UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

CCITT

COMITÉ CONSULTATIF
INTERNATIONAL
TÉLÉGRAPHIQUE ET TÉLÉPHONIQUE

LIVRE BLEU

TOME VIII – FASCICULE VIII.8

RÉSEAUX DE COMMUNICATIONS DE DONNÉES

ANNUAIRE

RECOMMANDATIONS X.500 À X.521

IX^e ASSEMBLÉE PLÉNIÈRE
MELBOURNE, 14-25 NOVEMBRE 1988

Genève 1989

UNION INTERNATIONALE DES TÉLÉCOMMUNICATIONS

CCITT

COMITÉ CONSULTATIF
INTERNATIONAL
TÉLÉGRAPHIQUE ET TÉLÉPHONIQUE

LIVRE BLEU

TOME VIII – FASCICULE VIII.8

RÉSEAUX DE COMMUNICATIONS DE DONNÉES

ANNUAIRE

RECOMMANDATIONS X.500 À X.521

IXe ASSEMBLÉE PLÉNIÈRE
MELBOURNE, 14-25 NOVEMBRE 1988

Genève 1989

ISBN 92-61-03732-1

Imprimé en Suisse

CONTENU DU LIVRE DU CCITT
EN VIGUEUR APRÈS LA NEUVIÈME ASSEMBLÉE PLÉNIÈRE (1988)

LIVRE BLEU

Tome I

FASCICULE I.1 – Procès-verbaux et rapports de l'Assemblée plénière.

Liste des Commissions d'études et des Questions mises à l'étude.

FASCICULE I.2 – Vœux et Résolutions.

Recommandations sur l'organisation du travail du CCITT (série A).

FASCICULE I.3 – Termes et définitions. Abréviations et acronymes. Recommandations sur les moyens d'expression (série B) et les Statistiques générales des télécommunications (série C).

FASCICULE I.4 – Index du Livre bleu.

Tome II

FASCICULE II.1 – Principes généraux de tarification – Taxation et comptabilité dans les services internationaux de télécommunications. Recommandations de la série D (Commission d'études III).

FASCICULE II.2 – Service téléphonique et RNIS – Exploitation, numérotage, acheminement et service mobile. Recommandations E.100 à E.333 (Commission d'études II).

FASCICULE II.3 – Service téléphonique et RNIS – Qualité de service, gestion du réseau et ingénierie du trafic. Recommandations E.401 à E.880 (Commission d'études II).

FASCICULE II.4 – Services de télégraphie et mobile. Exploitation et qualité de service. Recommandations F.1 à F.140 (Commission d'études I).

FASCICULE II.5 – Services de télématique, de transmission de données et de téléconférence – Exploitation et qualité de service. Recommandations F.160 à F.353, F.600, F.601, F.710 à F.730 (Commission d'études I).

FASCICULE II.6 – Services de traitement des messages et d'annuaire – Exploitation et définition du service. Recommandations F.400 à F.422, F.500 (Commission d'études I).

Tome III

FASCICULE III.1 – Caractéristiques générales des communications et des circuits téléphoniques internationaux. Recommandations G.100 à G.181 (Commissions d'études XII et XV).

FASCICULE III.2 – Systèmes internationaux analogiques à courants porteurs. Recommandations G.211 à G.544 (Commission d'études XV).

FASCICULE III.3 – Supports de transmission – Caractéristiques. Recommandations G.601 à G.654 (Commission d'études XV).

FASCICULE III.4 – Aspects généraux des systèmes de transmission numériques; équipements terminaux. Recommandations G.700 à G.795 (Commissions d'études XV et XVIII).

FASCICULE III.5 – Réseaux numériques, sections numériques et systèmes de ligne numérique. Recommandations G.801 à G.961 (Commissions d'études XV et XVIII).

FASCICULE III.6 – Utilisation des lignes pour la transmission des signaux autres que téléphoniques. Transmissions radiophoniques et télévisuelles. Recommandations des séries H et J (Commission d'études XV).

FASCICULE III.7 – Réseau numérique avec intégration des services (RNIS) – Structure générale et possibilités de service. Recommandations I.110 à I.257 (Commission d'études XVIII).

FASCICULE III.8 – Réseau numérique avec intégration des services (RNIS) – Aspects généraux et fonctions globales du réseau, interfaces usager-réseau RNIS. Recommandations I.310 à I.470 (Commission d'études XVIII).

FASCICULE III.9 – Réseau numérique avec intégration des services (RNIS) – Interfaces entre réseaux et principes de maintenance. Recommandations I.500 à I.605 (Commission d'études XVIII).

Tome IV

FASCICULE IV.1 – Principes généraux de maintenance, maintenance des systèmes de transmission internationaux et de circuits téléphoniques internationaux. Recommandations M.10 à M.782 (Commission d'études IV).

FASCICULE IV.2 – Maintenance des circuits internationaux télégraphiques, phototélégraphiques et loués. Maintenance du réseau téléphonique public international. Maintenance des systèmes maritimes à satellites et de transmission de données. Recommandations M.800 à M.1375 (Commission d'études IV).

FASCICULE IV.3 – Maintenance des circuits radiophoniques internationaux et transmissions télévisuelles internationales. Recommandations de la série N (Commission d'études IV).

FASCICULE IV.4 – Spécifications des appareils de mesure. Recommandations de la série O (Commission d'études IV).

Tome V – Qualité de la transmission téléphonique. Recommandations de la série P (Commission d'études XII).

Tome VI

FASCICULE VI.1 – Recommandations générales sur la commutation et la signalisation téléphoniques. Fonctions et flux d'information pour les services du RNIS. Suppléments. Recommandations Q.1 à Q.118 *bis* (Commission d'études XI).

FASCICULE VI.2 – Spécifications des Systèmes de signalisation n^{os} 4 et 5. Recommandations Q.120 à Q.180 (Commission d'études XI).

FASCICULE VI.3 – Spécifications du Système de signalisation n° 6. Recommandations Q.251 à Q.300 (Commission d'études XI).

FASCICULE VI.4 – Spécifications des Systèmes de signalisation R1 et R2. Recommandations Q.310 à Q.490 (Commission d'études XI).

FASCICULE VI.5 – Centraux numériques locaux, de transit, combinés et internationaux dans les réseaux numériques intégrés et les réseaux mixtes analogiques-numériques. Suppléments. Recommandations Q.500 à Q.554 (Commission d'études XI).

FASCICULE VI.6 – Interfonctionnement des systèmes de signalisation. Recommandations Q.601 à Q.699 (Commission d'études XI).

FASCICULE VI.7 – Spécifications du Système de signalisation n° 7. Recommandations Q.700 à Q.716 (Commission d'études XI).

FASCICULE VI.8 – Spécifications du Système de signalisation n° 7. Recommandations Q.721 à Q.766 (Commission d'études XI).

FASCICULE VI.9 – Spécifications du Système de signalisation n° 7. Recommandations Q.771 à Q.795 (Commission d'études XI).

FASCICULE VI.10 – Système de signalisation d'abonné numérique n° 1 (SAN 1), couche liaison de données. Recommandations Q.920 à Q.921 (Commission d'études XI).

FASCICULE VI.11 – Système de signalisation d'abonné numérique n° 1 (SAN 1), couche réseau, gestion usager-réseau. Recommandations Q.930 à Q.940 (Commission d'études XI).

FASCICULE VI.12 – Réseau mobile terrestre public, interfonctionnement du RNIS avec le RTPC. Recommandations Q.1000 à Q.1032 (Commission d'études XI).

FASCICULE VI.13 – Réseau mobile terrestre public. Sous-système application mobile et interface associées. Recommandations Q.1051 à Q.1063 (Commission d'études XI).

FASCICULE VI.14 – Interfonctionnement avec les systèmes mobiles à satellites. Recommandations Q.1100 à Q.1152 (Commission d'études XI).

Tome VII

FASCICULE VII.1 – Transmission télégraphique. Recommandations de la série R. Equipements terminaux pour les services de télégraphie. Recommandations de la série S (Commission d'études IX).

FASCICULE VII.2 – Commutation télégraphique. Recommandations de la série U (Commission d'études IX).

FASCICULE VII.3 – Equipements terminaux et protocoles pour les services de télématique. Recommandations T.0 à T.63 (Commission d'études VIII).

FASCICULE VII.4 – Procédures d'essai de conformité pour les Recommandations télétex. Recommandation T.64 (Commission d'études VIII).

FASCICULE VII.5 – Equipements terminaux et protocoles pour les services de télématique. Recommandations T.65 à T.101, T.150 à T.390 (Commission d'études VIII).

FASCICULE VII.6 – Equipements terminaux et protocoles pour les services de télématique. Recommandations T.400 à T.418 (Commission d'études VIII).

FASCICULE VII.7 – Equipements terminaux et protocoles pour les services de télématique. Recommandations T.431 à T.564 (Commission d'études VIII).

Tome VIII

FASCICULE VIII.1 – Communication de données sur le réseau téléphonique. Recommandations de la série V (Commission d'études XVII).

FASCICULE VIII.2 – Réseaux de communications de données: services et facilités, interfaces. Recommandations X.1 à X.32 (Commission d'études VII).

FASCICULE VIII.3 – Réseaux de communications de données: transmission, signalisation et commutation, réseau, maintenance et dispositions administratives. Recommandations X.40 à X.181 (Commission d'études VII).

FASCICULE VIII.4 – Réseaux de communications de données: interconnexion de systèmes ouverts (OSI) – Modèle et notation, définition du service. Recommandations X.200 à X.219 (Commission d'études VII).

FASCICULE VIII.5 – Réseaux de communications de données: interconnexion de systèmes ouverts (OSI) – Spécifications de protocole, essai de conformité. Recommandations X.220 à X.290 (Commission d'études VII).

FASCICULE VIII.6 – Réseaux de communications de données: interfonctionnement entre réseaux, systèmes mobiles de transmission de données, gestion inter-réseaux. Recommandations X.300 à X.370 (Commission d'études VII).

FASCICULE VIII.7 – Réseaux de communications de données: systèmes de messagerie. Recommandations X.400 à X.420 (Commission d'études VII).

FASCICULE VIII.8 – Réseaux de communications de données: annuaire. Recommandations X.500 à X.521 (Commission d'études VII).

Tome IX – Protection contre les perturbations. Recommandations de la série K (Commission d'études V). Construction, installation et protection des câbles et autres éléments d'installations extérieures. Recommandations de la série L (Commission d'études VI).

Tome X

FASCICULE X.1 – Langage de spécification et de description fonctionnelles (LDS). Critères d'utilisation des techniques de description formelles (TDF). Recommandation Z.100 et Annexes A, B, C et E, Recommandation Z.110 (Commission d'études X).

FASCICULE X.2 – Annexe D de la Recommandation Z.100: directives pour les usagers du LDS (Commission d'études X).

FASCICULE X.3 – Annexe F.1 de la Recommandation Z.100: définition formelle du LDS. Introduction (Commission d'études X).

FASCICULE X.4 – Annexe F.2 de la Recommandation Z.100: définition formelle du LDS. Sémantique statique (Commission d'études X).

FASCICULE X.5 – Annexe F.3 de la Recommandation Z.100: définition formelle du LDS. Sémantique dynamique (Commission d'études X).

FASCICULE X.6 – Langage évolué du CCITT (CHILL). Recommandation Z.200 (Commission d'études X).

FASCICULE X.7 – Langage homme-machine (LHM). Recommandations Z.301 à Z.341 (Commission d'études X).

TABLE DES MATIERES DU FASCICULE VIII.8 DU LIVRE BLEU

NOTES PRELIMINAIRES

1 Les questions confiées à chaque Commission d'études pour la période 1989-1992 figurent dans la contribution n° 1 de la Commission correspondante.

2 Dans ce fascicule, l'expression "Administration" est utilisée pour désigner de façon abrégée aussi bien une administration de télécommunications qu'une exploitation privée reconnue de télécommunications.

3 Les termes "annexe" et "appendice" aux Recommandations de la série X ont la signification suivante, sauf indication contraire:

- une annexe à une Recommandation fait partie intégrante de la Recommandation;
- un appendice à une Recommandation ne fait pas partie de la Recommandation, il contient seulement quelques explications ou informations complémentaires spécifiques à cette Recommandation.

4 Les Recommandations de la série X contenues dans le présent fascicule ont été élaborées en collaboration avec l'ISO/CEI. Le tableau ci-après donne la correspondance entre ces Recommandations et les normes ISO/CEI correspondantes.

Recommandations du CCITT	Normes ISO/CEI
X.500	ISO 9594-1, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 1: Présentation générale des concepts, modèles et services [a]
X.501	ISO 9594-2, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 2: Modèles [a]
X.509	ISO 9594-8, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 8: Cadre général d'authentification [a]
X.511	ISO 9594-3, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 3: Définition du service abstrait [a]
X.518	ISO 9594-4, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 4: Procédures de fonctionnement réparti [a]
X.519	ISO 9594-5, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 5: Spécifications du protocole [a]
X.520	ISO 9594-6, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 6: Types d'attribut sélectionné [a]
X.521	ISO 9594-7, Systèmes de traitement de l'information - Interconnexion de systèmes ouverts - L'annuaire - Partie 7: Catégorie d'objet sélectionné [a]

[a] Un scrutin relatif au Projet de Normes internationales (PNI): résultats du vote attendus sous peu.

FASCICULE VIII.8

Recommandations X.500 à X.521

RÉSEAUX DE COMMUNICATIONS DE DONNÉES: ANNUAIRE

PAGE INTENTIONALLY LEFT BLANK

PAGE LAISSEE EN BLANC INTENTIONNELLEMENT

L'ANNUAIRE - APERCU GENERAL DES CONCEPTS, MODELES ET SERVICES [1)]

(Melbourne, 1988)

SOMMAIRE

---

1) La Recommandation X.500 et la norme ISO 9594-1: annuaire-aperçu général des concepts, modèles et services, ont été élaborées en étroite collaboration et sont alignées du point de vue technique.

0 Introduction

0.1 La présente Recommandation, ainsi que les autres Recommandations de la même série, ont été élaborées pour faciliter l'interconnexion des systèmes de traitement de l'information et permettre ainsi d'assurer des services d'annuaire. L'ensemble de ces systèmes, ainsi que les informations d'annuaire qu'ils contiennent, peuvent être considérés comme un tout intégré, appelé l'"annuaire". Les informations contenues dans l'annuaire, appelées collectivement base de données de l'annuaire (DIB), sont généralement utilisées pour faciliter la communication entre des objets tels que entités d'application, individus, terminaux, listes de distribution, ainsi que les communications avec ces objets ou au sujet de ces objets.

0.2 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 de systèmes de traitement de l'information:

- provenant de divers fabricants;
- gérés différemment;
- de niveaux de complexité différents; et
- d'âge différent.

0.3 La présente Recommandation donne une présentation et des modèles des concepts de l'annuaire et de la DIB et décrit les services et les possibilités qu'ils offrent. D'autres Recommandations utilisent ces modèles pour définir le service abstrait fourni par l'annuaire, et pour spécifier les protocoles permettant d'obtenir ou de diffuser ce service.

1 Portée et domaine d'application

1.1 L'annuaire offre les possibilités d'annuaire requises par les applications OSI, les méthodes de gestion OSI, d'autres entités de couche OSI et des services de télécommunication. Parmi les possibilités qu'il offre, citons les désignations faciles à utiliser, c'est-à-dire des noms qui désignent des objets et que les utilisateurs peuvent nommer facilement (bien que tous les objets n'aient pas besoin d'avoir des noms faciles à utiliser); la mise en correspondance nom-adresse grâce à laquelle il existe un lien dynamique entre les objets et leurs emplacements. Cette dernière capacité permet aux réseaux OSI, par exemple, d'être autonomes dans le sens où une adjonction, une suppression ou une modification des emplacements d'objet n'affecte pas le fonctionnement du réseau OSI.

1.2 L'annuaire n'est pas censé être un système de base de données général, bien qu'il puisse être fondé sur ce type de système. On suppose par exemple, comme cela est caractéristique des annuaires de communication, qu'il y a beaucoup plus d'interrogations que de mises à jour. La fréquence des mises à jour dépend normalement de la dynamique des personnes et des organisations et non, par exemple, de la dynamique des réseaux. L'application globale instantanée des mises à jour n'est pas non plus nécessaire: des conditions transitoires dans lesquelles l'ancienne version et la nouvelle version de la même information coexistent, sont tout à fait acceptables.

1.3 Une caractéristique de l'annuaire est que les résultats des interrogations de l'annuaire ne dépendront ni de l'identité ni de l'emplacement du demandeur, sauf si cela découle de droits d'accès différents ou de mises à jour non diffusées. En raison de cette caractéristique, l'annuaire n'est pas approprié pour certaines applications des télécommunications, par exemple certains types d'acheminement.

2 Références

Recommandation X.200 - Modèle de référence pour l'interconnexion des systèmes ouverts.

Recommandation X.208 - Interconnexion des systèmes ouverts - Spécification de la notation 1 de la syntaxe abstraite (ASN.1).

Recommandation X.501 - L'annuaire - Modèles.

Recommandation X.509 - L'annuaire - Cadre d'authentification.

Recommandation X.511 - L'annuaire - Définition du service abstrait.

Recommandation X.518 - L'annuaire - Procédures pour l'exploitation répartie.

Recommandation X.519 - L'annuaire - Spécifications du protocole.

Recommandation X.520 - L'annuaire - Types d'attributs sélectionnés.

Recommandation X.521 - L'annuaire - Catégories d'objets sélectionnées.

Recommandation X.219 - Opérations à distance - Modèle, notation et définition des services.

Recommandation X.229 - Opérations à distance - Spécification du protocole.

3 Définitions

Les définitions contenues dans ce paragraphe utilisent les abréviations définies au § 4.

3.1 *Définitions du modèle de référence OSI*

La présente Recommandation est fondée sur les concepts énoncés dans la Recommandation X.200 et utilise les termes suivants qui y sont définis:

a) *entité d'application*;

b) *couche application*;

c) *processus d'application*;

d) *unité de données de protocole d'application*;

e) *élément de service d'application.*

3.2 *Définitions de base de l'annuaire*

a) *annuaire*: Ensemble de systèmes ouverts coopérant pour assurer des services d'annuaire;

b) *base de données d'annuaire* (*DIB*): Ensemble de données gérées par l'annuaire;

c) *utilisateur* (de l'*annuaire*): Utilisateur final de l'annuaire, c'est-à-dire l'entité ou la personne qui accède à l'annuaire.

