 – Procédure List (II)

19.3.2 Procédure Search

Le présent paragraphe spécifie la procédure d'évaluation spécifique aux opérations **search** et **chainedSearch**.

La procédure de vérification de règle de recherche (I) doit être suivie quand le composant **operationProgress.nameResolutionPhase** de demande de recherche est mis à **notStarted** ou **proceeding** et quand le DSA, après avoir effectué la résolution du nom, estime qu'il détient l'entrée d'objet cible. Si cette procédure retourne une erreur, retourner le résultat avec cette erreur. Dans le cas contraire, la procédure **Search (I)** doit être suivie.

La procédure de vérification de règle de recherche (II) doit être suivie quand le composant **nameResolutionPhase** de demande de recherche est mis à **completed**. Si cette procédure retourne une erreur, retourner le résultat avec cette erreur. Dans le cas contraire, la procédure **Search (II)** doit être suivie.

NOTE – Quand la variable **nameResolutionPhase** est à **completed**, on présume que l'objet cible est le supérieur immédiat d'un préfixe de contexte.

19.3.2.1 Paramètres de la procédure

19.3.2.1.1 Arguments

Les arguments qui sont utilisés pour cette procédure sont les suivants:

- **SearchArgument**;
- la DSE **e** cible;
- le paramètre **operationProgress** du composant **ChainingArguments**;
- le paramètre **exclusions** du composant **ChainingArguments** (une liste de RDN à exclure de la recherche);

- le paramètre **traceInformation** du composant **ChainingArguments**;
- le paramètre **searchRuleId** du composant **ChainingArguments**;
- le paramètre **chainedRelaxation** du composant **ChainingArguments**;
- le paramètre **relatedEntry** du composant **ChainingArguments**.

19.3.2.1.2 Résultats

Si cette procédure est correctement exécutée, elle retourne:

- un ensemble d'entrées mises en concordance et contenues dans le composant composite **searchResult.entryInformation**;
- **alreadySearched** dans **ChainingResults**;
- selon les conditions, un compte dans le qualifiant **partialOutcomeQualifier.entryCount**;
- un ensemble de références de continuation dans **SRcontinuationList**.

19.3.2.2 Définition de la procédure

19.3.2.2.1 Procédure de l'argument Related Entry

Cette procédure n'a de sens que si la demande **search** a un composant **joinArguments** et que **ChainingArguments** (s'il y en a) n'a pas de composant **relatedEntry**.

1) Si la demande **search** est protégée, générer une demande DSP pour chaque élément du composant **joinArguments** comportant chacun la demande DAP ou le LDAPMessage initial. Le composant **ChainingArguments** doit se présenter de la manière suivante:
   - si la demande entrante a un composant **ChainingArguments** ayant le composant **originator**, la valeur de celui-ci est copiée dans le composant **originator** des demandes qui ont été générées; sinon, l'emploi de ce composant est déterminé par la politique de sécurité locale;

     NOTE – Le DSA récepteur ne peut pas toujours utiliser le nom donné dans ce composant étant donné qu'il résulte d'un arbre DIT distinct.
   - le composant **operationProgress** doit être omis ou mis à la valeur par défaut;
   - les composants **traceInformation**, **aliasDereferenced**, **aliasedRDNs**, **returnCrossRefs**, **entryOnly**, **exclusions**, **nameResolutionOnMaster**, **searchRuleId** et **chainedRelaxation** doivent être omis;
   - le composant **relatedEntry** est mis à une valeur correspondant à la position relative de **JoinArgument** qui s'applique au DSA auquel la demande est envoyée; la valeur 0 étant attribuée au premier **JoinArgument**, la valeur 1 au suivant, etc.
2) Si la demande de recherche entrante n'est pas protégée, générer une demande DSP pour chaque élément du composant **jointArgument, SearchArgument** étant généré de la manière suivante:
   - le composant **baseObject** doit être copié du composant **joinBaseObject** du **JoinArgument** correspondant;
   - le composant **subset** doit être copié du composant **joinSubset** du **JoinArgument** correspondant;
   - le composant **filter** doit être copié du composant **filter** de **JoinArgument** correspondant;
   - les autres composants doivent être tels que dans la demande initiale, excepté que les composants **joinArgument** et **joinType** doivent être omis.

   Les arguments **ChainingArguments** doivent être comme ci-dessus pour des demandes protégées, sauf que le composant **relatedEntry** doit être omis.
3) Appeler l'aiguilleur **Operation Dispatcher** pour chaque demande qui doit être poursuivie localement.
4) Si l'aiguilleur **Operation Dispatcher** renvoie une erreur **referral**, d'occupation ou d'indisponibilité, ajouter (ou constituer et ajouter) la référence de continuation à **partialOutcomeQualifier** de **SearchResult**, puis retourner le résultat.
5) Si l'aiguilleur **Operation Dispatcher** retourne d'autres erreurs, les ignorer et retourner le résultat.
6) Si l'aiguilleur **Operation Dispatcher** retourne un résultat **SearchResult**:
   i) si ce résultat est signé, chiffré ou signé et chiffré, l'ajouter à **uncorrelatedSearchInfo** de **SearchResult;**
   ii) si ce résultat n'est pas signé, crypté ou signé et crypté, effectuer le processus de jonction comme indiqué dans la Rec. UIT-T X.511 | ISO/CEI 9594-3.

19.3.2.2.2 Procédure de vérification de règle de recherche Search-rule (I)

Cette procédure est uniquement pertinente si le DSA prend en compte des zones administratives de service.

Si le composant **searchRuleId** est présent dans le composant **ChainingArguments**, l'opération est le résultat d'une procédure de déréférenciation d'alias effectuée au cours d'une phase d'évaluation antérieure. Dans ce cas, si l'entrée DSE cible se situe dans une zone administrative de service présentant un **dmdId** différent ou si elle se situe en dehors d'une zone administrative de service, retourner le résultat avec une erreur de service **unwillingToPerform**. Dans le cas contraire, sélectionner la règle de recherche appropriée en vous appuyant sur des informations de **searchRuleId** et retourner le résultat.

NOTE 1 – L'administration de service a été définie comme étant une extension critique. Lorsqu'un DSA, qui ne prend pas en compte l'administration de service, reçoit une demande de recherche chaînée avec un composant **searchRuleId**, il doit retourner une erreur **serviceError** avec la cause **unavailableCriticalExtension**.

Si le composant **searchRuleId** n'est pas présent et que l'entrée DSE cible se situe en dehors d'une zone administrative de service ou à l'intérieur d'une telle zone mais qu'aucune sous-entrée n'est associée à cette zone, retourner le résultat.

Si l'entrée DSE cible se situe dans une zone administrative de service et que le paramètre **traceInformation** met en évidence que l'opération se trouvait dans une phase d'évaluation antérieure, retourner le résultat avec une erreur de service **unwillingToPerform**.

NOTE 2 – Il s'agit du cas de figure où une recherche a débuté son évaluation initiale en dehors d'une zone administrative de service et cherche maintenant à l'étendre à une autre zone administrative de service.

Dans le cas contraire, la procédure suivante est appliquée:

1) localiser toutes les règles de recherche associées à l'entrée DSE cible, c'est-à-dire toutes les règles de recherche des sous-entrées de service ayant l'entrée DSE cible dans leurs spécifications de sous-arbre (par exemple en utilisant l'attribut opérationnel **searchRulesSubentry**). Ces règles de recherche sont appelées ci-après *règles de recherche candidates*. Si de telles règles de recherche n'existent pas, générer une erreur de service avec la cause **requestedServiceNotAvailable**, inclure dans le composant **notification** du paramètre **CommonResults** un attribut **searchServiceProblem** avec la valeur **id-pr-unidentifiedOperation** et retourner le résultat;
2) si les commandes de service **serviceType** et/ou **userClass** font parties de la demande de recherche, éliminer toutes les règles de recherche non conformes aux commandes de service des règles de recherche candidates. Si la liste qui en résulte est vide, générer une erreur de service avec la cause **requestedServiceNotAvailable**, inclure dans le composant **notification** du paramètre **CommonResults** les renseignements détaillés ci-dessous et retourner:
   - un attribut **searchServiceProblem** avec la valeur **id-pr-unidentifiedOperation**;
   - si la commande de service **serviceType** faisait partie de la demande de recherche, un attribut **serviceType** avec la valeur de cette commande de service.
3) décomposer la liste des règles de recherche candidates en quatre listes (dont certaines peuvent être vides):
   - une liste **GoodPermittedSR** comprenant toutes les règles de recherche candidates pour lesquelles l'invocateur a demandé l'autorisation et qui sont conformes à la demande de recherche selon la procédure de validation de recherche décrite au § 13 de la Rec. UIT-T X.511 | ISO/CEI 9594-3;

     NOTE 3 – Si cette liste n'est pas vide, il n'y a pas lieu de créer les autres listes.
   - une liste **MatchProblemSR** comprenant toutes les règles de recherche candidates pour lesquelles l'invocateur a demandé l'autorisation et qui sont conformes à la demande de recherche à l'exception de la présence du composant matchingUse dans un ou plusieurs profils d'attribut de demande;
   - une liste **BadPermittedSR** comprenant toutes les règles de recherche candidates pour lesquelles l'invocateur a demandé l'autorisation mais qui ne sont pas conformes à la demande de recherche;
   - une liste **DeniedSR** comprenant toutes les règles de recherche candidates pour lesquelles l'invocateur n'a pas demandé d'autorisation.
4) si la liste **GoodPermittedSR** comporte une ou plusieurs règles de recherche vides, sélectionner au moyen d'un algorithme local l'une de ces règles de recherche vide comme règle de recherche dominante et faire retour;
5) si la liste **GoodPermittedSR** n'est pas vide, ignorer toutes les règles de recherche à l'exception de celles présentant l'indication **userClass** la plus élevée;
6) sélectionner l'une des règles de recherche restantes dans la liste **GoodPermittedSR** comme règle de recherche dominante au moyen d'un algorithme local et faire retour.

NOTE 4 – Si dans la liste ci-dessus il y a plusieurs règles de recherche parmi lesquelles effectuer la sélection, l'installation doit enregistrer l'incident pour le signaler à l'administration, dans la mesure où il est probable que les définitions des règles de recherche nécessitent d'être révisées.

7) si la liste **MatchProblemSR** n'est pas vide, sélectionner l'une de ses règles de recherche au moyen d'un algorithme similaire à celui indiqué aux points 5) et 6) ci-dessus, générer une erreur de service et des renseignements associés selon les indications fournies au § 13.4 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, puis faire retour;

8) si la liste **DeniedSR** est vide, poursuivre avec le point 10); dans le cas contraire, ignorer toute règle de recherche de la liste qui n'est pas conforme à la demande de recherche et toute règle de recherche vide. Si vous aboutissez à une liste vide, poursuivre avec le point 10; dans le cas contraire, générer une erreur de service avec la cause **requestedServiceNotAvailable**, insérer dans le composant **notification** du paramètre **CommonResults** les sous-composants énumérés ci-dessous et retourner:
   - un attribut **searchServiceProblem** avec la valeur **id-pr-unavailableOperation**;
   - si toutes les règles de recherche restantes de la liste **DeniedSR** présentent la même valeur pour le composant **serviceType**, un attribut **serviceType** de cette valeur;

9) si la liste **BadPermittedSR** est vide, générer une erreur de service avec la cause **requestedServiceNotAvailable**, insérer dans le composant **notification** du paramètre **CommonResults** les sous-composants énumérés ci-dessous et retourner:
   - un attribut **searchServiceProblem** avec la valeur **id-pr-unidentifiedOperation**;

10) prendre dans l'ordre chaque élément numéroté de la procédure décrite au § 13.1 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, confronter la demande de recherche aux règles de recherche restantes de la liste **BadPermittedSR**, puis pour chaque élément:
   - si la recherche est conforme à l'élément pour certaines règles de recherche mais pas toutes, ignorer les règles de recherche pour lesquelles elle n'est pas conforme;
   - si la liste **BadPermittedSR** ne comprend plus qu'une seule règle de recherche, appliquer la procédure spécifiée au § 13 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, et faire retour;
   - dans le cas contraire, vérifier l'élément suivant.

11) si la liste **BadPermittedSR** ne contient maintenant plus que des règles de recherche non conformes à la recherche d'après la procédure en cours, générer une erreur de service avec la cause **requestedServiceNotAvailable**, insérer dans le composant **notification** du paramètre **CommonResults** les sous-composants énumérés ci-dessous et retourner:
   - un attribut **searchServiceProblem** avec la valeur **id-pr-unidentifiedOperation**;
   - si toutes les règles de recherche de la liste **BadPermittedSR** indiquent le même type de service, un attribut **serviceType** ayant comme valeur ce type de service.

12) prendre dans l'ordre chaque élément numéroté du § 13.2 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, confronter la demande de recherche aux règles de recherche restantes de la liste **BadPermittedSR**, puis pour chaque élément:
   - si la recherche est conforme à l'élément pour certaines règles de recherche mais pas toutes, ignorer les règles de recherche pour lesquelles elle n'est pas conforme;
   - si la liste **BadPermittedSR** ne comprend plus qu'une seule règle de recherche, appliquer la procédure spécifiée au § 13 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, et faire retour;
   - dans le cas contraire, vérifier l'élément suivant;

13) prendre dans l'ordre chaque élément numéroté du § 13.3 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, confronter la demande de recherche aux règles de recherche restantes de la liste **BadPermittedSR**, puis pour chaque élément:
   - si la recherche est conforme à l'élément pour certaines règles de recherche mais pas toutes, ignorer les règles de recherche pour lesquelles elle n'est pas conforme;
   - si la liste **BadPermittedSR** ne comprend plus qu'une seule règle de recherche, appliquer la procédure spécifiée au § 13 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, et faire retour;
   - dans le cas contraire, vérifier l'élément suivant;

14) générer une erreur de service avec la cause **requestedServiceNotAvailable**, insérer dans le composant **notification** du paramètre **CommonResults** les sous-composants énumérés ci-dessous et retourner:

- un attribut **searchServiceProblem** avec la valeur **id-pr-unidentifiedOperation**;
- si toutes les règles de recherche de la liste **BadPermittedSR** indiquent le même type de service, un attribut **serviceType** ayant comme valeur ce type de service.

19.3.2.2.3 Procédure de vérification de règle de recherche (II)

Cette procédure n'est pertinente que si le DSA prend en compte les zones administratives de service.

Si le paramètre **searchRuleId** est absent et que toutes les entrées immédiatement subordonnées (préfixes de contexte) de l'entrée DSE cible sont des points administratifs de service, retourner le résultat avec une erreur de service **serviceError** avec la cause **unwillingToPerform**. Si, en revanche, certaines des entrées subordonnées ne sont pas des points administratifs de service, sélectionner alors les contextes de dénomination correspondant à l'évaluation de recherche et retourner le résultat.

Si le paramètre **searchRuleId** est présent, chaque entrée subordonnée de l'entrée DSE cible est vérifiée pour s'assurer qu'elle se trouve dans la même zone administrative de service que l'entrée DSE cible. Dans le cas contraire, le contexte de dénomination correspondant est exclus de la recherche. S'il reste des contextes de dénomination (y compris ceux du DSA exécutant) dans lesquels la recherche peut se poursuivre, sélectionner la règle de recherche identifiée dans **searchRuleId** et retourner le résultat. S'il ne reste plus de contextes de dénomination dans lesquels la recherche peut se poursuivre, générer une erreur de service **serviceError** avec la cause **unwillingToPerform** et retourner le résultat.

NOTE – Ce dernier cas de figure ne devrait pas se présenter si les informations de connaissance sont cohérentes entre le DSA et le DSA détenant le contexte de dénomination supérieur.

19.3.2.2.4 Sélection des informations d'entrée

Pour les entrées mises en concordance et pour les entrées sélectionnées dans une sélection hiérarchique, les informations d'attribut à retenir sont l'intersection:

a) des informations spécifiées par **searchArgument.selection**, éventuellement modifiées par les spécifications de contexte par défaut, et pour les entrées concordantes également spécifiées par **searchArgument.matchedValuesOnly**;

b) des informations fournies par la règle de recherche dominante (le cas échéant).

Ces informations d'entrée sont ajoutées à la liste des entrées dans **searchResult.entryInformation**.

Ajouter uniquement les attributs dont la taille (type et ensemble des valeurs) n'est pas supérieure à **attributeSizeLimit**.

19.3.2.2.5 Procédure Search (I)

Il s'agit d'une procédure récursive applicable à une demande de recherche qui débute à une entrée donnée **e**. Elle recherche l'entrée cible **e** et ensuite les DSE immédiatement subordonnées à l'entrée **e**. La procédure s'invoque d'elle-même de manière récursive si la recherche porte sur un sous-arbre complet. Elle est constituée des étapes suivantes, représentées dans la Figure 19:

1) si la DSE **e** est de type **cp** (c'est-à-dire une DSE avec préfixe de contexte), vérifier si un quelconque élément de l'argument **exclusions** est un préfixe du nom distinctif de l'entrée **e**.

   a) Dans l'affirmative, retourner le résultat.

   b) Dans le cas contraire, appeler la procédure check Suitability.

      i) Si l'entrée **e** ne convient pas, faire une référence **continuationReference** de la manière suivante et l'ajouter à **SRcontinuationList**:

         - **targetObject** mis au DN de la DSE **e**;
         - **operationProgress** avec **nameResolutionPhase** mis à **proceeding** et **nextRDNtoBeResolved** mis au nombre de RDN contenus dans **e**;
         - tous les autres composants de **continuationReference** sont inchangés.

         Ensuite faire retour.

         NOTE 1 – C'est le seul endroit où une sous-demande de recherche **search** est chaînée à un fournisseur d'informations miroirs. Autrement dit, l'objet cible d'une telle sous-demande chaînée est toujours un préfixe de contexte.

      ii) Par ailleurs, ajouter le nom DistinguishedName de **e** à **alreadySearched** dans **ChainingResults**;

         NOTE 2 – **alreadySearched** contient uniquement des préfixes de contexte.

2) Si **e** est du type **alias** et que **searchAliases** de **SearchArgument** soit à **TRUE**, appeler la procédure **Search Alias** et faire ensuite un retour.

3) Si **subset** est à **oneLevel**, passer à l'étape 6).

NOTE 3 – L'entrée **e** ne peut pas être une subordonnée incomplète à ce point étant donné que la procédure Check Suitability du préfixe de contexte a dû s'assurer que cela ne pouvait se produire.

4) Si le composant **subset** a la valeur **baseObject** ou si le composant **entryOnly** a la valeur **TRUE**, poursuivre cette étape; sinon, passer à l'étape 5).

Si l'une des conditions suivantes a la valeur VRAI:

a) **e** est du type **subentry** et la commande de service **subentry** est sélectionnée;

b) **e** n'est pas du type **subentry** et la commande de service **subentry** n'est pas sélectionnée, effectuer les opérations suivantes:

i) vérifier l'information ACI. Si l'opération est interdite, retour;

ii) appliquer l'argument de filtre spécifié dans **SearchArgument.filter** à la DSE **e**. S'assurer que l'accès à tous les attributs utilisant les filtres est permis comme indiqué dans la Rec. UIT-T X.501 | ISO/CEI 9594-2. S'il y a concordance, et si l'entrée n'est pas exclue du fait d'une sélection de hiérarchie, ajouter les informations d'attribut comme spécifié au § 19.3.2.2.3;

iii) si la commande de service **hierarchySelection** fait partie de la demande de recherche **search** (éventuellement modifiée par une spécification de règle de recherche), que l'entrée fait partie d'un groupe hiérarchique possédant plus d'un membre et que d'autres commandes que l'indication **self** sont activées, appeler la procédure **Hierarchy Selection (I)**.

Retour.

5) Si **subset** est à **subtree** (et si **entryOnly** n'est pas à **TRUE**), et qu'en plus il soit vérifié:

a) que l'entrée **e** est du type **subentry** et que la commande de service **subentry** est sélectionnée;

b) que l'entrée **e** n'est pas de type **subentry** et que la commande de service **subentry** n'est pas sélectionnée; procéder de la façon suivante:

i) vérifier l'information ACI. Si l'opération est interdite, passer à l'étape 6);

ii) appliquer l'argument de filtre spécifié dans **SearchArgument.filter** à la DSE **e**. S'assurer que l'accès à tous les attributs utilisant les filtres est permis, comme indiqué dans la Rec. UIT-T X.501 | ISO/CEI 9594-2. S'il y a concordance, et si l'entrée n'est pas exclue du fait d'une sélection de hiérarchie, ajouter les informations d'attribut comme spécifié au § 19.3.2.2.3;

iii) si la commande de recherche **hierarchySelection** fait partie de la demande de recherche **search** (éventuellement modifiée par une spécification de règle de recherche), que l'entrée fait partie d'un groupe hiérarchique possédant plus d'un membre et que d'autres commandes que l'indication **self** sont activées, appeler la procédure **Hierarchy Selection (I)**;

iv) Passer à l'étape 6).

6) Si **e** est de type **nssr**, ajouter une **continuationReference** à **SRcontinuationList** avec les composants suivants:

– **targetObject** mis au nom distinctif primaire de la DSE **e** (les RDN peuvent contenir des variantes de valeurs distinctives);

– **aliasedRDNs** absents;

– **operationProgress** avec **nameResolutionPhase** mis à **completed** et **nextRDNtoBeResolved** absent;

– **rdnsResolved** absent;

– **referenceType** mis à **nssr**;

– **accessPoints** mis à **AccessPointInformation** tirée de la ou des valeurs trouvées dans l'attribut **nonSpecificKnowledge**.

7) Traiter toutes les sous-entrées DSE **e'** immédiatement subordonnées à l'entrée cible DSE **e** jusqu'à ce que toutes les DSE subordonnées aient été traitées. Si **e** se situe dans une zone administrative de service, seules les DSE immédiatement subordonnées faisant partie de la même zone administrative de service doivent être traitées. Si **e** se situe en dehors d'une zone administrative de service, les DSE immédiatement subordonnées faisant partie d'une zone administrative de service ne doivent pas être traitées. Si, au cours de cette boucle, la liste des entrées concordantes dans **searchResult.entryInformation** dépasse la taille limite, le temps limite ou la limite administrative, sélectionner le fanion **limitProblem** correspondant dans **partialOutcomeQualifier** et faire retour.

NOTE 4 – Le contrôle de la limite de taille est appliqué de manière implicite à chaque mise à jour de **searchResult**.

a) Si la DSE **e'** est du type **subr**, n'est pas du type **cp**, et ne représente pas une entrée subordonnée qui est un point administratif de service, ajouter une **continuationReference** à **SRcontinuationList** avec les composants suivants:

- **targetObject** mis au nom distinctif primaire de la DSE **e** (les RDN peuvent contenir des variantes de valeurs distinctives);
- **aliasedRDNs** absents;
- **operationProgress** avec **nameResolutionPhase** mis à **completed** et **nextRDNToBeResolved** absent;
- **rdnsResolved** absent;
- **referenceType** mis à **subr**;
- **accessPoints** mis à l'information relative aux points d'accès contenue dans l'attribut **specificKnowledge** de la DSE **e'**;

NOTE 5 – Si **e'** est à la fois de type **cp** et **subr**, une sous-demande de recherche pourra éventuellement être émise à partir de la référence subordonnée ou de la base de connaissance du fournisseur, mais non des deux. La présente procédure utilise la seconde solution (références de fournisseur trouvées dans le préfixe cp).

b) Dans tous les cas:

i) si **subset** a la valeur **oneLevel**, mettre **entryOnly** à **TRUE**;

ii) exécuter par récurrence la procédure **Search (I)** pour l'entrée DSE cible **e'**.

8) Quand toutes les subordonnées ont été traitées, retourner à l'aiguilleur d'opérations pour la suite du traitement.

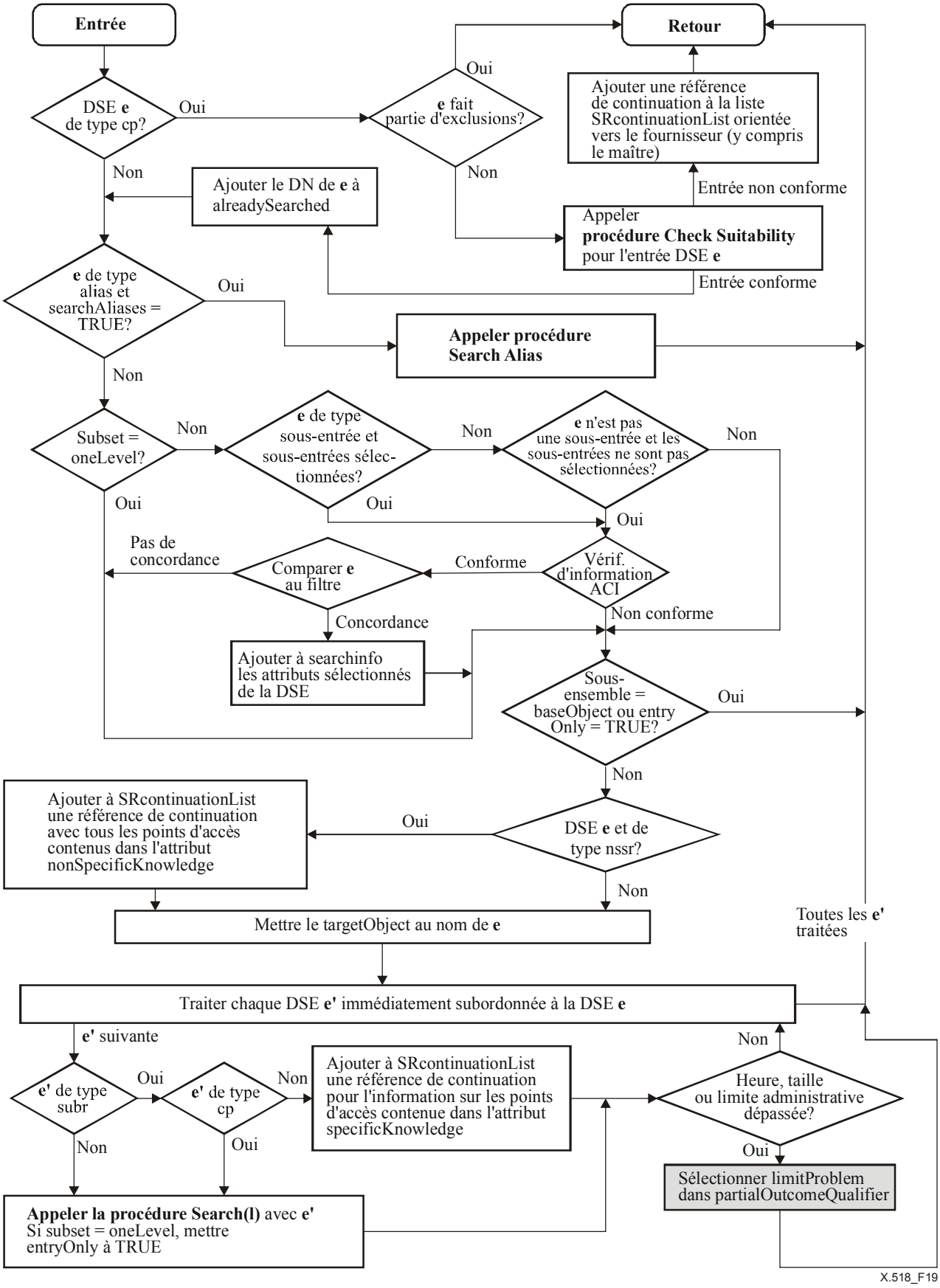


Figure 19 – Procédure Search (I)

19.3.2.2.6 Procédure Search (II)

Cette procédure doit être suivie en cas de traitement d'une demande de recherche provenant d'une décomposition de demande au niveau du DSA dont on a reçu la demande. Cette procédure traite les DSE qui dépendent de l'entrée cible DSE **e** et appelle la procédure **Search (I)** pour chaque entrée d'objet:

1) traiter toutes les sous-entrées DSE **e'** qui sont immédiatement subordonnées à l'entrée DSE **e** cible jusqu'à ce que toutes les DSE subordonnées aient été traitées. Lorsque toutes les entrées subordonnées ont été traitées, revenir à l'aiguilleur d'opérations pour la suite du traitement;
2) si l'entrée DSE n'est pas de type **cp**, ne pas en tenir compte et revenir à l'étape 1);
3) appeler la procédure **Check Suitability**. Si le résultat est favorable, passer à l'étape 4). Sinon, ignorer cette entrée et passer à l'étape 1);
4) exécuter la procédure **Search Procedure (I)** pour la DSE **e'** comme indiqué au § 19.3.2.2. Si la DSE est du type **alias** et que la valeur du paramètre **subset** est à **oneLevel**, mettre le composant **ChainingArguments.entryOnly** à **TRUE** à l'appel de la procédure **Search (I)**. Revenir à l'étape 1).

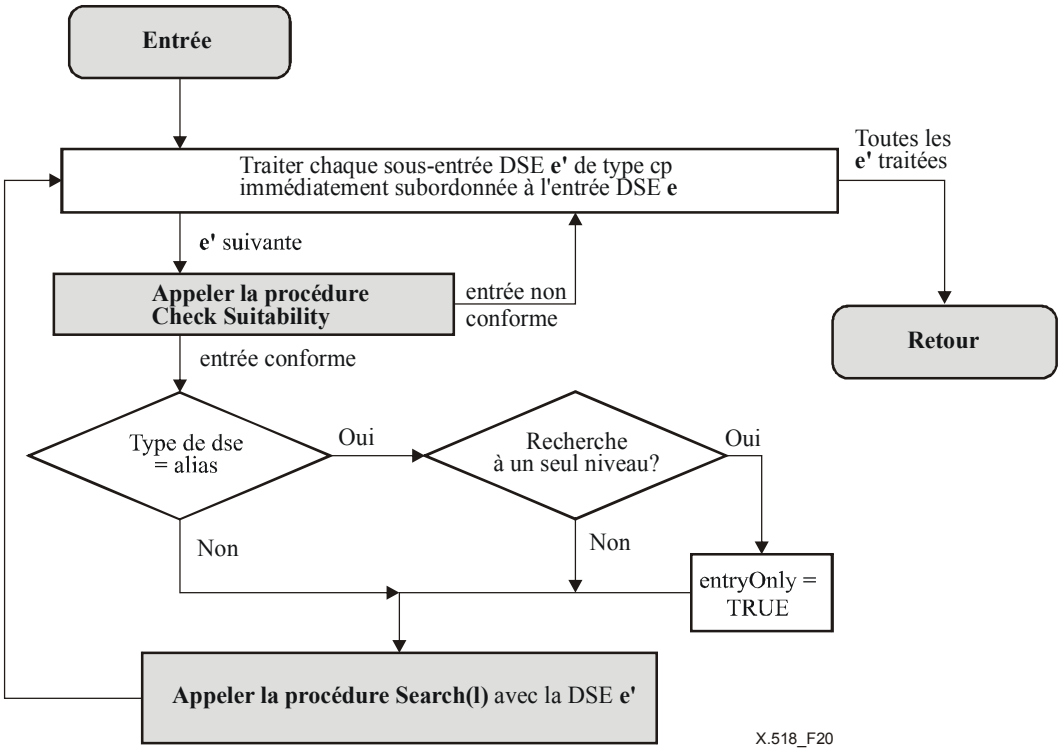


Figure 20 – Procédure Search (II)

19.3.2.2.7 Procédure Search Alias

Cette procédure est exécutée si une DSE de type **alias** a été rencontrée au cours du traitement d'une demande de recherche (voir Figure 21):

1) si le paramètre **subset** a la valeur **baseObject** ou **oneLevel**, passer à l'étape 4);
2) si le nom **aliasedEntryName** est un préfixe de **targetObject** ou **baseObject** ou de l'une quelconque des valeurs précédentes de l'objet cible contenu dans le composite **ChainingArguments.traceInformation**, l'alias est exclu de la recherche parce que cela entraînerait une recherche récursive produisant des résultats en double;
3) si **targetObject** ou **baseObject** ou l'une quelconque des valeurs précédentes de **targetObject** contenu dans le composite **ChainingArguments.traceInformation** est un préfixe du nom **aliasedEntryName**, aucun traitement spécifique de l'alias n'est requis étant donné que le sous-arbre d'alias sera recherché quoi qu'il arrive.

   NOTE – Dans les deux cas ci-dessus, **baseObject** peut ne pas être un préfixe de **targetObject** en raison du déréférencement d'alias.

4) si la recherche est effectuée dans une zone administrative de service et si le point administratif de service n'est pas un préfixe de **aliasedEntryName**, aucun traitement particulier de l'alias n'est nécessaire puisque l'entrée d'alias est en dehors de la zone administrative de service;

5) établir une demande de protocole **DSP** avec **targetObject** mis au nom **aliasedEntryName**. Si le paramètre **subset** est à **oneLevel**, mettre **entryOnly** à **TRUE**. Appeler l'aiguilleur d'opérations pour que la demande soit continuée localement;

6) si l'aiguilleur d'opérations retourne une erreur de renvoi de référence ou d'occupation ou d'indisponibilité, ajouter (ou constituer et ajouter) la référence de continuation au qualifiant **partialOutcomeQualifier** de **SearchResult**, puis retourner le résultat;

7) si l'aiguilleur d'opérations retourne d'autres erreurs, les ignorer et retourner le résultat;

8) si l'aiguilleur d'opérations retourne un résultat **SearchResult**:

   i) si ce résultat est signé, chiffré, ou signé et chiffré, l'ajouter aux informations **uncorrelatedSearchInfo** contenues dans le résultat **SearchResult**;

   ii) si ce résultat n'est ni signé, ni chiffré ni signé et chiffré, l'ajouter aux informations **searchInfo** contenues dans le résultat **SearchResult**;

   retourner l'ensemble.

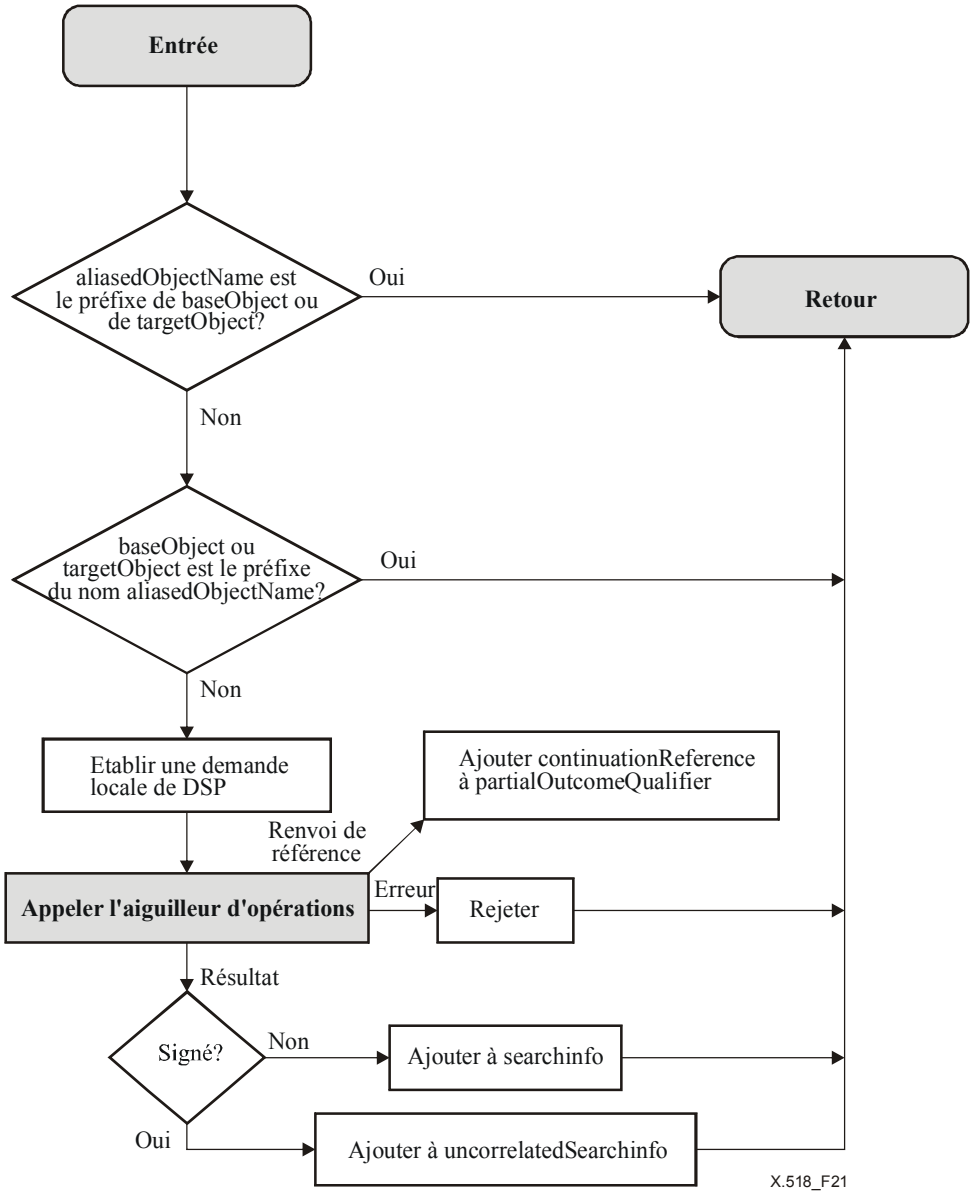


Figure 21 – Procédure Search alias

19.3.2.2.8 Procédure Hierarchy Selection (I)

Cette procédure est exécutée lorsqu'un membre d'un groupe hiérarchique a été rencontré au cours du traitement d'une demande de recherche spécifiant la sélection hiérarchique.

a) Si la sélection hiérarchique n'est pas prise en charge par le DSA, renvoyer:
   - une erreur **serviceError** avec la cause **requestedServiceNotAvailable**;
   - un attribut de notification **searchServiceProblem** avec la valeur **id-pr-unavailableHierarchySelect**;
   - un attribut de notification **serviceType** ayant comme valeur le composant **serviceType** de la règle de recherche;
   - un attribut de notification **hierarchySelectList** indiquant la ou les sélections non valables.

b) Dans les autres cas, ajouter toutes les entrées définies par la sélection hiérarchique comme il est indiqué au § 19.3.2.2.4. Si cela ne conduit pas à l'adjonction d'entrées, c'est-à-dire les sélections hiérarchiques n'indiquent que des entrées non existantes, fixer la variable globale **emptyHierarchySelect**.

20 Procédures de référence de continuation

Les procédures décrites dans le présent paragraphe sont appelées pour traiter la liste des références de continuation (**NRcontinuationList** ou **SRcontinuationList**) créée par d'autres procédures.

Les procédures de référence de continuation (**ContinuationReference**) comprennent les étapes présentées dans les Figures 24, 25 et 26. La première étape consiste à identifier, dans la liste de continuation, des ensembles de références de continuation qui ont un composant d'objet cible commun. Ces ensembles de références ont été créés à partir d'un ensemble de références subordonnées ou de références subordonnées non spécifiques associées à la même entrée dans le DIT. A l'intérieur de chacun de ces ensembles, il peut y avoir des références de continuation qui apparaissent plusieurs fois. Ces ensembles doivent être analysés et toute référence redondante trouvée doit être ignorée.

Ces ensembles (chacun avec un composant d'objet cible différent) peuvent être traités indépendamment, successivement ou parallèlement par le DSA, étant donné qu'il n'y a aucun risque que les mêmes résultats soient retournés par deux ensembles quelconques. Toutefois, le traitement de chaque référence de continuation dans un ensemble, de chaque information de point d'accès **AccessPointInformation** dans une référence de continuation et de chaque point d'accès dans une information de point d'accès **AccessPointInformation** doit être contrôlé; sinon on peut obtenir les résultats en double, comme décrit au § 20.1.

La procédure **APInfo** consiste à traiter un par un tous les points d'accès de l'ensemble contenu dans une information de point d'accès **AccessPointInformation** unique. Tous ces points désignent l'original (ou des copies) du même contexte de dénomination (ou éventuellement un ensemble de contextes de dénomination détenus par un même DSA dans le cas de NSSR). Si le premier point d'accès aboutit à un résultat ou à une erreur permanente, les autres points d'accès n'ont normalement pas besoin d'être traités. Toutefois, si l'erreur est une erreur temporaire, c'est-à-dire une erreur de service (**serviceError**) (**busy**, **unavailable**, **unwillingToPerform**, **invalidReference** ou **administrativeLimitExceeded**), le DSA peut choisir, à titre d'option locale, de traiter un autre point d'accès de l'ensemble.

Le traitement des valeurs **AccessPointInformation** dans un ensemble de références de continuation est effectué de façon uniforme, quelle que soit la référence de continuation dont elles proviennent. (Explication: deux DSE de type **subr** dépendant d'une entrée unique donneront deux références de continuation, chacune contenant une seule valeur **AccessPointInformation**, alors qu'une seule DSE de type référence **nssr** aux deux mêmes objets subordonnés (à supposer qu'ils soient détenus par des DSA différents) donnerait une référence de continuation contenant un ensemble de deux valeurs **AccessPointInformation**.)

Les valeurs de type **AccessPointInformation** peuvent être traitées successivement ou parallèlement, comme décrit au § 20.1. Il y a plus de risques d'obtenir des résultats en double avec la stratégie parallèle. Il faut toujours ignorer les résultats en double.

20.1 Stratégie de chaînage en présence de duplication miroir

En présence de duplication miroir, un DSA peut choisir entre différentes stratégies lorsqu'il doit multichaîner une demande à plusieurs DSA. Ce choix se présente toujours si le DSA doit traiter plusieurs références de continuation avec le même **targetObject**. Cette situation peut découler du multichaînage causé par décomposition de NSSR pendant la résolution du nom (voir Figure 22) ou d'une décomposition de demande pendant l'évaluation d'une opération portant sur plusieurs objets (Figure 23).

Le but de ces stratégies est de traiter le problème des résultats obtenus en double et du traitement en double lorsque des informations miroirs sont utilisées dans un multichaînage de demandes (causé par décomposition de NSSR ou de

demande). Par exemple dans la Figure 22, le DSA 1 multichaîne une demande à la fois vers les DSA 2 et 3 car la NSSR est détenue dans la DSE B. Si l'utilisation d'informations miroirs est autorisée, les DSA 2 et 3 peuvent appliquer l'opération chaînée aux deux sous-arbres commençant aux points X et Y.

De même, dans la Figure 23, le DSA 1 multichaîne une sous-demande (à la suite d'une décomposition de demande) aux deux références subordonnées détenues aux DSE X et Y. De nouveau, si l'utilisation d'informations miroirs est autorisée, les DSA 2 et 3 peuvent tous deux appliquer l'opération chaînée aux deux sous-arbres en commençant aux points X et Y.

Pour résoudre ce problème de duplication miroir, un DSA peut choisir l'une des stratégies suivantes en cas de multichaînage à des demandes de DSA multiples avec le même **targetObject**.

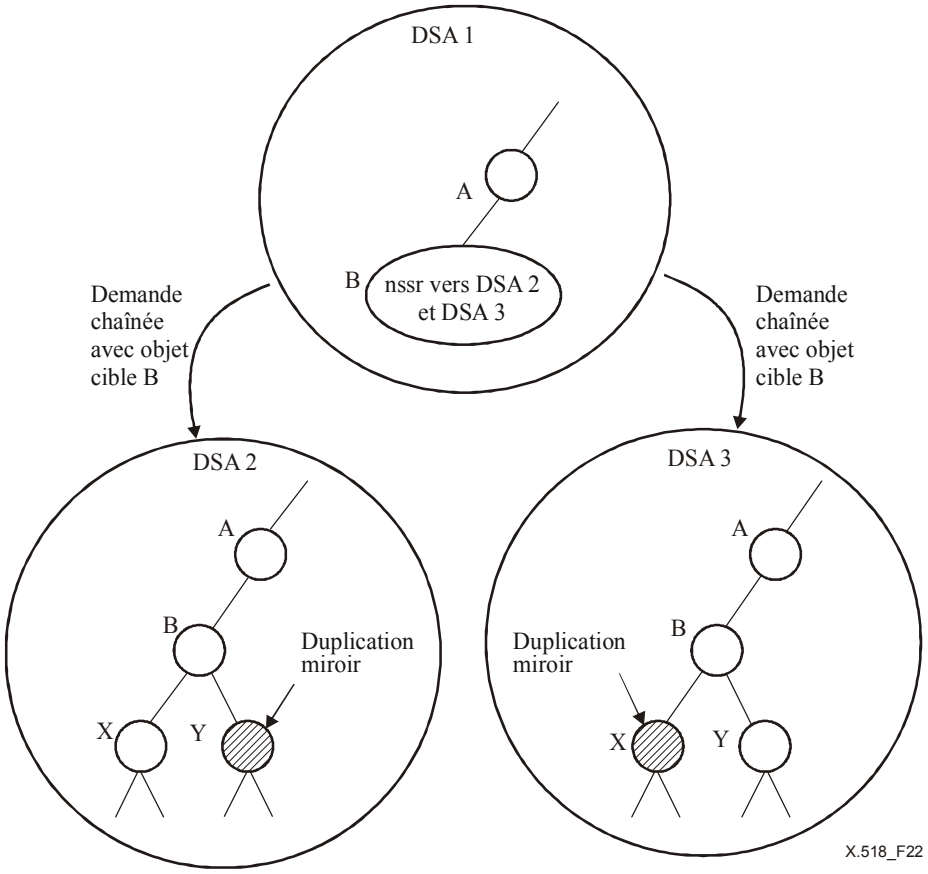


Figure 22 – Multichaînage causé par une NSSR pendant une résolution du nom

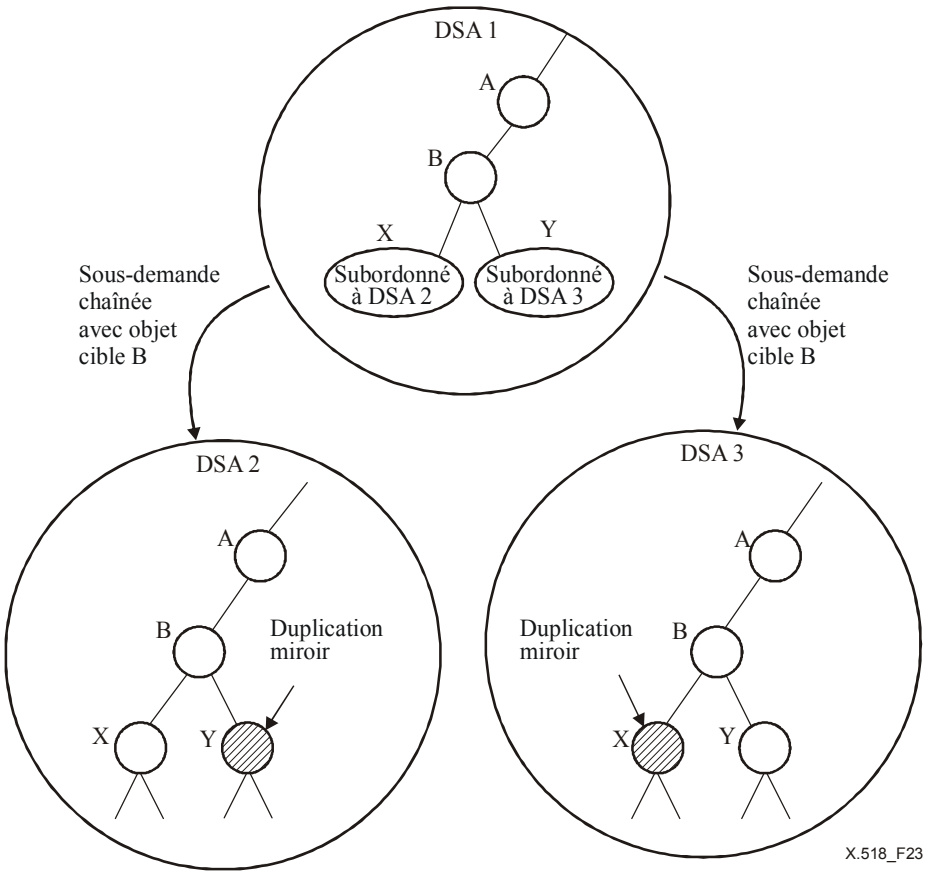


Figure 23 – Décomposition d'une demande de multichaînage à l'aide de références subordonnées

20.1.1 La stratégie maître seulement

Un DSA peut choisir cette stratégie pour éviter d'utiliser des informations miroirs lorsqu'il exécute un multichaînage parallèle ou séquentiel causé par une décomposition de NSSR ou par une décomposition de demande pendant une évaluation d'opération de recherche ou de liste. Pour cette stratégie, pendant une évaluation d'opération de recherche ou de liste, le composant **excludeShadows** de **ChainingArguments** est mis à **TRUE**. Si des NSSR sont rencontrées pendant la résolution du nom, un DSA peut mettre **nameResolveOnMaster** à **TRUE** pour s'assurer que seul un chemin unique est suivi. **nameResolveOnMaster** doit être mis à **TRUE** si des NSSR sont rencontrées et que l'opération soit l'une des opérations de modification de l'annuaire. Dans chaque cas, seuls le ou les DSA qui détiennent l'entrée (ou les entrées) (primaire) du maître concernant l'opération doivent effectuer cette opération. Cette stratégie maître seulement peut être utilisée pendant le multichaînage aussi bien parallèle que séquentiel.

NOTE – Le fait de mettre le composant **nameResolveOnMaster** à la valeur **TRUE** élimine la possibilité de chemins multiples au cours de la résolution du nom:

1) en ne tenant pas compte des entrées dupliquées et des copies programmables d'entrées;
2) en faisant en sorte qu'un seul agent DSA puisse procéder à la résolution du nom lorsque l'existence d'un arbre DIT de distribution complexe permettrait à plusieurs agents DSA de procéder à cette opération.

Pour cela, on ne permet de passer à la résolution du nom qu'à l'agent DSA qui détient l'entrée (primaire) maîtresse correspondant aux premiers noms RDN de type **nextRDNToBeResolved** du nom objet cible. Aucun autre agent DSA ne sera en mesure de procéder, même s'il possède des entrées maîtresses concordant avec un plus grand nombre d'éléments du nom objet cible.

20.1.2 La stratégie parallèle

Lorsqu'il utilise cette stratégie, un DSA envoie toutes les demandes chaînées sous forme de multichaînage parallèle. Cette stratégie peut être utilisée pendant l'évaluation de recherche ou de listage et la résolution du nom des NSSR. Cela permettra d'utiliser des informations miroirs pour traiter les demandes chaînées, mais cela risque d'entraîner une duplication des exécutions et des résultats pour l'opération. Si un DSA choisit cette stratégie, il devra supprimer du résultat de l'opération qu'il retourne les résultats obtenus en double.

Etant donné qu'il n'est pas possible de supprimer des résultats obtenus en double lorsqu'un résultat signé a été demandé, un DSA ne devra pas choisir cette stratégie si des résultats signés sont demandés pendant l'évaluation de la recherche, à moins que le composant **excludeShadows** ne soit aussi sélectionné.

20.1.3 La stratégie séquentielle

Cette stratégie évite d'obtenir des résultats en double grâce à l'utilisation du multichaînage séquentiel pour traiter les (sous-)demandes chaînées d'une décomposition de recherche ou de NSSR. Chaque demande chaînée est traitée séparément.

S'agissant de la décomposition d'une NSSR, si un résultat ou une erreur permanente fait suite à une demande, il n'est pas nécessaire de chaîner les autres demandes. Si une erreur temporaire est renvoyée, une demande ultérieure peut être chaînée, ou bien l'erreur temporaire peut être renvoyée au demandeur, selon la politique locale.

S'agissant d'une évaluation de recherche, le composant **exclusions** des **ChainingArguments** est mis sur l'ensemble des RDN qui ont déjà été traités. Pour cela, on incorpore les éléments de **ChainingResults.alreadySearched** dans l'argument **exclusions** de la demande chaînée suivante. C'est la seule stratégie qui évite complètement la duplication pendant l'évaluation de recherche.

Une stratégie séquentielle n'est pas définie pour l'évaluation de listage (bien que le multichaînage séquentiel puisse être utilisé), étant donné qu'un DSA supérieur ne peut pas empêcher que des subordonnés spécifiques soient renvoyés dans des sous-demandes de liste ultérieures (à noter que le composant **excludeShadows** n'exclut pas de subordonnés spécifiques mais est un moyen rapide d'exclure toutes les duplications miroirs et toutes les copies programmables).

20.2 Emission de sous-demandes chaînées vers un DSA distant

Avant l'émission d'une sous-demande, un DSA doit exécuter une opération de rattachement **dSABind** lorsqu'il doit établir une association avec le DSA distant. La gestion des associations n'entre pas dans le cadre des présentes Spécifications d'annuaire. Une association avec un autre DSA est considérée comme non disponible si l'association ne peut être établie ou si le DSA, pour des raisons locales, décide de ne pas établir d'association. En pareil cas, l'opération **dSABind** a échoué. C'est au niveau local que l'on décide du moment où l'on cesse d'essayer d'établir une association et où l'on déclare que l'association n'est pas disponible.

Lorsqu'un DSA essaie l'opération de rattachement **dSABind** pour établir une association avec un autre DSA et qu'il reçoit le message **directoryBindError**, l'émission de la sous-demande a échoué.

20.3 Paramètres des procédures

20.3.1 Arguments

Ces procédures utilisent les arguments suivants:

- la liste des références de continuation à traiter dans **NRcontinuationList** (pour la procédure **Name Resolution Continuation Reference**) et **SRcontinuationList** (respectivement pour la procédure **List Continuation Reference** et **Search Continuation Reference**);
- le composant **CommonArguments** de l'argument d'opération;
- le composant **ChainingArguments**.

20.3.2 Résultats

Ces procédures créent les résultats suivants:

- une liste des messages de résultats/erreurs reçus à la suite des demandes chaînées émises, si le chaînage a été choisi;
- une liste mise à jour des références de continuation non traitées, contenues dans **continuationList**.

20.3.3 Erreurs

Ces procédures peuvent retourner l'une des erreurs suivantes:

- **serviceError** avec la cause **outOfScope** si un renvoi de référence n'appartenant pas au **scopeOfReferral**, a été créé;
- **serviceError** avec la cause **ditError** lorsqu'une référence de connaissance non valide a été détectée;
- **nameError** avec la cause **noSuchObject** lorsque toutes les sous-demandes issues de la décomposition de NSSR ont retourné l'erreur **unableToProceed**;
- toute autre erreur qui est retournée par une sous-demande chaînée;
- **referral** lorsque le chaînage n'a pas été choisi et que **operationProgress.nameResolutionPhase** est mis à **notStarted** ou à **proceeding**.

20.4 Définition des procédures

Si le composant composite **operationProgress.nameResolutionPhase** est mis à **notStarted** ou à **proceeding**, la procédure décrite au § 20.4.1 (procédure **Name Resolution Continuation Reference**) doit être suivie. Les opérations d'interrogation à entrées multiples, listage et recherche, appellent respectivement les procédures décrites aux § 20.4.2 et 20.4.3.

20.4.1 Procédure de référence de continuation pour la résolution du nom

La procédure **Name Resolution Continuation Reference** comprend les étapes présentées à la Figure 24. Le principe de base de cette procédure consiste à traiter successivement l'ensemble des références de continuation créées pendant la résolution du nom. Les étapes suivantes doivent être exécutées pour chaque référence de continuation **C** contenue dans la **NRcontinuationList** dans un ordre donné jusqu'à ce que toutes les références aient été traitées ou qu'une erreur ou un résultat ait été retourné. Si toutes les références ont été traitées, retourner à **Operation Dispatcher** pour poursuivre la procédure **Result Merging** en vue de traiter le résultat ou le renvoi de référence reçu.

1) Vérifier si **chainingProhibited** est sélectionné. S'il est sélectionné, le DSA n'est pas autorisé à chaîner. Selon la politique locale (**local policy**), une **serviceError** avec la cause **chainingRequired** ou un **referral** est retourné à **Operation Dispatcher**.

2) Si **chainingProhibited** n'est pas sélectionné, vérifier si la politique locale (**local policy**) permet le chaînage. Si le chaînage n'est pas autorisé, retourner un **referral**. Si la politique locale (**local policy**) permet le chaînage, passer à l'étape suivante.

3) Traiter chacune des références de continuation de la liste des références de continuation figurant dans **NRcontinuationList**. S'il n'y a plus de références de continuation non traitées, retourner un message **serviceError**.

4) Traiter la référence de continuation **C** suivante de la **NRcontinuationList**. S'il s'agit d'une NSSR, passer à l'étape 5). S'il ne s'agit pas d'une NSSR, appeler la procédure **APInfo** pour la traiter. Il convient de faire une distinction entre les retours possibles par la procédure **APInfo**:

   – si la procédure **APInfo** retourne **null result**, revenir à l'étape 3) et traiter la référence de continuation suivante;

   – si la procédure **APInfo** retourne **error**, **referral** ou **result**, le renvoyer.