3.3 *Définitions du modèle d'annuaire*

La présente Recommandation utilise les termes définis dans la Recommandation X.501:

a) *domaine de gestion d'annuaire d'administration*;

b) *pseudonyme*;

c) *attribut*;

d) *type d'attribut*;

e) *valeur d'attribut*;

f) *arbre d'information de l'annuaire* (*DIT*);

g) *domaine de gestion d'annuaire* (*DMD*);

h) *agent de système d'annuaire* (*DSA*);

i) *agent d'utilisateur d'annuaire* (*DUA*);

j) *nom spécifique*;

k) *entrée*;

l) *nom*;

m) *objet* (*d'intérêt*);

n) *domaine de gestion privé de l'annuaire*;

o) *nom spécifique relatif*;

p) *racine*;

q) *schéma*;

r) *objet subordonné*;

s) *entrée supérieure*;

t) *objet supérieur*;

u) *arbre*.

3.4 *Définitions concernant l'exploitation répartie*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.518:

a) *chaînage*;

b) *destination multiple*;

c) *renvoi*.

4 Abréviations

ADDMD	Domaine de gestion d'annuaire d'Administration
DAP	Protocole d'accès à l'annuaire
DIB	Base de données d'annuaire
DIT	Arbre d'information de l'annuaire
DMD	Domaine de gestion d'annuaire
DSA	Agent du système d'annuaire
DSP	Protocole du système d'annuaire
DUA	Agent d'utilisateur de l'annuaire
OSI	Interconnexion des systèmes ouverts
PRDMD	Domaine de gestion d'annuaire privé
RDN	Nom spécifique relatif

5 Aperçu général de l'annuaire

5.1 L'*annuaire* est un ensemble de systèmes ouverts qui coopèrent pour établir une base de données logique contenant des informations sur un ensemble d'objets dans le monde réel. Les *utilisateurs* de l'annuaire, qu'il s'agisse de personnes ou de programmes d'ordinateurs, peuvent lire ou modifier l'information, ou une partie de celle-ci, à condition qu'ils soient autorisés à le faire. Pour accéder à l'annuaire, chaque utilisateur est représenté par un agent d'utilisateur d'annuaire (DUA), qui est considéré comme un processus d'application. Ces concepts sont illustrés à la figure 1/X.500.

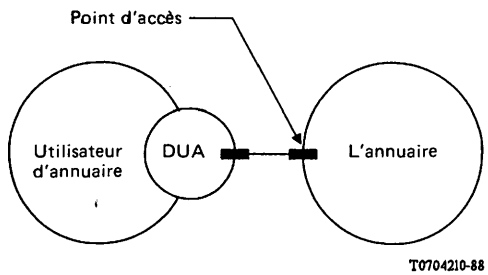


FIGURE 1/X.500

Accès à l'annuaire

Remarque - Cette série de Recommandations s'applique à l'annuaire au singulier et traduit l'intention de créer, par l'intermédiaire d'un espace de nom simple, unifié, un annuaire logique comprenant de nombreux systèmes et destiné à de nombreuses applications. La question de savoir si ces systèmes choisissent l'interfonctionnement dépendra des besoins des applications qu'ils assurent. Les applications traitant de mondes d'objets qui ne se croisent pas n'auront peut-être pas ce besoin. L'espace de nom unique facilite l'interfonctionnement ultérieur au cas où les besoins changeraient.

5.2 L'information contenue dans l'annuaire est appelée collectivement *base de données d'annuaire* (DIB). Le § 6 de la présente Recommandation donne un aperçu global de sa structure.

5.3 L'annuaire offre à ses utilisateurs un ensemble bien défini de capacités d'accès, appelé service abstrait de l'annuaire. Ce service, qui est décrit de façon générale au § 7 de la présente Recommandation, offre une capacité simple de modification et d'extraction. Elle peut être établie avec des fonctions DUA locales pour offrir les capacités requises par les utilisateurs finals.

5.4 Il est probable que l'annuaire sera distribué, peut-être même très largement, aux niveaux fonctionnel et organisationnel. Le § 8 décrit les modèles correspondants de l'annuaire. Ils ont été élaborés afin de fournir un cadre pour que la coopération des divers éléments forme un tout intégré.

5.5 La mise à disposition et l'utilisation des services d'annuaire exigent que les utilisateurs (en réalité les DUA) et les divers éléments fonctionnels de l'annuaire coopèrent les uns avec les autres. Très souvent, il sera nécessaire d'établir une coopération entre les processus d'application dans les différents systèmes ouverts, puis d'utiliser les protocoles d'application normalisés décrits au § 9, pour régir cette coopération.

5.6 L'annuaire a été conçu de façon à assurer des applications multiples, choisies parmi une vaste gamme de possibilités. La nature des applications assurées décidera des objets qui seront énumérés dans l'annuaire, des utilisateurs qui accéderont à l'information, et des types d'accès qui seront offerts. Les applications peuvent être très spécifiques (établissement de listes de distribution pour le courrier électronique) ou génériques (application d'annuaire de communications interpersonnelles). L'annuaire offre la possibilité d'exploiter des éléments communs aux différentes applications:

- un simple objet peut convenir pour plusieurs applications; peut-être même un même élément d'information concernant un même objet peut être approprié.

Pour cela, un certain nombre de catégories d'objet et de types d'attribut sont définis; ils seront utiles pour toute une gamme d'applications. Ces définitions figurent dans les Recommandations X.520 et X.521:

- certains schémas d'utilisation de l'annuaire seront communs à une gamme d'applications: ce sujet est étudié de façon plus approfondie à l'annexe A.

6 Base de données de l'annuaire (DIB)

Remarque - La DIB et sa structure sont définies dans la Recommandation X.501.

6.1 La DIB est un ensemble de données sur des objets. Elle est composée *d'entrées* (*d'annuaire*), chacune comprenant un ensemble d'informations sur un objet. Chaque entrée est composée *d'attributs*, chacun ayant un type et une ou plusieurs valeurs. Les types d'attribut qui sont présents dans une entrée donnée dépendent de la *catégorie* d'objet que l'entrée décrit.

6.2 Les entrées de la DIB sont présentées sous forme d'arbre, l'arbre d'information de l'annuaire (DIT) dont les sommets représentent les entrées. Les entrées se trouvant près de la racine de l'arbre représenteront souvent des objets tels que des pays ou des organisations, alors que les entrées plus éloignées de la racine représenteront des personnes ou des processus d'application.

Remarque - Les services définis dans la présente Recommandation ne fonctionnent que d'après une structure d'arbre (DIT). La présente Recommandation n'exclut pas l'existence, à l'avenir, d'autres structures selon les besoins.

6.3 Chaque entrée a un *nom spécifique*, qui identifie l'entrée de façon unique et non ambiguë. Les caractéristiques du nom spécifique découlent de la structure d'arbre de l'information. Le nom spécifique d'une entrée est composé du nom spécifique ou de son entrée supérieure, ainsi que des valeurs d'attribut spécialement désignées (les valeurs *spécifiques*) de l'entrée.

6.4 Certaines entrées se trouvant au niveau des feuilles de l'arbre sont des entrées pseudonymes alors que toutes les autres entrées sont des entrées objets. Les entrées pseudonymes annoncent les entrées d'objet, et constituent la base d'autres noms pour les objets correspondants.

6.5 L'annuaire applique un ensemble de règles pour s'assurer que la DIB reste bien formée face aux modifications qui interviennent dans le temps. Ces règles, appelées le *schéma de l'annuaire*, empêchent que les entrées aient des types d'attribut qui ne conviennent pas pour la classe d'objet, que les valeurs d'attribut aient une forme incorrecte pour le type d'attribut, et même que les entrées aient des entrées subordonnées de la mauvaise catégorie.

6.6 La figure 2/X.500 illustre les concepts ci-dessus de la DIB et de ses éléments.

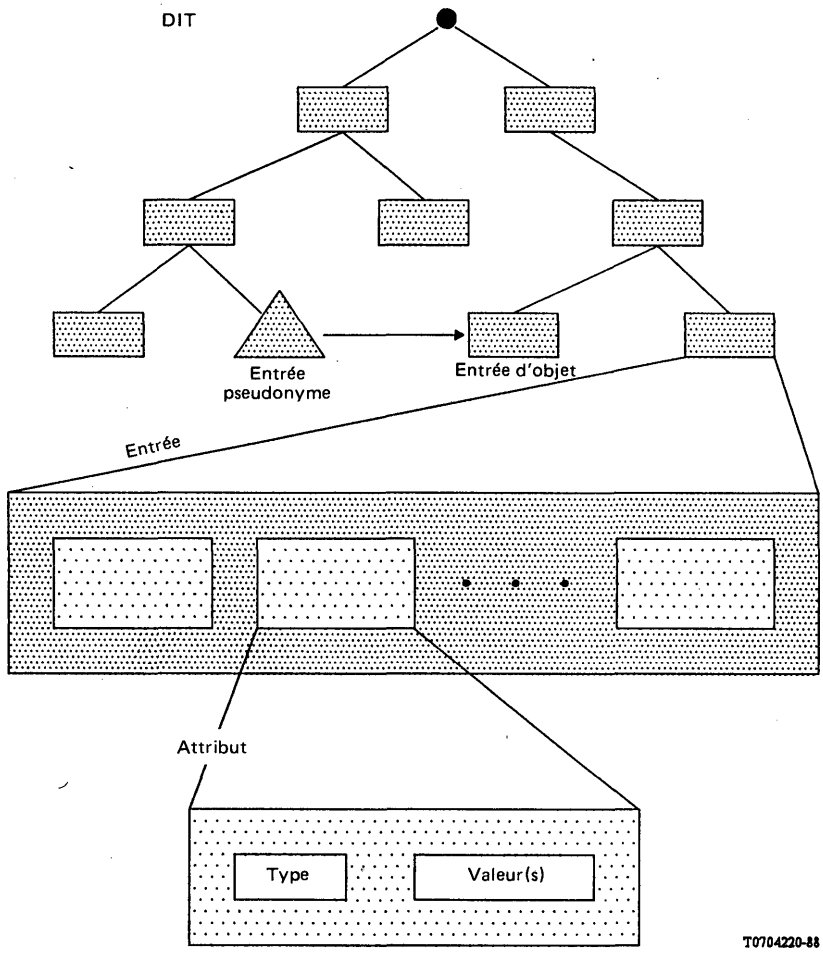


FIGURE 2/X.500

Structure du DIT et des entrées

6.7 La figure 3/X.500 donne un exemple hypothétique d'un DIT. L'arbre donne des exemples de certaines des catégories d'attributs utilisées pour identifier différents objets. Par exemple le nom:

{C = GB, L = Winslow, O = Graphic Services, CN = Laser Printer}

identifie l'entité d'application "imprimante à laser" qui a, dans son nom spécifique, l'attribution géographique de la localité. La personne résidentielle John Jones, dont le nom est:

{C = GB, L = Winslow, CN = John Jones}

a le même attribut géographique dans son nom spécifique.

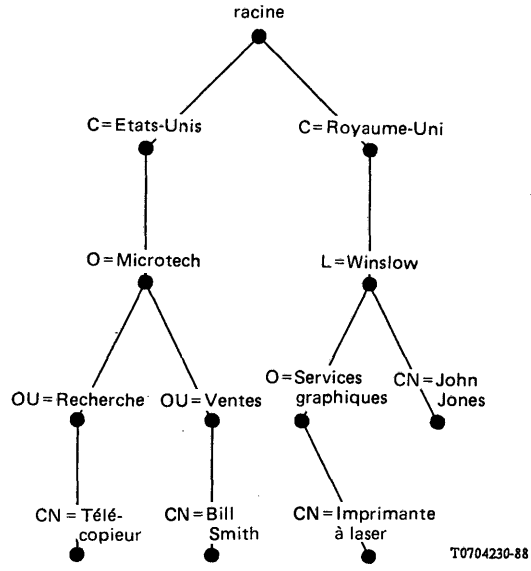


FIGURE 3/X.500

Arbre hypothétique d'information de l'annuaire

6.8 La croissance et la forme du DIT, la définition du schéma d'annuaire et la sélection des noms spécifiques pour des entrées, à mesure qu'elles sont ajoutées, relèvent de la compétence des diverses autorités, dont la relation hiérarchique est reflétée par la forme de l'arbre. Les autorités doivent s'assurer, par exemple, que toutes les entrées dépendant de leur juridiction, ont des noms spécifiques non ambigus, en gérant minutieusement les types d'attributs et les valeurs qui apparaissent dans ces noms. La responsabilité est transmise, comme le montre l'arbre, des autorités supérieures aux autorités subordonnées, le contrôle étant exercé au moyen du schéma.

7 Le service d'annuaire

Remarque - La définition du service abstrait de l'annuaire figure dans la Recommandation X.511.

7.1 *Introduction*

7.1.1 Ce paragraphe donne un aperçu général du service qu'offre l'annuaire aux usagers, représentés par leurs DUA. Tous les services sont fournis par l'annuaire en réponse aux demandes provenant des DUA. Il y a des demandes qui permettent l'interrogation de l'annuaire, comme décrit au § 7.3, et des demandes de modification, comme décrit au § 7.4. De plus, les demandes de service peuvent être qualifiées, comme indiqué au § 7.2. L'annuaire indique toujours les résultats de chaque demande faite. La forme des résultats normaux est propre à chaque demande et est évidente d'après la description de la demande. La plupart des résultats anormaux sont communs à plusieurs réponses. Les possibilités sont décrites au § 7.5.

7.1.2 Plusieurs aspects du service d'annuaire éventuel ne sont pas actuellement traités par les normes spécifiées dans la présente série de Recommandations. Les capacités correspondantes devront donc être fournies à titre de fonction locale jusqu'à ce qu'une solution normalisée soit disponible, à savoir:

- l'adjonction et la suppression d'entrées arbitraires, ce qui permet d'établir un annuaire réparti;
- la gestion de commande d'accès (c'est-à-dire octroi ou retrait de permission à un usager donné d'accéder à une information donnée);
- la gestion du schéma d'annuaire;
- la gestion de l'information de connaissance;
- la copie de parties de la DIB.

Remarque - Cette liste n'est pas nécessairement complète.

7.1.3 L'annuaire est conçu de façon que les modifications apportées à la DIB, qu'elles soient le résultat d'une demande de service d'annuaire ou d'autres moyens (locaux), permettent à celle-ci de continuer à respecter les règles du schéma d'annuaire.

7.1.4 Un utilisateur et l'annuaire sont liés pendant un certain temps à un point d'accès à l'annuaire. Au moment où ils se lient, l'utilisateur et l'annuaire peuvent, en option, vérifier leur identité respective.

7.2 *Qualifications du service*

7.2.1 *Commandes de services*

Diverses commandes peuvent être appliquées aux différentes demandes de service, avant tout pour permettre à l'utilisateur d'imposer à l'annuaire des limites à ne pas dépasser quant à l'utilisation des ressources. Des commandes sont prévues concernant notamment: la durée, l'ampleur des résultats, la portée de la recherche, les modes d'interaction et la priorité de la demande.

7.2.2 *Paramètres de sécurité*

Chaque demande peut être accompagnée de données fournies à l'appui des mécanismes de sécurité pour protéger l'information d'annuaire. Ces données peuvent inclure: la demande de divers types de protection faite par l'utilisateur, une signature numérique de la demande, ainsi que des données pour aider la partie correcte à vérifier la signature.

7.2.3 *Filtres*

Certaines demandes dont le résultat dépend d'informations provenant d'un certain nombre d'entrées ou les concernant, peuvent être accompagnées d'un filtre. Un filtre exprime une ou plusieurs conditions auxquelles une entrée doit satisfaire afin d'être retournée comme une partie du résultat. Cela permet de ne restituer que les entrées appropriées.

7.3 *Interrogation de l'annuaire*

7.3.1 *Lecture*

Une demande de lecture vise une entrée particulière et implique la restitution des valeurs de certains ou de l'ensemble des attributs de cette entrée. Lorsque seuls certains attributs doivent être retournés, le DUA fournit la liste des types d'attributs en question.

7.3.2 *Comparaison*

Une demande de comparaison vise un attribut particulier d'une entrée donnée et oblige l'annuaire à vérifier si une valeur donnée correspond à une valeur de cet attribut.

Remarque - Par exemple, on peut l'utiliser pour vérifier un mot de passe dans le cas où ce dernier, qui figure dans l'annuaire, risque d'être inaccessible pour la lecture, mais accessible pour la comparaison.

7.3.3 *Liste*

Une demande de liste oblige l'annuaire à restituer la liste des subordonnés immédiats d'une entrée désignée dans le DIT.

7.3.4 *Recherche*

Une demande de recherche oblige l'annuaire à restituer l'information provenant de toutes les entrées dans une certaine partie du DIT satisfaisant à un filtre. L'information provenant de chaque entrée comprend une partie ou l'ensemble des attributs de cette entrée, comme pour la lecture.

7.3.5 *Abandon*

Une demande d'abandon, appliquée à une demande d'interrogation en instance, informe l'annuaire que l'expéditeur de la demande ne désire plus qu'il soit donné suite à sa demande. L'annuaire peut, par exemple, arrêter le traitement de la demande et annuler les résultats déjà obtenus.

7.4 *Modification de l'annuaire*

7.4.1 *Adjonction d'entrée*

Une demande d'adjonction d'entrée entraîne l'adjonction au DIT d'une nouvelle entrée feuille (soit une entrée objet, soit une entrée pseudonyme).

Remarque - Tel qu'il se présente actuellement, ce service est destiné à ajouter des entrées qui resteront sous forme de feuilles, comme des entrées pour des personnes ou des entités d'application, et non à ajouter des sous-arbres entiers par des applications répétées de ce service. On envisage d'améliorer ce service à l'avenir en le généralisant.

7.4.2 *Suppression d'entrée*

Une demande de suppression d'entrée oblige à retirer l'entrée feuille du DIT.

Remarque - Comme pour l'adjonction d'entrée, ce service est utilisé actuellement pour les entrées "vraie feuille"; on l'améliorera à l'avenir en le généralisant.

7.4.3 *Modification d'entrée*

Une demande de modification d'entrée oblige l'annuaire à apporter une série de modifications à une entrée donnée. Il apporte soit toutes les modifications soit aucune modification et le DIB reste toujours dans un état compatible avec le schéma. Les modifications autorisées comprennent l'adjonction, la suppression ou le remplacement d'attributs ou de valeurs d'attribut.

7.4.4 *Modification du nom spécifique relatif*

Une demande de modification du nom spécifique relatif (RDN) a pour effet que le nom spécifique relatif d'une entrée feuille (entrée d'objet ou entrée pseudonyme) du DIT est modifié par la désignation de différentes valeurs d'attribut spécifique.

7.5 *Autres résultats*

7.5.1 *Erreurs*

Un service peut connaître une défaillance, par exemple en raison de problèmes posés par les paramètres fournis par l'utilisateur, auquel cas une erreur est signalée. L'information est retournée avec l'erreur, lorsque cela est possible, pour aider à résoudre le problème. Toutefois, en général, seule la première erreur rencontrée par l'annuaire est signalée. En dehors de l'exemple mentionné ci-dessus concernant les problèmes que posent les paramètres fournis par l'utilisateur (en particulier les noms non valables pour les entrées ou les types d'attributs non valables), les erreurs peuvent provenir de violations des principes de sécurité, des règles de schéma et des commandes de service.