5) En pareil cas, la référence de continuation est de type NSSR et le DSA a le choix entre un chaînage séquentiel ou parallèle, selon le choix local de stratégie (**local choice of strategy**). Si la NSSR doit être **processed sequentially**, passer à l'étape 6). Si elle doit être **processed in parallel**, alors pour chaque **AccessPointInformation (API)** de la NSSR, la procédure **APInfo** est appelée pour que ces informations soient traitées en parallèle. Attendre que toutes les API soient traitées, c'est-à-dire attendre que tous les appels à la procédure **APInfo** soient retournés. Vérifier tous les résultats provenant de ces appels dans l'ordre suivant:

   – si tous les appels retournent **serviceError** avec la cause **unableToProceed** et si **partialNameResolution** a la valeur **FALSE**, retourner **nameError**;

   – si tous les appels retournent **serviceError** avec la cause **unableToProceed** et si **partialNameResolution** a la valeur **TRUE**, alors mettre dans le résultat **partialName** à **TRUE**, **nameResolutionPhase** à **completed**, positionner **entry suitable** (relativement au **lastEntryFound**) et aller à l'opération d'évaluation qui convient;

   – si l'on reçoit un ou plusieurs **results**, **ignorer les duplications éventuelles** et retourner le **result**;

   – si l'on reçoit une **error** et qu'il n'est pas une **serviceError** (par exemple **nameError**), retourner **error**;

   – dans les autres cas, retourner **referral** ou **serviceError** vers **Operation Dispatcher**, selon le choix local.

6) Choisir l'**API** non traitée suivante parmi l'ensemble des **API** de la NSSR et passer à l'étape 7). Si toutes les **API** ont été traitées, vérifier que tous les appels à la procédure **APInfo** ont retourné le message **serviceError** avec la cause **unableToProceed**.

   – Si tel est le cas et si **partialNameResolution** a la valeur **FALSE**, alors l'entrée ne peut pas être trouvée et **nameError** est retourné; si tel est le cas et si **partialNameResolution** a la valeur **TRUE**, alors mettre dans le résultat **partialName** à **TRUE**, **nameResolutionPhase** à **completed**, positionner **entry suitable** (relativement au **lastEntryFound**) et aller à l'opération d'évaluation qui convient. Sinon, selon le choix local, retourner avec **referral** ou **serviceError**.

7) Appeler la procédure **APInfo**. Il convient de faire une distinction entre les résultats possibles des appels à la procédure **APInfo**:

   – si l'on reçoit **serviceError** avec la cause **unableToProceed**, essayer un autre point d'accès. Revenir à l'étape 6);

   – si l'on reçoit **serviceError** avec la cause **busy**, **unavailable**, **unwillingToPerform** ou **invalidReference**, la cause indiquée peut être de nature transitoire et c'est au niveau local que l'on choisit d'essayer de chaîner la demande vers un autre DSA. Si l'on choisit d'essayer un autre DSA, revenir à l'étape 6); sinon, retourner **referral** ou **serviceError**, selon le choix local;

   – si l'on reçoit une erreur autre que **serviceError** avec la cause **busy**, **unavailable**, **unwillingToPerform**, **invalidReference** ou **unableToProceed**, cette erreur doit être retournée à **Operation Dispatcher**. Si **serviceError** est **invalidReference**, une conversion en **ditError** est nécessaire avant le renvoi au demandeur;

   – si l'on reçoit **result** ou **referral**, le renvoyer à **Operation Dispatcher**.

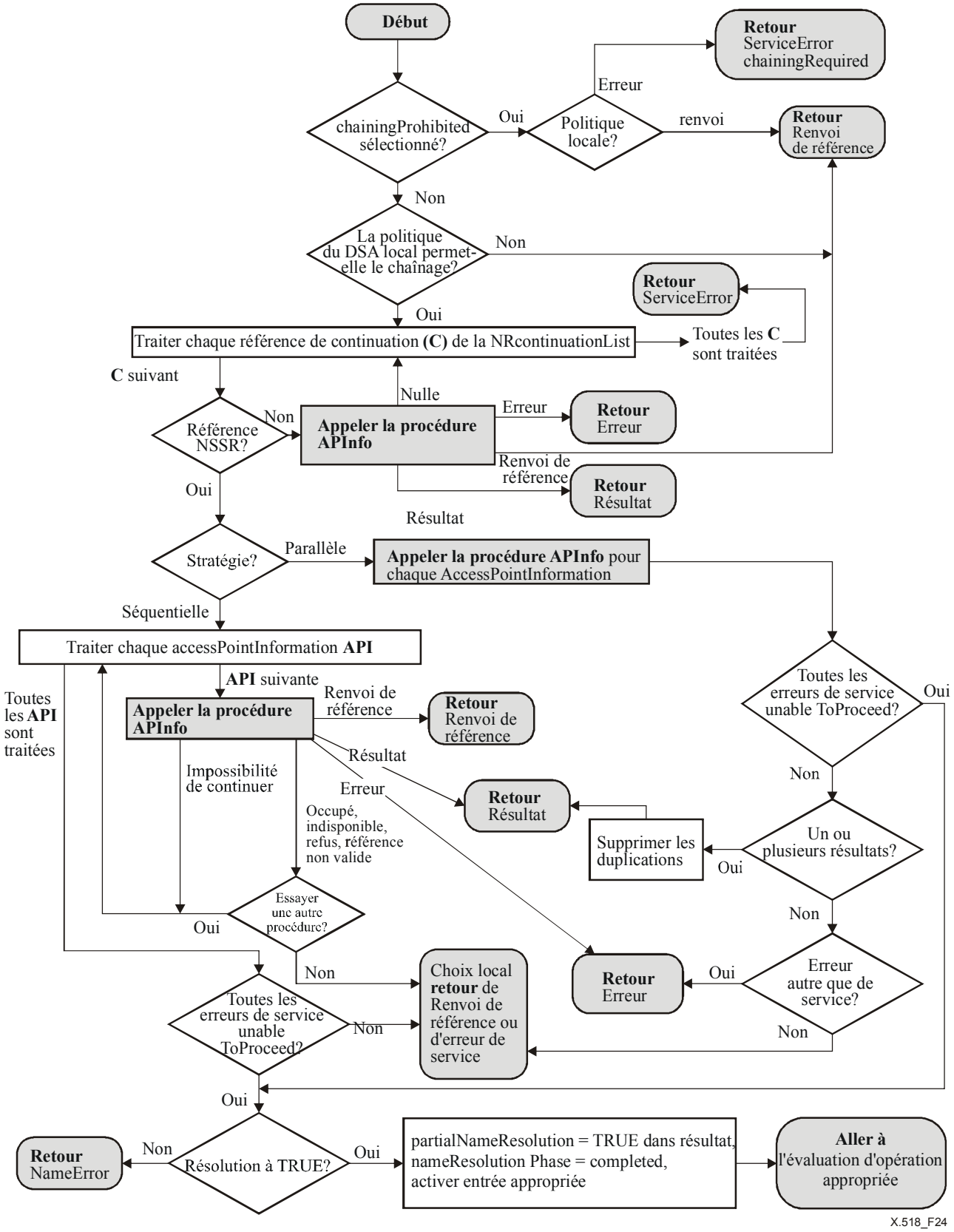


Figure 24 – Procédure de référence de continuation pour la résolution du nom

20.4.2 Procédure de référence de continuation pour le listage

La procédure **List Continuation Reference** comprend les étapes indiquées à la Figure 25. Elle est invoquée lorsqu'une demande de listage ne peut être satisfaite dans le DSA local et qu'un ensemble de références de continuation a été ajouté à **SRcontinuationList** pour le chaînage ou le renvoi de référence. Toutes ces références de continuation (CR) ont le même **targetObject**. Les CR avec **referenceType nssr** ont une ou plusieurs valeurs **AccessPointInformation** (**API**), alors que d'autres types de CR ont une seule **API**. Chacune de ces **API** est extraite et examinée en vue du chaînage ou du renvoi de référence.

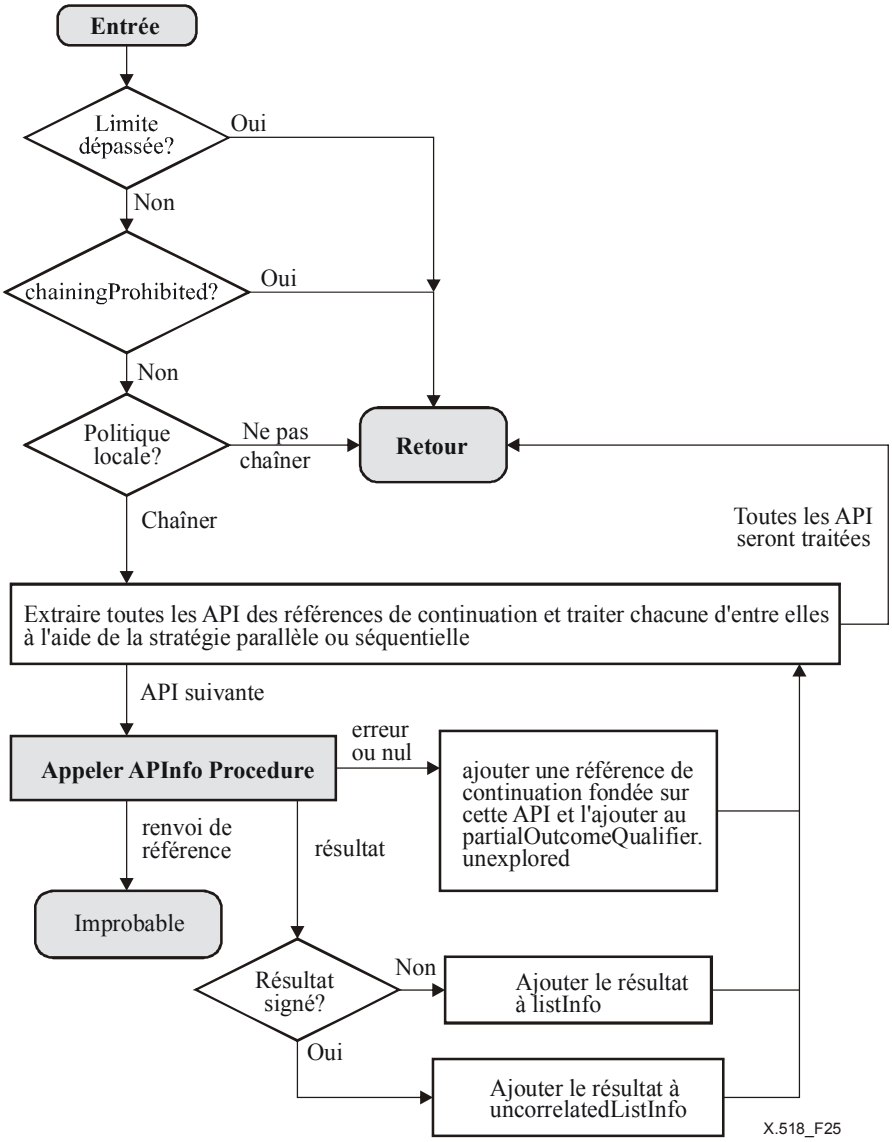


Figure 25 – Procédure de référence de continuation pour le listage

Il convient de suivre les étapes suivantes:

1) si une quelconque cause d'erreur par dépassement de limite a été détectée d'ici là, revenir à l'aiguilleur **Operation Dispatcher** pour poursuivre la procédure **Result Merging**;
2) si le fanion **chainingProhibited** dans **CommonArguments.serviceControls** est sélectionné ou si le DSA décide de ne pas faire de chaînage en raison de sa politique d'exploitation locale, le DSA doit revenir directement à l'aiguilleur **Operation Dispatcher** pour poursuivre la procédure de fusionnement **Result Merging**;
3) créer un ensemble de valeurs **AccessPointInformation** à partir du composant **accessPoints** de chaque référence de continuation de la **SRcontinuationList**.

Utiliser la stratégie parallèle ou séquentielle pour traiter chacune des **API** de la façon suivante:

i) appeler la procédure **APInfo** avec l'information **API** suivante de l'ensemble;

ii) si un résultat est retourné, l'ajouter à **listInfo** s'il n'est pas signé ou l'ajouter à **uncorrelatedListInfo** s'il est signé;

iii) si le retour est une erreur ou un résultat nul, cela signifie que la procédure **APInfo** a déjà essayé tous les points d'accès de l'**API** sans succès. Selon la politique de sécurité et d'exploitation locale, ne pas en tenir compte et passer à l'**API** suivante ou ajouter une référence de continuation fondée sur cette **API** à **partialOutcomeQualifier**;

NOTE – Il n'est pas plausible d'obtenir un renvoi de référence par la procédure **APInfo**. Tout "renvoi de référence" devrait se présenter sous la forme **unexplored** dans le qualificateur **partialOutcomeQualifier**.

4) lorsque toutes les **API** ont été traitées, revenir à l'aiguilleur **Operation Dispatcher**.

20.4.3 Procédure de référence de continuation pour la recherche

La procédure **Search Continuation Reference** comprend les étapes indiquées à la Figure 26. Elle est invoquée lorsqu'une demande de recherche ne peut être satisfaite dans le DSA local et qu'un ensemble de références de continuation a été ajouté à **SRcontinuationList** pour le chaînage ou le renvoi de référence. La procédure ressemble beaucoup à la procédure **List Continuation Reference**. La différence est que, dans ce cas, les références de continuation figurant dans **SRcontinuationList** peuvent avoir différentes valeurs du **targetObject**. En conséquence, les références de continuation sont classées en ensembles de références de continuation ayant le même **targetObject**. On définit aussi l'utilisation du paramètre **exclusions** dans les arguments de chaînage et du paramètre **alreadySearched** dans les résultats de chaînage, car il s'agit d'une importante stratégie pour la recherche. L'utilisation de **exclusions** et **alreadySearched** s'applique au traitement de chaque ensemble de références de continuation ayant le même **targetObject**.

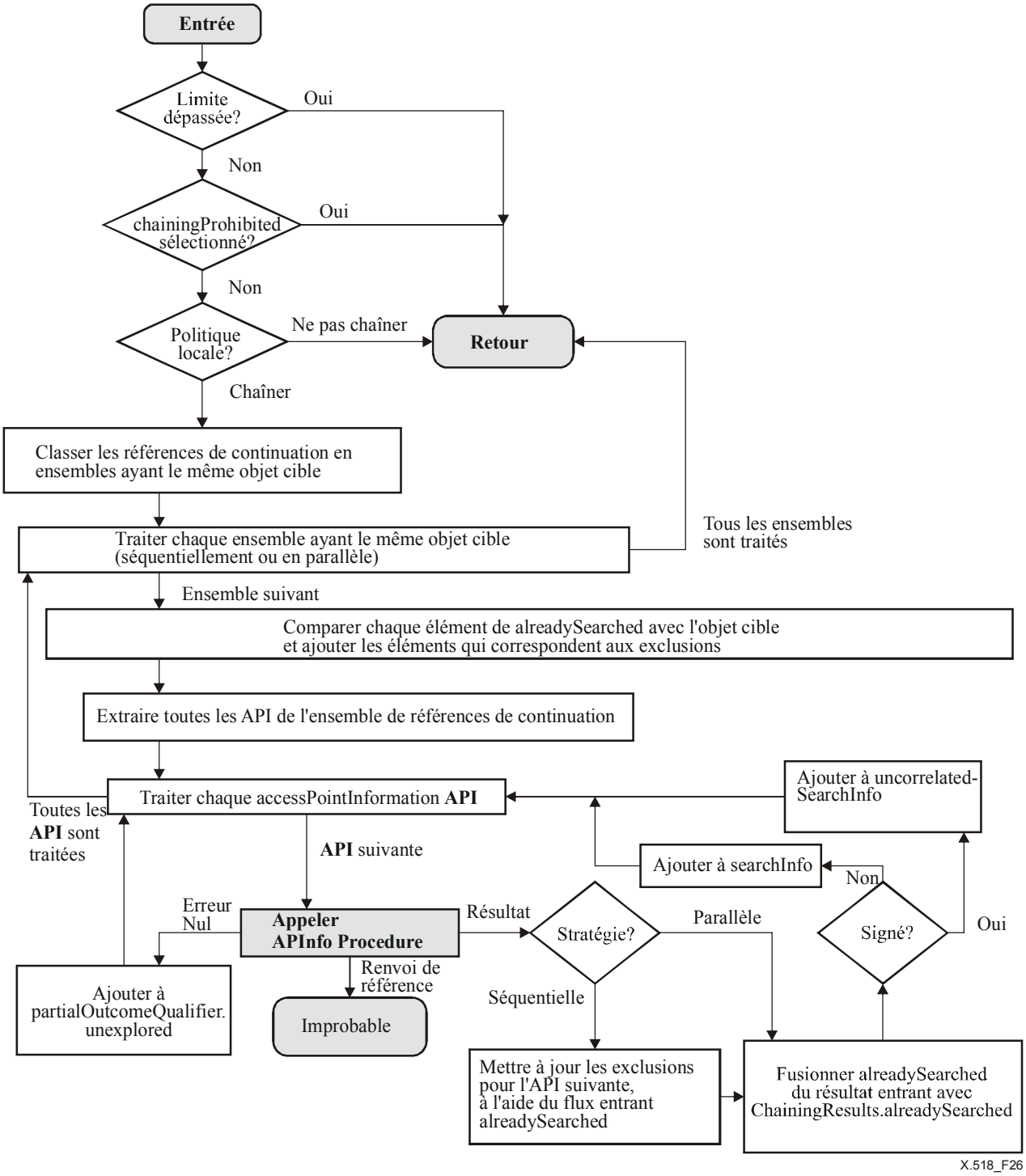


Figure 26 – Procédure de référence de continuation pour la recherche

Il convient de suivre les étapes suivantes:

1) si la limite a été dépassée, revenir à l'aiguilleur **Operation Dispatcher** pour passer à la procédure **Result Merging**;

2) si le fanion **chainingProhibited** dans **CommonArguments.serviceControls** est sélectionné ou si le DSA décide de ne pas effectuer de chaînage en raison de sa politique d'exploitation locale, le DSA reviendra directement à l'aiguilleur **Operation Dispatcher** pour passer à la procédure **Result Merging**;

3) classer les références de continuation figurant dans **SRcontinuationList** en ensembles ayant le même **targetObject**. Les références de continuation du type **ditBridge** ne figurent dans aucun de ces ensembles, mais chacune d'elles constitue un ensemble particulier. A l'intérieur de chaque ensemble, supprimer toutes les redondances.

   NOTE 1 – Si l'une au moins des valeurs de **targetObject** n'est pas un RDN primaire, ce classement risque d'être incorrect. Le classement doit prendre en compte les variantes de RDN distinctifs lorsqu'elles sont connues.

4) pour chaque sous-ensemble de références de continuation, créer un ensemble de valeurs **AccessPointInformation** à partir du composant **accessPoints** de chaque référence de continuation du sous-ensemble et choisir la stratégie séquentielle ou parallèle pour poursuivre le traitement. Si l'on choisit la stratégie parallèle, sauter les étapes indiquées ci-dessous, qui ne s'appliquent qu'à la stratégie séquentielle.

   a) Si l'on choisit la stratégie séquentielle, enregistrer une variable locale **localExclusions** pour chaque ensemble de références de continuation ayant le même **targetObject**. Au départ, la variable **localExclusions** est mise sur les **exclusions** de la demande de chaînage entrante (le cas échéant) et de tous les sous-arbres recherchés au niveau local qui dépendent directement du **targetObject**.

   b) Si l'on utilise la stratégie séquentielle, comparer le **targetObject** à tous les éléments de la variable **localExclusions** et supprimer les éléments qui ne contiennent pas comme préfixe **targetObject**. On obtient ainsi les exclusions pertinentes pour l'objet cible actuel.

   c) Extraire toutes les **API** de l'ensemble des références de continuation visant l'objet cible actuel.

   d) Effectuer l'itération de chaque **API**. Pour chaque **API**:

      i) appeler la procédure **APInfo**;

      ii) si un résultat est retourné, ajouter le résultat au composant **searchInfo** s'il n'est pas signé, ou l'ajouter au composant **uncorrelatedSearchInfo** s'il est signé. Si l'on utilise la stratégie séquentielle, mettre à jour la variable **localExclusions** à l'aide de l'élément **alreadySearched** dans la réponse entrante et fusionner **alreadySearched** avec la réponse entrant au composant **ChainingResults.alreadySearched** du DSA. Passer ensuite à l'**API** suivante;

      iii) si une erreur ou un résultat nul est retourné, cela signifie que la procédure **APInfo** a déjà essayé tous les points d'accès de l'**API** sans succès. Selon la politique de sécurité et d'exploitation locale, ne pas en tenir compte et passer à l'**API** suivante ou ajouter une référence de continuation fondée sur cette **API** à **partialOutcomeQualifier**.

         NOTE 2 – Il n'est pas plausible d'obtenir un renvoi de référence par la procédure **APInfo**. Tout "renvoi de référence" devrait se présenter sous la forme de l'élément **unexplored** dans le **partialOutcomeQualifier**.

   e) Lorsque toutes les **API** ont été traitées, passer à l'ensemble suivant de références de continuation ayant le même **targetObject**;

5) lorsque toutes les références de continuation sont traitées, revenir à l'aiguilleur **Operation Dispatcher**.

20.4.4 Procédure APInfo

Cette procédure est appelée pour traiter une API (**AccessPointInformation**), qui contient un ou plusieurs points d'accès (Figure 27). Les points sont traités un par un jusqu'à ce qu'un résultat ou une erreur soit retourné. Si l'erreur est une erreur de service qui permette d'essayer un autre point avec succès, on essaie d'autres points d'accès dans la mesure où la politique d'exploitation locale le permet:

1) procéder à la détection de boucle. Si une boucle est détectée, retourner le message **serviceError** avec la cause **loopDetected**. Sinon, passer à l'étape 2);

2) traiter chacun des points d'accès (AP) de l'information de point d'accès (API). Si tous les points ont été traités, retourner **null result**. S'il y a un point d'accès à traiter, passer à l'étape 3);

3) vérifier si la politique locale permet le chaînage à ce point d'accès. Cette vérification doit tenir compte des réglages des commandes de service et des arguments de chaînage (par exemple: **chainingProhibited**, **preferChaining**, que le point d'accès soit ou non dans **localScope**, **excludeShadows**). Si la politique locale ou le réglage des commandes de service respectives ne permet pas d'utiliser ce point d'accès donné, ne pas tenir compte du point d'accès et passer à l'étape 2). Si le point d'accès peut être utilisé, passer à l'étape 4);

4) si la politique locale a conduit à choisir la stratégie maître seulement, mettre l'argument de chaînage **excludeShadows** à **TRUE**;

   si **nameResolutionPhase** n'est pas **completed** et si la stratégie doit continuer la résolution du nom au sujet des entrées du maître, mettre **nameResolveOnMaster** sur **TRUE**.;

   l'argument de chaînage **nameResolveOnMaster** est mis sur **TRUE** si l'un des cas suivants se présente:

   – dans l'argument de chaînage entrant, **nameResolutionPhase** se poursuit et **nameResolveOnMaster** est mis sur **TRUE**;

   – l'opération est l'une des opérations de modification, le **referenceType** de la demande de chaînage à émettre est une NSSR, et l'on utilise une stratégie parallèle.

     NOTE – L'utilisation du composant **nameResolveOnMaster** vise à empêcher d'effectuer plusieurs fois les opérations de modification en raison de la présence d'une NSSR.

5) établir une demande chaînée et essayer de l'émettre:

   a) appliquer la procédure de prévention de boucle en vérifiant si un item ayant le même **targetObject** et **operationProgress** apparaît dans **traceInformation** du **ChainingArguments** reçu. Si la demande qui en résulte [comme décrit à l'étape 5) c)] aboutit à une boucle, le DSA doit retourner le message **serviceError** avec la cause **loopDetected** au DUA/client LDAP/DSA demandeur ou ne pas tenir compte du point d'accès et essayer le point d'accès suivant en revenant à l'étape 2);

   b) si la demande ou sous-demande à chaîner est le résultat de l'exécution d'un renvoi de référence, il est alors nécessaire d'effectuer une vérification supplémentaire pour éviter une boucle. On vérifie si le composant **referralRequests** contient un item ayant le même **targetObject**, la même **operationProgress** et le même agent DSA cible. Si c'est le cas, effectuer l'action spécifiée en a); si ce n'est pas le cas, ajouter un nouvel élément **TraceItem** au composant **referralRequests**, avec les composants suivants:

      – **targetObject** et **operationProgress** mis à la valeur de la demande/sous-demande chaînée;

      – **dsa** mis au nom de l'agent DSA avec lequel la demande/sous-demande doit être chaînée;

   c) à la suite d'un rattachement d'association réussi, le DSA doit émettre une opération chaînée du même type que l'opération qui est traitée avec les paramètres suivants:

      – l'argument d'opération dans l'opération chaînée, mis sur la même valeur que l'argument d'opération reçu;

      – **ChainingArguments.originator**, mis à la valeur reçue;

      – **ChainingArguments.targetObject** mis au **targetObject** de la référence de continuation;

      – **ChainingArguments.operationProgress** mis à la valeur de **operationProgress** de la référence de continuation;

      – **ChainingArguments.traceInformation** mis à l'information de trace, telle qu'elle est mise à jour par la procédure **Request Validation**, si la référence de continuation n'est pas du type **ditBridge**; sinon le composant est absent;

      – **ChainingArguments.aliasDereferenced** mis à la valeur actualisée de **aliasDereferenced** mis à jour au niveau local;

      – **ChainingArguments.returnCrossRefs** mis à choix local;

      – **ChainingArguments.referenceType** mis à la valeur du **referenceType** de la référence de continuation;

      – **ChainingArguments.timeLimit** mis à la valeur de **timeLimit** reçue;

      – **ChainingArguments.exclusions** soit mis aux exclusions pertinentes pour l'objet cible établi lors d'un appel par la procédure de référence de continuation pour la recherche, soit absent si la procédure **APInfo** a été appelée par les procédures de continuation pour la résolution du nom et pour le listage;

      – **SecurityParameters** mis à la valeur des **SecurityParameters** reçus.

6) si la demande n'a pas pu aboutir, passer à l'étape 7); si elle a pu aboutir, passer à l'étape 8);

7) on décide au niveau local de continuer ou pas. Si le DSA choisit de continuer, on ne tient pas compte de l'erreur et l'on essaie le point d'accès suivant. Revenir à l'étape 2). Si le DSA décide de ne pas essayer d'autre point d'accès, on décidera, selon la politique locale, de retourner ou non un message **referral** ou **serviceError**, selon le cas, au demandeur de la procédure;

8) si la demande a pu aboutir, le DSA attend la réponse et la traite:

   a) si l'on reçoit un résultat **result**, celui-ci est retourné au demandeur de la procédure;

   b) si l'on reçoit l'erreur **serviceError** avec la cause **busy**, **unavailable**, **unwillingToPerform** ou **invalidReference**, revenir à l'étape 7);

   c) si l'on reçoit le renvoi **referral** et que **returnToDUA** est mis sur **TRUE**, le DSA qui reçoit ne donne pas suite au **referral**, mais le retourne au demandeur;

   d) si l'on reçoit le renvoi **referral** et que **returnToDUA** est mis sur **FALSE**, il convient d'appliquer les mêmes considérations de politique locale qu'à l'étape 3) (compte tenu des commandes de service, des arguments de chaînage, de la stratégie de chaînage, etc.). Si l'on décide de ne pas déréférencer le renvoi **referral**, retourner celui-ci au demandeur. S'il est décidé de référencer le renvoi **referral**, vider la liste **NRcontinuationList**, placer la référence de continuation (telle qu'elle a été reçue dans la liste renvoi **referral**) dans **NRcontinuationList** et appeler la procédure **Name Resolution Continuation Reference**. Cela peut aboutir à un **result**, un **referral**, une **serviceError** ou à une autre **error**. Quel que soit le résultat de l'appel de la procédure **Name Resolution Continuation Reference**, ce résultat sera communiqué au demandeur;

   e) si une autre **error** se produit, elle sera renvoyée au demandeur.