7.5.2 *Renvois*

Un service peut échouer parce que le point d'accès auquel le DUA est lié n'est pas celui qui convient le mieux pour exécuter la demande, par exemple du fait que l'information affectée par la demande est (logiquement) très éloignée du point d'accès. En pareil cas, l'annuaire peut retourner un renvoi, qui suggère un point d'accès de remplacement auquel le DUA peut faire sa demande.

Remarque - L'annuaire et le DUA peuvent avoir chacun une préférence quant à l'utilisation des renvois ou au *chaînage* des demandes (voir le § 8.3.3.2). Le DUA peut exprimer sa préférence au moyen de commandes de service. L'annuaire décide finalement de la solution à appliquer.

8 Annuaire réparti

Remarque - Les modèles d'annuaire sont définis dans la Recommandation X.501, alors que les procédures d'exploitation de l'annuaire réparti sont spécifiées dans la Recommandation X.518.

8.1 *Modèle fonctionnel*

Le modèle fonctionnel des annuaires est présenté dans la figure 4/X.500.

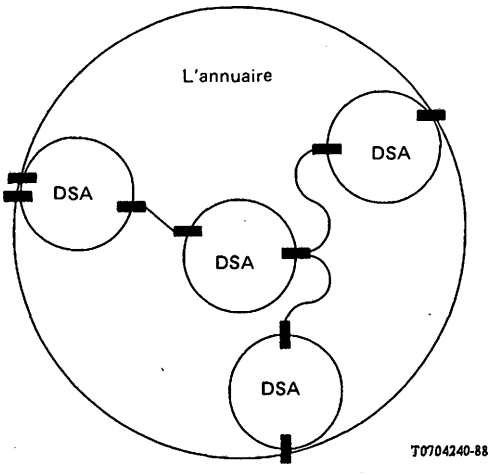


FIGURE 4/X.500

Modèle fonctionnel de l'annuaire

Un *agent de système d'annuaire* (DSA) est un processus d'application OSI qui fait partie de l'annuaire et dont le rôle est d'assurer un accès à la DIB, aux DUA et/ou à d'autres DSA. Un DSA peut utiliser l'information enregistrée dans sa base de données locale, ou interagir avec d'autres DSA pour effectuer des demandes. Par ailleurs, le DSA peut diriger un demandeur vers un autre DSA qui peut aider à effectuer la demande. Les bases de données locales dépendent entièrement de la mise en oeuvre.

8.2 *Modèle organisationnel*

8.2.1 Un ensemble d'un ou de plusieurs DSA et d'aucun ou de plusieurs DUA gérés par une organisation simple peut former un domaine de gestion d'annuaire (DMD). L'organisation en question peut choisir d'utiliser ou non cette série de Recommandations pour régir les communications entre les éléments fonctionnels dans le DMD.

8.2.2 Les Recommandations subséquentes de la présente série spécifient certains aspects du comportement des DSA. A cet égard, un groupe de DSA dans un DMD peut, selon l'option de l'organisation qui dirige le DMD, se comporter comme un simple DSA.

8.2.3 Un DMD peut être un DMD d'Administration (ADDMD), ou un DMD privé (PRDMD), selon qu'il est exploité ou non par une entreprise de télécommunications publique.

Remarque - Il convient de reconnaître que l'appui fourni par les membres du CCITT pour les systèmes d'annuaire privés entre dans le cadre des réglementations nationales. Ainsi, les possibilités techniques décrites peuvent être offertes ou non par une Administration qui assure des services d'annuaire. Le fonctionnement interne et la configuration des DMD privés n'entrent pas dans le cadre des Recommandations prévues par le CCITT.

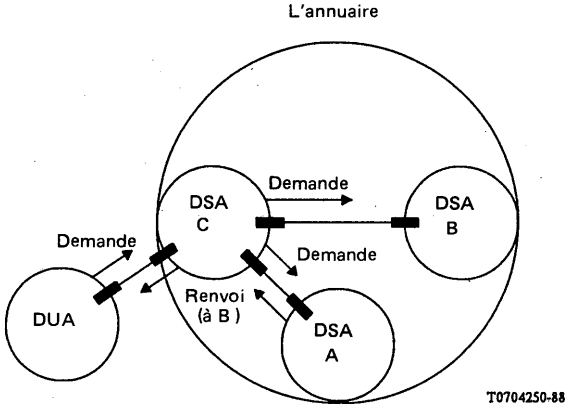
8.3 *Fonctionnement du modèle*

8.3.1 Le DUA dialogue avec l'annuaire en communiquant avec un ou plusieurs DSA. Un DUA n'a pas besoin d'être lié à un DSA particulier. Il peut entrer en interaction directe avec divers DSA pour faire des demandes. Pour des raisons administratives, une interaction directe avec le DSA qui a besoin d'exécuter la demande est parfois impossible, par exemple pour retourner une information d'annuaire. Il peut aussi se faire que le DUA accède à l'annuaire par un simple DSA. A cette fin, une interaction des DSA entre eux est nécessaire.

8.3.2 Le DSA est chargé de mener à bien les demandes des DUA et d'obtenir les informations nécessaires dont il ne dispose pas. Il peut prendre la responsabilité d'obtenir les informations en interagissant avec d'autres DSA pour le compte du DUA.

8.3.3 Plusieurs cas de traitement de demandes ont été recensés (voir les figures 5/X.500 à 7/X.500) et décrits ci-dessous.

8.3.3.1 Dans la figure 5a/X.500, le DSA C reçoit un renvoi d'un DSA A et est chargé d'acheminer la demande directement au DSA B (désigné dans le renvoi du DSA A) ou d'acheminer le renvoi au DUA d'origine.



Remarque - Si le DSA C retourne le renvoi au DUA, la "demande (à B)" n'a pas lieu. De même, si le DSA C transmet la demande au DSA B, il ne retournera pas un renvoi au DUA.

FIGURE 5a/X.500

Renvois

Dans la figure 5b/X.500, le DUA reçoit le renvoi du DSA C et il est chargé de réémettre la demande directement au DSA A (désigné dans le renvoi du DSA C).

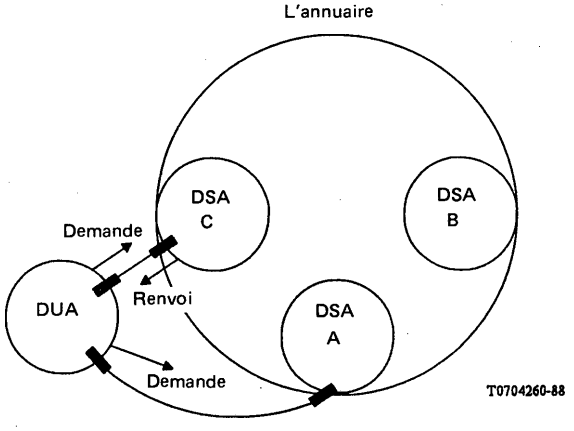


FIGURE 5b/X.500

Renvois

8.3.3.2 La figure 6/X.500 montre le chaînage DSA, qui permet à la demande d'être transmise à travers plusieurs DSA avant que la réponse soit envoyée.

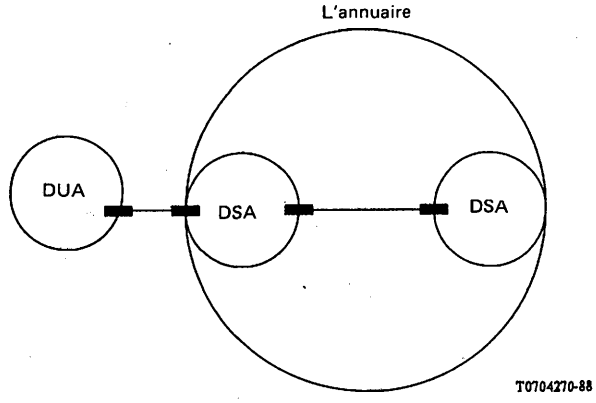


FIGURE 6/X.500

Chaînage

8.3.3.3 La figure 7/X.500 montre la destination multiple dans laquelle le DSA associé au DUA exécute la demande en l'envoyant à deux autres DSA ou plus, la demande à chaque DSA étant identique.

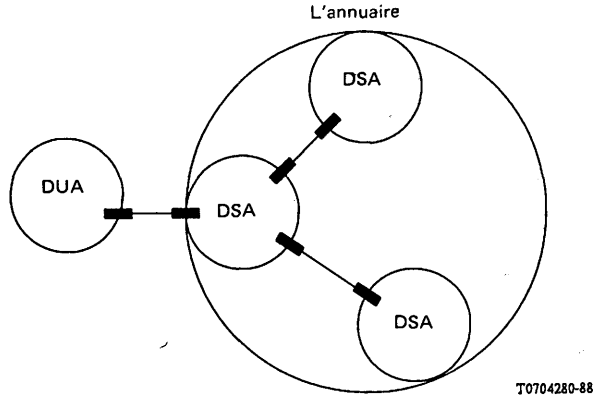


FIGURE 7/X.500

Destination multiple

8.3.4 Toutes les solutions ont leurs avantages. Par exemple, la méthode décrite à la figure 5/X.500 peut être utilisée lorsqu'il est souhaitable d'alléger la charge du DSA local. Dans d'autres cas, une solution hybride, combinant un ensemble plus complexe d'interactions fonctionnelles, peut être nécessaire pour satisfaire la demande de l'expéditeur, comme le montre la figure 8/X.500.

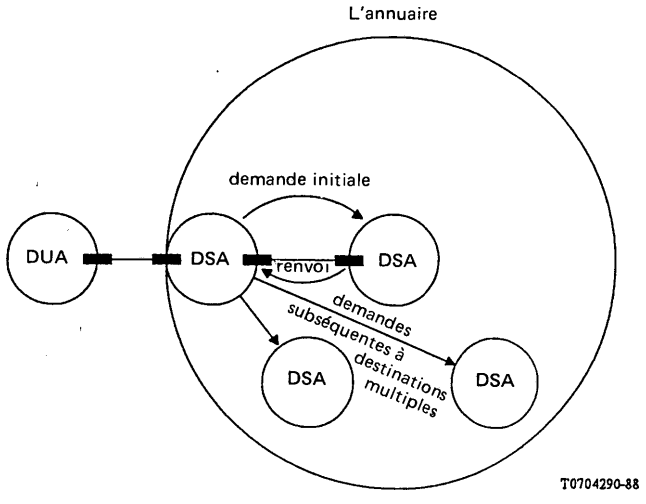


FIGURE 8/X.500

Solutions hybrides des modes mixtes

9 Protocoles d'annuaire

Remarque - Les protocoles de couche application OSI définis pour permettre aux DUA et aux DSA dans un système ouvert différent de coopérer, sont spécifiés dans la Recommandation X.519.

9.1 Il existe deux protocoles d'annuaire:

- le protocole d'accès à l'annuaire (DAP), qui définit l'échange de demandes et de réponses entre un DUA et un DSA;
- le protocole de système d'annuaire (DSP), qui définit l'échange de demandes et de réponses entre deux DSA.

9.2 Chaque protocole est défini par un contexte d'application, chacun contenant un ensemble d'éléments de protocole. Par exemple, le DAP contient des éléments de protocole associés à l'interrogation et à la modification de l'annuaire.

9.3 Chaque contexte d'application se compose d'éléments de service d'application, lesquels sont définis pour l'utilisation du service d'opérations à distance (ROS) de la Recommandation X.219 pour structurer et assurer leurs interactions. Ainsi, le DAP et le DSP sont définis comme des ensembles d'opérations à distance et d'erreurs utilisant la notation ROS.

ANNEXE A

(à la Recommandation X.500)

Application de l'annuaire

Cette annexe ne fait pas partie intégrante de la présente Recommandation.

A.1 *L'environnement de l'annuaire*

Remarque - Dans la présente annexe, le terme "réseau" est utilisé dans un sens général pour indiquer l'ensemble des systèmes reliés et des processus concernant tout service de télécommunications et non un seul système ou processus lié à la couche réseau OSI.

L'environnement dans lequel l'annuaire offre des services est le suivant:

a) de nombreux réseaux de télécommunications seront établis à grande échelle et subiront constamment des changements:

   1) des objets de divers types entreront, seuls ou en groupe, dans le réseau et le quitteront, sans avertissement;

   2) la connectivité des objets (en particulier des noeuds de réseau) changera, en raison de l'adjonction ou de la suppression de trajets entre ces objets;

   3) les diverses caractéristiques des objets, telles que leurs adresses, leur disponibilité et leurs emplacements physiques peuvent changer à tout moment;

b) bien que les changements soient fréquents, la durée de vie utile d'un objet donné n'est pas courte. Un objet interviendra beaucoup plus souvent dans les communications qu'il ne changera d'adresse, de disponibilité, d'emplacement physique, etc.;

c) les objets qui interviennent dans les services de télécommunications actuels sont généralement identifiés par des numéros ou d'autres chaînes de symboles, choisis pour leur facilité d'attribution ou de traitement mais pas pour leur facilité d'utilisation par des personnes.

A.2 *Caractéristiques du service d'annuaire*

Les capacités d'annuaire sont nécessaires pour les raisons suivantes:

a) le désir d'isoler (autant que possible) l'utilisateur du réseau des changements fréquents apportés à ce dernier. Pour ce faire, on peut prévoir une "zone de flou" entre les utilisateurs et les objets avec lesquels ils traitent. Cela signifie que les utilisateurs se réfèrent aux objets par leur nom et non, par exemple, par l'adresse. L'annuaire assure le service de mise en correspondance nécessaire;

b) le désir de donner l'image d'un réseau plus facile à utiliser. Par exemple, l'utilisation de pseudonymes, la mise à disposition des "pages jaunes" (voir A.3.5), etc., facilitent la recherche et l'utilisation d'informations de réseau.

L'annuaire permet aux utilisateurs d'obtenir diverses informations sur le réseau et prévoit la maintenance, la distribution et la sécurité de cette information.

A.3 *Schémas d'utilisation de l'annuaire*

Remarque - Il ne s'agit ici que de la recherche dans l'annuaire: on suppose que les services de modification d'annuaire ne servent qu'à maintenir la DIB dans la forme nécessaire à l'application dans le temps.

A.3.1 *Introduction*

Le service d'annuaire est défini dans ces normes en termes de demandes particulières qu'un DUA peut formuler, et de paramètres correspondants. Toutefois, un concepteur d'application pensera vraisemblablement en termes plus orientés vers des objectifs, lorsqu'il étudiera les besoins de recherche d'information pour l'utilisation de l'annuaire dans cette application. En conséquence, la présente section décrit un certain nombre de schémas de haut niveau d'utilisation du service d'annuaire qui sont susceptibles de convenir pour de nombreuses applications.

A.3.2 *Recherche*

La recherche directe d'annuaire, qui sera vraisemblablement le type d'interrogation le plus fréquent, fait intervenir le DUA qui fournit le nom spécifique d'un objet, ainsi qu'un type d'attribut. L'annuaire renverra une ou plusieurs valeurs correspondant à ce type d'attribut. Il s'agit d'une généralisation de la fonction d'annuaire classique, que l'on obtient lorsque le type d'attribut demandé correspond à un type particulier d'adresse. Les types d'attributs pour divers types d'adresses sont normalisés, y compris l'adresse OSI PSAP, l'adresse O/R de traitement de message et les numéros de téléphone et télex.

La recherche est assurée par le service de lecture, qui fournit aussi les autres généralisations suivantes:

- la recherche peut être fondée sur des noms autres que le nom spécifique de l'objet, par exemple des pseudonymes;
- les valeurs provenant d'un nombre de types d'attributs peuvent être obtenues par une simple demande: le cas extrême étant qu'il faille retourner les valeurs de tous les attributs dans l'entrée.

A.3.3 *Désignation facile à utiliser*

On peut donner aux objets des noms que les utilisateurs puissent trouver (ou mémoriser) facilement. Les noms de ce type seront composés généralement d'attributs qui sont en quelque sorte inhérents à l'objet, et non fabriqués à cette fin. Le nom d'un objet sera commun à toutes les applications qui s'y rapportent.

A.3.4 *Recherche rapide*

Dans de nombreuses utilisations de l'annuaire conçues pour les individus, il se peut que l'utilisateur (ou DUA) ne puisse pas citer directement un nom, facile à utiliser ou non, concernant l'objet sur lequel il recherche des informations. Toutefois, l'utilisateur le reconnaîtra peut-être lorsqu'il le verra. La recherche rapide permettra à l'utilisateur de parcourir la DIB pour rechercher les entrées appropriées.

La recherche rapide est une combinaison des services de listage et de recherche, assurée éventuellement en conjonction avec la lecture (bien que le service de recherche offre la capacité de lecture).

A.3.5 *"Pages jaunes"*

Il y a différentes façons d'assurer une capacité de type "pages jaunes". La plus simple est fondée sur le filtrage et utilise des assertions sur des attributs particuliers dont les valeurs sont les catégories (par exemple le type d'attribut "catégorie affaires" défini dans la Recommandation X.520). Cette méthode ne nécessite pas l'établissement dans le DIT d'informations spéciales, sauf pour s'assurer que les attributs requis sont présents. Toutefois, en général, il peut être onéreux de faire des recherches s'il y a une vaste population, car le filtrage nécessite la production de l'ensemble universel qui doit être filtré.

Une autre méthode est possible; elle est fondée sur l'établissement de sous-arbres spéciaux, dont les structures de désignation sont conçues spécialement pour la recherche de type "pages jaunes". La figure A-1/X.500 donne un exemple de sous-arbre "pages jaunes" peuplé d'entrées pseudonymes uniquement. En réalité, les entrées dans les sous-arbres "pages jaunes" peuvent être un mélange d'entrées d'objet d'entrées pseudonymes, du moment qu'il n'existe qu'une entrée d'objet pour chaque objet enregistré dans l'annuaire.

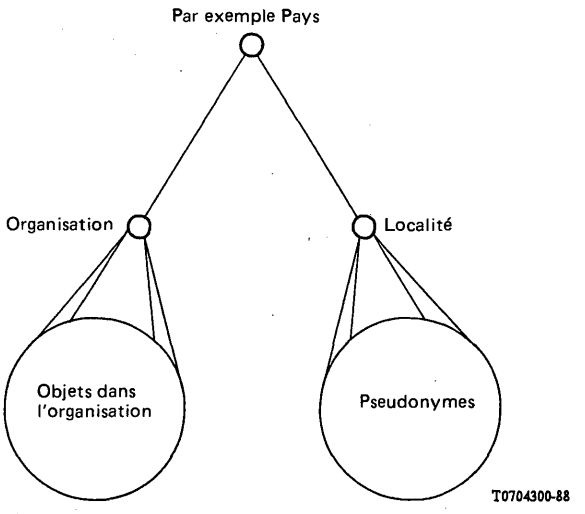


FIGURE A-1/X.500

Méthode concernant les "pages jaunes"

A.3.6 *Groupes*

Un groupe est un ensemble dont les membres peuvent changer avec le temps par adjonction et suppression explicites des membres. Le groupe est un objet, tout comme ses membres. Il peut être demandé à l'annuaire:

- d'indiquer si un objet particulier est membre ou non d'un groupe;
- d'énumérer les membres d'un groupe.

Les groupes sont admis de la façon suivante: l'entrée contient un attribut "membre" à valeurs multiples (ce type d'attribut est défini dans la Recommandation X.520). Les deux capacités mentionnées peuvent être appliquées respectivement par comparaison et lecture.

Un membre d'un groupe pourrait lui-même être un groupe si cela est important pour l'application. Toutefois, les services de vérification et d'expansion récurrents nécessaires devront être créés par le DUA en dehors des versions non récurrentes fournies.

A.3.7 *Authentification*

Dans de nombreuses applications, il faut que les objets qui y participent donnent une preuve de leur identité avant d'être autorisés à effectuer une action. L'annuaire aide à assurer ce processus d'authentification. (Indépendamment de cela, l'annuaire demande à ses utilisateurs de s'authentifier eux-mêmes, de façon à assurer la commande d'accès.)

Dans la méthode d'authentification la plus directe, appelée "authentification simple", l'annuaire contient un attribut "mot de passe d'utilisateur" dans l'entrée pour tout utilisateur qui désire s'authentifier auprès d'un service. A la demande du service, l'annuaire confirmera ou niera qu'une valeur particulière fournie est réellement le mot de passe des utilisateurs. Cela évite à l'utilisateur d'avoir besoin d'un mot de passe différent pour chaque service. Dans les cas où l'échange des mots de passe dans un contexte local qui repose sur une authentification simple est jugé inapproprié, l'annuaire fournit en option des moyens de protéger ces mots de passe contre une réutilisation ou une utilisation erronée par une fonction à sens unique.

La méthode la plus complexe, appelée "authentification renforcée" est fondée sur une cryptographie publique à clé, dans laquelle l'annuaire agit comme un dépôt des clés de chiffrement publiques des utilisateurs, convenablement protégées contre la fraude. Les étapes que les utilisateurs peuvent suivre pour obtenir les clés publiques les uns des autres à partir de l'annuaire, puis authentifier ceux qui les utilisent, sont décrites en détail dans la Recommandation X.509.

A.4 *Applications génériques*

A.4.1 *Introduction*

On peut imaginer qu'un certain nombre d'applications génériques sont assurées implicitement par l'annuaire: les applications qui ne sont pas propres à un service de télécommunications particulier. Deux de ces applications sont décrites dans le présent document: l'annuaire de communications interpersonnelles et l'annuaire de communications intersystèmes (pour OSI).

Remarque - L'authentification, décrite dans le paragraphe précédent comme un "schéma d'accès", pourrait aussi être considérée comme une application d'annuaire générique.

A.4.2 *Communications interpersonnelles*

Le but de cette application est d'offrir aux individus ou à leurs agents des informations sur la façon de communiquer avec d'autres individus ou d'autres groupes.

Les classes d'objet suivantes sont certainement utilisées: personne, rôle organisationnel et groupe. De nombreuses autres catégories interviennent aussi peut-être de façon moins directe, à savoir: pays, organisation, unité organisationnelle.

Les types d'attribut concernés, autres que ceux utilisés dans la désignation, sont généralement les attributs d'adressage. Généralement, l'entrée pour une personne particulière aura les adresses correspondant à chacune des méthodes de communication par lesquelles cette personne peut être atteinte; ces dernières sont choisies parmi une liste non exhaustive comprenant au moins: la téléphonie, le courrier électronique, le télex, le RNIS, la remise physique (par exemple le système postal), la télécopie. Dans certains cas, comme pour le courrier électronique, l'entrée aura des informations supplémentaires telles que les types d'information que l'équipement d'utilisateur peut traiter. Si l'authentification doit être assurée, le mot de passe d'utilisateur et/ou l'identité seront nécessaires.

Les schémas de désignation utilisés pour les diverses catégories d'objet devraient être faciles à utiliser, avec des pseudonymes établis le cas échéant pour donner d'autres noms et assurer la continuité après une modification de nom, etc.

Les schémas d'accès suivants seront présents dans cette application: recherche, désignation facile à utiliser, recherche rapide, "pages jaunes" et groupes, à divers degrés, l'authentification sera aussi utilisée.

A.4.3 *Communications intersystèmes (pour OSI)*

Conformément au modèle de référence OSI, deux fonctions d'annuaire sont nécessaires: l'une dans la couche application qui met en correspondance les applications-titres avec les adresses de présentation, et l'autre dans la couche réseau, qui met en correspondance les adresses NSAP et les adresses SNPA (SNPA = point d'attachement du sous-réseau).

Remarque - Dans le reste de ce paragraphe, seul le cas de la couche application est traité.

Pour accomplir cette fonction, on consulte l'annuaire si l'information nécessaire pour assurer la mise en correspondance n'est pas disponible au niveau local.

Les utilisateurs sont des entités-applications et les catégories d'objet présentant un intérêt sont aussi des entités-applications ou des sous-catégories de celles-ci.

Le principal type d'attribut concerné, autre que ceux utilisés pour la désignation, est l'adresse de présentation. D'autres types d'attribut, qui ne sont pas considérés comme nécessaires pour la fonction d'annuaire proprement dite, pourraient assurer la vérification ou la recherche du type d'entité d'application, ou des listes des contextes d'application, des syntaxes abstraites, etc. Les types d'attribut liés à l'authentification pourraient aussi être appropriés.

Le principal schéma d'accès qui doit être présenté sera la recherche.

Recommandation X.501

L'ANNUAIRE - MODELES [1)]

(Melbourne, 1988)

SOMMAIRE

---

1) La Recommandation X.501 et ISO 9594-2, l'annuaire-modèles ont été mis au point en étroite collaboration et sont alignés du point de vue technique.

0 Introduction

0.1 Ce document, ainsi que les autres documents de la même série, ont été établis de manière à faciliter l'interconnexion de systèmes de traitement de l'information afin d'assurer des services d'annuaire. L'ensemble de ces systèmes, ainsi que les informations d'annuaire qu'ils contiennent, peuvent être considérés comme un tout intégré, appelé *"l'annuaire"*. Les informations contenues dans l'annuaire, collectivement appelées "base de données de l'annuaire" (DIB), sont généralement utilisées pour faciliter la communication entre, avec ou sur des objets tels que entités d'application, personnes, terminaux, listes de diffusion.

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

- type de fabrication;
- méthode de gestion;
- niveau de complexité;
- ancienneté.

0.3 La présente Recommandation fournit un certain nombre de modèles pour l'annuaire afin qu'ils servent de cadre aux autres Recommandations. Il s'agit du modèle général (fonctionnel), du modèle organisationnel, du modèle de sécurité et du cadre d'information. Ce dernier décrit la manière dont l'annuaire organise les informations qu'il contient, par exemple comment les informations sur les objets sont groupées en vue de former des entrées d'annuaire pour ces objets et comment ces informations fournissent des noms pour les objets.

0.4 L'annexe A résume la terminologie mathématique associée aux structures arborescentes.

0.5 L'annexe B résume les modalités d'utilisation des identificateurs d'objets ASN.1 dans cette série de Recommandations.

0.6 L'annexe C présente le module ASN.1 qui contient toutes les définitions associées au cadre d'information.

0.7 L'annexe D énumère alphabétiquement les termes définis dans ce document.

0.8 L'annexe E décrit quelques critères qui peuvent être pris en considération dans la conception des noms.

0.9 L'annexe F donne les principes directeurs de la commande d'accès.

1 Portée et domaine d'application

1.1 Les modèles présentés dans cette Recommandation fournissent un cadre conceptuel et terminologique pour les autres Recommandations qui définissent divers aspects de l'annuaire.

1.2 Les modèles fonctionnel et organisationnel définissent la façon dont l'annuaire peut être réparti, tant du point de vue fonctionnel que du point de vue administratif.

1.3 Le modèle de sécurité définit le cadre dans lequel les fonctions de sécurité, telles que la commande d'accès, sont assurées dans l'annuaire.

1.4 Le modèle d'information décrit la structure logique de la DIB. De ce point de vue, le fait que l'annuaire soit réparti, au lieu d'être centralisé, n'est pas visible. Les autres Recommandations de la série utilisent les concepts du cadre d'information, notamment:

a) le service offert par l'annuaire est décrit (dans la Recommandation X.511) en fonction des concepts du cadre d'information: cela permet au service assuré d'être relativement indépendant de la répartition physique de la DIB;

b) l'exploitation répartie de l'annuaire est spécifiée (dans la Recommandation X.518) afin d'assurer ce service et, donc, de maintenir cette structure d'information logique, étant donné que la DIB est en fait très largement répartie.

2 Références

Recommandation X.200 - Interconnexion de systèmes ouverts - Modèle de référence de base.

Recommandation X.500 - L'annuaire - Aperçu général des concepts, modèles et services.

Recommandation X.509 - L'annuaire - Cadre d'authentification.

Recommandation X.511 - L'annuaire - Définition du service abstrait.

Recommandation X.518 - L'annuaire - Procédures de fonctionnement réparti.

Recommandation X.519 - L'annuaire - Spécifications du protocole.

Recommandation X.520 - L'annuaire - Types d'attributs sélectionnés.

Recommandation X.521 - L'annuaire - Catégories d'objets sélectionnées.

3 Définitions

Les définitions des termes sont, le cas échéant, données au début des divers paragraphes. Un index de ces termes est fourni, pour plus de commodité, dans l'annexe D.

4 Abréviations

ADDMD	Domaine de gestion d'annuaire d'Administration
AVA	Assertion de valeur d'attribut
DIB	Base de données de l'annuaire
DIT	Arbre d'information de l'annuaire
DMD	Domaine de gestion d'annuaire
DSA	Agent de système d'annuaire
DUA	Agent d'usager d'annuaire

PRDMD Domaine de gestion d'annuaire privé

RDN Nom spécifique relatif

SECTION 1 - *Modèle d'annuaire*

5 Modèle d'annuaire

5.1 *Définitions*

a) *point d'accès:* Le point auquel un service abstrait est obtenu;

b) *domaine de gestion d'annuaire d'Administration (ADDMD):* DMD géré par une Administration;

Remarque - Le terme "Administration" désigne une Administration publique de télécommunication ou une autre organisation offrant des services publics de télécommunication.

c) *autorité administrative:* Entité exerçant un contrôle administratif sur toutes les entrées mises en mémoire dans un seul agent de système d'annuaire;

d) *l'annuaire:* Dépôt d'informations sur des objets; offre à ses usagers des services d'annuaire permettant d'accéder aux informations;

e) *domaine de gestion d'annuaire (DMD):* Ensemble d'un ou plusieurs DSA, avec ou sans DUA, géré par une seule organisation;

f) *agent de système d'annuaire (DSA):* Processus d'application OSI qui fait partie de l'annuaire;

g) *usager (de l'annuaire):* Usager final de l'annuaire, c'est-à-dire entité ou personne qui accède à l'annuaire;

h) *agent d'usager d'annuaire (DUA):* Processus d'application OSI qui représente un usager dans l'accès à l'annuaire;

Remarque - Les DUA peuvent offrir également diverses facilités locales pour aider les usagers à composer les demandes et à interpréter les réponses.

i) *domaine de gestion d'annuaire privé (PRDMD):* DMD géré par une organisation autre qu'une Administration.

5.2 *L'annuaire et ses usagers*

5.2.1 Un usager d'annuaire (par exemple une personne ou un processus d'application) est un usager qui obtient des services d'annuaire accédant à l'*annuaire*. Plus précisément, il s'agit d'un *agent d'usager d'annuaire* (DUA) qui accède effectivement à l'annuaire et entre en interaction avec lui pour obtenir le service au nom d'un usager donné. L'annuaire offre un ou plusieurs *points d'accès* auxquels cet accès peut s'effectuer. Ces principes sont illustrés par la figure 1/X.501.

5.2.2 Les services offerts par l'annuaire sont définis dans la Recommandation X.511.

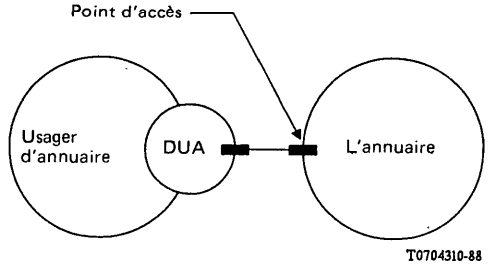


FIGURE 1/X.501

Accès à l'annuaire

5.2.3 L'annuaire est un dépôt d'informations sur des objets et les services d'annuaire qu'il offre à ses usagers impliquent divers types d'accès à ces informations. L'information est collectivement connue sous le nom de *base de données d'annuaire (DIB)*. Un modèle de DIB est défini dans la section 2 de la présente Recommandation.

5.2.4 Un DUA se manifeste sous la forme d'un processus d'application. Chaque DUA représente précisément un usager d'annuaire.

Remarque 1 - Certains systèmes ouverts peuvent offrir une fonction DUA centralisée qui extrait l'information pour les utilisateurs (processus d'application, personnes, etc.). Ce processus est transparent pour l'annuaire.

Remarque 2 - Les fonctions DUA et un DSA (voir le § 5.3.1) peuvent se trouver dans le même système ouvert et le fait de rendre visibles dans l'environnement OSI un ou plusieurs DUA constitue un choix d'application.

Remarque 3 - Un DUA aura probablement, au niveau local, un comportement et une structure qui sortent du cadre des Recommandations envisagées. Par exemple, un DUA qui représente un usager d'annuaire en tant que personne physique peut offrir diverses facilités locales pour aider cet usager à composer des demandes et à interpréter les réponses.

5.3 *Modèle fonctionnel*

5.3.1 L'annuaire se présente sous la forme d'un ensemble d'un ou de plusieurs processus d'application connus sous le terme d'"*agents de système d'annuaire (DSA)*", chacun d'eux offrant ou non un ou plusieurs points d'accès, comme l'illustre la figure 2/X.501. Lorsque l'annuaire se compose de plusieurs DSA, on dit qu'il est réparti. Les procédures d'exploitation de l'annuaire, lorsque celui-ci est réparti, sont spécifiées dans la Recommandation X.518.

Remarque - Un DSA aura probablement, au niveau local, un comportement et une structure qui sortent du cadre des Recommandations envisagées. Par exemple, un DSA chargé de détenir une partie ou la totalité des informations de la DIB le fera normalement à l'aide d'une base de données dont l'interface relève de l'autorité locale.

5.3.2 Deux processus d'application donnés qui doivent entrer en interaction lors de la fourniture de services d'annuaire (DUA et DSA, ou deux DSA) peuvent être situés dans des systèmes ouverts différents. Une telle interaction est effectuée à l'aide de protocoles d'annuaire OSI, comme spécifié dans la Recommandation X.519.

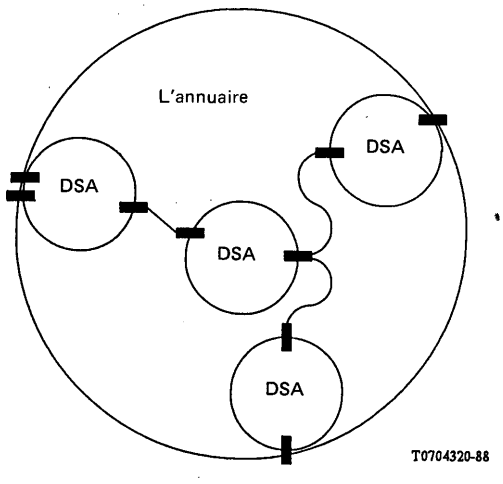


FIGURE 2/X.501

L'annuaire offert par des DSA multiples

5.4 *Modèle organisationnel*

5.4.1 Un ensemble d'un ou de plusieurs DSA sans DUA ou avec plusieurs DUA, géré par une seule organisation, peut former un *domaine de gestion d'annuaire (DMD)*.

Remarque - L'organisation qui gère un DMD peut être une Administration (c'est-à-dire une Administration publique de télécommunication ou une autre organisation offrant des services publics de télécommunication), auquel cas le DMD est appelé DMD d'Administration (ADDMD); sinon, il s'agit d'un DMD privé (PRDMD). Il convient de noter que l'installation de systèmes d'annuaire privés par les membres du CCITT entre dans le cadre des réglementations nationales. Ainsi, les possibilités techniques décrites peuvent être offertes ou non par une Administration qui assure des services d'annuaire. Le fonctionnement interne et la configuration des DMD privés n'entrent pas dans le cadre des Recommandations du CCITT envisagées.

5.4.2 La gestion d'un DUA par un DMD implique que le DMD est chargé d'assurer le service de ce DUA, par exemple la maintenance ou, dans certains cas, que ce DUA lui appartient.

5.4.3 L'organisation concernée peut ou non décider d'utiliser cette série de Recommandations pour régir toute interaction éventuelle entre les DUA et les DSA situés en totalité dans le DMD.

5.4.4 Chaque DSA est administré par une autorité administrative. Cette entité exerce un contrôle sur toutes les entrées d'objet et les entrées pseudonymes mises en mémoire par ce DSA. Cette fonction de contrôle inclut notamment les responsabilités relatives au schéma d'annuaire utilisé pour guider la création et la modification des entrées (voir le § 9). La structure et l'attribution de noms relèvent d'une autorité de désignation [voir le § 8.1 f)] et le rôle de l'autorité administrative est de mettre en oeuvre ces structures de désignation dans le schéma.