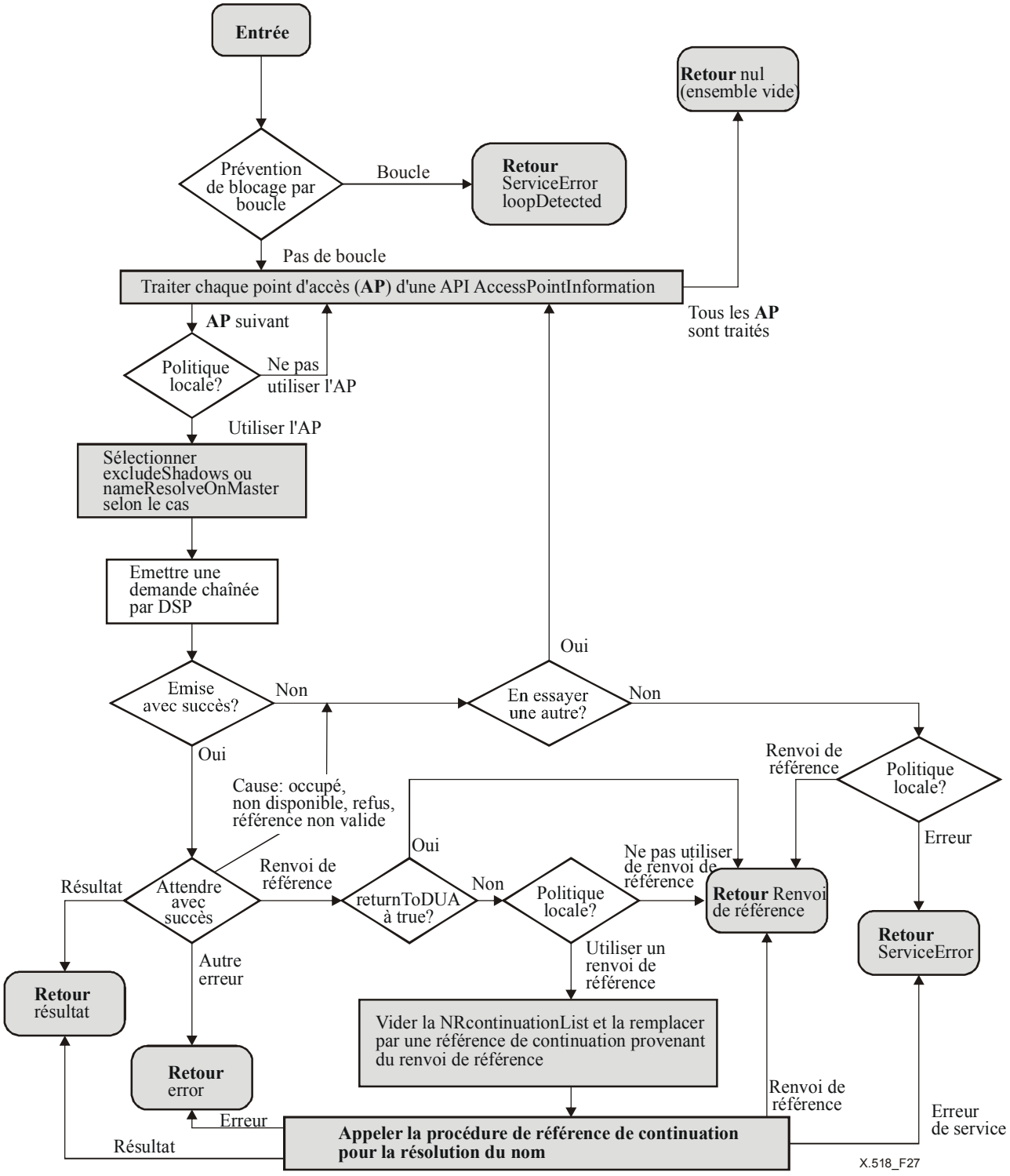


Figure 27 – Procédure APInfo

20.5 Procédure d'abandon

Cette procédure est invoquée si l'on reçoit une demande d'abandon. Elle comprend les étapes suivantes, indiquées à la Figure 28:

1) lorsque l'on reçoit une demande d'abandon qui se réfère à une opération inconnue, il faut retourner au demandeur une erreur **abandonFailed** avec une valeur **noSuchOperation**;

2) s'il a déjà été répondu à la demande d'abandon et si le DSA a gardé les informations permettant de le savoir, une erreur **abandonError** avec un motif **tooLate** peut être retournée au demandeur;

3) si la demande d'abandon n'est pas valide, c'est-à-dire si elle vise à abandonner une demande qui n'est pas une demande d'interrogation, il faut retourner au demandeur un message d'erreur **abandonFailed** avec la cause **cannotAbandon** (abandon impossible);

4) si un DSA a des (sous-)demandes chaînées en suspens lorsqu'il reçoit une demande d'abandon valide pour la demande originale, et s'il décide d'essayer d'abandonner, il peut envoyer des demandes d'abandon pour certaines ou toutes les (sous-)demandes en suspens pour l'opération en question (ou pour aucune) puis attendre les réponses à la demande d'abandon et aux (sous-)demandes en suspens. A tout moment au cours de cette opération, le DSA peut envoyer un résultat d'abandon et une erreur **abandonFailed** au demandeur puis supprimer les réponses aux demandes d'abandon émises et aux (sous-)demandes en suspens au fur et à mesure qu'elles arrivent;

   si le DSA décide de ne pas envoyer de réponses au demandeur jusqu'à ce qu'il n'y ait plus de (sous-)demandes en suspens, il peut, à titre facultatif, envoyer une erreur **abandonedFailed** au demandeur s'il a été répondu à toutes les demandes d'**abandon** émises avec des erreurs **abandonedFailed** et si aucune opération d'abandon locale n'a été effectuée;

   si une erreur **abandonedFailed** est retournée au demandeur, la demande originale doit être traitée comme si la demande d'abandon n'avait jamais été reçue.

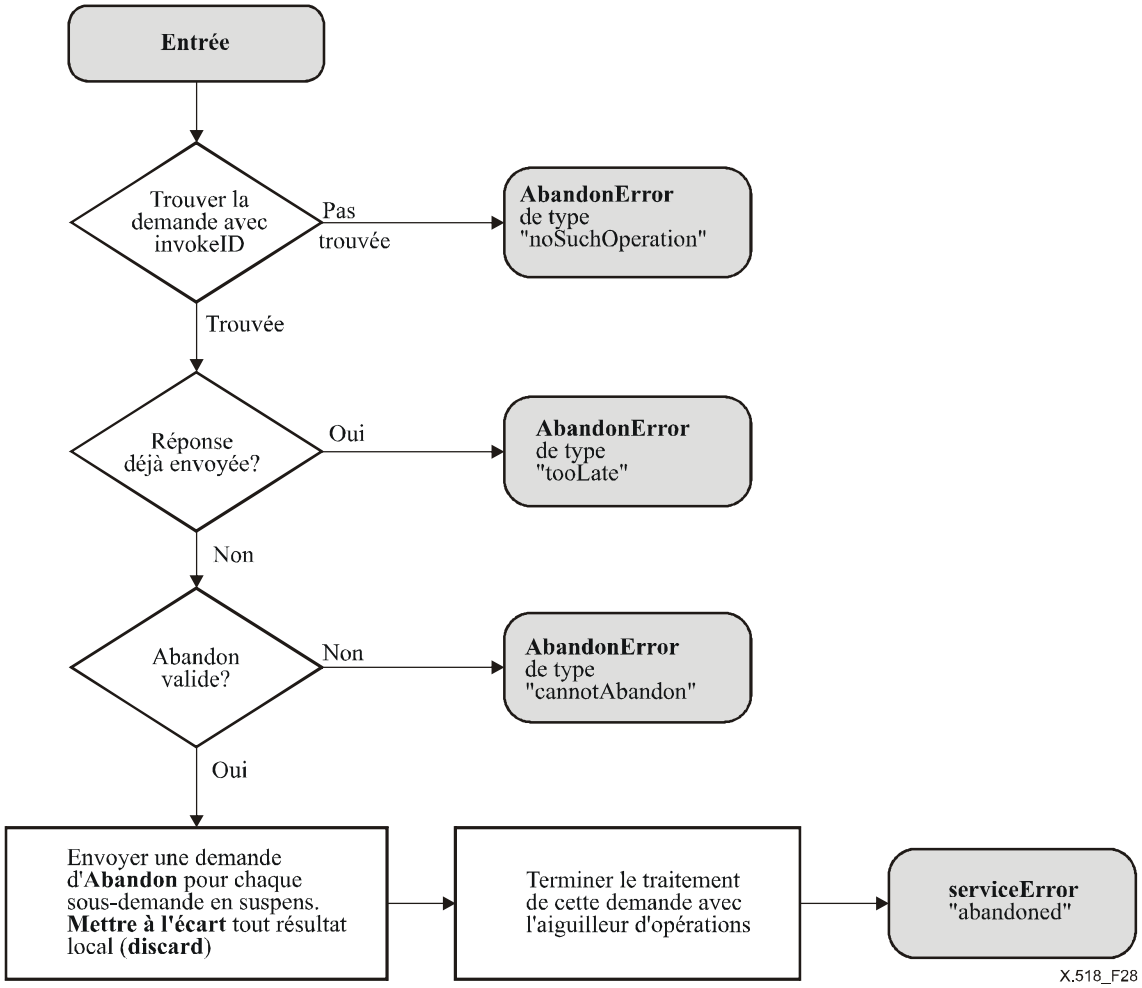


Figure 28 – Procédure d'abandon

21 Procédure de fusionnement des résultats

La procédure de fusionnement **Result Merging** présentée à la Figure 29 est appelée à la suite d'une des procédures **Continuation Reference**. Cette procédure supprime les duplications si le résultat n'est pas signé et s'il y a des références de continuation supplémentaires dans l'élément **partialOutcomeQualifier.unexplored**. Puis la ou les procédures **Continuation Reference** pertinentes sont appelées si la politique d'exploitation locale le permet:

1) si l'opération est une opération de listage, passer à l'étape 2); si l'opération est une opération de recherche, passer à l'étape 3); sinon, retourner le résultat qui a été fourni comme paramètre d'entrée dans la procédure **Result Merging**;

2) l'opération est une opération de listage. Supprimer toutes les duplications, en donnant la priorité aux informations maîtresses par rapport aux informations miroirs;

   si le résultat de l'opération a été émis au niveau local et s'il contient des références de continuation, celles-ci ne seront pas utilisées pour le chaînage mais fournies à l'utilisateur. En pareil cas, passer à l'étape 6);

   si le résultat de l'opération a été reçu à l'issue d'une opération de listage chaînée Chained List, il peut contenir des références de continuation. En pareil cas, vérifier si la commande de service **preferChaining** a été sélectionnée. Si elle est mise à **TRUE**, les références de continuation devraient être utilisées pour le chaînage par le DSA. Passer à l'étape 4);

3) l'opération est une opération de recherche. Supprimer toutes les duplications, en donnant la priorité aux informations maîtresses par rapport aux informations miroirs. S'il y a un problème de limite, retourner le résultat. Sinon, revenir à l'étape 4);

4) traiter chaque référence de continuation figurant dans l'élément **partialOutcomeQualifier.unexplored** du résultat de chaque opération chaînée. Si la politique locale décide de ne pas l'utiliser pour le chaînage, ne pas tenir compte de cette référence de continuation et en choisir une autre. Si la politique locale permet de l'utiliser pour le chaînage, procéder comme suit:

   vérifier la phase de résolution **nameResolutionPhase** fournie dans la référence de continuation. Si elle est **notStarted** ou **proceeding**, l'ajouter à la liste des références de continuation qui sera fournie à la procédure **Name Resolution Continuation** (**NRcontinuationList**). Si la phase **nameResolutionPhase** est **completed**, ajouter la référence de continuation à la liste des références de continuation qui est fournie à la procédure de **Continuation** de sous-demande (**SRcontinuationList**);

   continuer jusqu'à ce que toutes les références de continuation aient été traitées;

5) si des références de continuation doivent être traitées dans **SRcontinuationList**, vérifier l'opération. Si l'opération est une opération de listage, appeler la procédure **List Continuation Reference** et revenir à l'étape 2). Si l'opération est une opération de recherche, appeler la procédure **Search Continuation Reference** et revenir à l'étape 3);

   si **SRcontinuationList** est à "empty", vérifier s'il y a des références de continuation dans **NRcontinuationList**. Dans l'affirmative, appeler la procédure **Name Resolution Continuation Reference** et passer à l'étape 3);

   si les deux listes de continuation sont à "empty", passer à l'étape 6);

6) vérifier si le résultat a la valeur "empty". Si ce n'est pas le cas, le retourner. Si c'est le cas, retourner un **résultat "null"** si la commande d'accès et la politique locale le permettent, ou retourner une erreur appropriée.

Si un DSA reçoit des résultats de recherche ou de liste d'autres DSA et que ces résultats contiennent des paramètres inconnus du DSA, les résultats sans corrélation sont retournés. Dans le cas contraire, le DSA effectue le fusionnement si les résultats des recherches ne sont pas signés ou si le DSA est un exécuteur initial habilité à retirer les signatures (voir § 7.9 de la Rec. UIT-T X.511 | ISO/CEI 9594-3).

Un DSA ayant reçu des résultats non signés et non corrélés d'un DSA qui n'est pas en mesure d'en effectuer le regroupement doit effectuer un fusionnement s'il possède les connaissances appropriées relatives à tous les paramètres des résultats non corrélés.

Si un DSA reçoit des résultats non signés d'autres DSA, que, éventuellement, il possède aussi un résultat local, en générant un compte d'entrées à renvoyer dans le composant **entryCount** du qualificateur **PartialOutcomeQualifier** généré par le DSA, le DSA doit prendre la somme de toutes les valeurs **entryCount** reçues, le résultat local et le nombre d'entrées reçues des DSA qui n'ont pas renvoyé une valeur **entryCount**, puis compenser en fonction des entrées dupliquées. Si le DSA est l'exécuteur initial et si des résultats paginés ont été demandés, il doit alors inclure les comptes d'entrées relatifs aux résultats signés d'autres DSA.

En cas de demande de resultats paginés et si aucun DSA n'a eu de problème de limite, il doit alors opter pour la valeur **exact** du paramètre **entryCount**. La même valeur est attribuée à chaque page renvoyée.

Si un ou plusieurs DSA ont eu un problème de limite, alors:

– si tous les DSA qui ont eu un problème de limite ont renvoyé un composant **entryCount** comportant le choix de la valeur **exact** ou **bestEstimate**, il doit retenir la valeur **bestEstimate** si un seul DSA a fait ce choix; sinon il doit choisir la valeur **exact** ;

– si un seul des DSA qui ont eu un problème de limite a renvoyé un composant **entryCount** comportant le choix de la valeur **lowEstimate** ou n'a pas renvoyé de composant **entryCount**, il doit choisir la valeur **lowEstimate**.

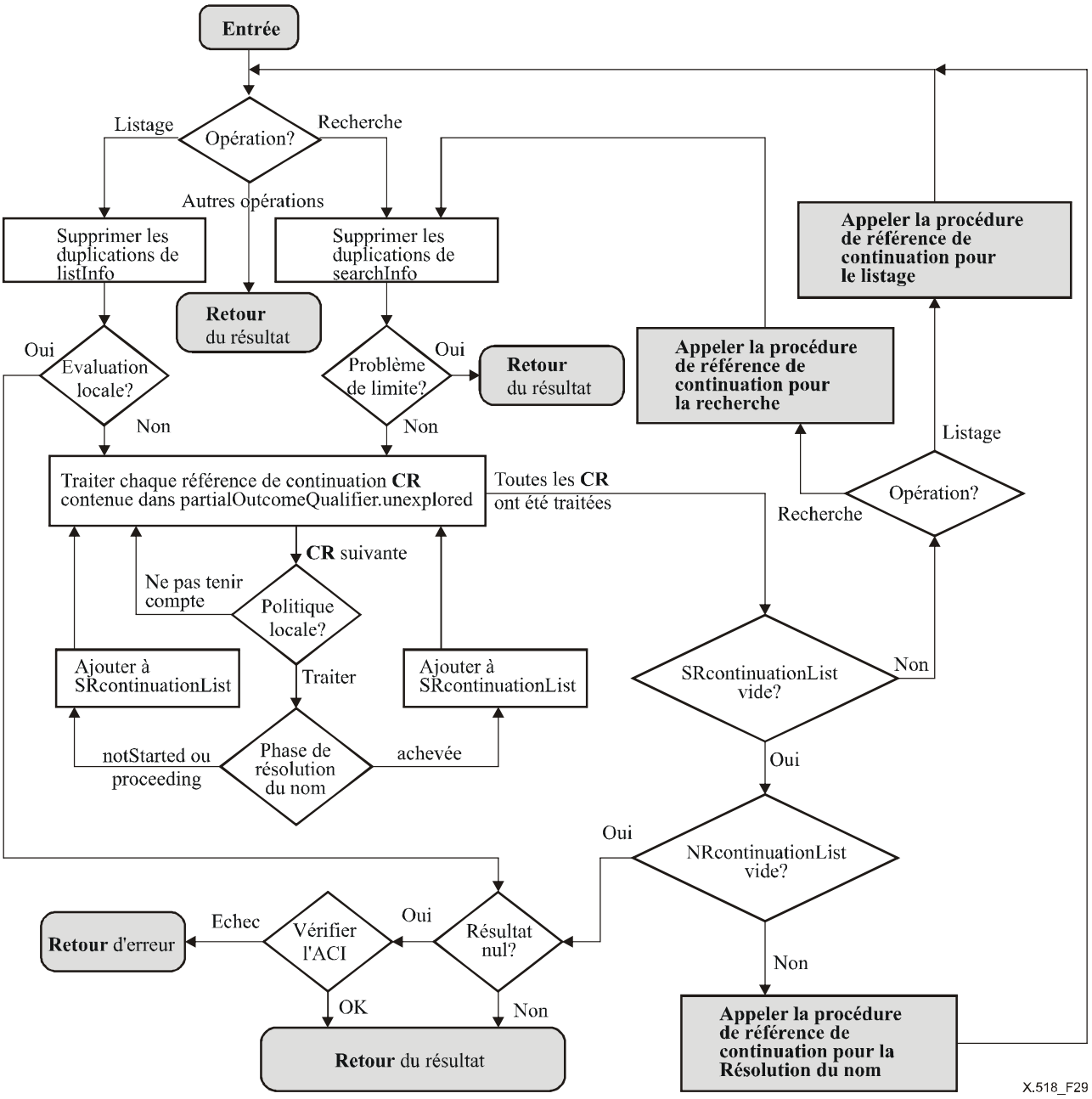


Figure 29 – Procédure de fusionnement des résultats

22 Procédures d'authentification répartie

Le présent paragraphe spécifie les procédures nécessaires pour assurer les services d'authentification répartie de l'annuaire. Ces services, et donc ces procédures, relèvent des catégories suivantes:

- authentification de l'expéditeur qui est assurée d'une manière non protégée (fondée sur la simple identité) ou sûre (fondée sur des signatures numériques);
- authentification des résultats, qui est protégée de la même façon (fondée également sur des signatures numériques).

22.1 Authentification de l'expéditeur

22.1.1 Authentification fondée sur l'identité (Identity-based)

Le service d'authentification fondée sur l'identité permet aux DSA d'authentifier le demandeur initial d'informations en vue d'effectuer des contrôles d'accès locaux. Les DSA qui désirent exploiter ce service doivent adopter la procédure suivante:

- dans le cas d'un DSA demandant l'authentification d'une demande de protocole d'accès à l'annuaire (DAP, *directory access protocol*) ou de protocole LDAP, le DSA se procure le nom distinctif du demandeur via les procédures Bind au moment de l'établissement d'une association par le DUA (DUA à DSA) ou d'une association par le client LDAP (client LDAP à DSA). Le succès de ces procédures ne préjuge en aucune façon le niveau d'authentification qui pourra être requis par la suite pour traiter des opérations en utilisant cette association;
- le DSA avec lequel le DUA ou le client LDAP établit l'association doit insérer le nom distinctif du demandeur dans le champ initiateur de **ChainingArguments** pour toutes les futures opérations chaînées vers d'autres DSA;
- un DSA, à la réception d'une opération chaînée, peut satisfaire cette opération ou non, selon les droits d'accès établis (par un mécanisme défini au niveau local). Si le résultat n'est pas satisfaisant, une erreur **securityError** peut être renvoyée avec la cause **insufficientAccessRights**.

22.1.2 Authentification de l'expéditeur fondée sur la signature

Le service d'authentification de l'expéditeur fondée sur la signature permet à un DSA d'authentifier (de manière sûre) l'expéditeur d'une demande de service particulière. Les procédures à exécuter par un DSA pour réaliser ce service sont décrites dans le présent paragraphe.

Le service d'authentification fondée sur la signature est invoqué par un DUA par utilisation de la variante **PROTECTED** d'une demande de service avec option de protection dont le composant **DirQOP** prend la valeur **signed** ou **signedAndEncrypted**.

A la réception d'une demande signée de la part d'un autre DSA, un DSA doit enlever la signature avant de traiter l'opération. En supposant que le résultat de toute vérification de signature s'avérera satisfaisant, le DSA continue à traiter l'opération. Si, durant ce traitement, le DSA a besoin d'effectuer un chaînage, l'ensemble d'arguments de chaque opération chaînée associée sera construit comme suit:

- le DSA forme un ensemble d'arguments qui peut être signé (à titre facultatif); l'ensemble d'arguments comprend l'ensemble des arguments signés entrants ainsi qu'un **ChainingArguments** modifié.

Si le DSA peut fournir des informations pour la réponse, l'authentification de l'expéditeur, fondée sur la demande de service signée, peut être utilisée pour la détermination des droits d'accès à ces informations.

Si un DSA reçoit une demande de service non signée pour des informations dont la diffusion est soumise à l'authentification de l'expéditeur, une erreur **securityError** sera retournée avec un motif mise à **protectionRequired**.

22.2 Authentification des résultats

Ce service est fourni pour permettre aux demandeurs d'opérations d'annuaire (des DUA, des clients LDAP ou des DSA) de vérifier (de manière sûre, en utilisant des techniques de signature numériques) l'origine des résultats. Le service d'authentification des résultats peut être demandé, que le service d'authentification de l'expéditeur soit utilisé ou non.

On lance le service d'authentification des résultats en utilisant la valeur signée du composant **protectionRequest** contenu dans l'ensemble d'arguments des opérations d'annuaire; un DSA recevant une opération pour laquelle cette option a été choisie pourra alors signer (à titre facultatif) tous les résultats subséquents. L'option signée de la demande de protection sert à indiquer au DSA la préférence des demandeurs; le DSA peut signer ou non un quelconque résultat subséquent.

Si un DSA effectue un chaînage, il dispose d'un certain nombre d'options en ce qui concerne la forme des résultats à renvoyer au demandeur, à savoir:

a) renvoyer une réponse composite (signée ou non signée) au demandeur;

b) renvoyer au demandeur un ensemble de deux ou plus de deux réponses partielles non collationnées (signées ou non signées), dans cet ensemble, zéro ou plusieurs éléments peuvent être signés et zéro ou un seul élément peut être non signé. Si un résultat partiel non signé est présent, cet élément peut être en fait une collation d'une ou de plusieurs réponses partielles non signées, provenant d'autres DSA, fournies par ce DSA ou provenant de ces deux sources.

Si un DSA effectue une jonction d'entrées associées, il peut signer le résultat.

SECTION 6 – ADMINISTRATION DES CONNAISSANCES

23 Aperçu général de l'administration des connaissances

Pour exploiter un annuaire largement réparti avec un niveau acceptable de cohérence et de performance, des procédures sont nécessaires pour créer, tenir à jour et étendre les connaissances détenues par chaque DSA. Les mécanismes suivants sont utilisés pour administrer les connaissances d'un DSA.

a) *Rattachements opérationnels hiérarchiques (HOB, hierarchical operational binding) et hiérarchiques non spécifiques (NHOB)* – Ces procédures et protocoles sont définis aux § 24 et 25. Ils servent à créer et à tenir à jour des références subordonnées, des références subordonnées non spécifiques, des références de supérieurs immédiats, ainsi que l'information du préfixe de contexte pour les contextes de dénomination. Ces rattachements opérationnels sont établis entre des DSA maîtres détenant des contextes de dénomination hiérarchiquement reliés les uns aux autres de subordonné immédiat à supérieur immédiat. Le déclenchement des procédures peut être un effet secondaire de la modification du RDN, de l'adjonction ou de la suppression d'une entrée dont le supérieur immédiat n'est pas détenu dans le même DSA que celui qui détient l'entrée.

b) *Rattachements opérationnels de duplication miroir* – Ces procédures et protocoles sont définis dans la Rec. UIT-T X.525 | ISO/CEI 9594-9. Ils sont utilisés pour créer et tenir à jour les références de connaissance de deux façons. Premièrement, l'effet secondaire de l'établissement (ou de la terminaison) d'accords de duplication miroir se traduit par l'adjonction (ou la suppression) de points d'accès dans les attributs opérationnels **consumerKnowledge** et, à titre facultatif, **secondaryShadow**. Cette information peut ensuite être utilisée par les procédures et les protocoles susmentionnés pour mettre à jour la référence subordonnée dans le DSA maître supérieur et la référence supérieure immédiate dans le DSA maître subordonné. Deuxièmement, le DISP propage les références de connaissance détenues par les DSA maîtres aux DSA consommateurs d'informations miroirs.

c) *Références croisées* – La distribution des références croisées est une fonction du DSP. La façon dont elle est utilisée pour créer et tenir à jour les références croisées est résumée au § 23.2.

NOTE – Les mécanismes utilisés pour établir et tenir à jour la référence supérieure et **myAccessPoint** n'entrent pas dans le cadre de la présente Spécification d'annuaire.

23.1 Administration des références de connaissances

Le présent paragraphe décrit comment le DOP est utilisé pour tenir à jour les attributs opérationnels de DSA qui expriment les connaissances. Un exemple simple illustrant la relation entre les attributs de connaissances et les protocoles utilisés pour les tenir à jour est décrit à l'Annexe E.

23.1.1 Administration des connaissances de consommateur par les DSA fournisseurs et maîtres

Une référence de consommateur est exprimée par une valeur de l'attribut **consumerKnowledge**, que détient un DSA fournisseur d'informations miroirs et que l'on associe au préfixe de contexte pour un contexte de dénomination; une référence de fournisseur est exprimée par une valeur de l'attribut **supplierKnowledge**, que détient un DSA consommateur d'informations miroirs et que l'on associe au préfixe de contexte pour un contexte de dénomination. Les deux attributs sont détenus dans des DSE de type **cp**. Une valeur de chacun de ces attributs est créée à l'établissement du rattachement opérationnel de duplication miroir et mise à jour au moment de la modification de ce rattachement.

Un DSA fournisseur peut obtenir les informations nécessaires pour construire les valeurs de l'attribut **secondaryShadows** si le composant facultatif **secondaryShadows** de son accord **ShadowingAgreementInfo** avec un consommateur a la valeur **TRUE**. Dans ce cas, chaque fois que le DSA du consommateur détectera une modification (par addition, modification ou suppression de points d'accès) de l'ensemble des DSA détenant des copies de la zone dupliquée d'usage commun (ses consommateurs ou, tout à tour, les consommateurs de ses consommateurs, etc., jusqu'au niveau le plus bas auquel des duplications miroirs peuvent être effectuées), cet agent communiquera ces nouvelles informations (sous forme d'ensemble de fournisseurs et de consommateurs **SuppliersAndConsumers**) au moyen d'une opération de type **modifyOperationalBinding** comme décrit dans la Rec. UIT-T X.525 | ISO/CEI 9594-9.

Un DSA fournisseur tient à jour son propre attribut **secondaryShadows** associé au préfixe de contexte de la manière suivante:

a) l'ensemble de fournisseurs et de consommateurs (**SupplierAndConsumers**) reçu d'un consommateur au moyen d'une opération de type **modifyOperationalBinding** peut servir à créer ou à remplacer des valeurs de l'attribut **secondaryShadows**. Le composant supplier de l'ensemble **SupplierAndConsumers** représente le point d'accès d'un DSA de consommateur (ou de ses consommateurs, etc., selon le niveau le plus bas des duplications miroirs secondaires); le composant consumers de l'ensemble

SupplierAndConsumers représente l'ensemble des consommateurs du consommateur (ou de leurs consommateurs, etc., selon le niveau le plus bas des duplications miroirs secondaires);

b) chaque consommateur qui communique à son fournisseur une opération de type **modifyOperationalBinding** contenant un ensemble du type **SupplierAndConsumers** présentera les valeurs de son attribut **secondaryShadows** et une valeur nouvellement construite au moyen de son propre point d'accès: **myAccessPoint** (en tant que composant supplier) et au moyen des valeurs des points d'accès des consommateurs, contenues dans l'attribut **consumerKnowledge**, qui représentent les consommateurs détenant (en tant que composant consumers) des duplications miroirs d'usage commun.

L'itération de cette procédure permet à un DSA maître, pour un contexte de dénomination, de connaître tous ses DSA consommateurs d'informations miroirs secondaires détenant des zones dupliquées d'usage commun obtenues à partir du contexte de dénomination. Cette information est ensuite disponible pour l'administration des références subordonnées, subordonnées non spécifiques et supérieures immédiates.

23.1.2 Administration des connaissances subordonnées et supérieures immédiates dans les DSA maîtres

Une référence subordonnée est exprimée par une valeur de l'attribut **specificKnowledge**, détenue dans une DSE de type **subr** par le DSA qui détient le contexte de dénomination immédiatement supérieur par rapport à celui qui est référencé; une référence de supérieur immédiat est exprimée par une valeur de l'attribut **specificKnowledge**, détenue dans une DSE de type **immSupr** par le DSA qui détient le contexte de dénomination immédiatement subordonné à celui qui est référencé. Une valeur de chacun de ces attributs est déterminée dans les DSA maîtres supérieur et subordonné à l'établissement du HOB, et elle est actualisée au moment de la modification de ce rattachement.