SECTION 2 - *Modèle d'information*

6 Base de données de l'annuaire

6.1 *Définitions*

a) *entrée pseudonyme:* Entrée de la catégorie "pseudonyme" qui contient une information servant à donner un nom de remplacement à un objet;

b) *base de données de l'annuaire (DIB):* Ensemble complet des informations auxquelles l'annuaire assure l'accès; il comprend toutes les informations qui peuvent être lues ou manipulées à l'aide des opérations de l'annuaire;

c) *arbre d'information de l'annuaire (DIT):* DIB considérée comme un arbre dont les sommets (autres que la racine) sont les entrées de l'annuaire;

Remarque - Le terme DIT n'est utilisé, au lieu de DIB, que lorsqu'il s'agit d'une structure arborescente de l'information.

d) *entrée (d'annuaire):* Partie de la DIB qui contient des informations sur un objet;

e) *supérieur immédiat* (nom): Relatif à une entrée ou à un objet particulier (le contexte doit préciser clairement de quoi il s'agit): entrée ou objet immédiatement supérieur;

f) *immédiatement supérieur (entrée):* Par rapport à une entrée donnée: entrée qui est au sommet initial d'un arc du DIT dont le sommet final est celui de l'entrée considérée;

(objet): par rapport à un objet donné: Objet dont l'entrée d'*objet* est le supérieur immédiat de *l'une* quelconque des entrées (objet ou pseudonyme) pour le second objet;

g) *objet (d'intérêt):* Tout ce qui, dans un certain "monde", généralement le monde des télécommunications et du traitement de l'information ou une partie de celui-ci, est identifiable (peut être désigné) et sur quoi il est intéressant de détenir des informations dans la DIB;

h) *catégorie d'objets:* Famille identifiée d'objets (ou d'objets concevables) qui ont en commun certaines caractéristiques;

i) *entrée d'objet:* Entrée qui constitue la collecte primaire d'informations sur un objet dans la DIB et dont on peut dire par conséquent qu'elle représente cet objet dans la DIB;

j) *sous-catégorie:* Par rapport à une super-catégorie: catégorie d'objets dérivée d'une super-catégorie. Les membres de la sous-catégorie possèdent toutes les caractéristiques d'une autre catégorie d'objets (super-catégorie), ainsi que des caractéristiques supplémentaires que ne possède aucun des membres de cette catégorie d'objets (la super-catégorie);

k) *inférieur subordonné:* Le contraire de supérieur;

l) *super-catégorie:* Par rapport à une sous-catégorie: catégorie d'objets dont est dérivée une sous-catégorie;

m) *supérieur:* (applicable à une entrée ou à un objet) Immédiatement supérieur ou supérieur à une entrée ou à un objet qui est immédiatement supérieur (récurrence).

6.2 *Objets*

6.2.1 Le but de l'annuaire est de détenir des informations sur des *objets d'intérêt (objets)* qui existent dans un certain "monde" et de fournir l'accès à ces informations. Un objet peut être tout ce qui, dans ce monde, est identifiable (peut être désigné).

Remarque 1 - Le "monde" est généralement celui des télécommunications et du traitement de l'information, ou une partie de celui-ci.

Remarque 2 - Les objets connus de l'annuaire peuvent ne pas correspondre exactement à l'ensemble des choses "réelles" dans le monde. Par exemple, en ce qui concerne l'annuaire, une personne du monde réel peut être considérée comme deux objets différents, un homme d'affaires et un abonné résidentiel. La mise en correspondance n'est pas définie dans la présente Recommandation; il s'agit d'une question qui relève des usagers et des prestataires de l'annuaire dans le cadre de leurs applications.

6.2.2 L'ensemble complet des informations auxquelles l'annuaire permet d'accéder est connu sous le nom de *base de données de l'annuaire* (DIB). Toutes les informations qui peuvent être lues ou manipulées par les opérations de l'annuaire sont considérées comme incluses dans la DIB.

6.2.3 Une *catégorie d'objets* est une famille identifiée d'objets (ou d'objets concevables) qui ont en commun certaines caractéristiques. Chaque objet appartient au moins à une catégorie. Une catégorie d'objets peut être une *sous-catégorie* d'une autre catégorie d'objets, auquel cas les membres de la première sont également considérés comme des membres de la seconde super-catégorie. Il peut y avoir des sous-catégories de sous-catégories, etc., jusqu'à un rang hiérarchique arbitraire.

6.3 *Entrées d'annuaire*

6.3.1 La DIB est formée d'*entrées d'annuaire (entrées)*, dont chacune contient des informations sur (décrit) un seul objet.

6.3.2 Pour tout objet particulier, il y a exactement une *entrée d'objet*, celle-ci étant la collecte primaire d'informations sur cet objet dans la DIB. On dit que l'entrée d'objet représente l'objet.

6.3.3 Pour tout objet particulier, il peut y avoir ou non, en plus de l'entrée d'objet, une ou plusieurs *entrées pseudonymes* pour cet objet, qui sont utilisées pour fournir d'autres noms (voir le § 8.5).

6.3.4 La structure des entrées d'annuaire est illustrée par la figure 3/X.501 et décrite au § 7.2.

6.3.5 Chaque entrée contient une indication de la catégorie et de la super-catégorie d'objets auxquelles l'entrée est associée. Dans le cas d'une entrée d'objet, celle-ci indique la ou les catégories auxquelles l'objet appartient. Dans le cas d'une entrée pseudonyme, celle-ci indique à l'aide d'une classe d'objet spéciale, "pseudonyme" (définie au § 9.4.8.2), qu'il s'agit en fait d'une entrée pseudonyme et peut aussi indiquer à quelle(s) sous-catégorie(s) de la catégorie d'objets pseudonymes l'entrée appartient.

6.4 *L'arbre d'information d'annuaire (DIT)*

6.4.1 Pour répondre aux besoins de la répartition et de la gestion d'une DIB potentiellement très large et pour faire en sorte que les objets puissent être désignés sans ambiguïté (voir le § 8) et que leurs entrées puissent être trouvées, il n'est guère possible d'adopter une structure uniforme des entrées. En conséquence, on peut exploiter la relation hiérarchique qui existe généralement entre les objets (par exemple, une personne travaille pour un département, qui appartient à une organisation, laquelle a son siège dans un pays) en disposant les entrées selon une structure arborescente connue sous le nom d'*arbre d'information d'annuaire (DIT)*.

Remarque - On trouvera, en annexe A, une introduction aux concepts et à la terminologie des structures arborescentes.

6.4.2 Les éléments constitutifs du DIT sont définis comme suit:

a) les sommets sont les entrées. Les entrées d'objet peuvent être soit des sommets-feuilles, soit des sommets non-feuilles, alors que les entrées pseudonymes sont toujours des sommets-feuilles. La racine n'est pas une entrée en tant que telle mais peut, lorsque cela est commode [par exemple dans les définitions des alinéas b) et c) ci-dessous], peut être considérée comme une entrée d'objet nulle [voir d) ci-dessous];

b) les arcs définissent la relation entre les sommets (donc entre les entrées). Un arc allant du sommet A au sommet B signifie que l'entrée en A est l'*entrée immédiatement supérieure (le supérieur immédiat)* de l'entrée en B et, inversement, que l'entrée en B est une *entrée immédiatement subordonnée (le subordonné immédiat)* de l'entrée en A. Les *entrées supérieures (les supérieurs)* d'une entrée donnée sont ses supérieurs immédiats, avec *ses* supérieurs (récurrence). Les *entrées subordonnées (subordonnés)* d'une entrée donnée sont ses subordonnées immédiates, avec leurs subordonnés (récurrence);

c) l'objet représenté par une entrée est l'autorité de désignation (voir le § 8) pour ses subordonnés, ou est étroitement associé à cette autorité;

d) la racine représente le niveau le plus élevé de l'autorité de désignation pour la DIB.

6.4.3 Une relation supérieur/subordonné entre les objets peut être déduite de celle qui existe entre les entrées. Un objet est un *objet immédiatement supérieur (supérieur immédiat)* d'un autre objet si, et seulement si, l'entrée d'objet pour le premier objet est le supérieur immédiat de l'une quelconque des entrées pour le second objet. Les termes *"objet immédiatement subordonné", "subordonné immédiat", "supérieur"* et *"subordonné"* (appliqués aux objets) ont des significations analogues.

6.4.4 Les relations supérieur/subordonné autorisées entre objets sont régies par les définitions de la structure DIT (voir le § 9.2).

7 Entrées de l'annuaire

7.1 *Définitions*

a) *attribut:* Information d'un type particulier concernant un objet et figurant dans une entrée qui décrit cet objet dans la DIB;

b) *type d'attribut:* Composante d'un attribut qui indique la classe d'information donnée par cet attribut;

c) *valeur d'attribut:* Exemple particulier de la classe d'information indiquée par un type d'attribut;

d) *assertion de valeur d'attribut:* Proposition qui peut être vraie, fausse, ou indéfinie, concernant les valeurs (ou peut-être seulement les valeurs spécifiques) d'une entrée;

Remarque - Dans le présent document, la notation "stringl = string2" est utilisée pour donner des exemples d'assertion de valeur d'attribut. Dans cette notation, "stringl" est une abréviation pour le "nom" du type d'attribut et "string2" est une représentation textuelle d'une valeur appropriée. Bien que les types d'attribut dans les exemples soient souvent fondés sur des types réels, tels que ceux définis dans la Recommandation X.520 (par exemple, "C" désigne "pays", CN désigne "nom commun"), cela n'est pas strictement nécessaire pour les besoins du présent document, l'annuaire ne connaissant généralement pas les significations des types d'attribut utilisés.

e) *valeur spécifique:* Valeur d'attribut dans une entrée qui a été désignée pour figurer dans le nom spécifique relatif de l'entrée.

7.2 *Structure générale*

7.2.1 Comme le montre la figure 3/X.501, une entrée est formée d'un ensemble d'*attributs*.

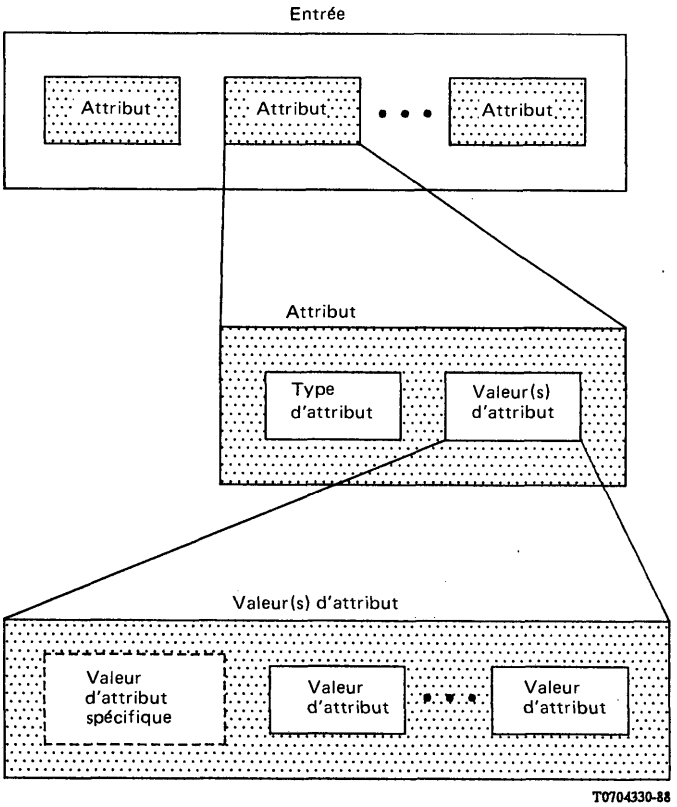


FIGURE 3/X.501

Structure d'une entrée

7.2.2 Chaque attribut fournit une information sur (ou décrit) une caractéristique particulière de l'objet auquel l'entrée correspond.

Remarque - A titre d'exemple d'attributs qui peuvent être présents dans une entrée, on peut citer notamment les informations de désignation telles que le nom personnel de l'objet et les informations d'adressage telles que son numéro de téléphone.

7.2.3 Un attribut est formé d'un *type d'attribut*, qui identifie la classe d'information donnée par un attribut, et de *la ou des valeurs d'attribut* correspondantes qui sont les exemples particuliers de cette classe figurant dans l'entrée.

```
Attribute ::=
    SEQUENCE{
        type        Attribute Type
        values      SET OF AttributeValue
        -- at least one value is required --}
```

7.3 *Types d'attribut*

7.3.1 Certains types d'attribut seront normalisés à l'échelon international. D'autres types d'attribut seront définis par les autorités administratives nationales et les organisations privées. Cela implique qu'un certain nombre d'autorités séparées seront chargées d'assigner des types de telle sorte que chacun soit distinct de tous les autres types assignés. A cet effet, on identifie chaque type d'attribut par un identificateur d'objet lorsque le type est défini (comme indiqué au § 9.5):

Attribute Type ::= OBJECT IDENTIFIER

7.3.2 Tous les attributs dans une entrée doivent être d'un type d'attribut différent.

7.3.3 Il existe un certain nombre de types d'attribut que l'annuaire connaît et utilise pour ses propres besoins. Il s'agit des types d'attribut suivants:

a) **Catégorie d'objets.** Un attribut de ce type figure dans chaque entrée; il indique la catégorie et la ou les super-catégories d'objets auxquelles l'objet appartient.

b) **Nom d'objet-pseudonyme.** Un attribut de ce type figure dans chaque entrée pseudonyme; il contient le nom spécifique (voir le § 8.5) de l'objet que cette entrée pseudonyme décrit.

Ces attributs sont (en partie) définis au § 9.5.4.

7.3.4 Les types d'attribut qui doivent ou peuvent figurer dans une entrée (autres que ceux mentionnés au § 7.3.3) sont régis par les règles applicables à la ou aux catégories d'objets indiquées.

7.4 *Valeurs d'attribut*

7.4.1 Lorsqu'on définit un type d'attribut (voir le § 9.5), il faut également spécifier la syntaxe et donc le type de données, auxquels chaque valeur dans ces attributs doit se conformer. Il peut s'agir d'un type quelconque de données:

AttributeValue ::= ANY

7.4.2 Une seule des valeurs d'un attribut peut être désignée comme *valeur spécifique*, auquel cas la valeur d'attribut figure dans le nom spécifique relatif (voir le § 8.3) de l'entrée.

7.4.3 Une *assertion de valeur d'attribut (AVA)* est une proposition qui peut être vraie, fausse, ou indéfinie, concernant les valeurs (ou peut-être seulement les valeurs spécifiques) d'une entrée. Elle implique un type d'attribut et une valeur d'attribut.

AttributeValueAssertion ::=
SEQUENCE {AttributeType, AttributeValue}

et est:

a) indéfinie si l'une des conditions suivantes est remplie:

i) le type d'attribut est inconnu;

ii) la syntaxe d'attribut pour le type n'a pas de règle de correspondance d'égalité;

iii) la valeur n'est pas conforme au type de données de la syntaxe d'attribut;

Remarque - Les éléments ii) et iii) indiquent normalement une AVA défectueuse; toutefois, l'élément i) peut se produire dans une situation locale (par exemple, un DSA particulier n'a pas enregistré ce type particulier d'attribut).

b) vraie, si l'entrée contient un attribut de ce type, dont l'une des valeurs correspond à cette valeur (si l'assertion concerne uniquement des valeurs spécifiques, la valeur en correspondance doit être la valeur spécifique);

Remarque - La mise en correspondance des valeurs est fondée sur l'égalité; elle implique l'utilisation de la règle de mise en correspondance associée à la syntaxe d'attribut.

c) fausse, dans les autres cas.

8 Noms

8.1 *Définitions*

a) *pseudonyme, nom-pseudonyme:* Nom d'un objet obtenu par l'utilisation d'une ou de plusieurs entrées pseudonymes dans le DIT;

b) *déréférençage:* Remplacement du nom-pseudonyme d'un objet par le nom spécifique de cet objet;

c) *nom spécifique* (d'un objet): L'un des noms de l'objet, formé à partir de la séquence des RDN de l'entrée d'objet et de chacune de ses entrées supérieures;

d) *nom (d'annuaire):* Structure qui distingue un objet particulier parmi tous les autres objets. Un nom ne doit pas être ambigu (c'est-à-dire qu'il ne doit désigner qu'un seul objet), mais il ne doit pas nécessairement être unique (c'est-à-dire être le seul nom qui désigne l'objet sans ambiguïté);

e) *nom visé:* Structure qui est, du point de vue syntaxique, un nom mais qui n'est pas (encore) apparue comme un nom valable;

f) *autorité de désignation:* Autorité responsable de l'attribution de noms. Chaque objet dont l'entrée d'objet est située à un sommet non-feuille dans le DIT est, ou est étroitement, associé à une autorité de désignation;

g) *nom spécifique relatif (RDN):* Ensemble d'assertions de valeur d'attribut dont chacune est vraie, concernant les valeurs spécifiques d'une entrée particulière.

8.2 *Noms en général*

8.2.1 Un *nom (d'annuaire)* est une structure qui identifie un objet particulier parmi l'ensemble de tous les objets. Un nom ne doit pas être ambigu, c'est-à-dire qu'il ne doit désigner qu'un seul objet. Cependant, un nom ne doit pas être nécessairement unique, c'est-à-dire être le seul nom qui désigne l'objet sans ambiguïté.

8.2.2 Du point de vue syntaxique, chaque nom d'un objet est une séquence ordonnée de noms spécifiques relatifs (voir le § 8.3).

NAME ::=
CHOICE { *--only one possibility for now--*
RDNSequence}

RDNSequence ::= SEQUENCE OF RelativeDistinguishedName

DistinguishedName ::= RDNSequence

Remarque - Les noms qui sont formés d'une autre manière que celle décrite ici, constituent une extension future éventuelle.

8.2.3 La séquence nulle est le nom de la racine de l'arbre.

8.2.4 Chaque sous-séquence initiale du nom d'un objet est également le nom d'un objet. La séquence d'objets ainsi identifiés, commençant par la racine se terminant par l'objet désigné, est telle que chacun d'eux est le supérieur immédiat de celui qui le suit dans la séquence.

8.2.5 Un *nom visé* est une structure qui du point de vue syntaxique est un nom mais qui n'est pas (encore) apparue comme un nom valable.

8.3 *Noms spécifiques relatifs*

8.3.1 Chaque entrée a un *nom spécifique relatif* unique *(RDN).* Un RDN est formé d'une séquence particulière d'assertions de valeur d'attribut dont chacune est vraie, concernant les valeurs spécifiques de l'entrée.

RelativeDistinguishedName ::=
SET OF AttributeValueAssertion

L'ensemble contient exactement une assertion sur chaque valeur spécifique de l'entrée.

8.3.2 Les RDN de toutes les entrées ayant un supérieur immédiat particulier sont distincts. Il incombe à l'autorité de désignation compétente pour cette entrée de veiller à ce qu'il en soit ainsi en assignant d'une manière appropriée des valeurs d'attribut spécifiques.

Remarque - Souvent, une entrée contiendra une valeur spécifique unique (et le RDN comprendra donc une seule AVA); cependant, dans certaines circonstances (pour différencier), des valeurs additionnelles (et donc des AVA additionnelles) peuvent être utilisées.

8.3.3 Le RDN pour une entrée est choisi lorsque l'entrée est créée. Une valeur unique d'un quelconque type d'attribut peut faire partie du RDN, selon la nature de la catégorie d'objets désignée. L'attribution de RDN est considérée comme une fonction administrative qui peut ou non nécessiter une négociation entre les organisations ou les Administrations concernées. La présente Recommandation ne décrit pas un tel mécanisme de négociation et n'émet aucune hypothèse quant à la manière de le mettre en oeuvre. Le RDN peut, le cas échéant, être modifié par un remplacement complet.

Remarque - Les RDN sont conçus pour une longue durée de vie afin que les usagers de l'annuaire puissent mettre en mémoire les noms spécifiques d'objets (par exemple, dans l'annuaire lui-même) sans se soucier de leur obsolescence. Les RDN doivent donc être modifiés avec prudence.