Un DSA maître subordonné fournit à un DSA maître supérieur l'information requise pour déterminer la référence subordonnée par l'intermédiaire du composant **accessPoints** du paramètre **SubordinateToSuperior** qu'il transmet au supérieur dans le DOP. L'information figurant dans **accessPoints** est déterminée par les valeurs des attributs que détient le DSA subordonné, de la manière suivante:

a) la valeur de l'attribut **myAccessPoint** (que détient la DSE racine) est utilisée pour constituer l'élément dans **accessPoints** avec la valeur **master** pour **category**;

b) les valeurs de **consumerKnowledge** et **secondaryShadows** (que détient la DSE du préfixe de contexte subordonné) sont utilisées pour former des éléments supplémentaires dans **accessPoints**, avec la valeur **shadow** pour **category**.

Un DSA maître supérieur fournit à un DSA maître subordonné l'information requise pour construire sa référence de supérieur immédiat à l'aide du composant **contextPrefixInfo** du paramètre **SuperiorToSubordinate** qu'il transfère au subordonné dans le DOP. Ce composant est une valeur du type **SEQUENCE OF Vertex**, qui contient une séquence d'éléments correspondant au trajet depuis la racine du DIT jusqu'au préfixe de contexte subordonné. Pour l'un de ces éléments, correspondant au préfixe du contexte de dénomination immédiatement supérieur, le composant facultatif **accessPoints** est présent. Le DSA subordonné détient cette information sous la forme d'un attribut **specificKnowledge** dans la DSE, de type **immSupr**, correspondant à cet élément de **contextPrefixInfo**. L'information incluse dans **accessPoints** par le DSA supérieur est déterminée par les valeurs des attributs que détient le DSA supérieur, de la manière suivante:

a) la valeur de l'attribut **myAccessPoint** (que détient la DSE racine) est utilisée pour constituer l'élément dans **accessPoints** avec la valeur **master** pour **category**;

b) les valeurs de **consumerKnowledge** et **secondaryShadows** (que détient la DSE du préfixe de contexte subordonné) sont utilisées pour former des éléments supplémentaires dans **accessPoints**, avec la valeur **shadow** pour **category**.

NOTE – Seuls les points d'accès correspondant aux DSA consommateurs recevant des zones dupliquées d'usage commun doivent être choisis par les DSA supérieurs et subordonnés à partir de leurs attributs **consumerKnowledge** pour être inclus dans **accessPoints**. Les procédures de détermination de **secondaryShadows** garantissent que ces points d'accès identifieront les DSA de duplication miroir détenant des zones dupliquées d'usage commun.

23.1.3 Administration des connaissances subordonnées et supérieures immédiates dans les DSA consommateurs

Un DSA consommateur d'informations miroirs concluant un accord avec son fournisseur pour recevoir les connaissances supérieures immédiates et subordonnées associées avec une unité de duplication, s'engage de telle sorte que ses références de supérieurs immédiats et subordonnées soient tenues à jour par son DSA fournisseur d'informations miroirs via le DISP.

NOTE – Pour certaines spécifications d'unités de duplication, il peut être nécessaire que le DSA consommateur conclue un accord pour recevoir **extendedKnowledge** afin que des connaissances subordonnées puissent lui être fournies par son fournisseur.

23.2 Demande de références croisées

Pour améliorer les performances du système d'annuaire, l'ensemble local de références croisées peut être étendu par l'utilisation des opérations d'annuaire ordinaires. Si un DSA supporte le DSP, il peut demander à un autre DSA (qui supporte également le DSP) de retourner les références de connaissance contenant des informations sur l'emplacement des contextes de dénomination relatifs au nom d'objet cible d'une opération d'annuaire ordinaire.

Si le composant **returnCrossRefs** de **ChainingArguments** est mis à **TRUE**, le composant **crossReferences** de **ChainingResults** peut être présent; il comprend une séquence d'éléments de références croisées.

Si un DSA ne peut pas chaîner une demande au DSA suivant, un renvoi de référence est retourné au DSA d'origine. Si le composant **returnCrossRefs** de **ChainingArguments** est mis à **TRUE**, le renvoi de référence peut contenir en outre le préfixe du contexte de dénomination auquel se réfère le renvoi de référence. Le composant **contextPrefix** est absent si le renvoi de référence est fondé sur une référence subordonnée non spécifique. La référence croisée retournée par un renvoi de référence est fondée sur les connaissances détenues par le DSA qui émet le renvoi de référence.

Dans les deux cas (résultat du chaînage et renvoi de référence), une autorité administrative peut, par le biais de son DSA, choisir d'ignorer la demande de retour de références croisées.

23.3 Incohérences de connaissances

L'annuaire doit disposer de mécanismes de contrôle de cohérence, afin de garantir un certain niveau de cohérence des connaissances.

NOTE – Dans certaines circonstances, les références de connaissances sont précises (il ne s'agit pas de références non valides telles qu'elles sont décrites ci-après), mais elles ne sont pas valides pour être utilisées par un DSA parce que le DMD du DSA auquel il est fait référence ne souhaite pas du tout que le DSA référençant entre en contact avec lui (par exemple DSA ayant acquis d'une manière ou d'une autre une référence croisée au DSA référencé) ou ne souhaite pas que l'on entre en relation avec lui pour jouer un rôle particulier (par exemple DSA maître pour un contexte de dénomination).

23.3.1 Détection des incohérences de connaissances

Les incohérences et leur détection dépendent des types de références de connaissances:

a) *références croisées et subordonnées* – Ce type de référence n'est pas valide si le DSA référencé ne détient pas de contexte de dénomination ou de zone reproduite obtenue à partir du contexte de dénomination dont le préfixe de contexte est contenu dans la référence. Cette incohérence sera détectée durant le processus de résolution du nom, par inspection des composants **operationProgress** et **referenceType** de **ChainingArguments**;

b) *références subordonnées non spécifiques* – Ce type de référence n'est pas valide si le DSA référencé ne détient pas de contexte de dénomination local dont le préfixe de contexte est contenu dans la référence moins le dernier RDN. Le contrôle de cohérence est appliqué comme indiqué ci-dessus;

c) *références supérieures* – Une référence supérieure non valide est une référence qui ne fait pas partie d'un chemin de référence vers la racine. La tenue à jour des références supérieures doit être assurée par des moyens extérieurs et n'entre pas dans le cadre de la présente Spécification d'annuaire.

NOTE – Il n'est pas toujours possible de détecter une référence supérieure non valide.

d) *références supérieures immédiates* – Ce type de référence n'est pas valide si le DSA référencé ne détient pas un contexte de dénomination ou une zone reproduite que l'on obtient à partir du contexte de dénomination avec le préfixe de contexte contenu dans la référence. En outre, l'utilisation d'un tel type de référence est uniquement valide lorsque le composant **operationProgress** de **ChainingArguments** a la valeur **notStarted** ou **proceeding**. Cette incohérence sera détectée pendant le processus de résolution du nom par inspection des composants **operationProgress** et **referenceType** de **ChainingArguments**;

e) *références du fournisseur* – Ce type de référence, qui identifie le fournisseur d'une zone reproduite et, facultativement, le maître pour le contexte de dénomination à partir duquel est obtenue la zone reproduite, n'est pas valide si le DSA référencé n'est pas le fournisseur d'informations miroirs pour le DSA utilisant la référence (lorsque le composant **referenceType** de **ChainingArguments** a la valeur **supplier**), ou si le DSA référencé n'est pas le maître pour le contexte de dénomination (lorsque **referenceType** a la valeur **master**). Cette incohérence sera détectée pendant les phases de résolution du nom et d'évaluation du traitement des opérations par inspection du composant **referenceType** de **ChainingArguments**.

23.3.2 Signalisation des incohérences de connaissances

Si un chaînage est utilisé pour traiter une demande présentée à l'annuaire, toutes les incohérences de connaissances seront détectées par le DSA qui détient la référence de connaissance non valide, par la réception d'une **serviceError**, avec la cause **invalidReference**.

Si un DSA retourne un renvoi de référence fondé sur une référence de connaissance non valide, une **serviceError** sera retournée au demandeur avec la cause **invalidReference**, s'il utilise le renvoi de référence. La façon dont l'état d'erreur sera communiqué au DSA qui enregistre la référence non valide n'entre pas dans le cadre de la présente Spécification d'annuaire.

23.3.3 Traitement des références de connaissances incohérentes

Quand un DSA a détecté une référence non valide, il doit essayer de rétablir la cohérence des connaissances. Pour cela, il peut par exemple supprimer simplement une référence croisée non valide, ou la remplacer par une référence croisée correcte, qui peut être obtenue à l'aide des mécanismes **returnCrossRefs**.

La façon dont un DSA traite effectivement les références non valides relève d'une initiative locale et n'entre pas dans le cadre de la présente Spécification d'annuaire.

23.4 Références de connaissances et contextes

Les noms contenus dans les références doivent être les noms distinctifs primaires. Il peut y avoir aussi des variantes de valeurs distinctives et des informations contextuelles contenues, comme l'indique le § 9.3 de la Rec. UIT-T X.501 | ISO/CEI 9594-2, dans le composant **valuesWithContext** de n'importe lequel des attributs constitutifs d'un quelconque RDN.

Selon la manière dont est obtenue une référence de connaissance et, en particulier, si un DSA antérieur à 1997 est soit le dépositaire de cette information, soit un maillon de la chaîne qui a permis de l'atteindre, il peut arriver que la référence de connaissance ne contienne pas toutes les variantes possibles de noms distinctifs. Le risque est que le possesseur de la référence de connaissance ne reconnaisse pas le nom prétendu comme le même nom, entraînant des étapes supplémentaires dans la résolution du nom sinon, dans certains cas, des résultats incohérents ou l'échec de la résolution du nom. L'usage généralisé des noms distinctifs primaires, lorsqu'ils sont connus, pousse à leur maximum les capacités de l'annuaire à traiter des variantes contextuelles de noms.

24 Rattachements opérationnels hiérarchiques

Un rattachement opérationnel hiérarchique (HOB) est utilisé pour représenter la relation entre deux DSA détenant deux contextes de dénomination, l'un étant immédiatement subordonné à l'autre. S'agissant d'un HOB, le DSA supérieur détient une référence subordonnée au contexte de dénomination que possède le DSA subordonné; le DSA subordonné détient une référence de supérieur immédiat au contexte de dénomination que possède le DSA supérieur. Le rattachement opérationnel garantit que les informations de connaissances appropriées seront échangées et tenues à jour entre les deux DSA de sorte qu'ils puissent agir pendant le processus de résolution du nom et d'évaluation de l'opération, comme défini dans les § 18 et 19.

24.1 Caractéristiques du type de rattachement opérationnel

24.1.1 Symétrie et rôles

Le rattachement opérationnel hiérarchique est de type asymétrique. Les deux rôles d'un rattachement de ce type sont les suivants:

a) le rôle du DSA maître pour le contexte de dénomination supérieur, *le DSA supérieur* (associé au rôle abstrait "A");

b) le rôle du DSA maître pour le contexte de dénomination subordonné, *le DSA subordonné* (associé au rôle abstrait "B").

24.1.2 Accord

L'information d'accord échangée pendant l'établissement du rattachement opérationnel hiérarchique est une valeur de **HierarchicalAgreement** qui contient le nom distinctif relatif du nouveau préfixe de contexte (le composant **rdn**) et le nom distinctif de l'entrée immédiatement supérieure au nouveau contexte de dénomination (le composant **immediateSuperior**). Cette information est transmise par le DSA qui établit le HOB.

```
HierarchicalAgreement ::= SEQUENCE {
        rdn                 [0]  RelativeDistinguishedName,
        immediateSuperior   [1]  DistinguishedName }
```

rdn doit être le RDN primaire, et **immediateSuperior** doit être le nom distinctif primaire. Selon le § 9.3 de la Rec. UIT-T X.501 | ISO/CEI 9594-2, les informations contextuelles et toutes les variantes de valeurs distinctives doivent être incluses dans le composant **valuesWithContext** de **AttributeTypeAndDistinguishedValue** d'un quelconque RDN.

24.1.3 Initiateur

24.1.3.1 Etablissement

L'établissement d'un rattachement opérationnel hiérarchique peut être entrepris par l'un ou l'autre DSA. L'établissement par le DSA supérieur peut résulter d'une opération adjonction d'entrée avec la spécification du DSA subordonné dans l'extension **targetSystem**, ou bien d'une intervention administrative. L'établissement par le DSA subordonné (qui connecte au DIT global une entrée ou une sous-arborescence existant au niveau local) résulte d'une intervention administrative.

24.1.3.2 Modification

La modification d'un rattachement opérationnel hiérarchique peut être entreprise par l'un ou l'autre DSA. Le DSA supérieur peut apporter cette modification à la suite d'une modification de l'information du préfixe de contexte supérieur. Cela peut résulter de l'une quelconque des opérations de modification, ou d'une intervention administrative.

Chaque DSA peut modifier l'accord suite à une modification du RDN de l'entrée du préfixe de contexte du contexte de dénomination subordonné. Le DSA supérieur procède à cette modification en raison d'un changement de nom distinctif relatif plus haut dans le DIT ou à cause d'une intervention administrative. Le DSA subordonné procède à la modification de l'accord à cause d'une opération ModifyDN (modifier nom distinctif) effectuée sur un préfixe de contexte ou suite à une intervention administrative.

Chaque DSA peut également modifier le HOB si l'information de point d'accès pour son contexte de dénomination change.

24.1.3.3 Terminaison

La terminaison d'un rattachement opérationnel hiérarchique peut être effectuée par l'un ou l'autre DSA. La terminaison par le DSA supérieur peut résulter d'une intervention administrative. La terminaison par le DSA subordonné peut résulter d'une opération de suppression d'entrée supprimant l'entrée du préfixe de contexte du contexte de dénomination subordonné, ou bien d'une intervention administrative.

24.1.4 Paramètres d'établissement

Les paramètres d'établissement pour les deux rôles d'un HOB (DSA supérieur et DSA subordonné) sont différents. Le paramètre d'établissement pour le rôle de DSA supérieur est une valeur de **SuperiorToSubordinate**; pour le DSA subordonné, le paramètre d'établissement est une valeur de **SubordinateToSuperior**.

24.1.4.1 Paramètre d'établissement de DSA supérieur

Le paramètre d'établissement émis par le DSA supérieur, qui est une valeur de **SuperiorToSubordinate**, fournit au DSA subordonné des informations concernant les nœuds du DIT supérieurs au préfixe de contexte du nouveau contexte de dénomination (qui comprend la référence de supérieur immédiat) et, facultativement, des attributs d'utilisateur et opérationnels pour l'entrée du préfixe de contexte subordonné ainsi que des copies d'attributs d'utilisateur et opérationnels provenant de l'entrée immédiatement supérieure au nouveau préfixe de contexte.

```
SuperiorToSubordinate ::= SEQUENCE {
        contextPrefixInfo           [0]  DITcontext,
        entryInfo                   [1]  SET SIZE (1..MAX) OF Attribute OPTIONAL,
        immediateSuperiorInfo       [2]  SET SIZE (1..MAX) OF Attribute OPTIONAL }
```

Le **rdn** de **Vertex** ou de **SubentryInfo** doit être le RDN primaire. Selon le § 9.3 de la Rec. UIT-T X.501 | ISO/CEI 9594-2, les informations contextuelles et toutes les variantes de valeurs distinctives doivent appartenir au composant **AttributeTypeAndDistinguishedValue** du RDN.

24.1.4.1.1 Information du préfixe de contexte

Le composant **contextPrefixInfo** de **SuperiorToSubordinate** est une valeur du type **DITcontext**, qui correspond à une séquence de valeurs **Vertex**.

```
DITcontext ::= SEQUENCE OF Vertex

Vertex ::= SEQUENCE {
        rdn             [0]  RelativeDistinguishedName,
        admPointInfo    [1]  SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries      [2]  SET SIZE (1..MAX) OF SubentryInfo OPTIONAL,
        accessPoints    [3]  MasterAndShadowAccessPoints OPTIONAL }
```

Le composant **contextPrefixInfo** est la séquence de noms RDN qui forme le nom distinctif du supérieur immédiat du nouveau préfixe de contexte, chaque RDN (donné par le composant **rdn**) étant facultativement accompagné par des informations supplémentaires.

Le composant facultatif **admPointInfo** d'un **Vertex** indique que le nœud du DIT est un point administratif et fournit au moins son attribut opérationnel **administrativeRole**.

L'information de sous-entrée associée à un point administratif est fournie par le composant **subentries** d'un nœud **Vertex**, à savoir une série d'une ou plusieurs valeurs **SubentryInfo**. Chaque valeur **SubentryInfo** est composée du RDN de la sous-entrée (composant **rdn**) et des attributs de la sous-entrée (composant **info**).

```
SubentryInfo ::= SEQUENCE {
        rdn     [0]  RelativeDistinguishedName,
        info    [1]  SET OF Attribute }
```

Le composant facultatif **accessPoints** d'un **Vertex** indique que ce nœud correspond au préfixe de contexte du contexte de dénomination immédiatement supérieur. Le supérieur utilise ce composant pour fournir au subordonné l'information requise pour sa référence de supérieur immédiat.

NOTE – Le point d'accès maître contenu dans le paramètre **accessPoints** est le même que celui qui est transmis dans le paramètre **accessPoint** des opérations d'établissement et de modification de rattachement opérationnel.

24.1.4.1.2 Information d'entrée

Le composant facultatif **entryInfo** de **SuperiorToSubordinate** est une série d'attributs établissant le contenu de la nouvelle entrée du préfixe de contexte.

24.1.4.1.3 Information d'entrée supérieure immédiate

Le composant facultatif **immediateSuperiorInfo** de **SuperiorToSubordinate** est une copie d'une série d'attributs, en particulier **objectClass** et **entryACI**, provenant de l'entrée immédiatement supérieure au nouveau préfixe de contexte.

NOTE – Ce composant peut être utilisé par le subordonné pour optimiser l'évaluation d'une demande de listage qui produit un résultat **ListResult** de valeur empty pour une entrée d'objet de base qui est l'entrée immédiatement supérieure au préfixe de contexte subordonné (voir la Note du § 19.3.1.2.2, alinéa 2)).

24.1.4.2 Paramètre d'établissement du DSA subordonné

Le paramètre d'établissement émis par le DSA subordonné, qui est une valeur de **SubordinateToSuperior**, fournit au DSA supérieur des informations concernant le contexte de dénomination subordonné.

```
SubordinateToSuperior ::=       SEQUENCE {
        accessPoints    [0]     MasterAndShadowAccessPoints OPTIONAL,
        alias           [1]     BOOLEAN DEFAULT FALSE,
        entryInfo       [2]     SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries      [3]     SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }
```

Le composant **accessPoints** de **SubordinateToSuperior** est utilisé par le subordonné en vue de fournir au supérieur les informations requises pour sa référence subordonnée.

NOTE 1 – Le point d'accès maître contenu dans le paramètre **accessPoints** est le même que celui qui est transmis dans le paramètre **accessPoint** des opérations d'établissement et de modification de rattachement opérationnel.

Le composant **alias** de **SubordinateToSuperior** est utilisé pour indiquer au supérieur que le contexte de dénomination subordonné comprend une entrée d'alias unique.

Le composant **entryInfo** de **SubordinateToSuperior** comprend une copie d'une série d'attributs, en particulier **objectClass** et **entryACI,** mais aussi, le cas échéant, l'attribut opérationnel **administrativeRole**, provenant de la nouvelle entrée du préfixe de contexte.

NOTE 2 – Les deux premiers attributs peuvent être utilisés par le supérieur pour optimiser l'évaluation de listage ou de recherche à un niveau dont l'objet de base est l'entrée immédiatement supérieure au préfixe de contexte subordonné, alors que le dernier attribut est utilisé pour éviter la progression non désirée d'une opération de recherche dans une zone administrative de service ou en dehors de celle-ci.

Le composant **subentries** de l'élément **SubordinateToSuperior** est utilisé par l'entrée subordonnée pour transmettre à l'entrée supérieure des sous-entrées contenant des informations ACI prescriptives.

24.1.5 Paramètres de modification

Pour les modifications d'un HOB, le paramètre de modification de DSA supérieur, **SuperiorToSubordinateModification**, est **SuperiorToSubordinate**, à une restriction près qui est l'absence éventuelle du composant **entryInfo**; le paramètre de modification du rôle de DSA subordonné est **SubordinateToSuperior**.

```
SuperiorToSubordinateModification ::= SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})
```

Ces paramètres sont identiques (moyennant la restriction susmentionnée) aux paramètres d'établissement correspondants qui sont utilisés pour indiquer les modifications apportées aux informations fournies dans les paramètres d'établissement suite à l'établissement d'un HOB.

Si un composant quelconque de **SuperiorToSubordinate** (ou, ultérieurement, **SuperiorToSubordinateModification**) ou de **SubordinateToSuperior** fait l'objet d'une modification (par exemple composant **contextPrefixInfo** de **SuperiorToSubordinate**), le composant correspondant du paramètre de modification (par exemple composant **contextPrefixInfo** de **SuperiorToSubordinateModification**) doit être fourni dans son intégralité à l'intérieur de la modification du rattachement opérationnel.

24.1.6 Paramètres de terminaison

Pour la terminaison d'un HOB, aucun des deux DSA ne fournit de paramètre de terminaison.

24.1.7 Identification du type

Le rattachement opérationnel hiérarchique est identifié par l'identificateur d'objet assigné lors de la définition de l'objet d'information **hierarchicalOperationalBinding OPERATIONAL-BINDING**, au § 24.2.

24.2 Définition de la classe d'objets d'information de type operational binding

Le présent paragraphe définit le type de rattachement opérationnel hiérarchique en utilisant le modèle de classe d'objets d'information **OPERATIONAL-BINDING** défini dans la Rec. UIT-T X.501 | ISO/CEI 9594-2.

```
hierarchicalOperationalBinding OPERATIONAL-BINDING ::= {
        AGREEMENT       HierarchicalAgreement
        APPLICATION CONTEXTS {
                        {directorySystemAC} }
        ASYMMETRIC
            ROLE-A {        -- DSA supérieur
                ESTABLISHMENT-INITIATOR         TRUE
                ESTABLISHMENT-PARAMETER         SuperiorToSubordinate
                MODIFICATION-INITIATOR          TRUE
                MODIFICATION-PARAMETER          SuperiorToSubordinateModification
                TERMINATION-INITIATOR           TRUE }
            ROLE-B {        -- DSA supérieur
                ESTABLISHMENT-INITIATOR         TRUE
                ESTABLISHMENT-PARAMETER         SubordinateToSuperior
                MODIFICATION-INITIATOR          TRUE
                MODIFICATION-PARAMETER          SubordinateToSuperior
                TERMINATION-INITIATOR           TRUE }
        ID                      id-op-binding-hierarchical }
```

24.3 Procédures de DSA pour la gestion des rattachements opérationnels hiérarchiques

Dans les procédures suivantes, une nouvelle DSE ou une marque (c'est-à-dire une indication d'état associée à un élément d'information donné) créée par un DSA doit être stockée dans une mémoire stable. Ce faisant, il devient possible pour les deux DSA suivant les procédures ci-après de maintenir une compréhension cohérente des paramètres du HOB en cas de défaillance de la communication et des systèmes/terminaux.

Dans la procédure d'établissement **establishment** et de modification **modification** décrite ci-après, le DSA jouant le rôle de celui qui répond (c'est-à-dire n'entreprenant pas l'établissement ou la modification) peut fournir au DSA qui joue le rôle initiateur des informations (par exemple attributs opérationnels) qui ne sont pas acceptables pour telle ou telle raison. En pareil cas, le DSA initiateur peut mettre fin au rattachement opérationnel.

24.3.1 Procédure d'établissement

24.3.1.1 Etablissement entrepris par un DSA supérieur

Si un DSA évalue une opération Add Entry avec un DSA différent, spécifié dans l'extension **targetSystem**, il devra établir un rattachement opérationnel hiérarchique conformément à la procédure suivante. Si un DSA, pour des raisons administratives, désire établir un HOB avec un DSA subordonné et s'il admet le DOP par HOB, la procédure ci-après doit être suivie:

1) le DSA supérieur crée une nouvelle DSE de type **subr** avec le nom de la nouvelle entrée, et marque cette nouvelle DSE comme *étant ajoutée*. Le DSA supérieur émet une identification **bindingID** unique et l'enregistre avec la nouvelle DSE;
2) le DSA supérieur doit envoyer au DSA subordonné une opération établissement de rattachement opérationnel contenant les paramètres suivants:
   a) type de rattachement **bindingType** mis à **hierarchicalOperationalBindingID**;
   b) paramètre d'établissement **SuperiorToSubordinate** avec les composants **contextPrefixInfo** et **entryInfo** présents; tous les autres paramètres sont facultatifs;
   c) **HierarchicalAgreement** avec le composant **immediateSuperior** mis au nom distinctif du supérieur immédiat de la nouvelle entrée et le composant **rdn** mis au RDN de la nouvelle entrée;
   d) paramètres **bindingID**, **myAccessPoint** et **valid**, le cas échéant;
3) si le DSA subordonné accepte l'opération, il crée les DSE requises de type **glue**, **subentry**, **admPoint**, **rhob** et **immSupr**, selon le cas, pour représenter l'information **contextPrefixInfo**, une DSE de type **cp** et **entry** ou **alias** pour représenter le nouvel objet du préfixe de contexte ou l'entrée d'alias; et, le cas échéant, une DSE de type **rhob** et **entry** pour représenter l'information **immediateSuperiorInfo**. Il enregistre **bindingID** avec la DSE de la nouvelle entrée du préfixe de contexte et retourne un paramètre **SubordinateToSuperior** au DSA supérieur;

   si le DSA subordonné refuse l'opération, il renvoie une erreur de rattachement opérationnel avec la valeur de cause appropriée;

   si le contexte de dénomination existe déjà et si les valeurs de l'identificateur **bindingID** sont les mêmes pour le contexte existant et pour le nouveau contexte, c'est que le DSA subordonné a déjà créé le contexte de dénomination demandé, auquel cas il doit retourner un résultat au DSA supérieur. Si les valeurs ne sont pas les mêmes, une erreur de rattachement opérationnel avec la cause **invalidAgreement** est envoyée; cela signifie que le DSA supérieur a une incohérence de connaissances permanente qui doit être corrigée par un administrateur;
4) si le DSA supérieur reçoit une erreur, il supprime la DSE marquée du type **subr** et retourne une erreur pour l'opération Add Entry;

   si le DSA supérieur reçoit un résultat, il supprime l'indication de la DSE qui représente le **subr** et retourne un résultat pour l'opération Add Entry;

   en cas de défaillance (par exemple, communication du système terminal), le DSA supérieur doit répéter les opérations à partir de l'étape 2) jusqu'à l'obtention d'un résultat ou d'une erreur pour chaque établissement en suspens d'un rattachement opérationnel hiérarchique dont il est l'initiateur. Si l'établissement résulte d'une opération Add Entry et si le demandeur interrompt l'opération (par exemple en libérant ou en interrompant l'association d'application) avant que l'établissement soit terminé, le DSA supérieur ne doit pas tenir compte de cet événement et doit terminer l'établissement (qui peut réussir ou non). En pareil cas, l'utilisateur ne sera pas informé du résultat de l'opération Add Entry.

   NOTE 1 – Le marquage de la référence subordonnée aide à gérer les reprises et les annulations. Un autre utilisateur ne peut ajouter une entrée qui est déjà marquée et le DSA répète l'établissement de rattachement opérationnel pour toutes les références subordonnées marquées, après une défaillance.

   NOTE 2 – Compte tenu de la procédure décrite ci-dessus, les connaissances n'ont qu'une incohérence transitoire. La façon dont le DSA supérieur traite les opérations indépendantes qui lisent la référence subordonnée lorsqu'elle est marquée, relève d'une initiative locale.