8.4 *Noms spécifiques*

8.4.1 Le *nom spécifique* d'un objet donné est défini comme étant la séquence des RDN de l'entrée qui représente l'objet et de ceux de toutes les entrées supérieures de celui-ci (dans l'ordre décroissant). Compte tenu de la correspondance biunivoque entre les objets et les entrées d'objet, on peut considérer que le nom spécifique d'un objet identifie également l'entrée de l'objet.

Remarque 1 - Il est préférable que les noms spécifiques d'objets qui concernent des individus soient facilement utilisables par l'usager.

Remarque 2 - Le document ISO 7498/3 définit le concept d'un nom primitif. Un nom spécifique peut être utilisé comme nom primitif pour l'objet qu'il identifie en raison du fait: a) qu'il n'est pas ambigu, b) qu'il est unique et c) que la sémantique de sa structure interne (une séquence de RDN) ne doit pas nécessairement (mais peut naturellement) être comprise par l'usager de l'annuaire.

Remarque 3 - L'entrée de l'objet et ses supérieurs étant seuls concernés, les noms spécifiques d'objets ne peuvent jamais s'appliquer à des entrées pseudonymes.

8.4.2 Il est commode de définir le "nom spécifique" de la racine et d'une entrée pseudonyme, bien que, ni dans un cas ni dans l'autre, le nom ne soit également le nom spécifique d'un objet. Le nom spécifique de la racine est défini comme étant la séquence nulle. Le nom spécifique d'une entrée pseudonyme est défini comme étant la séquence de RDN de l'entrée pseudonyme et de ceux de toutes les entrées supérieures de celle-ci (dans l'ordre décroissant).

8.4.3 Un exemple qui illustre les concepts de RDN et de nom spécifique est donné dans la figure 4/X.501.

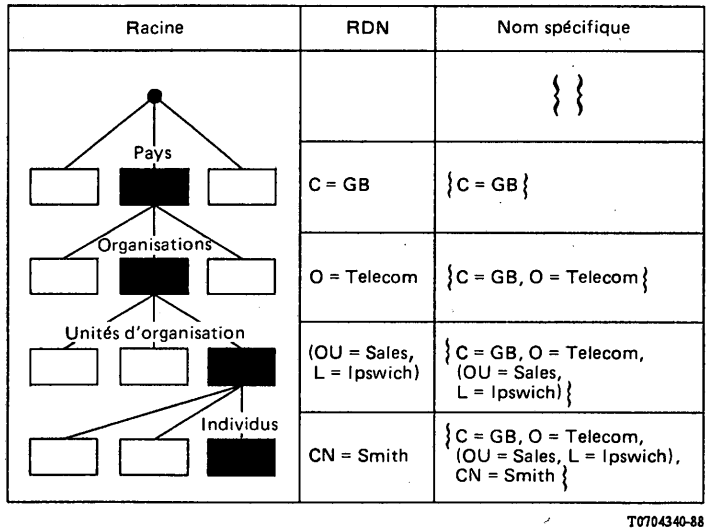


FIGURE 4/X.501

Détermination des noms spécifiques

8.5 *Noms-pseudonymes*

8.5.1 Un *pseudonyme* ou *nom-pseudonyme* pour un objet est un nom dont au moins l'un des RDN est celui d'une entrée pseudonyme. Les pseudonymes permettent aux entrées d'objet d'avoir des supérieurs immédiats multiples. Ils constituent donc la base d'un système de noms de remplacement.

8.5.2 De même que le nom spécifique d'un objet exprime sa relation principale avec une certaine hiérarchie d'objets, de même, un pseudonyme exprime (dans le cas général) une relation de remplacement avec une hiérarchie d'objets différente.

8.5.3 Dans le DIT, une entrée d'objet peut avoir zéro pseudonyme ou plusieurs pseudonymes. Il s'ensuit que plusieurs entrées pseudonymes peuvent faire référence à une entrée d'objet qui n'est pas une entrée feuille. Seules les entrées d'objet peuvent avoir des pseudonymes. Autrement dit, les pseudonymes de pseudonymes ne sont pas autorisés.

8.5.4 Une entrée pseudonyme n'a pas de subordonnées, ce qui signifie qu'une entrée pseudonyme est une entrée feuille.

8.5.5 L'annuaire utilise l'attribut de nom d'objet avec pseudonyme dans une entrée pseudonyme pour identifier et trouver l'entrée d'objet correspondante.

9 Schéma de l'annuaire

9.1 *Définitions*

a) *schéma d'annuaire:* Ensemble de définitions et de contraintes concernant la structure du DIT, les définitions de catégories d'objets, les types d'attribut et les syntaxes qui caractérisent la DIB;

b) *règle de structure du DIT:* Règle, faisant partie du schéma d'annuaire, qui met en relation une catégorie d'objets (subordonnés) avec une autre catégorie d'objets (supérieurs) et qui permet à une entrée de la première catégorie à être immédiatement subordonnée à une entrée des secondes catégories dans le DIT. Cette règle régit aussi le ou les types d'attribut autorisés à figurer dans le RDN de l'entrée (subordonnée) et peut imposer des conditions supplémentaires. Le schéma peut contenir de nombreuses règles de ce type.

9.2 *Aperçu général*

9.2.1 Le schéma d'annuaire est un ensemble de définitions et de contraintes concernant la structure du DIT et les modalités possibles de désignations des entrées, l'information qui peut figurer dans une entrée et les attributs utilisés pour représenter cette information.

Remarque 1 - Par exemple, le schéma permet au système d'annuaire:

- d'empêcher la création d'entrées subordonnées d'une catégorie d'objets incorrecte (par exemple un pays comme subordonné d'une personne);
- d'empêcher l'adjonction de types d'attribut à une entrée inappropriée à la catégorie d'objets (par exemple, un numéro de série à une entrée de personne);
- d'empêcher l'adjonction d'une valeur d'attribut d'une syntaxe ne correspondant pas à celle définie pour le type d'attribut (par exemple, une chaîne imprimable à une chaîne binaire).

Remarque 2 - Les mécanismes dynamiques pour la gestion du schéma d'annuaire ne sont pas actuellement prévus par cette série de Recommandations.

9.2.2 Du point de vue formel, le schéma d'annuaire se compose d'un ensemble:

a) de définitions (règles) de la *structure du DIT* qui définissent les noms spécifiques que peuvent avoir les entrées et les modalités des relations mutuelles entre ces noms par l'intermédiaire du DIT;

b) de définitions des *catégories d'objets* qui définissent l'ensemble des attributs obligatoires et facultatifs qui doivent être présents, et qui peuvent être présents, respectivement, dans une entrée d'une catégorie donnée (voir le § 6.2.3 de la présente Recommandation);

c) de définitions de *types d'attribut* qui identifient l'identificateur d'objet par lequel un attribut est connu, sa syntaxe et qui indiquent si cet attribut peut prendre plusieurs valeurs;

d) de définitions de *syntaxe d'attribut* qui définissent, pour chaque attribut, le type de données ASN.1 et les règles de mise en correspondance.

La figure 5/X.501 récapitule les relations qui existent entre les définitions du schéma, d'une part, le DIT, les entrées d'annuaire, les attributs et les valeurs d'attribut, d'autre part.

9.2.3 Le schéma d'annuaire est du type réparti, comme la DIB elle-même. Chaque autorité administrative établit les parties du schéma qui s'appliqueront aux parties de la DIB qu'elle administre.

Remarque - La répartition de l'information du schéma parmi les DSA gérés par des autorités administratives différentes n'est pas acceptée dans la présente série de Recommandations. Cette répartition est traitée administrativement par des accords bilatéraux.

9.2.4 La spécification des éléments impliqués dans la définition de la structure DIT, des catégories d'objets, des types d'attribut et des syntaxes d'attribut figure aux § 9.3 à 9.6 respectivement.

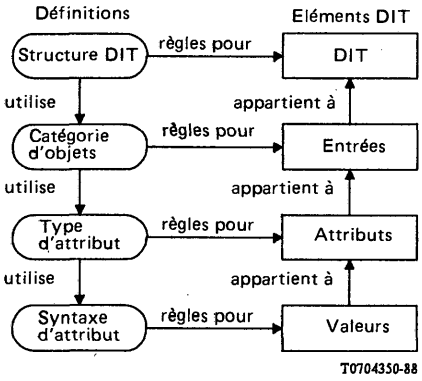


FIGURE 5/X.501

Aperçu du schéma d'annuaire

9.3 *Définition de la structure DIT*

9.3.1 Une règle de structure de DIT définit les relations hiérarchiques autorisées entre entrées ainsi que leurs RDN autorisés. La définition de la règle de structure du DIT implique:

- l'identification des catégories d'objets subordonné et supérieur;
- l'identification des types d'attribut qui peuvent intervenir dans les RDN des entrées subordonnées;
- (facultativement) une information complémentaire.

9.3.2 L'annuaire permet à une entrée de se tenir dans la relation de subordonné immédiat à une autre (son supérieur immédiat) seulement s'il existe une définition de structure de DIT contenue dans le schéma (voir le § 9.2.3) applicable à la portion de la DIB qui contiendrait l'entrée et pour laquelle:

- l'entrée est de la catégorie de l'objet subordonné;
- le supérieur immédiat de l'entrée est de la catégorie d'objet supérieur;
- le ou les types d'attribut constituant le RDN de l'entrée sont parmi ceux autorisés; et
- les conditions éventuellement imposées par un élément d'ensemble de l'information complémentaire sont satisfaites.

Remarque 1 - Les techniques à appliquer pour documenter la structure DIT ou pour représenter les règles de structuration dans la DIB ne sont pas encore spécifiées dans la présente série de Recommandations.

Remarque 2 - Si une règle de structure DIT autorise des subordonnés ou supérieurs immédiats appartenant à une catégorie donnée, elle autorise aussi implicitement (à moins de dérogation explicite) des subordonnés ou supérieurs appartenant à toute catégorie d'objets dérivée de cette catégorie (voir le § 9.4).

9.3.3 L'annuaire applique les règles de structuration définies pour chaque entrée du DIT. Toute tentative pour modifier le DIT en violation des règles de structuration applicables est vouée à l'échec.

9.3.4 Une règle de structure DIT dans laquelle une catégorie d'objet est le subordonné s'appelle une *liaison de nom* pour cette catégorie d'objet.

9.3.5 Pour qu'une catégorie d'objet soit représentée par des entrées dans une partie de la DIB, une liaison de nom au moins pour cette catégorie d'objet doit être contenue dans la partie applicable du schéma. Ce schéma contient des liaisons de nom supplémentaires selon les besoins.

Remarque - Il est concevable qu'une catégorie d'objet intervenant dans deux schémas distincts puisse avoir des liaisons de nom distinctes dans chaque schéma.

9.4 *Définition de catégorie d'objets*

9.4.1 La définition d'une catégorie d'objets implique:

a) facultativement, l'assignation d'un identificateur d'objet pour la catégorie d'objets;

b) l'indication des catégories dont cette catégorie doit être une sous-catégorie;

c) l'énumération des types d'attribut *obligatoires* qu'une entrée de la catégorie d'objets doit contenir, en plus des types d'attribut obligatoires de toutes ses super-catégories;

d) l'énumération des types d'attribut *facultatifs* qu'une entrée de la catégorie d'objets peut contenir, en plus des attributs facultatifs de toutes ses super-catégories.

Remarque - Une catégorie d'objets dépourvue d'identificateur d'objet assigné est destinée à l'usage local, comme moyen d'ajouter commodément de nouveaux types d'attribut à une super-catégorie prédéfinie. Cette adjonction offre plusieurs possibilités. Ainsi, une autorité administrative peut définir une catégorie d'objet non enregistrée pour permettre à un usager d'ajouter, s'il y a lieu, un attribut enregistré à l'entrée. L'autorité administrative peut limiter les attributs d'une entrée pour une catégorie d'objet donnée à ceux figurant sur une liste établie localement. Elle peut aussi rendre certains attributs obligatoires pour une catégorie d'objet donnée, en plus de ceux qu'exige la définition de la catégorie d'objet enregistrée.

9.4.2 Il existe une catégorie d'objets spéciale, dont chaque autre catégorie est une sous-catégorie. Cette catégorie d'objets, appelée "Top", est définie au § 9.4.8.1.

9.4.3 Chaque entrée contient un attribut de **Catégorie d'Objets** de type, pour identifier la catégorie et les super-catégories d'objets auxquelles appartient l'entrée. La définition de cet attribut est donnée au § 9.5.4. L'attribut a plusieurs valeurs. Il existe une valeur unique de l'attribut pour la catégorie d'objets et chacune de ses super-catégories pour lesquelles un identificateur d'objet est défini, à ceci près que la valeur de **"Top"** n'a pas besoin d'être présente tant qu'une autre valeur est présente.

Remarque 1 - La condition de présence de l'attribut de **Catégorie d'Objets** dans chaque entrée est exprimée dans la définition de **"Top"**.

Remarque 2 - Etant donné qu'une catégorie d'objets est censée appartenir à toutes ses super-catégories, chaque membre de la chaîne des super-catégories jusqu'à "Top" est représenté par une valeur de l'attribut de catégorie d'objet (et chaque valeur présente dans la chaîne peut être modélisée par un filtre).

L'attribut de **Catégorie d'Objets** est géré par l'annuaire, ce qui signifie qu'il ne peut pas être modifié par l'usager.

9.4.4 L'annuaire met en oeuvre la catégorie d'objets définie pour chaque entrée présente dans la DIB. Toute tentative de modifier une entrée en violation de la définition de la catégorie d'objets correspondant à l'entrée est vouée à l'échec.

Remarque - L'annuaire empêchera notamment:

a) que des types d'attribut absents de la définition de la catégorie d'objets soient ajoutés à une entrée de cette catégorie d'objets;

b) qu'une entrée soit créée avec un ou plusieurs types d'attribut obligatoires absents pour la catégorie d'objets correspondant à l'entrée;

c) qu'un type d'attribut obligatoire pour la catégorie d'objets correspondant à l'entrée soit supprimé.

9.4.5 La catégorie d'objets spéciale **Pseudonyme** est définie au § 9.4.8.2. Chaque entrée pseudonyme a une catégorie d'objets qui est une sous-catégorie de cette catégorie.

Remarque - Le déréférençage des entrées pseudonymes par l'annuaire a pour résultat que les valeurs de l'attribut de **Catégorie d'Objets** d'une entrée pseudonyme sont rarement perçues. Il est recommandé que des catégories d'objets pseudonymes appropriées soient dérivées de **Pseudonyme**, sans assignation d'un identificateur d'objet.

9.4.6 La macro ASN.1 suivante peut (mais ne doit pas nécessairement) être utilisée pour définir une catégorie d'objets. La production vide de **SubclassOf** est autorisée uniquement pour la définition de Top:

```
OBJECT-CLASS MACRO ::=
BEGIN

TYPENOTATION ::= SubclassOf
                 MandatoryAttributes
                 OptionalAttributes
```

VALUENOTATION ::=
value(VALUE OBJECT IDENTIFIER)

SubclassOf ::=
"SUBCLASS OF" Subclasses |
empty

Subclasses ::= Subclass | Subclass "," Subclasses
Subclass ::= value (OBJECT-CLASS)

MandatoryAttributes ::=
"MUST CONTAIN {"Attributes"}" | empty

OptionalAttributes ::=
"MAY CONTAIN {"Attributes"}" | empty

Attributes ::= AttributeTerm | AttributeTerm "," Attributes

AttributeTerm ::= Attribute | AttributeSet

Attribute ::= value(ATTRIBUTE)

AttributeSet ::= value(ATTRIBUTE-SET)

END

La correspondance entre les parties de la définition, telles qu'énumérées au § 9.4.1, et les divers éléments de la notation introduite par la macro est la suivante:

a) l'identificateur d'objet pour la catégorie d'objets est la valeur fournie dans l'assignation d'une valeur à la macro;

b) les super-catégories dont cette catégorie d'objets est une sous-catégorie, sont celles qui sont identifiées par la production **SubclassOf**, c'est-à-dire celles qui suivent **"SUBCLASS OF"**;

c) les attributs obligatoires sont ceux identifiés par la liste des identificateurs d'objet produits par la production des **Attributs Obligatoires**, c'est-à-dire ceux qui suivent **"MUST CONTAIN"**;

d) les attributs facultatifs sont ceux identifiés par la liste des identificateurs d'objet produits par la production des **Attributs Facultatifs**, c'est-à-dire ceux qui suivent **"MAY CONTAIN"**.

Remarque 1 - Les identificateurs d'objets visés en c) et d) identifient aussi bien des attributs individuels que des ensembles d'attributs (voir le § 9.4.7). Dans les deux cas, la liste effective est leur union, au sens de la théorie des ensembles. Si un attribut figure à la fois dans l'ensemble obligatoire et dans l'ensemble facultatif, il est considéré comme obligatoire.

Remarque 2 - La macro est utilisée pour définir certaines catégories d'objets dans la Recommandation X.521.

Si tous les éléments de notation introduits par la macro et décrits en b), c) et d) ci-dessus sont vides, la notation qui en résulte (**"OBJECT-CLASS"**) peut être utilisée pour désigner toute catégorie d'objet possible.

9.4.7 Un *ensemble attributs* est un ensemble d'attributs identifiés par un identificateur d'objet. La définition d'un ensemble attributs implique:

a) l'assignation d'un identificateur d'objet à l'ensemble;

b) l'énumération des identificateurs d'objet des attributs et des autres ensembles attributs dont la réunion des membres forme l'ensemble.

La macro ASN.1 suivante peut (mais ne doit pas nécessairement) être utilisée pour définir un ensemble d'attributs destiné à être utilisé avec la macro **CATEGORIE D'OBJETS**:

ATTRIBUTE-SET-MACRO ::=

BEGIN

TYPE NOTATION ::= "CONTAINS" "{"Attributes"}" | empty

VALUE NOTATION ::= value(VALUE OBJECT IDENTIFIER)

Attributes ::=

AttributeTerm | AttributeTerm "," Attributes

AttributeTerm ::= Attribute | AttributeSet

Attribute ::= value(ATTRIBUTE)

AttributeSet ::= value(ATTRIBUTE-SET)

END

La correspondance entre les parties de la définition d'un ensemble attributs et la notation introduite par la macro est la suivante:

a) l'identificateur d'objet assigné à l'ensemble attributs est la valeur fournie dans l'assignation d'une valeur à la macro;

b) l'ensemble d'attributs incluant l'ensemble attributs est celui formé par l'union au sens de la théorie des ensembles des attributs et des ensembles d'attributs identifiés par la production **Attributs**, c'est-à-dire celui qui suit "**CONTAINS**".

Si la version "**vide**" de la notation est choisie, la notation qui en résulte ("**ATTRIBUTE-SET**") peut être utilisée pour désigner tout ensemble attributs possible.