24.3.1.2 Etablissement entrepris par un DSA subordonné

Le DSA subordonné peut entreprendre un rattachement opérationnel hiérarchique. Cela peut faire suite au désir d'un administrateur de relier, à un certain point du DIT global, un sous-arbre d'entrées détenues dans le DSA. En pareil cas, ce DSA subordonné doit établir un HOB conformément à la procédure suivante:

1) le DSA subordonné a une DSE de type **cp** en tant que partie d'un contexte de dénomination existant, ou il en crée une nouvelle. Il marque la DSE *ajoutée*, émet un identificateur **bindingID** unique et l'enregistre avec la DSE du préfixe de contexte;

2) le DSA subordonné envoie une opération d'établissement de rattachement opérationnel au DSA supérieur, contenant les paramètres suivants:

   a) **bindingType** mis à **hierarchicalOperationalBindingID**;

   b) paramètre d'établissement **SubordinateToSuperior**, le cas échéant;

   c) **HierarchicalAgreement** avec le composant **immediateSuperior** mis au nom distinctif du supérieur immédiat de la nouvelle entrée et le composant **rdn** mis au RDN de la nouvelle entrée;

   d) paramètres **bindingID, myAccessPoint** et **valid**, le cas échéant;

   si le DSA supérieur refuse l'opération, il retourne une erreur de type Operational Binding avec la valeur de cause appropriée;

3) le DSA supérieur vérifie qu'il est maître pour le supérieur immédiat de la nouvelle entrée du préfixe de contexte ou retourne une erreur **operationalBindingError** avec la cause **roleAssignment**;

4) le DSA supérieur vérifie que le RDN demandé pour le nouveau préfixe de contexte n'est pas déjà utilisé. Si l'on ne trouve pas de RDN concordant en utilisant l'information détenue localement, mais que la DSE immédiatement supérieure soit du type **nssr**, la procédure décrite au § 19.1.5 est appliquée. Si l'on ne trouve toujours pas de RDN concordant en utilisant cette procédure, le DSA supérieur crée une DSE de type **subr**, enregistre avec cette entrée **bindingID** et retourne un résultat;

   si l'on trouve une référence subordonnée avec ce RDN, on compare les deux valeurs de **bindingID**. Si ces valeurs sont les mêmes, un résultat est retourné. Le paramètre **SuperiorToSubordinate** retourné par le DSA supérieur ne doit pas contenir le composant **entry**. Si les deux valeurs de **bindingID** ne sont pas identiques, une erreur **operationalBindingError** est envoyée avec la cause **invalidAgreement**; cela signifie que le DSA supérieur a une incohérence de connaissances permanente qui doit être corrigée par un administrateur;

   si un RDN concordant est trouvé dans l'examen d'une NSSR, une erreur de rattachement opérationnel **operationalBindingError** est envoyée avec la cause **invalidAgreement**; cela signifie également que le DSA supérieur a une incohérence de connaissances permanente qui doit être corrigée par un administrateur;

5) si le DSA subordonné reçoit une erreur, il supprime la nouvelle DSE du préfixe de contexte et sa marque. Le sort de l'information d'entrée à partir de laquelle a été obtenue la DSE du préfixe de contexte doit être déterminé localement;

   si le DSA subordonné reçoit un résultat, il ajoute les DSE nécessaires de type **glue**, **subentry**, **admPoint**, **rhob** et **immSupr**, selon le cas, pour représenter l'information **contextPrefixInfo** et, le cas échéant, une DSE de type **rhob** et **entry** pour représenter l'information **immediateSuperiorInfo**. La marque de la DSE du préfixe de contexte est supprimée;

   en cas de défaillance (par exemple communication du système terminal), le DSA subordonné reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations d'établissement en suspens d'un HOB dont il est l'initiateur.

24.3.2 Procédure de modification

Les procédures ci-après sont définies pour la modification d'un HOB entrepris à l'aide de la procédure décrite au § 24.3.1.

24.3.2.1 Procédure de modification entreprise par un DSA supérieur

On peut invoquer cette procédure à la suite des opérations de modification, comme décrit au § 19.1, ou à la suite d'une intervention administrative (par exemple pour transmettre les modifications relatives aux paramètres **myAccessPoint**, **agreement** ou **valid** du HOB). Par ailleurs, si un DSA supérieur détecte des modifications dans les composants **contextPrefixInfo** ou **immediateSuperiorInfo** de la valeur **SuperiorToSubordinate** qu'il a fournie au DSA subordonné, il doit faire suivre ces nouvelles informations au DSA subordonné en utilisant la procédure ci-après:

1) marquer la DSE de type **subr** comme *étant modifiée* et, si cette modification résulte d'une modification apportée au RDN de l'entrée du préfixe de contexte subordonné, une nouvelle DSE de type **subr** est ajoutée et marquée comme *étant ajoutée*;

2) le DSA supérieur produit une nouvelle valeur de **bindingID** à partir de la valeur existante en augmentant son composant **version**. En utilisant ce nouveau **bindingID**, il envoie au DSA subordonné une opération de modification du rattachement opérationnel avec le paramètre de modification **SuperiorToSubordinateModification**;

3) le DSA subordonné vérifie le composant **identifier** de **bindingID**. S'il n'a pas d'accord de ce genre avec le DSA supérieur, ou si le composant **version** a une valeur inférieure à celle de la version du HOB, il doit retourner une erreur **operationalBindingError** avec la cause **invalidAgreement**;

4) le DSA subordonné peut accepter la modification du HOB, modifier ou reconstruire les DSE représentant les informations du préfixe de contexte, actualiser le composant **version** de son **bindingID** et retourner un résultat. Sinon, il peut retourner une erreur et mettre fin à l'accord;

5) si le DSA supérieur reçoit un résultat, la modification est achevée. Si cette modification résulte d'une modification apportée au RDN de l'entrée du préfixe de contexte subordonné, on supprime la marque de la nouvelle DSE de type **subr** marquée comme *étant ajoutée*, et l'ancienne DSE marquée comme *étant modifiée* est supprimée. Sinon, la marque "*étant modifiée*" est supprimée;

   si le DSA supérieur reçoit une erreur, la modification a échoué. La marque "*étant modifiée*" est supprimée. Si cette modification résulte d'une modification apportée au RDN de l'entrée du préfixe de contexte subordonné, la nouvelle DSE de type **subr**, marquée comme *étant ajoutée* est supprimée. Dans le cas contraire, les mesures qu'il faut prendre n'entrent pas dans le cadre de la présente Spécification d'annuaire;

   en cas de défaillance (par exemple communication du système terminal), le DSA supérieur reprend la procédure à partir de l'étape 2) jusqu'à ce qu'un résultat ou une erreur ait été reçu pour chaque modification en suspens d'un rattachement opérationnel hiérarchique dont il est l'initiateur. Si la modification résulte d'une opération ModifyDN modifiant le RDN de l'entrée du préfixe de contexte subordonné et si le demandeur interrompt l'opération (par exemple en libérant ou en interrompant l'association d'application) avant l'achèvement de la modification, le DSA supérieur ne tient pas compte de cet événement et achève la modification (qui peut aboutir ou non). En pareil cas, l'utilisateur ne sera pas informé du résultat de l'opération ModifyDN.

24.3.2.2 Procédure de modification entreprise par un DSA subordonné

Cette procédure peut être invoquée suite à une intervention administrative (par exemple pour transmettre des modifications aux paramètres **myAccessPoint**, **agreement** ou **valid** du HOB). Par ailleurs, si un DSA subordonné détecte des modifications relatives à la valeur de **SubordinateToSuperior** qu'il a fournie au DSA supérieur, il diffuse la nouvelle information au DSA supérieur en utilisant la procédure ci-après:

1) marquer la DSE de type **cp** comme *étant modifiée*;

2) le DSA subordonné produit une nouvelle valeur de **bindingID** à partir de la valeur existante en augmentant son composant **version**. En utilisant ce nouveau **bindingID**, il envoie au DSA supérieur une opération de modification du rattachement opérationnel avec le paramètre de modification **SubordinateToSuperior**;

3) le DSA supérieur vérifie le composant **identifier** de **bindingID**. S'il n'a pas d'accord de ce genre avec le DSA subordonné, ou si le composant **version** a une valeur inférieure à celle de la version du HOB, il doit retourner une erreur **operationalBindingError** avec la cause **invalidAgreement**;

4) le DSA supérieur peut accepter la modification du HOB, modifier la DSE représentant la référence subordonnée et retourner un résultat. Sinon, il peut retourner une erreur et mettre fin à l'accord;

   de plus, si la DSE supérieure de la DSE (de type **subr**) à renommer est du type **nssr**, le DSA doit, avant de répondre à la demande de modification du HOB, suivre la procédure définie au § 19.1.5 (concernant les opérations de modification et les NSSR) afin de garantir que le nouveau nom de l'entrée n'est pas ambigu;

5) si le DSA subordonné reçoit un résultat, la modification est achevée, et il supprime la marque. S'il reçoit une erreur, les mesures qu'il faut prendre n'entrent pas dans le cadre de la présente Spécification d'annuaire;

   en cas de défaillance (par exemple communication du système terminal), le DSA subordonné reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations de modification en suspens d'un HOB dont il est l'initiateur.

24.3.3 Procédure de terminaison

Les procédures suivantes sont définies pour mettre fin à un HOB qui a été entrepris par la procédure décrite en détail au § 24.3.1.

24.3.3.1 Terminaison entreprise par un DSA supérieur

La terminaison d'un rattachement opérationnel hiérarchique n'est entreprise par le DSA supérieur qu'à la suite d'une intervention administrative. Il convient de suivre la procédure ci-après:

1) le DSA supérieur marque la DSE représentant la référence subordonnée comme étant "*supprimée*", de façon que la référence subordonnée ne soit plus utilisée pendant la résolution du nom;
2) le DSA supérieur envoie au DSA subordonné une opération de terminaison de rattachement opérationnel pour le rattachement opérationnel hiérarchique. Le composant **version** de **bindingID** est omis par le DSA supérieur;
3) lorsque le DSA subordonné reçoit la terminaison de rattachement opérationnel, il supprime toute information concernant le rattachement opérationnel hiérarchique et envoie un résultat, à moins que le composant **identifier** de **bindingID** ne soit pas connu, auquel cas une erreur **operationalBindingError** avec la cause **invalidID** est retournée. Ce qui se passe pour toute information d'entrée associée au contexte de dénomination subordonné relève d'une initiative locale;
4) si le DSA supérieur reçoit un résultat ou une erreur **operationalBindingError** avec la cause **invalidID**, il supprime la DSE marquée *supprimée* qui représente la référence subordonnée associée au rattachement opérationnel hiérarchique et supprime toute information concernant le rattachement opérationnel;

   en cas de défaillance (par exemple communication du système terminal), le DSA supérieur reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations de terminaison en suspens d'un HOB dont il est l'initiateur.

24.3.3.2 Terminaison entreprise par un DSA subordonné

La terminaison entreprise par le DSA subordonné peut résulter d'une opération de suppression d'entrée qui supprime la dernière entrée dans le contexte de dénomination subordonné, l'entrée du préfixe de contexte, ou bien d'une intervention administrative. Il convient d'appliquer la procédure ci-après:

1) le DSA subordonné marque la DSE du préfixe de contexte pour le contexte de dénomination comme étant *supprimée*;
2) le DSA subordonné envoie au DSA supérieur une opération terminaison de rattachement opérationnel pour le rattachement opérationnel hiérarchique. Le composant **version** de **bindingID** est omis par le DSA subordonné;
3) lorsque le DSA supérieur reçoit la terminaison de rattachement opérationnel, il supprime la DSE qui représente la référence subordonnée associée au rattachement opérationnel et hiérarchique, supprime toute information concernant le rattachement opérationnel et envoie un résultat, à moins que le composant **identifier** de **bindingID** ne soit pas connu, auquel cas une erreur **operationalBindingError** est retournée avec la cause **invalidID**;
4) si le DSA subordonné reçoit un résultat ou une erreur **operationalBindingError** avec la cause **invalidID**, il supprime toute information concernant le rattachement opérationnel.

   NOTE – Ce qui arrive à l'information d'entrée du contexte de dénomination relève d'une initiative locale au niveau du DSA subordonné. Etant donné que le fait de renommer (par exemple par déplacement) un contexte de dénomination n'est pas autorisé par l'opération ModifyDN, un administrateur pourrait, par exemple, mettre fin à l'HOB, choisir un autre préfixe de contexte pour le contexte de dénomination et le reconnecter à une autre partie du DIT (c'est-à-dire en établissant un nouvel HOB).

   En cas de défaillance (par exemple communication du système terminal), le DSA subordonné reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations de terminaison en suspens d'un HOB dont il est l'initiateur.

24.4 Procédures pour les opérations

Les opérations qui peuvent être exécutées dans l'état coopératif d'un rattachement opérationnel hiérarchique sont définies dans le contexte d'application **directorySystemAC**.

Les procédures que doit suivre le DSA intervenant dans un rattachement opérationnel hiérarchique, sont définies dans les § 16 à 22.

24.5 Utilisation des contextes d'application

Pour établir, modifier ou terminer un rattachement opérationnel hiérarchique à l'aide du protocole et des procédures de la présente Spécification d'annuaire, un DSA doit utiliser le contexte d'application **operationalBindingManagementAC**.

25 Rattachement opérationnel hiérarchique non spécifique

Un rattachement opérationnel hiérarchique non spécifique (NHOB) est utilisé pour représenter la relation entre deux DSA détenant deux contextes de dénomination dont l'un est immédiatement subordonné à l'autre. S'agissant d'un NHOB, le DSA supérieur détient une référence subordonnée non spécifique au contexte de dénomination que possède le DSA subordonné; le DSA subordonné détient une référence supérieure immédiate au contexte de dénomination que possède le DSA supérieur. Le rattachement opérationnel garantit que les informations de connaissances appropriées seront échangées et tenues à jour entre les deux DSA de façon que ces deux DSA puissent agir pendant le processus de résolution du nom et d'évaluation des opérations, comme défini dans les § 18 et 19.

25.1 Caractéristiques typiques du rattachement opérationnel

25.1.1 Symétrie et rôles

Le rattachement opérationnel hiérarchique est du type asymétrique. Dans un rattachement de ce type, les deux rôles sont les suivants:

a) le rôle de DSA maître pour le contexte de dénomination supérieur, *le DSA supérieur* (associé au rôle abstrait "A");

b) le rôle de DSA maître pour le contexte de dénomination subordonné, *le DSA subordonné* (associé au rôle abstrait "B").

25.1.2 Accord

L'information d'accord échangée pendant l'établissement du rattachement opérationnel hiérarchique non spécifique, qui est une valeur de **NonSpecificHierarchicalAgreement**, contient uniquement le nom distinctif de l'entrée immédiatement supérieure au nouveau contexte de dénomination (composant **immediateSuperior**). Cette information doit être fournie par le DSA qui entreprend le NHOB.

```
NonSpecificHierarchicalAgreement ::= SEQUENCE {
        immediateSuperior      [1]   DistinguishedName }
```

NOTE – La façon dont le DSA subordonné détermine que le nom du nouveau contexte de dénomination n'est pas ambigu n'entre pas dans le cadre de la présente Recommandation | Norme internationale. Le nom ne sera pas ambigu s'il est correctement assigné par l'autorité chargée de la dénomination correspondante et si aucun autre DSA ne détient le même nom comme entrée principale.

25.1.3 Initiateur

25.1.3.1 Etablissement

L'établissement d'un rattachement opérationnel hiérarchique non spécifique ne peut être entrepris que par le DSA subordonné. Cette action du DSA subordonné (qui connecte une ou plusieurs entrées ou sous-arbres existant localement au DIT global) est déclenchée par une intervention administrative.

25.1.3.2 Modification

La modification d'un rattachement opérationnel hiérarchique non spécifique peut être entreprise par l'un ou l'autre DSA. Le DSA supérieur peut émettre la modification suite à une modification de l'information du préfixe de contexte supérieur. Cela peut résulter de l'une quelconque des opérations de modification, ou d'une intervention administrative.

Par ailleurs, chaque DSA peut modifier le NHOB si l'information de point d'accès pour son contexte de dénomination (ou l'un de ses contextes de dénomination immédiatement subordonnés dans le cas du DSA subordonné) est modifiée.

25.1.3.3 Terminaison

La terminaison d'un rattachement opérationnel hiérarchique peut être entreprise par l'un ou l'autre DSA. Une telle action de la part du DSA supérieur peut résulter d'une intervention administrative et, dans le cas du DSA subordonné, peut résulter d'une opération de suppression d'entrée qui supprime l'entrée du préfixe de contexte final que détient le subordonné immédiatement subordonné au composant **immediateSuperior** de l'accord, ou bien d'une intervention administrative.

25.1.4 Paramètres d'établissement

Le paramètre d'établissement émis par le DSA supérieur, qui est une valeur de **NHOBSuperiorToSubordinate**, équivaut au paramètre d'établissement de HOB correspondant, mais sans la présence du composant **entryInfo**.

```
NHOBSuperiorToSubordinate ::= SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})
```

Le paramètre d'établissement émis par le DSA subordonné, qui est une valeur de **NHOBSubordinateToSuperior**, équivaut au paramètre d'établissement de HOB correspondant, mais sans la présence des composants **alias** et **entryInfo**.

```
NHOBSubordinateToSuperior  ::= SEQUENCE {
        accessPoints        [0]  MasterAndShadowAccessPoints OPTIONAL,
        subentries          [3]  SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }
```

25.1.5 Paramètres de modification

Ces paramètres sont identiques aux paramètres d'établissement correspondants et sont utilisés pour mettre en évidence les modifications apportées aux informations fournies dans les paramètres d'établissement après l'établissement du NHOB.

Si un composant quelconque de **NHOBSuperiorToSubordinate** ou de **NHOBSubordinateToSuperior** subit une modification (par exemple le composant **contextPrefixInfo** de **NHOBSuperiorToSubordinate**), le composant correspondant du paramètre de modification (par exemple le composant **contextPrefixInfo** de **NHOBSuperiorToSubordinate**) est fourni dans son intégralité dans la modification du rattachement opérationnel.

25.1.6 Paramètres de terminaison

Aucun des deux DSA ne fournit de paramètre de terminaison en mettant fin à un NHOB.

25.1.7 Identification du type

Le rattachement opérationnel hiérarchique non spécifique est identifié par l'identificateur d'objet assigné lorsque l'on définit l'objet d'information **nonSpecificHierarchicalOperationalBinding OPERATIONAL-BINDING** (voir § 25.2).

25.2 Définition de la classe d'objets d'information de rattachement opérationnel

Le présent paragraphe définit le type de rattachement opérationnel hiérarchique non spécifique à l'aide du modèle de la classe d'objets d'information **OPERATIONAL-BINDING** défini dans la Rec. UIT-T X.501 | ISO/CEI 9594-2.

```
nonSpecificHierarchicalOperationalBinding  OPERATIONAL-BINDING  ::=  {
        AGREEMENT       NonSpecificHierarchicalAgreement
        APPLICATION CONTEXTS {
                        { directorySystemAC } }
        ASYMMETRIC
            ROLE-A  {           -- DSA  supérieur
                ESTABLISHMENT-PARAMETER     NHOBSuperiorToSubordinate
                MODIFICATION-INITIATOR      TRUE
                MODIFICATION-PARAMETER      NHOBSuperiorToSubordinate
                TERMINATION-INITIATOR       TRUE }
            ROLE-B  {           -- DSA subordonné
                ESTABLISHMENT-INITIATOR     TRUE
                ESTABLISHMENT-PARAMETER     NHOBSubordinateToSuperior
                MODIFICATION-INITIATOR      TRUE
                MODIFICATION-PARAMETER      NHOBSubordinateToSuperior
                TERMINATION-INITIATOR       TRUE }
        ID              id-op-binding-non-specific-hierarchical }
```

25.3 Procédures relatives au DSA pour la gestion des rattachements opérationnels non spécifiques

Dans les procédures suivantes, comme pour les procédures décrites au § 24.3, une nouvelle marque ou une nouvelle DSE créée par un DSA sera stockée dans une mémoire stable.

Dans la procédure d'établissement et de modification décrite ci-après, le DSA jouant le rôle de celui qui répond (c'est-à-dire n'entreprenant pas l'établissement ou la modification) peut fournir au DSA qui joue le rôle initiateur des informations (par exemple attributs opérationnels) qui ne sont pas acceptables pour telle ou telle raison. En pareil cas, le DSA initiateur peut mettre fin au rattachement opérationnel.

25.3.1 Procédure d'établissement

Seul le DSA subordonné peut entreprendre un rattachement opérationnel hiérarchique. Cela pourrait résulter du fait qu'un administrateur souhaite connecter un ou plusieurs sous-arbres d'entrée que possède le DSA à un certain point dans le DIT global. En pareil cas, le DSA subordonné établit un NHOB conformément à la procédure suivante:

1) le DSA subordonné a une DSE de type **cp** dans le cadre d'un contexte de dénomination existant ou crée une nouvelle entrée. Il marque la DSE comme *étant ajoutée*, émet un **bindingID** unique et l'enregistre avec la DSE du préfixe de contexte;

2) le DSA subordonné envoie une opération d'établissement de rattachement opérationnel au DSA supérieur, contenant les paramètres suivants:

   a) **bindingType** mis à **nonSpecificHierarchicalOperationalBindingID**;

   b) paramètre d'établissement **NHOBSubordinateToSuperior**, le cas échéant;

   c) **NonSpecificHierarchicalAgreement** avec le composant **immediateSuperior** mis au nom distinctif du supérieur immédiat de la nouvelle entrée;

   d) paramètres **bindingID**, **myAccessPoint** et **valid**, le cas échéant;

3) le DSA supérieur vérifie qu'il est le maître pour le supérieur immédiat de la nouvelle entrée du préfixe de contexte ou retourne une erreur **operationalBindingError** avec la cause **roleAssignment**;

4) le DSA supérieur ajoute la DSE du type **nssr** (et l'information d'attribut **nonSpecificKnowledge**) à la DSE du supérieur immédiat de la nouvelle entrée, enregistre **bindingID** avec elle et retourne un résultat;

5) si le DSA subordonné reçoit une erreur, il supprime la nouvelle DSE du préfixe de contexte et sa marque. Ce qui arrive à l'information d'entrée à partir de laquelle a été obtenue la DSE du préfixe de contexte relève d'une initiative locale;

   si le DSA subordonné reçoit un résultat, il ajoute les DSE de type **glue**, **subentry**, **admPoint**, **rhob** et **immSupr**, selon le cas, pour représenter l'information **contextPrefixInfo**, et, le cas échéant, une DSE de type **rhob** et **entry** pour représenter l'information **immediateSuperiorInfo**. La marque de la DSE du préfixe de contexte est supprimée;

   en cas de défaillance (par exemple communication du système terminal), le DSA subordonné reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations d'établissement en suspens d'un HOB dont il est l'initiateur.

25.3.2 Procédure de modification

Si le DSA supérieur détecte des modifications dans l'information **NHOBSuperiorToSubordinate** qu'il a fournie à un DSA subordonné dans un rattachement opérationnel hiérarchique non spécifique, il doit diffuser l'information modifiée au DSA subordonné. Si le NHOB a été établi à l'aide des procédures décrites au § 25.3.1, il doit être modifié conformément aux procédures définies pour la modification du HOB au § 24.3.2.1 (en substituant **NHOBSuperiorToSubordinate** à **SuperiorToSubordinateModification**).

De même, si le DSA subordonné détecte des modifications dans l'information **NHOBSubordinateToSuperior** qu'il a fournie à un DSA supérieur, il doit diffuser les modifications au DSA supérieur. Si le NHOB a été établi à l'aide des procédures décrites au § 25.3.1, il doit être modifié conformément aux procédures définies pour la modification du HOB au § 24.3.2.2 (en substituant **NHOBSubordinateToSuperior** à **SubordinateToSuperior**).

25.3.3 Procédure de terminaison

Les procédures suivantes sont définies pour mettre fin à un NHOB qui a été établi à l'aide des procédures décrites au § 25.3.1.

25.3.3.1 Terminaison entreprise par le DSA supérieur

La terminaison d'un rattachement opérationnel hiérarchique n'est entreprise par le DSA supérieur qu'à la suite d'une intervention administrative. La procédure ci-après doit être appliquée:

1) le DSA supérieur marque comme *étant supprimée* la valeur correspondant au DSA subordonné dans l'attribut **nonSpecificKnowledge** que détient la DSE de l'entrée immédiatement supérieure;

2) le DSA supérieur envoie au DSA subordonné une opération de terminaison de rattachement opérationnel pour le NHOB. Le composant **version** de **bindingID** est omis par le supérieur;

3) lorsque le DSA subordonné reçoit la terminaison de rattachement opérationnel, il supprime toute information concernant le NHOB et envoie un résultat, à moins que le composant **identifier** de **bindingID** ne soit inconnu, auquel cas une erreur de rattachement opérationnel est retournée avec la cause **invalidID**. Ce qui arrive à toute information d'entrée associée au contexte de dénomination subordonné relève d'une initiative locale;

4) si le DSA supérieur reçoit un résultat ou une erreur **operationalBindingError** avec la cause **invalidID**, il doit supprimer la valeur de l'attribut **nonSpecificKnowledge** marquée comme *étant supprimée* qui représente l'information de point d'accès associée avec le NHOB, et il supprime toute information concernant le rattachement opérationnel. S'il s'agissait de la dernière valeur de l'attribut **nonSpecificKnowledge**, il supprime l'attribut **nonSpecificKnowledge** et la DSE de type **nssr** dans la DSE;

en cas de défaillance (par exemple communication du système terminal), le DSA supérieur reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations de terminaison en suspens d'un NHOB dont il est l'initiateur.

25.3.3.2 Terminaison entreprise par le DSA subordonné

La terminaison entreprise par le DSA subordonné peut résulter d'une opération de suppression d'entrée qui supprime la dernière entrée dans le contexte de dénomination subordonné (l'entrée du préfixe de contexte) du dernier contexte de dénomination subordonné détenu par le DSA subordonné, ou elle peut résulter d'une intervention administrative. Il convient d'appliquer la procédure ci-après:

1) le DSA subordonné marque la DSE du préfixe de contexte de dénomination comme *étant supprimée*;
2) le DSA subordonné envoie au DSA supérieur une opération de terminaison de rattachement opérationnel pour le HOB. Le composant **version** de **bindingID** est omis par le subordonné;
3) lorsque le DSA supérieur reçoit la terminaison de rattachement opérationnel, il supprime la valeur de l'attribut **nonSpecificKnowledge** qui représente l'information de point d'accès associée avec le NHOB, supprime toute information concernant le rattachement opérationnel, supprime l'attribut **nonSpecificKnowledge** et le type de DSE du type **nssr** dans la DSE immédiatement supérieure au contexte de dénomination subordonné (si la valeur supprimée était la dernière valeur de l'attribut **nonSpecificKnowledge**) et envoie un résultat, à moins que le composant **identifier** de **bindingID** ne soit inconnu, auquel cas une erreur de rattachement opérationnel est retournée avec la cause **invalidID**;
4) si le DSA subordonné reçoit un résultat ou une erreur **operationalBindingError** avec la cause **invalidID**, il supprime toute information concernant le rattachement opérationnel. Ce qui arrive à toute information d'entrée associée au contexte de dénomination subordonné relève d'une initiative locale;

   en cas de défaillance (par exemple communication du système terminal), le DSA subordonné reprend la procédure à partir de l'étape 2) jusqu'à ce qu'il ait reçu un résultat ou une erreur pour toutes les opérations de terminaison en suspens d'un NHOB dont il est l'initiateur.

25.4 Procédures applicables aux opérations

Les opérations qui peuvent être exécutées dans l'état coopératif d'un rattachement opérationnel hiérarchique non spécifique sont définies dans le contexte d'application **directorySystemAC**.

Les procédures que le DSA intervenant dans un rattachement opérationnel hiérarchique non spécifique doit suivre sont définies dans les § 16 à 22.

25.5 Utilisation des contextes d'application

Pour établir, modifier ou mettre fin à un rattachement opérationnel hiérarchique non spécifique à l'aide du protocole et des procédures de la Spécification d'annuaire, un DSA doit utiliser le contexte d'application **operationalBindingManagementAC**.