9.4.8 Les catégories d'objets précédemment mentionnées sont définies aux § 9.4.8.1 et 9.4.8.2.

Remarque - Il s'agit de définitions partielles: les identificateurs d'objet sont effectivement attribués pour ces catégories d'objets dans la Recommandation X.521 de telle sorte qu'il n'y ait qu'un seul point d'attribution de ces identificateurs d'objet dans cette série de Recommandations.

9.4.8.1 La catégorie d'objets "**Top**" est définie comme suit:

Top ::=
OBJECT-CLASS
MUST CONTAIN {ObjectClass}

9.4.8.2 La catégorie d'objets **pseudonyme** (**Alias**) est définie comme suit:

Alias ::=
OBJECT-CLASS
SUBCLASS OF top
MUST CONTAIN {aliasedObjectName}

Remarque 1 - La catégorie d'objets **pseudonyme** ne spécifie pas de types d'attribut appropriés pour le RDN d'une entrée pseudonyme. Les autorités administratives peuvent désigner des sous-catégories de la catégorie **pseudonyme** qui spécifient des types d'attribut utiles pour les RDN d'entrées pseudonymes (voir la Recommandation X.521).

Remarque 2 - Les entrées d'une sous-catégorie de la catégorie **pseudonyme** sont des entrées pseudonymes.

9.5 *Définition de type d'attribut*

9.5.1 La définition d'un type d'attribut implique:

a) l'assignation d'un identificateur d'objet au type d'attribut;

b) l'indication ou la définition de la syntaxe d'attribut pour le type d'attribut;

c) l'indication qu'un attribut de ce type peut avoir une seule valeur ou peut avoir plus d'une valeur (récurrence).

9.5.2 L'annuaire est conçu de telle sorte que la syntaxe d'attribut indiquée est utilisée pour chaque attribut de ce type et que les attributs de ce type ont une valeur et une seule dans les entrées s'ils sont définis comme n'ayant qu'une seule valeur.

9.5.3 La macro ASN.1 macro suivante peut (mais ne doit pas nécessairement) être utilisée pour définir un type d'attribut:

ATTRIBUTE MACRO ::=
BEGIN

TYPENOTATION ::= AttributeSyntax Multivalued | empty

```
VALUENOTATION ::= value (VALUE OBJECT IDENTIFIER)

AttributeSyntax      ::=
    "WITH ATTRIBUTE-SYNTAX" SyntaxChoice

Multivalued          ::= "SINGLE VALUE"
                         | "MULTIVALUE" | empty

SyntaxChoice         ::= value(ATTRIBUTE-SYNTAX)
                         Constraint | type MatchTypes

Constraint           ::= "("ConstraintAlternative")" | empty

ConstraintAlternative ::= StringConstraint | IntegerConstraint

StringConstraint     ::= "SIZE"  "("SizeConstraint")"

SizeConstraint       ::= SingleValue | Range

SingleValue          ::= value(INTEGER)

Range                ::= value(INTEGER) ".." value
                         (INTEGER)

IntegerConstraint    ::= Range

MatchTypes           ::= "MATCHES FOR"  Matches | empty

Matches              ::= Match Matches | Match

Match                ::= "EQUALITY" | "SUBSTRINGS" |
                         "ORDERING"

END
```

La correspondance entre les parties de la définition, telles qu'elles sont énumérées au § 9.5.1, et les divers éléments de la notation introduite par la macro est la suivante:

a) l'identificateur d'objet assigné au type d'attribut est la valeur fournie dans l'assignation de valeur de la MACRO;

b) la syntaxe d'attribut pour le type d'attribut est celle identifiée par la production **AttributeSyntax**. Ou bien elle dénote soit une syntaxe d'attribut définie séparément ou bien elle définit explicitement une syntaxe d'attribut en lui donnant son type ASN.1 et les règles de correspondance (voir le § 9.6). Si on emploie une syntaxe d'attribut identifiée séparément, on peut indiquer, facultativement, une contrainte de taille pour les types de chaîne sous-jacents ou une gamme de valeurs pour un type d'entier sous-jacent;

c) l'attribut a une valeur unique si la production **Multivalued** est "SINGLE VALUE"; il peut avoir une valeur unique ou plusieurs valeurs si cette production est "MULTI VALUE" ou vide.

Remarque - La macro est utilisée pour définir certains types d'attribut dans la Recommandation X.520.

Si l'on choisit la solution "**vide**" pour la notation de type, la notation qui en résulte ("**ATTRIBUTE**") peut être utilisée pour désigner tout type d'attribut possible.

9.5.4 Les types d'attribut identifiés au § 7.3.3, que l'annuaire connaît et utilise pour ses propres besoins, sont définis comme suit:

```
ObjectClass ::= ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax

AliasedObjectName ::= ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
    SINGLE VALUE
```

Remarque 1 - Il s'agit de définitions partielles: les identificateurs d'objet sont effectivement attribués pour ces types d'attribut dans la Recommandation X.520 de telle sorte qu'il n'y ait qu'un seul point d'attribution de ces identificateurs d'objet dans cette série de Recommandations.

Remarque 2 - Les syntaxes d'attribut auxquelles se réfèrent ces définitions sont elles-mêmes définies au § 9.6.5.

9.6 *Définition de syntaxe d'attribut*

9.6.1 La définition d'une syntaxe d'attribut implique:

a) facultativement, l'assignation d'un identificateur d'objet à la syntaxe d'attribut;

b) l'indication du type de données, en ASN.1, de la syntaxe d'attribut;

c) la définition de règles appropriées pour la mise en correspondance d'une valeur présentée et d'une valeur d'attribut cible contenue dans la DIB. Aucune des règles de mise en correspondance, certaines d'entre elles ou la totalité d'entre elles peuvent être définies pour une syntaxe d'attribut donnée:

i) égalité. Applicable à toute syntaxe d'attribut. La valeur présentée doit être conforme au type de données de la syntaxe d'attribut;

ii) sous-chaînes. Applicable à toute syntaxe d'attribut avec un type de données en chaîne. La valeur présentée doit être une séquence ("**SEQUENCE OF**"), dont chacun des éléments est conforme au type de données;

iii) hiérarchisation. Applicable à toute syntaxe d'attribut pour laquelle il est possible de définir une règle qui permettra de spécifier une valeur présentée comme étant inférieure, égale ou supérieure à une valeur cible. La valeur présentée doit être conforme au type de données de la syntaxe d'attribut.

9.6.2 En l'absence de définition d'une règle de mise en correspondance avec égalité, l'annuaire:

a) traite les valeurs comme des attributs de cette syntaxe d'attribut qui sont du type **ANY**, c'est-à-dire que l'annuaire ne vérifie pas que ces valeurs sont conformes au type de données indiqué pour la syntaxe d'attribut;

b) ne cherchera pas à mettre en correspondance les valeurs présentées avec les valeurs cibles du type d'attribut considéré.

Remarque - Il s'ensuit que l'annuaire n'autorisera pas l'emploi d'un tel attribut dans un nom spécifique, ni la modification d'une valeur spécifique.

9.6.3 Si une règle de mise en correspondance avec égalité est définie, l'annuaire:

a) traite les valeurs des attributs de la syntaxe d'attribut considérée comme étant du type **DEFINI PAR (ANY DEFINED BY)** le type de données indiqué pour la syntaxe d'attribut;

b) n'établit une correspondance que conformément aux règles de mise en correspondance définies pour la syntaxe d'attribut considérée;

c) n'effectuera la mise en correspondance d'une valeur présentée, relevant d'un type de données approprié, que conformément aux spécifications du § 9.6.1 c).

9.6.4 La macro ASN.1 suivante peut, mais ne doit pas nécessairement, être utilisée pour définir les syntaxes d'attribut:

```
ATTRIBUTE-SYNTAX MACRO ::=
BEGIN

TYPE NOTATION ::= Syntax
                  MatchTypes | empty

VALUE NOTATION ::=
    value (VALUE OBJECT IDENTIFIER)

Syntax ::= type

MatchTypes ::= "MATCHES FOR" Matches | empty

Matches ::= Match Matches | Match

Match ::= "EQUALITY" | "SUBSTRINGS" | "ORDERING"

END
```

La correspondance entre les parties de la définition, telles qu'elles sont énumérées au § 9.6.1, et les divers éléments de la notation introduite par la macro est la suivante:

a) l'identificateur d'objet assigné à la syntaxe d'attribut est une valeur fournie dans l'assignation de valeurs de la macro;

b) le type de données de la syntaxe d'attribut est celui identifié par la production **Syntax**, c'est-à-dire celui qui suit le nom de la macro;

c) les règles de mise en correspondance définies sont l'égalité si "**EQUALITY**" figure dans la production **MatchTypes**; sous-chaînes si "**SUBSTRINGS**" y figure; et hiérarchisation si "**ORDERING**" y figure. Si la production est vide, aucune règle de mise en correspondance n'est définie.

Si l'on choisit la solution "**vide**" de la notation, la notation qui en résulte ("**ATTRIBUTE-SYNTAX**") peut être utilisée pour désigner toute syntaxe d'attribut possible.

Remarque 1 - Aucun moyen n'est prévu dans la macro pour définir effectivement les règles de mise en correspondance elles-mêmes: il convient à cet effet d'utiliser le langage naturel ou d'autres moyens.

Remarque 2 - La macro est utilisée pour définir certaines syntaxes d'attribut dans la Recommandation X.520.

9.6.5 Les syntaxes d'attribut utilisées au § 9.5.4 sont définies aux § 9.6.5.1 et 9.6.5.2.

Remarque - Il s'agit de définitions partielles: les identificateurs d'objet sont effectivement attribués pour ces syntaxes d'attribut dans la Recommandation X.520, de telle sorte qu'il n'y ait qu'un seul point d'attribution de ces identificateurs d'objet dans cette série de Recommandations.

9.6.5.1 La **Syntaxe d'Identificateur d'Objet** est définie comme suit:

```
ObjectIdentifierSyntax ::=
    ATTRIBUTE-SYNTAX
        OBJECT IDENTIFIER
        MATCHES FOR EQUALITY
```

La règle de mise en correspondance pour égalité est inhérente à la définition de l'identificateur d'objet type ASN.1.

9.6.5.2 La **Syntaxe de Nom Spécifique** est définie comme suit:

```
DistinguishedNameSyntax ::=
    ATTRIBUTE-SYNTAX
        DistinguishedName
        MATCHES FOR EQUALITY
```

Une valeur de nom spécifique présentée est égale à une valeur de nom spécifique cible si, et seulement si, tous les éléments suivants s'appliquent:

a) le nombre de RDN dans chaque valeur est le même;

b) les RDN correspondants ont le même nombre d'AVA;

c) les AVA correspondantes ont des types d'attribut identiques; les AVA correspondantes (c'est-à-dire celles ayant des types d'attribut identiques) ont des valeurs d'attribut assorties sur le plan de l'égalité (dans cet assortiment, les valeurs d'attribut jouent le même rôle - c'est-à-dire comme valeur présentée ou valeur cible - que celui que joue, dans l'assortiment général, le nom spécifique qui les contient).

SECTION 3 - *Modèle de sécurité*

10 Sécurité

10.1 L'annuaire existe dans un environnement où plusieurs autorités assurent l'accès à leur section respective de la DIB. Cet accès est réalisé en conformité avec la politique de sécurité (voir la Recommandation X.509) dans le domaine de sécurité où se trouve la section de DIB.

10.2 On examinera ici deux composantes spécifiques d'une politique de sécurité:

a) la définition d'une politique d'autorisation;

b) la définition d'une politique d'authentification.

10.3 La définition de l'autorisation, dans le contexte de l'annuaire, englobe les méthodes qui permettent:

a) de spécifier les droits d'accès;

b) de mettre en pratique les droits d'accès (commande d'accès);

c) de maintenir les droits d'accès.

10.4 La définition de l'authentification, dans le contexte de l'annuaire, englobe les méthodes qui permettent de vérifier:

a) l'identité des DSA et des usagers de l'annuaire;

b) l'identité de l'origine de l'information reçue en un point d'accès.

L'intégrité de l'information reçue relève de l'autorité locale; elle doit être en conformité avec la politique de sécurité en vigueur.

10.5 La présente Recommandation ne définit pas une politique de sécurité.

10.6 L'annexe F énonce des principes directeurs pour spécifier les droits d'accès.

10.7 La Recommandation X.509 définit des procédures d'authentification. Le DAP et le DSP peuvent fournir une authentification poussée du demandeur par la signature de la demande, l'intégrité des données de la demande par la signature de la demande, l'authentification poussée de la réponse et l'intégrité du résultat par la signature du résultat. Le DAP peut fournir une authentification simple entre un DUA et un DSA. Le DSP peut fournir une authentification simple entre deux DSA.

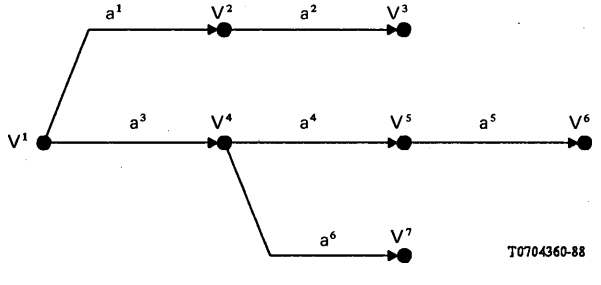
10.8 Les autorités administratives compétentes pour des applications qui utilisent l'annuaire peuvent appliquer leur politique de sécurité propre. L'annuaire peut mettre en oeuvre des applications en conservant une information d'authentification (par exemple, noms spécifiques, mots de passe, certificats) sur les entités de communication. Ce point est traité plus en détail dans la Recommandation X.509.

ANNEXE A

(à la Recommandation X.501)

La mathématique des arbres

La présente annexe ne fait pas partie intégrante de la présente Recommandation.



Un arbre est un ensemble de points, appelés *sommets*, et un ensemble de lignes orientées, appelées *arcs*; chaque arc va d'un sommet V à un sommet V'. Par exemple, l'arbre de la figure a sept sommets numérotés V^1 à V^7 et six arcs numérotés a^1 à a^6.

On dit que deux sommets V et V' sont respectivement les sommets *initial* et *final* d'un arc a allant de V à V'. Par exemple, V^2 et V^3 sont respectivement les sommets initial et final de l'arc a^2. Plusieurs arcs différents peuvent avoir le même sommet initial mais pas le même sommet final. Par exemple, les arcs a^1 et a^3 ont le même sommet initial, V^1, mais il n'y a pas deux arcs dans la figure qui aient le même sommet final.

Le sommet qui n'est pas le sommet final d'un arc quelconque est souvent appelé sommet *racine* ou même plus simplement la "racine" de l'arbre. Par exemple, dans la figure, V^1 est la racine.

Un sommet qui n'est pas le sommet initial d'un arc quelconque est souvent appelé un sommet *feuille*, ou même plus simplement une "feuille" de l'arbre. Par exemple, les sommets V^3, V^6 et V^7 sont des feuilles.

Un *trajet orienté* qui va d'un sommet V à un sommet V' est un ensemble d'arcs $(a^1, a^2, ..., a^n)$ $(n \geq 1)$ tel que V est le sommet initial de l'arc a^1, V' est le sommet final de l'arc a^n et le sommet final de l'arc a^k est également le sommet initial de l'arc a^{k+1}, avec $1 \leq k < n$. Par exemple, le trajet orienté du sommet V^1 au sommet V^6 est l'ensemble d'arcs (a^3, a^4, a^5). Il faut entendre par "trajet", un trajet orienté qui va de la racine à un sommet.

ANNEXE B

(à la Recommandation X.501)

Utilisation de l'identificateur d'objet

La présente annexe fait partie intégrante de la présente Recommandation.

Elle documente les parties supérieures du sous-arbre d'identificateurs d'objet où résident tous les identificateurs d'objet assignés dans cette série de Recommandations. A cet effet, elle fournit un module ASN.1 appelé **"Définitions utiles"** dans lequel des noms sont assignés à tous les noeuds non-feuilles du sous-arbre.

```
UsefulDefinitions    {joint-iso-ccitt ds(5) modules(1)
                     usefulDefinitions(0)}
DEFINITIONS ::=
BEGIN

EXPORTS
     module, serviceElement, applicationContext, attributeType, attributeSyntax, objectClass,
     algorithm, abstractSyntax, attributeSet,

     usefulDefinitions, informationFramework, directoryAbstractService,
     directoryObjectIdentifiers, algorithmObjectIdentifiers, distributedOperations,
     protocolObjectIdentifiers, selectedAttributeTypes, selectedObjectClasses,
     authenticationFramework, upperBounds,
     dap,dsp,

     id-ac, id-ase, id-as, id-ot, id-pt;

ds                OBJECT IDENTIFIER ::= {joint-iso-ccitt ds(5)}

-- categories of information object --

module            OBJECT IDENTIFIER ::= {ds 1}
serviceElement    OBJECT IDENTIFIER ::= {ds 2}
applicationContext OBJECT IDENTIFIER ::= {ds 3}
attributeType     OBJECT IDENTIFIER ::= {ds 4}
attributeSyntax   OBJECT IDENTIFIER ::= {ds 5}
objectClass       OBJECT IDENTIFIER ::= {ds 6}
attributeSet      OBJECT IDENTIFIER ::= {ds 7}
algorithm         OBJECT IDENTIFIER ::= {ds 8}
abstractSyntax    OBJECT IDENTIFIER ::= {ds 9}
object            OBJECT IDENTIFIER ::= {ds 10}
port              OBJECT IDENTIFIER ::= {ds 11}
```

```
-- modules --

usefulDefinitions                   OBJECT IDENTIFIER ::= {module 0}
informationFramework                OBJECT IDENTIFIER ::= {module 1}
directoryAbstractService            OBJECT IDENTIFIER ::= {module 2}
distributedOperations               OBJECT IDENTIFIER ::= {module 3}
protocolObjectIdentifier            OBJECT IDENTIFIER ::= {module 4}
selectedAttributeTypes              OBJECT IDENTIFIER ::= {module 5}
selectedObjectClasses               OBJECT IDENTIFIER ::= {module 6}
authenticationFramework             OBJECT IDENTIFIER ::= {module 7}
algorithmObjectIdentifiers          OBJECT IDENTIFIER ::= {module 8}
directoryObjectIdentifiers          OBJECT IDENTIFIER ::= {module 9}
upperBounds                         OBJECT IDENTIFIER ::= {module 10}
dap                                 OBJECT IDENTIFIER ::= {module 11}
dsp                                 OBJECT IDENTIFIER ::= {module 12}
distributedDirectoryObjectIdentifier OBJECT IDENTIFIER ::= {module 13}

-- synonyms --

id-ac     OBJECT IDENTIFIER ::= applicationContext
id-ase    OBJECT IDENTIFIER ::= serviceElement
id-as     OBJECT IDENTIFIER ::= abstractSyntax
id-ot     OBJECT IDENTIFIER ::= object
id-pt     OBJECT IDENTIFIER ::= port

END
```

ANNEXE C

(à la Recommandation X.501)

Cadre d'information dans l'ASN.1

La présente annexe fait partie intégrante de la présente Recommandation.