Annexe A

ASN.1 pour les opérations réparties

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

La présente annexe comprend toutes les définitions de type, de valeur et de macro ASN.1 contenues dans la présente Spécification d'annuaire, sous la forme du module ASN.1 **DistributedOperations**.

```
DistributedOperations  {joint-iso-itu-t ds(5) module(1) distributedOperations(3) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTE TOUT --
-- Les types et les valeurs définis dans ce module sont exportés afin d'être utilisés dans les autres modules ASN.1
-- contenus dans les Spécifications d'annuaire et dans d'autres applications qui s'en serviront pour accéder au service
-- d'annuaire. D'autres applications pourront les utiliser à leurs propres fins mais ces utilisations n'obligeront pas à
-- apporter les extensions et les modifications nécessaires pour tenir à jour ou améliorer le service d'annuaire.

IMPORTS


-- de la Rec. UIT-T X.501 | ISO/CEI 9594-2

        basicAccessControl, commonProtocolSpecification, directoryAbstractService, enhancedSecurity,
        informationFramework,selectedAttributeTypes, serviceAdministration
            FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        DistinguishedName, Name, RDNSequence
            FROM InformationFramework informationFramework

        MRMapping, SearchRuleId
            FROM ServiceAdministration serviceAdministration

        AuthenticationLevel
            FROM BasicAccessControl basicAccessControl

        OPTIONALLY-PROTECTED{ }
            FROM EnhancedSecurity enhancedSecurity

-- de la Rec. UIT-T X.511 | ISO/CEI 9594-3

        abandon, addEntry, CommonResults, compare, directoryBind, list,
        modifyDN, modifyEntry, read, referral, removeEntry, search, SecurityParameters
            FROM DirectoryAbstractService directoryAbstractService

-- de la Rec. UIT-T X.519 | ISO/CEI 9594-5

        ERROR, id-errcode-dsaReferral, OPERATION
            FROM CommonProtocolSpecification commonProtocolSpecification

-- de la Rec. UIT-T X.520 | ISO/CEI 9594-6

        DirectoryString{}, PresentationAddress, ProtocolInformation, UniqueIdentifier
            FROM SelectedAttributeTypes selectedAttributeTypes

        ub-domainLocalID, ub-labeledURI
            FROM UpperBounds upperBounds;

-- type paramétré pour définir des opérations chaînées --
```

```
chained { OPERATION : operation } OPERATION  ::=  {
        ARGUMENT  OPTIONALLY-PROTECTED {
            SET {
                chainedArgument ChainingArguments,
                argument        [0]   operation.&ArgumentType } }
        RESULT    OPTIONALLY-PROTECTED {
            SET {
                chainedResult   ChainingResults,
                result          [0]   operation.&ResultType } }
        ERRORS{ operation.&Errors EXCEPT referral | dsaReferral }
        CODE    operation.&operationCode }

-- opérations de rattachement et de détachement --

dSABind             OPERATION  ::=  directoryBind

--dSAUnbind         OPERATION  ::=  directoryUnbind

-- opérations chaînées --

chainedRead         OPERATION  ::=  chained { read }

chainedCompare      OPERATION  ::=  chained { compare }

chainedAbandon      OPERATION  ::=  abandon

chainedList         OPERATION  ::=  chained { list }

chainedSearch       OPERATION  ::=  chained { search }

chainedAddEntry     OPERATION  ::=  chained { addEntry }

chainedRemoveEntry  OPERATION  ::=  chained { removeEntry }

chainedModifyEntry  OPERATION  ::=  chained { modifyEntry }

chainedModifyDN     OPERATION  ::=  chained { modifyDN }

-- erreurs et paramètres --

dsaReferral   ERROR  ::=  {
        PARAMETER OPTIONALLY-PROTECTED {
            SET {
                reference       [0]   ContinuationReference,
                contextPrefix   [1]   DistinguishedName OPTIONAL,
                COMPONENTS OF CommonResults } }
        CODE        id-errcode-dsaReferral }

-- arguments communs et résultats  --

ChainingArguments  ::=  SET {
        originator                  [0]   DistinguishedName OPTIONAL,
        targetObject                [1]   DistinguishedName OPTIONAL,
        operationProgress           [2]   OperationProgress
                                    DEFAULT { nameResolutionPhase notStarted },
        traceInformation            [3]   TraceInformation,
        aliasDereferenced           [4]   BOOLEAN DEFAULT FALSE,
        aliasedRDNs                 [5]   INTEGER OPTIONAL,
                                -- présent uniquement dans les systèmes première édition
        returnCrossRefs             [6]   BOOLEAN DEFAULT FALSE,
        referenceType               [7]   ReferenceType DEFAU       LT superior,
        info                        [8]   DomainInfo OPTIONAL,
        timeLimit                   [9]   Time OPTIONAL,
        securityParameters          [10]  SecurityParameters DEFAULT { },
        entryOnly                   [11]  BOOLEAN DEFAULT FALSE,
        uniqueIdentifier            [12]  UniqueIdentifier OPTIONAL,
        authenticationLevel         [13]  AuthenticationLevel OPTIONAL,
        exclusions                  [14]  Exclusions OPTIONAL,
```

```
        excludeShadows          [15] BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster     [16] BOOLEAN DEFAULT FALSE,
        operationIdentifier     [17] INTEGER OPTIONAL,
        searchRuleId            [18] SearchRuleId  OPTIONAL,
        chainedRelaxation       [19] MRMapping     OPTIONAL,
        relatedEntry            [20] INTEGER  OPTIONAL,
        dspPaging               [21] BOOLEAN DEFAULT FALSE,
        nonDapPdu               [22] ENUMERATED { ldap (0) } OPTIONAL,
        streamedResults         [23] INTEGER OPTIONAL,
        excludeWriteableCopies       [24] BOOLEAN DEFAULT FALSE }

Time ::= CHOICE {
        utcTime         UTCTime,
        generalizedTime   GeneralizedTime }

DomainInfo   ::=   ABSTRACT-SYNTAX.&Type

ChainingResults ::= SET {
        info                    [0]   DomainInfo OPTIONAL,
        crossReferences         [1]   SEQUENCE SIZE (1..MAX) OF CrossReference OPTIONAL,
        securityParameters      [2]   SecurityParameters DEFAULT { },
        alreadySearched         [3]   Exclusions OPTIONAL }

CrossReference ::= SET {
        contextPrefix     [0]   DistinguishedName,
        accessPoint       [1]   AccessPointInformation }

OperationProgress ::= SET {
        nameResolutionPhase     [0]   ENUMERATED {
            notStarted    (1),
            proceeding    (2),
            completed     (3) },
        nextRDNToBeResolved     [1]   INTEGER OPTIONAL }

TraceInformation ::= SEQUENCE OF TraceItem

TraceItem ::= SET {
      dsa                 [0]   Name,
      targetObject        [1]   Name  OPTIONAL,
      operationProgress   [2]   OperationProgress }

ReferenceType ::= ENUMERATED {
        superior                (1),
        subordinate             (2),
        cross                   (3),
        nonSpecificSubordinate  (4),
        supplier                (5),
        master                  (6),
        immediateSuperior       (7),
        self                    (8),
        ditBridge               (9) }

AccessPoint ::= SET {
      ae-title            [0]   Name,
      address             [1]   PresentationAddress,
      protocolInformation [2]   SET SIZE (1..MAX) OF ProtocolInformation OPTIONAL,
      labeledURI          [6]   LabeledURI OPTIONAL }

LabeledURI ::= DirectoryString{ub-labeledURI}

MasterOrShadowAccessPoint ::= SET {
      COMPONENTS OF       AccessPoint,
      category            [3]   ENUMERATED {
            master              (0),
            shadow              (1) } DEFAULT master,
      chainingRequired    [5]   BOOLEAN  DEFAULT FALSE }

MasterAndShadowAccessPoints ::= SET SIZE (1..MAX) OF MasterOrShadowAccessPoint
```

```
AccessPointInformation ::= SET {
        COMPONENTS OF           MasterOrShadowAccessPoint ,
        additionalPoints   [4]  MasterAndShadowAccessPoints OPTIONAL }

DitBridgeKnowledge ::= SEQUENCE {
        domainLocalID           DirectoryString{ub-domainLocalID} OPTIONAL,
        accessPoints       MasterAndShadowAccessPoints }

Exclusions ::= SET SIZE (1..MAX) OF RDNSequence

ContinuationReference ::= SET {
        targetObject            [0]  Name,
        aliasedRDNs             [1]  INTEGER OPTIONAL, -- présent uniquement dans les systèmes
        operationProgress       [2]  OperationProgress,
        rdnsResolved            [3]  INTEGER OPTIONAL,
        referenceType           [4]  ReferenceType,
        accessPoints            [5]  SET OF AccessPointInformation,
        entryOnly               [6]  BOOLEAN DEFAULT FALSE,
        exclusions              [7]  Exclusions OPTIONAL,
        returnToDUA             [8]  BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster     [9]  BOOLEAN DEFAULT FALSE }

END -- DistributedOperations
```

Annexe B

Exemple de résolution répartie du nom

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

L'exemple donné à la Figure B.1 montre comment la résolution répartie du nom est utilisée pour traiter différentes demandes adressées à l'annuaire. L'exemple est fondé sur le DIT hypothétique et sur la ou les configurations correspondantes de DSA décrites à l'Annexe O de la Rec. UIT-T X.501 | ISO/CEI 9594-2. Cet arbre est reproduit ci-dessous pour faciliter les consultations.

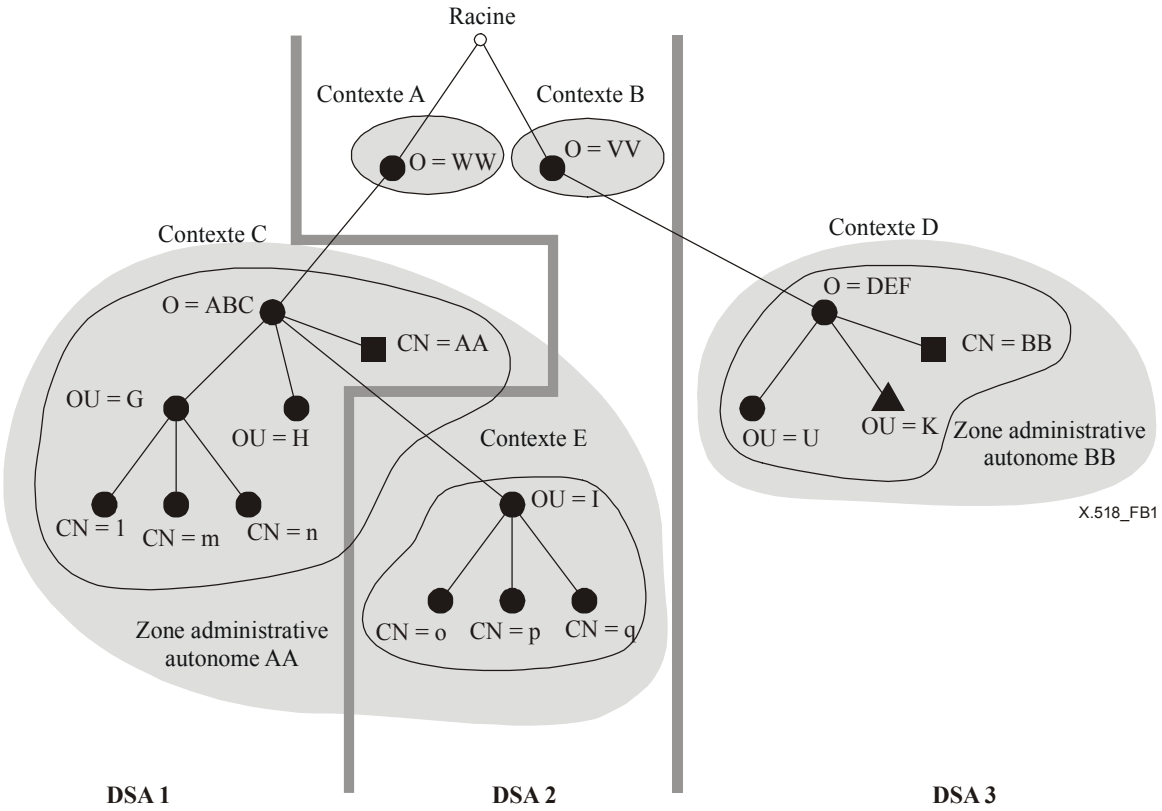


Figure B.1 – DIT hypothétique mappé sur trois DSA

Avec un mode de propagation par chaînage, les demandes suivantes adressées au DSA 1 seront traitées comme suit:

1) une demande avec un nom distinctif {C = WW, O = ABC, OU = G, CN = 1}
   - la résolution du nom assurera la mise en correspondance de chaque RDN dans le nom cible, les DSE étant détenues par le DSA 1, jusqu'à ce que la DSE cible soit localisée;
2) une demande avec un nom distinctif {C = WW, O = JPR}
   - la procédure de résolution du nom dans le DSA 1 mettra en correspondance la DSE C = WW et ne pourra pas assurer d'autres mises en correspondance. A ce stade, le DSA 1 trouve potentiellement deux références qui l'aideront à poursuivre: l'une est la référence **immSupr** dans la DSE C = WW, et l'autre est la référence **supr** dans la DSE racine. Dans cet exemple hypothétique, les deux références donneraient comme indication le DSA 2. En conséquence, la demande est chaînée au DSA 2;
   - dans le DSA 2, la procédure **Name Resolution** assurera la mise en correspondance de la DSE C = WW et ne pourra pas assurer d'autres mises en correspondance. En l'occurrence, étant donné que la DSE C = WW est **cp** et **entry**, et que le DSA 2 est le DSA maître pour cette entrée, et que, par ailleurs, il n'y a pas de **nssr** à C = WW, le DSA 2 peut déterminer qu'il n'y a pas de tel nom dans l'annuaire. Une erreur **nameError** avec un motif **noSuchObject** est retournée;

3) une demande avec un nom distinctif {C = VV, O = DEF, OU = K}

   – la procédure **Name Resolution** dans le DSA 1 ne sera pas en mesure d'assurer la mise en correspondance d'une DSE quelconque. La seule référence disponible est la référence **supr** dans la DSE racine, qui donne comme indication le DSA 2. Ainsi, la demande est chaînée au DSA 2;

   – dans le DSA 2, la procédure **Name Resolution** assurera la mise en correspondance de la DSE C = VV, puis de la DSE O = DEF, et ne pourra pas assurer d'autres mises en correspondance. Etant donné que la DSE O = DEF est du type **subr**, la référence de connaissances spécifique qui donne comme indication le DSA 3, est utilisée et la demande est chaînée au DSA 3;

   – dans le DSA 3, la procédure **Name Resolution** met en correspondance l'intégralité du nom objet cible et constate que la DSE localisée est du type **alias**. En supposant que les alias doivent être déréférencés dans ce cas, on construira un nouveau nom en utilisant le **aliasedEntryName** contenu dans la DSE mise en correspondance. Le DSA 3 reprendra le traitement de la procédure **Name Resolution** pour poursuivre.

Annexe C

Utilisation répartie de l'authentification

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

C.1 Résumé

Le modèle de sécurité est défini au § 17 de la Rec. UIT-T X.501 | ISO/CEI 9594-2. Les principales caractéristiques du modèle sont résumées ci-après:

a) le DSP prend en charge l'authentification renforcée, par signature de la demande, du résultat et des erreurs;

b) le DSP prend en charge la signature de la demande, du résultat et des erreurs.

La présente annexe décrit la manière de réaliser ces caractéristiques dans un annuaire réparti. Elle utilise la terminologie et la notation définies dans la Rec. UIT-T X.509 | ISO/CEI 9594-8.

C.2 Modèle de protection répartie

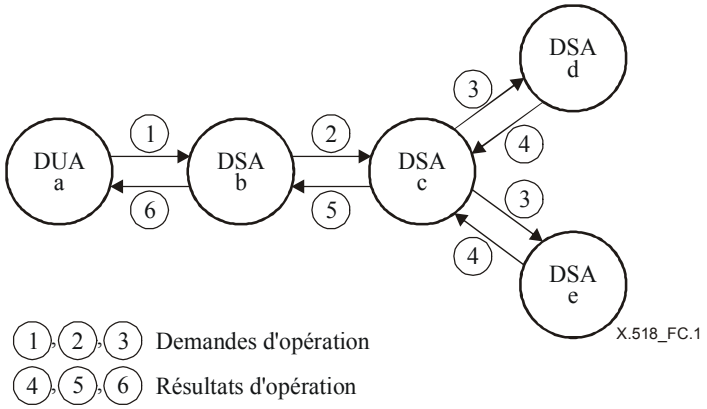


Figure C.1 – Modèle de protection répartie

La Figure C.1 illustre le modèle à utiliser pour spécifier les procédures de protection répartie. Le modèle présente les séquences de flux informationnels dans le cas général d'une opération List ou Search. L'opération est sensée partir du DUA 'a', nommant un objet cible situé dans le DSA 'c'. Les DSA 'b', 'c', 'd' et 'e' sont à impliquer dans le déroulement de l'opération.

Le DUA 'a' commence par entrer en contact avec un DSA, le DSA 'b', qui n'héberge pas l'objet cible, mais qui est en mesure de trouver un chemin, par chaînage, jusqu'au DSA 'c' qui contient l'objet cible. Si tous les DSA travaillaient en mode renvoi de référence, le modèle en serait simplifié de manière significative, chaque échange de DSA à DSA se ramenant, en termes de protection, à une interaction entre le DUA 'a' et le DSA 'b'.

C.2.1 Qualité de protection

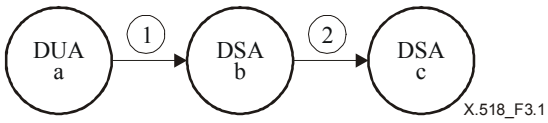
Le niveau de qualité de protection applicable pendant la durée de l'association d'application est déterminé au cours de l'opération Directory Bind. C'est la politique propre au système qui décide du niveau de protection auquel doivent obéir le DUA et les DSA. La classe d'objets d'information **DIRQOP** donne le moyen de spécifier le niveau de protection à associer à chaque opération (demande, résultat ou erreur). Le DUA transmet les objets d'information de la classe **DIRQOP** dans **DirectoryBindArgument**. Le DSA accepte le niveau dans **DirectoryBindResult**. La qualité de protection permet de mettre en œuvre l'un des modes de protection suivants: signé (signed), chiffré (encrypted), ou signé et chiffré (signed and encrypted).

C.3 Opérations chaînées signées

Si les opérations chaînées signées sont prises en charge, le DUA porte la responsabilité de la vérification des signatures renvoyées par le DSA avec le résultat d'une opération List ou Search. Si ces résultats ont été élaborés dans un environnement réparti, il faut que le DUA soit capable de vérifier les signatures de plus d'un DSA. Il est de la responsabilité du DUA d'effectuer la corrélation des résultats des opérations List et Search. Les DSA ne devraient pas fusionner ces résultats pour le compte du DUA. Il peut arriver que le DUA reçoive des informations en provenance de DSA qui traitent des niveaux différents de signature et d'authentification. En cas de non-validité de la signature, c'est le DUA qui décide s'il y a lieu d'utiliser ou non les informations reçues.

C.3.1 Arguments chaînés signés

Si un argument du DAP est signé par le DUA, il convient de conserver la signature pendant toute la durée de vie de la demande. Cette signature peut être vérifiée par les DSA qui peuvent l'utiliser lorsqu'ils exécutent des vérifications de contrôle d'accès. Si le DSA découvre qu'il y a lieu de chaîner la demande pour traitement par un autre DSA, il doit inclure la demande signée du DUA parmi les arguments de chaînage dont il a besoin. Si le DSA était prêt à prendre en charge, de DSA à DSA, des opérations de DSP signées, les justificatifs d'identité du DSA pourraient servir à signer les **chainingArguments** du DSP. Il faudrait alors conserver la signature du DUA en même temps que la demande DAP initiale.

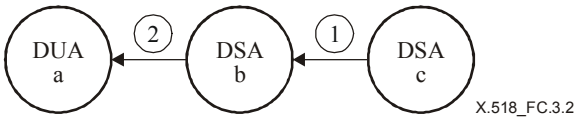


1 L'utilisateur du DUA 'a' scelle la demande DAP

2 Le DSA 'b' scelle le Chaining Argument du DSP (DAP request scellé par l'utilisateur du DUA 'a')

C.3.2 Résultats chaînés signés

Si l'utilisateur du DUA souhaite recevoir de l'annuaire des résultats signés, il faut donner la valeur **SIGNED** au champ **SecurityParameters.ProtectionRequest**. Il importe que le DSA distant soit configurable de telle sorte qu'il puisse signer les **ChainingResults** qu'il envoie. Le DSA distant peut facultativement signer le résultat DAP Result et les **ChainingResults** du DSP, prenant ainsi en charge les signatures de bout en bout. Le DSA 'b' aura la responsabilité de la vérification de la signature DSP du DSA distant, tandis que le DUA 'a' portera celle de la vérification de la signature du résultat DAP du DSA.

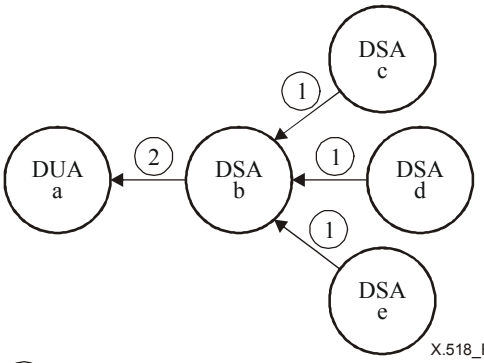


1 Le DSA 'c' scelle le Chaining Result du DSP et le résultat DAP

2 Le DSA 'b' renvoie le résultat DAP scellé par le DSA 'c'

C.3.3 Fusionnement des résultats signés de List ou de Search

Si les résultats de List ou de Search ont été élaborés dans un environnement réparti, il faut que le DUA soit capable de vérifier des signatures provenant de plus d'un DSA. Il est de la responsabilité du DSA d'effectuer la corrélation des résultats de List et de Search. Les DSA ne devraient pas s'en charger pour le compte de l'utilisateur du DUA. Il peut arriver que le DUA reçoive des informations en provenance de DSA qui traitent des niveaux différents de signature et d'authentification. En cas de non-validité de la signature, c'est le DUA qui décide s'il y a lieu d'utiliser ou non les informations reçues.



(1) Les DSA 'c', 'd' et 'e' scellent Chaining Result (le résultat DAP est scellé par les DSA 'c', 'd' et 'e')

(2) Le DSA 'b' renvoie les résultats DAP partiels scellés par les DSA 'c', 'd' et 'e', le DSA 'b' ne fusionne pas les résultats DAP

NOTE – Le protocole DSP de DSA à DSA est aussi susceptible de signature, de chiffrement, ou de signature et de chiffrement.

C.3.4 Demande en chaînage multiple

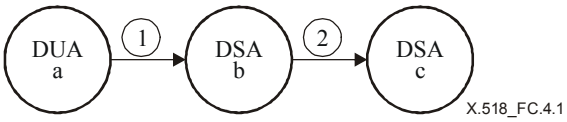
Si le DSA découvre qu'il faut chaîner la demande vers une pluralité d'autres DSA, il peut effectuer le chaînage multiple en parallèle ou en série. Deux modes de décomposition ont été décrits, références subordonnées non spécifiques (NSSR, *non-specific subordinate references*) et décomposition de demande. Dans le cas de NSSR, le DSA expédie la demande identiquement aux autres DSA. Dans le cas de décomposition de demande, il envoie successivement à chacun des autres DSA des demandes partielles qui peuvent être différentes les unes des autres.

C.4 Opérations chaînées chiffrées

Si le chiffrement est pris en charge, il y a lieu de fournir une protection équivalente entre chacun des composants de l'annuaire. Il sort du domaine d'application de la présente Spécification d'étudier les correspondances nécessaires pour établir une équivalence entre les différentes politiques.

C.4.1 Chiffrement point à point (DUA à DSA ou DSA à DSA) de la demande

Si l'utilisateur veut chiffrer la demande DAP, le chiffrement n'est possible que point à point. Le DUA chiffre la demande DAP à destination du DSA 'b', mais l'utilisateur ne sait pas si par la suite la demande sera ou ne sera pas chaînée pour être traitée par un autre DSA. Le DSA 'b' déchiffre la demande et tente de la satisfaire. S'il décide qu'il faut la chaîner pour qu'elle soit traitée par un autre DSA, le DSA 'c' dans l'exemple, il chiffre les opérations chaînées lancées vers le DSA 'c'. Le choix de protection point à point des demandes et des réponses (chained operation arguments and results) est indiqué dans le composant **dirqop** du Bind du DSP échangé entre le DSA 'b' et le DSA 'c'.

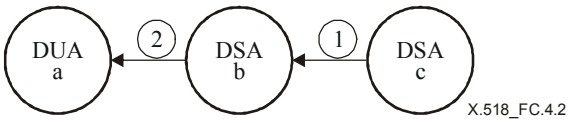


(1) L'utilisateur du DUA 'a' chiffre la demande DAP envoyée au DSA 'b'

(2) Le DSA 'b' chiffre l'argument chained operation du DSP

C.4.2 Chiffrement point à point (DSA à DUA ou DSA à DSA) du résultat

Si l'utilisateur du DUA souhaite recevoir de l'annuaire des résultats ou des indications d'erreur chiffrés, il faut donner la valeur **ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de chained operation Arguments doit refléter la valeur du **DIRQOP** de **BindArgument** du DAP. Il importe que le DSA distant, 'c', soit configurable de telle sorte qu'il puisse chiffrer le chained operation Results qu'il envoie. Dans ce scénario, le DSA 'c' décide qu'il peut satisfaire la demande. Il crée le résultat DAP et le chained operation Results du DSP. Pour réaliser le chiffrement point à point, le DSA 'c' chiffre le chained operation Results du DSP à l'intention du DSA 'b'. Celui-ci peut le déchiffrer et chiffrer le résultat DAP à l'intention de l'utilisateur du DUA 'a'. On a obtenu ainsi le chiffrement point à point du résultat. Le DUA 'a' est chargé de déchiffrer le résultat DAP que lui transmet son DSA local, le DSA 'b'.

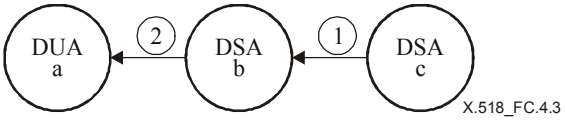


(1) Le DSA 'c' chiffre le chaining operation result

(2) Le DSA 'b' chiffre le résultat DAP à l'intention du DUA 'a'

C.4.3 Chiffrement de bout en bout de DAP Result et chiffrement point à point de DSP chaining Result

Si l'utilisateur du DUA 'a' souhaite recevoir de l'annuaire des résultats ou des indications d'erreur chiffrés, il faut donner la valeur **ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de chained operation Arguments doit refléter la valeur du **DIRQOP** du Bind du DAP. Il importe que le DSA distant, 'c', soit configurable de telle sorte qu'il puisse chiffrer le chained operation Results qu'il envoie. Dans ce scénario, le DSA 'c' décide qu'il peut satisfaire la demande. Il crée un résultat DAP qu'il chiffre de bout en bout à l'intention de l'utilisateur du DUA 'a' et un chained operation Results du DSP qu'il chiffre de point à point. Le DSA 'c' est en mesure d'effectuer un chiffrement de bout en bout car il sait quel est l'utilisateur prévu du DUA 'a'. Le chiffrement point à point des chained operation Results du DSP est effectué par le DSA 'c' à l'intention du DSA 1. DSA 'b' déchiffre le DSP et retransmet au DUA 'a' le résultat DAP chiffré. Le DUA 'a' aura la responsabilité du déchiffrement du résultat DAP qu'il reçoit du DSA 'c' via le DSA 'b'.

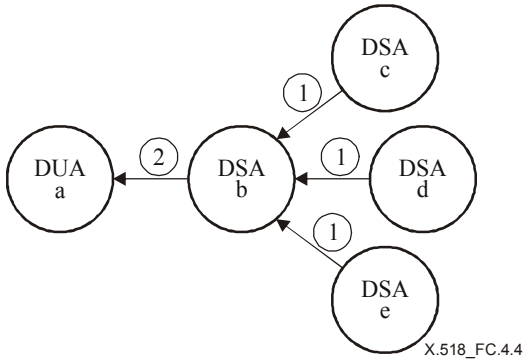


(1) Le DSA 'c' chiffre chained operation result du DSP à l'intention du DSA 'b', qui contient le résultat DAP du DSA 'c' qui a été chiffré à l'intention du DUA 'a'

(2) Le DSA 'b' renvoie le résultat DAP chiffré par le DSA 'c' à l'intention du DUA 'a'

C.4.4 Fusionnement des résultats de List et de Search (fusionnement avec rechiffrement par le DSA 1)

Si l'utilisateur du DUA 'a' souhaite recevoir de l'annuaire des résultats ou des indications d'erreur chiffrés résultant d'opérations List ou Search, il faut donner la valeur **ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de chained operation Arguments doit refléter la valeur du **DIRQOP** du Bind du DAP. Le DSA local, DSA 'b', peut choisir d'effectuer un chaînage multiple, en parallèle ou en série, de la demande de List ou de Search vers plusieurs autres DSA. Il importe que les DSA distants, 'c', 'd' et 'e' soient configurables de telle sorte qu'ils puissent chiffrer les résultats chaînés de List ou de Search qu'ils envoient. Dans ce modèle, chacun des DSA distants 'c', 'd' et 'e' répond à la demande en créant un résultat DAP et un chained operation Results du DSP chiffré. Les chained operation Results que créent les DSA distants 'c', 'd' et 'e' sont transmis au DSA 'b'. Celui-ci reçoit chacun des chained operation Results, les déchiffre et procède à leur assemblage ou à leur fusionnement pour former un résultat commun qu'il chiffre et envoie au DUA utilisateur 'a'. Le chiffrement point à point résulte du chiffrement par les DSA distants des chained operation Results à l'intention du DSA 'b' et du chiffrement par le DSA 'b' du résultat DAP à l'intention de l'utilisateur du DUA 'a'. Le DUA sera responsable du déchiffrement du résultat DAP fusionné unique.



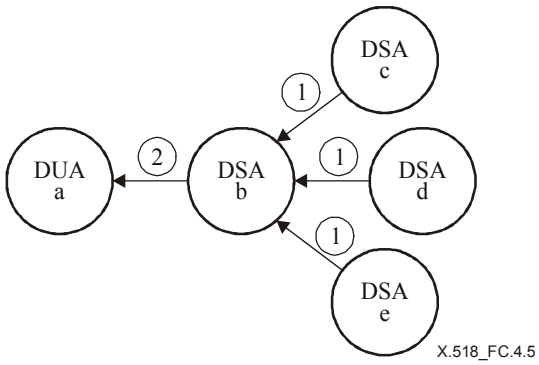
(1) Les DSA 'c', 'd' et 'e' chiffrent les chained operation results du DSP (contenant les résultats DAP)

(2) Le DSA 'b' déchiffre les chained operation results du DSP émis par les DSA 'c', 'd' et 'e', puis fusionne les résultats DAP et rechiffre le résultat DAP à l'intention du DUA 'a'

C.4.5 Interdiction de fusionnement des résultats de List et de Search

(Le DSA 'b' n'exécutant pas le fusionnement, le chiffrement des résultats de List et de Search est de bout en bout)

Si l'utilisateur du DUA souhaite recevoir de l'annuaire des résultats ou des indications d'erreur chiffrés résultant d'opérations List ou Search, il faut donner la valeur **ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de chained operation Arguments doit refléter la valeur du **DIRQOP** du Bind du DAP. Le DSA local, DSA 'b', peut choisir d'effectuer un chaînage multiple, en parallèle ou en série, de la demande de List ou de Search vers plusieurs autres DSA. Il importe que les DSA distants, 'c', 'd' et 'e' soient configurables de telle sorte qu'ils puissent chiffrer les résultats chaînés de List ou de Search qu'ils envoient. Dans ce scénario, chacun des DSA distants 'c', 'd' et 'e' répond à la demande en créant un résultat DAP chiffré à l'intention de l'utilisateur du DUA 'a' et un chained operation Results du DSP chiffré à l'intention du DSA 'b'. Les chained operation Results que créent les DSA distants 'c', 'd' et 'e' sont transmis au DSA 'b'. Celui-ci reçoit chacun des chained operation Results, les déchiffre mais NE procède PAS à l'assemblage et au fusionnement des résultats. Le DSA 'b' retransmet sans changements au DUA 'a' les résultats de List ou de Search que les DSA 'c', 'd' et 'e' ont chiffrés. Le chiffrement de bout en bout résulte du chiffrement par les DSA distants des List/Search Results du DAP à l'intention de l'utilisateur du DUA 'a' et le chiffrement point à point du chiffrement par les DSA distants des chained operation Results à l'intention du DSA 'b'. Le DUA 'a' sera responsable du déchiffrement de chacun des résultats de List ou de Search du DAP.

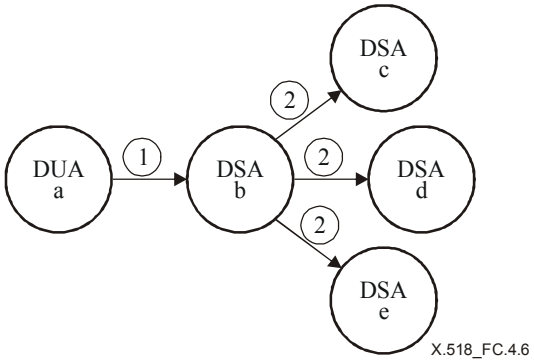


(1) Les DSA 'c', 'd' et 'e' chiffrent les chained operation Results du DSP à l'intention du DSA 'b', contenant le résultat DAP qui a été chiffré à l'intention de l'utilisateur du DUA 'a'

(2) Le DSA 'b' déchiffre les chained operation Results du DSP émis par les DSA 'c', 'd' et 'e', puis retransmet au DUA 'a', sans les déchiffrer ni les fusionner, les résultats DAP qui ont été chiffrés par les DSA 'c', 'd' et 'e'

C.4.6 Chaînage multiple d'une demande DAP avec utilisation d'une clé de chiffrement (clé de réseau)

Si l'utilisateur du DUA 'a' souhaite recevoir de l'annuaire des résultats ou des indications d'erreur chiffrés, il faut donner la valeur **ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de chained operation Arguments doit refléter la valeur du **DIRQOP** du Bind du DAP. Le DSA local, DSA 'b', peut choisir d'effectuer un chaînage multiple, parallèle ou série, de la demande de List ou de Search vers plusieurs autres DSA. Il se peut que la configuration du DSA local, le DSA 'b', lui permette de prendre en charge une clé de chiffrement, la clé de réseau (net-key). Il s'agit d'une clé de chiffrement symétrique commune à tous les DSA de la chaîne. Son emploi permet au DSA 'b' de ne procéder qu'une seule fois au chiffrement de la demande chaînée. Chacun des DSA distants, connaissant la clé de réseau, est en mesure de l'utiliser pour déchiffrer le chained operation Argument du DSP. Dans ce scénario, le chiffrement point à point résulte du chiffrement de la demande DAP par l'utilisateur du DUA 'a' à l'intention du DSA 'b' alors que celui-ci se sert de la clé de réseau pour le chiffrement à l'intention des DSA distants.



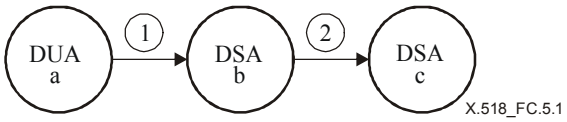
(1) Le DUA 'a' chiffre une demande DAP à l'intention du DSA 'b'

(2) Le DSA 'b' déchiffre la demande et tente d'y répondre. S'il ne le peut, il se sert d'une clé de réseau pour chiffrer la demande chained operation (qui contient la demande DAP). Chained request est envoyé aux DSA 'c', 'd' et 'e'

C.5 Opérations réparties signées et chiffrées

C.5.1 Signature de bout en bout avec chiffrement point à point

Si un utilisateur du DUA 'a' souhaite signer et chiffrer la demande DAP, il est possible de fournir la signature de bout en bout, alors que le chiffrement ne peut s'effectuer que point à point. Le DUA 'a' peut signer et chiffrer la demande DAP à l'intention du DSA 'b', mais l'utilisateur du DUA 'a' ne sait pas si la demande sera ou ne sera pas finalement chaînée pour être traitée par un DSA distant (le DSA 'c'). Le DSA 'b' déchiffre la demande, dont il vérifie la signature. Il tente alors d'y répondre. S'il découvre la nécessité de la chaîner pour qu'un autre DSA ('c') en assure le traitement, il chiffre les **ChainingArguments** du DSP à l'intention du DSA 'c'. La demande DAP d'origine peut être conservée et transmise en même temps que les **ChainingArguments** chiffrés.

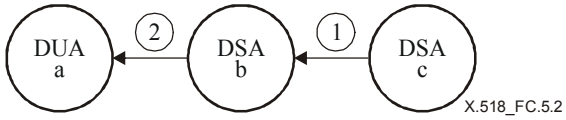


(1) Le DUA 'a' scelle et chiffre la demande DAP à l'intention du DSA 'b'

(2) Le DSA 'b' chiffre la demande DAP et en vérifie le sceau. Après avoir tenté de répondre localement à la demande, le DSA 'b' décide qu'il faut la chaîner vers le DSA 'c'. Le DSA 'b' envoie la demande DAP avec son sceau d'origine (provenant de l'utilisateur du DUA 'a'); il crée et chiffre le Chaining Argument du DSP à l'intention du DSA 'c'

C.5.2 Signature et chiffrement de bout en bout du résultat DAP, signature et chiffrement point à point du DSP

Si l'utilisateur du DUA 'a' souhaite recevoir de l'annuaire des résultats signés et chiffrés, il faut donner la valeur **SIGNED-AND-ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de **ChainingArguments** doit refléter la valeur du **DIRQOP** du Bind du DAP. Il importe que le DSA distant, 'c', soit configurable de telle sorte qu'il puisse signer et chiffrer les opérations chaînées qu'il envoie. Dans ce scénario, le DSA 'c' est capable de satisfaire la demande. Il crée un résultat DAP et exécute un chiffrement de bout en bout à l'intention de l'utilisateur du DUA 'a' et un chiffrement point à point du DSP. Le DSA 'c' est en mesure d'élaborer une signature et un chiffrement de bout en bout car il sait quel est l'utilisateur prévu du DUA 'a'. La signature et le chiffrement point à point des **ChainingResults** du DSP sont effectués par le DSA 'c' à l'intention du DSA 'b'. Celui-ci peut déchiffrer les **ChainingResults** du DSP, vérifier la signature du DSA 'c' et retransmettre le résultat DAP signé et chiffré vers le DUA 'a'. Ce DUA sera responsable du déchiffrement et de la vérification de la signature apposée sur le résultat DAP qu'il reçoit du DSA 'c' via le DSA 'b'.

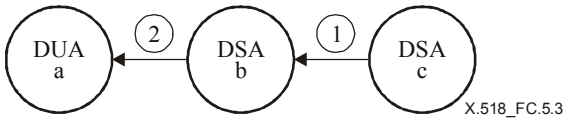


(1) Le DSA 'c' scelle et chiffre à l'intention du DSA 'b' le Chained Result du DSP, contenant les résultats DAP qui sont scellés et chiffrés à l'intention de l'utilisateur DUA 'a'

(2) Le DSA 'b' déchiffre le Chained Result du DSP reçu du DSA 'c' et retransmet au DUA 'a' le résultat DAP scellé et chiffré

C.5.3 Signature de bout en bout du DAP, chiffrement point à point du DSP et du résultat DAP

Si l'utilisateur du DUA 'a' souhaite recevoir de l'annuaire des résultats signés et chiffrés, il faut donner la valeur **SIGNED-AND-ENCRYPTED** au champ **SecurityParameters.ProtectionRequest**. En cas d'absence de ce champ, le champ **SecurityParameters.ProtectionRequest** de **ChainingArguments** doit refléter la valeur du **DIRQOP** du Bind du DAP. Il importe que le DSA distant, 'c', soit configurable de telle sorte qu'il puisse signer et chiffrer les opérations chaînées qu'il envoie. Dans ce modèle, le DSA 'c' est capable de satisfaire la demande. Il crée un résultat DAP signé et élabore une signature et un chiffrement point à point du résultat DAP et des **ChainingResults** du DSP à l'intention du DSA 'b'. Celui-ci peut déchiffrer les **ChainingResults** du DSP, vérifier la signature du DSA 'c' et rechiffrer le résultat DAP signé par le DSA 'c' à l'intention de l'utilisateur du DUA 'a'. Ce DUA sera responsable du déchiffrement du résultat DAP reçu du DSA 'b' et de la vérification de la signature apposée sur le résultat DAP qu'il reçoit du DSA 'c' via le DSA 'b'.



(1) Le DSA 'c' scelle et chiffre à l'intention du DSA 'b' le ChainedResult du DSP, contenant les résultats DAP

(2) Le DSA 'b' déchiffre le Chained Result du DSP reçu du DSA 'c' ainsi que le résultat DAP reçu dans le Chained Result du DSP et retransmet au DUA 'a' le résultat DAP scellé

Annexe D

Spécification des types de rattachement opérationnel hiérarchique et de rattachement opérationnel hiérarchique non spécifique

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

La présente annexe comprend les définitions des classes d'objets d'information contenues dans la présente Spécification d'annuaire sous la forme du module ASN.1 **HierarchicalOperationalBindings**.

```
HierarchicalOperationalBindings
        {joint-iso-itu-t ds(5) module(1) hierarchicalOperationalBindings(20) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTE TOUT --
-- Les types et les valeurs définis dans ce module sont exportés afin d'être utilisés dans les autres modules ASN.1
-- contenus dans les Spécifications d'annuaire et dans d'autres applications qui s'en serviront pour accéder au service
-- d'annuaire. D'autres applications pourront les utiliser à leurs propres fins mais ces utilisations n'obligeront pas à
-- apporter les extensions et les modifications nécessaires pour tenir à jour ou améliorer le service d'annuaire.

IMPORTS

-- de la Rec. UIT-T X.501 | ISO/CEI 9594-2

        directoryOperationalBindingTypes, directoryOSIProtocols, distributedOperations,
        informationFramework, opBindingManagement
            FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        Attribute, DistinguishedName, RelativeDistinguishedName
            FROM InformationFramework informationFramework

        OPERATIONAL-BINDING
            FROM OperationalBindingManagement opBindingManagement

-- de la Rec. UIT-T X.518 | ISO/CEI 9594-4

        MasterAndShadowAccessPoints
            FROM DistributedOperations distributedOperations

-- de la Rec. UIT-T X.519 | ISO/CEI 9594-5

        directorySystemAC
            FROM DirectoryOSIProtocols directoryOSIProtocols

        id-op-binding-hierarchical, id-op-binding-non-specific-hierarchical
            FROM DirectoryOperationalBindingTypes directoryOperationalBindingTypes;

-- types --

HierarchicalAgreement ::= SEQUENCE {
        rdn                 [0]  RelativeDistinguishedName,
        immediateSuperior   [1]  DistinguishedName }

SuperiorToSubordinate ::= SEQUENCE {
        contextPrefixInfo       [0]  DITcontext,
        entryInfo               [1]  SET SIZE (1..MAX) OF Attribute OPTIONAL,
        immediateSuperiorInfo       [2]  SET SIZE (1..MAX) OF Attribute OPTIONAL }

DITcontext ::= SEQUENCE OF Vertex

Vertex ::= SEQUENCE {
        rdn         [0]  RelativeDistinguishedName,
```

```
        admPointInfo [1]   SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries   [2]   SET SIZE (1..MAX) OF SubentryInfo OPTIONAL,
        accessPoints [3]   MasterAndShadowAccessPoints OPTIONAL }

SubentryInfo  ::=  SEQUENCE {
        rdn  [0]   RelativeDistinguishedName,
        info [1]   SET OF Attribute }

SubordinateToSuperior  ::= SEQUENCE  {
        accessPoints [0]   MasterAndShadowAccessPoints OPTIONAL,
        alias              [1]   BOOLEAN DEFAULT FALSE,
        entryInfo    [2]   SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries         [3]   SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }

SuperiorToSubordinateModification  ::=  SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})

NonSpecificHierarchicalAgreement  ::=  SEQUENCE {
        immediateSuperior     [1]   DistinguishedName }

NHOBSuperiorToSubordinate  ::=  SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})

NHOBSubordinateToSuperior  ::= SEQUENCE {
        accessPoints [0]   MasterAndShadowAccessPoints OPTIONAL,
        subentries         [3]   SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }
```

-- objets informationnels de rattachement opérationnel --

```
hierarchicalOperationalBinding  OPERATIONAL-BINDING  ::=  {
        AGREEMENT       HierarchicalAgreement
        APPLICATION CONTEXTS {
                        {directorySystemAC} }
        ASYMMETRIC
            ROLE-A  {           -- DSA supérieur
                ESTABLISHMENT-INITIATOR      TRUE
                ESTABLISHMENT-PARAMETER      SuperiorToSubordinate
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       SuperiorToSubordinateModification
                TERMINATION-INITIATOR        TRUE }
            ROLE-B  {           -- DSA subordonné
                ESTABLISHMENT-INITIATOR      TRUE
                ESTABLISHMENT-PARAMETER      SubordinateToSuperior
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       SubordinateToSuperior
                TERMINATION-INITIATOR        TRUE }
        ID                  id-op-binding-hierarchical }

nonSpecificHierarchicalOperationalBinding  OPERATIONAL-BINDING  ::=  {
        AGREEMENT       NonSpecificHierarchicalAgreement
        APPLICATION CONTEXTS {
                        { directorySystemAC } }
        ASYMMETRIC
            ROLE-A  {           -- DSA supérieur
                ESTABLISHMENT-PARAMETER      NHOBSuperiorToSubordinate
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       NHOBSuperiorToSubordinate
                TERMINATION-INITIATOR        TRUE }
            ROLE-B  {           -- DSA subordonné
                ESTABLISHMENT-INITIATOR      TRUE
                ESTABLISHMENT-PARAMETER      NHOBSubordinateToSuperior
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       NHOBSubordinateToSuperior
                TERMINATION-INITIATOR        TRUE }
        ID                  id-op-binding-non-specific-hierarchical }

END -- HierarchicalOperationalBindings
```

Annexe E

Exemple d'administration des connaissances

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

La présente annexe illustre l'administration des connaissances (définie au § 23) avec un exemple unique. La Figure E.1 contient les symboles utilisés pour décrire les arbres d'information de DSA de cinq DSA.

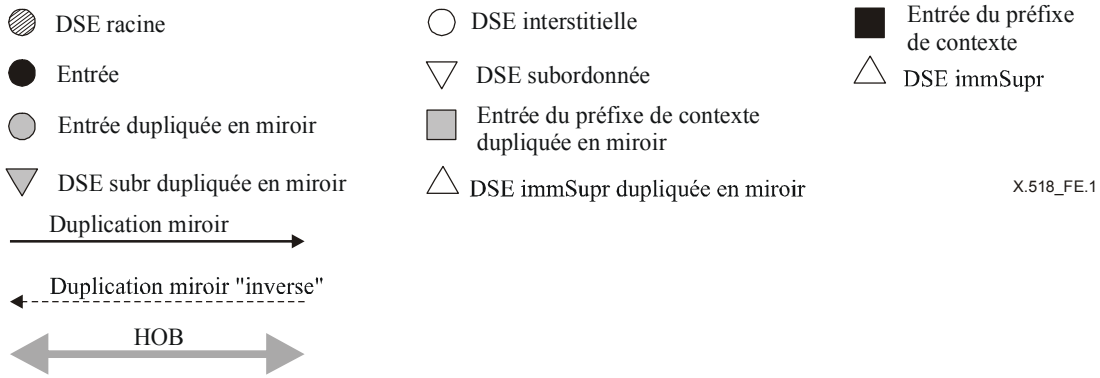


Figure E.1 – Symboles utilisés pour décrire les arbres d'information de DSA

Dans la Figure E.2, le DSA 1 est le maître pour le contexte de dénomination {A}, avec les deux entrées {A} et {A, B}. Le DSA 1 détient une référence subordonnée pour le contexte de dénomination {A, B, C}, dont l'administration est assurée via un HOB avec le DSA 3. Le DSA 1 est un fournisseur d'informations miroirs pour le DSA 2: il lui transmet les copies de l'information d'utilisateur du contexte de dénomination {A} et la référence subordonnée au contexte de dénomination {A, B, C} qui identifie les points d'accès des DSA 3, DSA 4 et DSA 5, le premier étant le maître pour le contexte de dénomination subordonné.

Le DSA 3 est le maître pour le contexte de dénomination {A, B, C}. Cet agent détient non seulement l'entrée unique {A, B, C} du contexte de dénomination, mais encore une référence supérieure immédiate pour le contexte de dénomination {A} dont l'administration est assurée via un HOB avec le DSA 1. Le DSA 3 est un fournisseur d'informations miroirs pour le DSA 4: il lui transmet les copies de l'information d'utilisateur du contexte de dénomination {A, B, C} et la référence supérieure immédiate au contexte de dénomination {A} qui identifie les points d'accès des DSA 1 et DSA 2, ce dernier étant le maître pour le contexte de dénomination supérieur. Le DSA 4 est un fournisseur d'informations miroirs (secondaires) pour le DSA 5, auquel il transmet une copie de l'information qu'il reçoit de la part du DSA 3.

La Figure E.2 illustre les attributs opérationnels de DSA utilisés pour représenter les connaissances et en assurer l'administration.

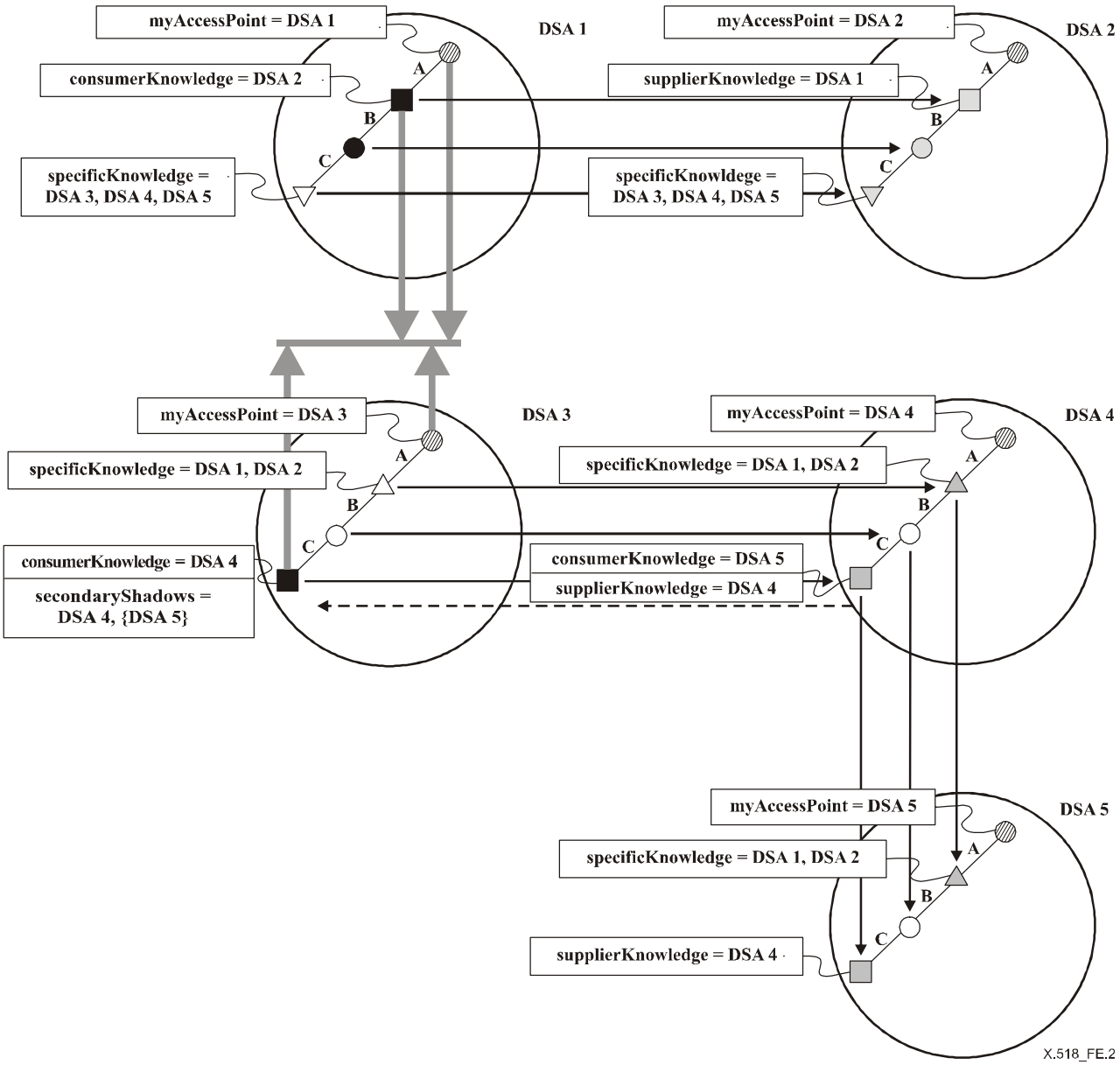


Figure E.2 – Exemple d'administration des connaissances

Le DSA 1 utilise la valeur de son attribut **myAccessPoint** (associé avec sa DSE racine) et les valeurs d'usage commun de son attribut **consumerKnowledge** (associé au préfixe de contexte {A}) pour former une valeur du type **MasterAndShadowAccessPoints** destinée à être utilisée dans les interactions de HOB avec le DSA 3. A son tour, le DSA 3 utilise la valeur de son attribut **myAccessPoint** (associé avec sa DSE racine) et les valeurs d'usage commun de son attribut **consumerKnowledge** ainsi que de son attribut **secondaryShadows** (tous deux associés au préfixe de contexte {A, B, C}) pour former une valeur du type **MasterAndShadowAccessPoints** destinée à être utilisée dans ses interactions de HOB avec le DSA 1. Ensemble, les deux DSA utilisent le DOP pour assurer l'administration d'une référence subordonnée détenue par le DSA 1 et une référence supérieure immédiate détenue par le DSA 3. La référence subordonnée du DSA 1, exprimée par un attribut **specificKnowledge** associé avec une DSE à {A, B, C}, est fondée sur la valeur **MasterAndShadowAccessPoints** qu'elle reçoit de la part du DSA 3; la référence supérieure immédiate du DSA 3, exprimée par un attribut **specificKnowledge** associé avec une DSE à {A}, est également fondée sur la valeur **MasterAndShadowAccessPoints** qu'elle reçoit de la part du DSA 1.

Les DSA 1 et DSA 2 utilisent leurs valeurs de **myAccessPoint** dans les interactions de rattachement opérationnel de duplication miroir pour tenir à jour une valeur de **consumerKnowledge** dans le DSA 1 (identifiant le point d'accès du DSA 2) et de **supplierKnowledge** dans le DSA 2 (identifiant le point d'accès du DSA 1), les deux attributs étant associés au préfixe de contexte {A}. Ensemble les deux DSA utilisent le DOP pour tenir à jour la référence du consommateur détenue par le DSA 1 et la référence du fournisseur détenue par le DSA 2.

Le DSA 2 reçoit une copie de l'attribut **specificKnowledge** associé au préfixe de contexte {A, B, C} de la part du DSA 1 dans les interactions de DISP avec le DSA 1. Ces interactions servent à tenir à jour la référence subordonnée du DSA 2 au préfixe de contexte {A, B, C}.

Les DSA 3 et DSA 4 (et, par la même occasion, les DSA 4 et DSA 5) tiennent à jour les références du consommateur et du fournisseur, respectivement, de manière analogue à l'interaction qui existe entre les DSA 1 et DSA 2.

Le DSA 4 reçoit une copie de l'attribut **specificKnowledge** associé au préfixe de contexte {A4} de la part du DSA 3 dans les interactions de DISP avec le DSA 3. Ces interactions servent à tenir à jour la référence supérieure immédiate du DSA 4 au préfixe de contexte {A}.

Le DSA 4 communique au DSA 3 toutes modifications apportées à ses attributs **myAccessPoint** et **consumerKnowledge** (et de l'attribut **secondaryShadows**, qui est nul dans cet exemple) en utilisant l'opération de modification de rattachement opérationnel du DOP. Le DSA 4 fournit au DSA 3 une valeur de l'ensemble **SupplierAndConsumers**, contenant uniquement les valeurs de l'attribut **consumerKnowledge** qui identifient les points d'accès des DSA qui détiennent des duplications miroirs d'usage commun; les valeurs de l'attribut **secondaryShadows** fournies par le DSA 4, selon le cas, seraient toutes intrinsèquement utilisables de manière générale (dans cet exemple, le DSA 5 est supposé détenir une copie d'usage commun du contexte de dénomination à {A, B, C}). Le DSA 3 utilise cette information pour tenir à jour une valeur de son attribut **secondaryShadows** associé au préfixe de contexte {A, B, C}. Cet attribut, comme indiqué ci-dessus, est utilisé dans les interactions de DOP avec le DSA 1 pour tenir à jour la référence subordonnée du DSA 1 au préfixe de contexte {A, B, C}.

Le DSA 5 tient à jour sa référence supérieure immédiate au préfixe de contexte {A} en utilisant les interactions de DISP avec le DSA 4, de manière analogue aux interactions qui existent entre les DSA 3 et DSA 4.

Annexe F

Amendements et corrigenda

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

Cette édition de la présente Spécification d'annuaire comprend les projets d'amendements suivants qui avaient été votés et approuvés par l'ISO/CEI:

- Amendement 1 pour les extensions de prise en charge des résultats paginés sur le protocole DSP.
- Amendement 3 pour un alignement optimal entre X.500 et protocole LDAP.

Cette édition de la présente Spécification d'annuaire comprend les corrigenda techniques qui corrigent les défauts signalés dans le rapport 307.

SÉRIES DES RECOMMANDATIONS UIT-T

Série A	Organisation du travail de l'UIT-T
Série D	Principes généraux de tarification
Série E	Exploitation générale du réseau, service téléphonique, exploitation des services et facteurs humains
Série F	Services de télécommunication non téléphoniques
Série G	Systèmes et supports de transmission, systèmes et réseaux numériques
Série H	Systèmes audiovisuels et multimédias
Série I	Réseau numérique à intégration de services
Série J	Réseaux câblés et transmission des signaux radiophoniques, télévisuels et autres signaux multimédias
Série K	Protection contre les perturbations
Série L	Construction, installation et protection des câbles et autres éléments des installations extérieures
Série M	Gestion des télécommunications y compris le RGT et maintenance des réseaux
Série N	Maintenance: circuits internationaux de transmission radiophonique et télévisuelle
Série O	Spécifications des appareils de mesure
Série P	Qualité de transmission téléphonique, installations téléphoniques et réseaux locaux
Série Q	Commutation et signalisation
Série R	Transmission télégraphique
Série S	Equipements terminaux de télégraphie
Série T	Terminaux des services télématiques
Série U	Commutation télégraphique
Série V	Communications de données sur le réseau téléphonique
Série X	**Réseaux de données, communication entre systèmes ouverts et sécurité**
Série Y	Infrastructure mondiale de l'information, protocole Internet et réseaux de prochaine génération
Série Z	Langages et aspects généraux logiciels des systèmes de télécommunication

Imprimé en Suisse
Genève, 2006