Elle fournit un résumé de toutes les définitions des termes "Type ASN.1", "Valeur ASN.1" et "ASN.1 macro" contenus dans cette Recommandation. Ces définitions constituent le module ASN.1 "Cadre d'information".

```
InformationFramework    {joint-iso-ccitt ds(5) modules(1)
                         informationFramework(1)}
DEFINITIONS ::=
BEGIN

EXPORTS
    Attribute, AttributeType, AttributeValue, AttributeValueAssertion,
    DistinguishedName, Name, RelativeDistinguishedName,
    OBJECT-CLASS,ATTRIBUTE,ATTRIBUTE-SET,ATTRIBUTE-SYNTAX,
    Top, Alias,
    ObjectClass, AliasedObjectName,
    ObjectIdentifierSyntax, DistinguishedNameSyntax;

IMPORTS
    selectedAttributeTypes, selectedObjectClasses
        FROM UsefulDefinitions {joint-iso-ccitt ds(5) modules(1)
            usefulDefinitions(0)}
    top
        FROM SelectedObjectClasses selectedObjectClasses
    objectIdentifierSyntax, distinguishedNameSyntax, objectClass, aliasedObjectName
        FROM SelectedAttributeTypes selectedAttributeTypes;
```

```
-- attribute data types --

Attribute               ::=  SEQUENCE{
                                 type          AttributeType
                                 values SET OF AttributeValue
                                 -- at least one value is required --}

AttributeType           ::=  OBJECT IDENTIFIER

AttributeValue          ::=  ANY

AttributeValueAssertion ::=  SEQUENCE {AttributeType, AttributeValue}

-- naming data types --

Name            ::=  CHOICE {-- only one possibility for now --
                             RDNSequence}

RDNSequence                  ::=  SEQUENCE OF
                                      RelativeDistinguishedName

DistinguishedName            ::=  RDNSequence

RelativeDistinguishedName    ::=  SET OF AttributeValueAssertion

-- macros --

OBJECT-CLASS MACRO  ::=
BEGIN
    TYPENOTATION        ::=  SubclassOf MandatoryAttributes
                                 OptionalAttributes
    VALUENOTATION       ::=  value (VALUE OBJECT IDENTIFIER)

    SubclassOf          ::=  "SUBCLASS OF" Subclasses | empty
    Subclasses          ::=  Subclass | Subclass "," Subclasses
    Subclass            ::=  value (OBJECT-CLASS)
    MandatoryAttributes ::=  "MUST CONTAIN {"Attributes"}" | empty
    OptionalAttributes  ::=  "MAY CONTAIN {"Attributes"}" | empty

    Attributes          ::=  AttributeTerm | AttributeTerm ","
                             Attributes
    AttributeTerm       ::=  Attribute | AttributeSet
    Attribute           ::=  value(ATTRIBUTE)
    AttributeSet        ::=  value(ATTRIBUTE-SET)

END

ATTRIBUTE-SET-MACRO ::=
    BEGIN
    TYPE NOTATION       ::=  "CONTAINS" "{"Attributes"}" | empty

    VALUE NOTATION      ::=  value(VALUEOBJECTIDENTIFIER)

    Attributes          ::=  AttributeTerm | AttributeTerm "," Attributes

    AttributeTerm       ::=  Attribute | AttributeSet
    Attribute           ::=  value(ATTRIBUTE)
    AttributeSet        ::=  value(ATTRIBUTE-SET)

    END

ATTRIBUTE MACRO     ::=
BEGIN
TYPENOTATION        ::=  AttributeSyntax Multivalued | empty
VALUENOTATION       ::=  value(VALUE OBJECT IDENTIFIER)

AttributeSyntax     ::=  "WITH ATTRIBUTE-SYNTAX"  SyntaxChoice

Multivalued         ::=  "SINGLE VALUE" | "MULTI VALUE" | empty

SyntaxChoice        ::=  value(ATTRIBUTE-SYNTAX)
                         Constraint | type Match Types
```

```
Constraint              ::=  "("ConstraintAlternative")" | empty
ConstraintAlternative   ::=  StringConstraint | IntegerConstraint
StringConstraint        ::=  "SIZE" "("SizeConstraint")"
SizeConstraint          ::=  SingleValue | Range
SingleValue             ::=  value(INTEGER)
Range                   ::=  value(INTEGER) ".." value(INTEGER)
IntegerConstraint       ::=  Range
MatchTypes              ::=  "MATCHES FOR"  Matches | empty
Matches                 ::=  Match Matches | Match
Match                   ::=  "EQUALITY" | "SUBSTRINGS" | "ORDERING"
END

ATTRIBUTE-SYNTAX MACRO ::=
BEGIN
    TYPENOTATION    ::=  Syntax MatchTypes | empty
    VALUENOTATION   ::=  value(VALUE OBJECT IDENTIFIER)

    Syntax          ::=  type

    MatchTypes      ::=  "MATCHES FOR" Matches | empty
    Matches         ::=  Match Matches | Match
    Match           ::=  "EQUALITY" | "SUBSTRINGS" | "ORDERING"
END


-- object classes --

Top     ::=     OBJECT-CLASS
                    MUST CONTAIN {objectClass}
Alias   ::=     OBJECT-CLASS
                    SUBCLASS OF top
                    MUST CONTAIN {aliasedObjectName}


-- attribute types --

ObjectClass         ::=  ATTRIBUTE
                             WITH ATTRIBUTE-SYNTAX objectIdentifierSyntax
AliasedObjectName   ::=  ATTRIBUTE
                             WITH ATTRIBUTE-SYNTAX distinguishedNameSyntax
                             SINGLE VALUE


-- attribute syntaxes --

ObjectIdentifierSyntax ::=
    ATTRIBUTE-SYNTAX
    OBJECT IDENTIFIER
    MATCHES FOR EQUALITY
DistinguishedNameSyntax ::=
    ATTRIBUTE-SYNTAX
    DistinguishedName
    MATCHES FOR EQUALITY
END
```

ANNEXE D

(à la Recommandation X.501)

Index alphabétique des définitions

Cette annexe ne fait pas partie intégrante de la présente Recommandation.

Cette annexe énumère, dans l'ordre alphabétique, tous les termes définis dans la présente Recommandation, avec un renvoi aux paragraphes où ils sont définis.

ANNEXE E

(à la Recommandation X.501)

Critères de conception des noms

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

Le cadre d'information est très général et permet d'introduire un nombre arbitraire d'entrées et d'attributs dans le DIT. Les noms étant, comme indiqué par ailleurs, étroitement liés aux trajets à travers le DIT, il s'ensuit qu'un nombre arbitraire de noms est possible. La présente section propose des critères à prendre en considération pour la conception des noms. Les critères appropriés ont été utilisés dans la conception des formes de noms recommandées qui figurent plus loin dans le présent document. Il est proposé que les critères soient également utilisés, le cas échéant, pour la conception de noms pour des objets auxquels les formes de noms recommandées ne s'appliquent pas.

Actuellement, un seul critère est pris en considération, celui de la facilité d'utilisation pour l'usager.

Remarque - Il n'est pas nécessaire que les noms soient tous faciles à utiliser par l'usager.

E.1 *Facilité d'utilisation pour l'usager*

Les noms auxquels les individus ont affaire directement doivent être faciles à utiliser par l'usager. Un nom facile à utiliser par l'usager est un nom qui tient compte du point de vue de l'usager et non de celui de l'ordinateur. C'est un nom que les individus peuvent facilement déduire, se rappeler et comprendre plutôt qu'un nom que les ordinateurs peuvent facilement interpréter.

On peut exprimer d'une manière un peu plus précise, l'objectif de cette facilité d'utilisation en fonction des deux principes suivants:

- Un individu doit généralement pouvoir deviner correctement le nom, commode pour l'usager, d'un objet, à partir des informations qu'il possède naturellement sur l'objet. Par exemple, il devrait être possible de deviner le nom d'un usager d'affaires uniquement sur la base des informations recueillies occasionnellement sur cette personne dans le cadre d'une association normale d'affaires.
- Lorsqu'un nom d'objet est spécifié d'une manière ambiguë, l'annuaire doit reconnaître ce fait au lieu de conclure que le nom identifie un objet particulier. Par exemple, lorsque deux personnes ont le même nom de famille, ce nom seul doit être considéré comme une identification inadéquate de l'un ou l'autre abonné.

Les sous-objectifs suivants résultent de l'objectif de facilité d'utilisation par l'usager:

a) Les noms ne doivent pas artificiellement supprimer les ambiguïtés naturelles. Par exemple, si deux personnes partagent le même nom de famille "Jones", ni l'une ni l'autre ne doit être tenue de répondre au nom "WJones" ou "Jones2". Au contraire, la convention de désignation de nom doit offrir un moyen commode pour l'usager d'établir une discrimination entre les entités. Par exemple, elle pourrait exiger, en plus du nom de famille, le premier prénom et l'initiale du second prénom.

b) Les noms doivent permettre des abréviations courantes et des variations courantes d'orthographe. Par exemple, si une personne est employée par la Conway Steel Corporation et si le nom de l'employeur figure dans le nom de cette personne, l'un des noms "Conway Steel Corporation", "Conway Steel Corp", "Conway Steel", et "CSC" devrait suffire à identifier l'organisation en question.

c) Dans certains cas, des noms pseudonymes peuvent être utilisés: pour orienter la recherche d'une entrée particulière, faciliter l'utilisation par l'usager ou réduire l'étendue d'une recherche. L'exemple suivant illustre l'utilisation d'un nom-pseudonyme pour un tel objectif: comme indiqué sur la figure E-1/X.501, la succursale d'Osaka peut être également identifiée par le nom {C = Japon, L = Osaka, 0 = ABC, OU = Osaka-branch}.

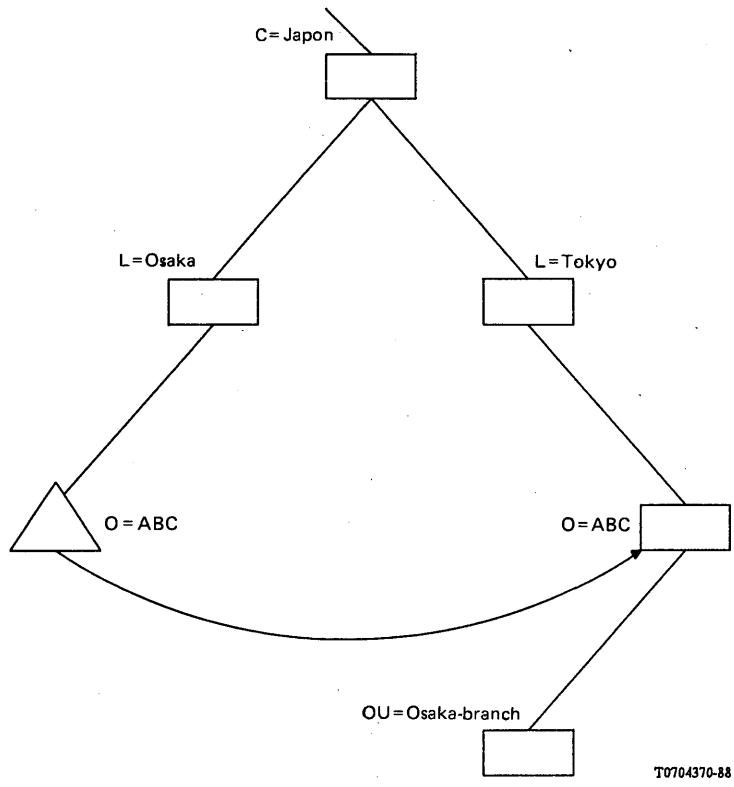


FIGURE E-1/X.501

Exemple d'attribution de pseudonymes

d) Si des noms sont formés de plusieurs éléments, le nombre d'éléments obligatoires et le nombre d'éléments facultatifs doivent être relativement faibles et il doit donc être facile de les mémoriser.

e) Si les noms sont formés de plusieurs éléments, l'ordre exact dans lequel ces éléments apparaissent doit être généralement immatériel.

f) Les noms faciles à utiliser par l'usager ne doivent pas inclure d'adresses d'ordinateur.

ANNEXE F

(à la Recommandation X.501)

Commande d'accès

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

F.1 *Introduction*

Les utilisateurs de l'annuaire peuvent avoir accès à l'information contenue dans la DIB en vertu de leurs droits de commande d'accès, conformément à la politique de commande d'accès en vigueur qui protège cette information.

Dans la présente série de Recommandations, la commande d'accès est laissée à l'initiative de l'autorité locale. Il est admis, cependant, que les applications devront introduire des moyens de commande d'accès et que les versions futures de la présente série de Recommandations définiront sans doute des moyens normalisés permettant de créer, de tenir à jour et d'appliquer des informations de commande d'accès. La présente annexe énonce les principes de base de la commande d'accès et décrit schématiquement deux méthodes possibles de commande d'accès.

F.2 *Principes*

Les deux principes sur lesquels sera fondée la mise en place de procédures pour gérer la commande d'accès sont les suivants:

a) il doit exister des moyens de protéger l'information présente dans l'annuaire contre une détection, une consultation et une modification non autorisées, y compris la protection du DIT contre une modification non autorisée;

b) l'information nécessaire pour déterminer les droits d'un usager d'effectuer une opération donnée doit être disponible pour le ou les DSA qui entre(nt) en jeu pour effectuer ladite opération, cela afin de ne pas être obligé d'effectuer d'autres opérations à distance à seule fin de déterminer ces droits.

F.3 *Eléments protégés*

Les niveaux suivants de protection sont identifiés actuellement:

a) protection d'un sous-arbre entier du DIT;

b) protection d'une entrée individuelle;

c) protection d'un attribut entier à l'intérieur d'une entrée;

d) protection d'exemples choisis de valeurs d'attribut.

F.4 *Catégories d'accès*

On prévoit qu'il est nécessaire d'avoir au moins cinq catégories d'accès. Si l'accès à un élément protégé n'est accordé dans aucune catégorie, l'annuaire répond, dans la mesure du possible, comme si l'élément protégé n'existait pas.

Les catégories d'accès sont spécifiées dans le tableau F-1/X.501. La colonne "Eléments" indique si l'élément pouvant être protégé selon ces modalités est une entrée (E), un attribut (A) ou les deux (EA).

TABLEAU F-1/X.501

Catégories d'accès

Catégorie	Eléments	Description
détection	A	Permet la détection de l'élément protégé.
comparaison	A	Permet de comparer une valeur présentée avec l'élément protégé.
lecture	A	Permet la lecture de l'élément protégé.
modification	A	Permet la mise à jour de l'élément protégé.
adjonction/ suppression	EA	Permet la création et la suppression de nouvelles composantes (attributs ou valeurs d'attribut) à l'intérieur de l'élément protégé.
désignation	E	Permet la modification du nom spécifique relatif, ainsi que la création et la suppression, d'entrées qui sont immédiatement subordonnées à l'entrée protégée.

F.5 *Détermination des droits d'accès*

Une modalité possible de la gestion de la commande d'accès associe à chaque élément protégé, explicitement ou implicitement, une liste de droits d'accès. Dans cette liste, chaque élément réalise un doublet entre un ensemble d'usagers et un ensemble de catégories d'accès.

L'information fournie avec la demande doit permettre de déterminer si un usager est inclus dans un (ou plusieurs) de ces ensembles: soit d'après l'identité authentifiée et l'accréditation de l'usager, fournies dans BIND; soit d'après l'information contenue dans l'argument de l'opération.

Il existe au moins deux possibilités:

a) Les ensembles sont décrits sur la base des noms spécifiques des usagers qu'ils identifient: il s'agit soit du nom spécifique de l'usager, soit du nom spécifique d'un supérieur avec un drapeau qui indique que le sous-arbre tout entier est inclus.

b) Les ensembles spécifient seulement une capacité et incluent implicitement tous les usagers qui ont cette capacité. Dans cette modalité, la capacité d'usager en question doit être disponible localement, ou être contenue dans BIND ou dans l'argument de l'opération. Dans ce dernier cas, il peut être nécessaire de prévoir une extension aux protocoles définis actuellement.

Recommandation X.509

ANNUAIRE - CADRE D'AUTHENTIFICATION [1)]

(Melbourne, 1988)

SOMMAIRE

---

1) La Recommandation X.509 et l'ISO 9594-8, Systèmes de traitement de l'information - Interconnexion des systèmes ouverts - L'annuaire - Cadre d'authentification, ont été élaborées en étroite collaboration et sont alignées du point de vue technique.

0 Introduction

0.1 La présente Recommandation, de même que les autres de la série, a été élaborée pour faciliter l'interconnexion des systèmes de traitement de l'information en vue d'assurer la prestation des services d'annuaire. L'ensemble de tous ces systèmes de même que l'information d'annuaire qu'ils conservent, peuvent être considérés comme un tout intégré appelé l'*annuaire*. Cette information détenue par l'annuaire connue généralement sous le nom de base d'information d'annuaire (DIB) est en général utilisée pour faciliter la communication entre, avec ou au sujet d'objets tels que des entités d'application OSI, des personnes, des terminaux et des listes de diffusion.

0.2 L'annuaire joue un rôle important dans l'interconnexion des systèmes ouverts, dont le but est de permettre sur une base de concertation technique minimale - mises à part les normes d'interconnexion elles-mêmes - l'interconnexion des systèmes de traitement de l'information:

- provenant de fabricants différents;
- placés sous des gestions différentes;
- de niveaux de complexité différents; et
- d'âge différent.

0.3 Un grand nombre d'applications comportent des exigences de sécurité pour assurer leur protection contre les dangers susceptibles de porter atteinte à la communication de l'information. L'annexe A contient une brève description des menaces généralement connues ainsi que les services et les mécanismes de sécurité qu'on peut utiliser pour la protection contre ces dernières. En fin de compte, tous les services de sécurité reposent sur la fiabilité de la connaissance des identités des parties en communication, c'est-à-dire sur leur authentification.

0.4 La présente Recommandation définit un cadre d'authentification pour assurer la prestation des services d'authentification par l'annuaire à ses utilisateurs. Ces utilisateurs comprennent l'annuaire lui-même ainsi que d'autres applications et services. L'annuaire peut utilement contribuer à répondre à leurs besoins en authentification et en d'autres services de sécurité car c'est un emplacement naturel à partir duquel les parties en communication peuvent obtenir l'information d'authentification des uns et des autres: connaissance sur laquelle repose l'authentification. L'annuaire est l'emplacement naturel du fait qu'il détient d'autres informations qui sont nécessaires à la communication et qui sont obtenues avant l'établissement de la communication. L'obtention de l'information d'authentification d'un partenaire d'une communication potentielle à partir de l'annuaire est, avec cette approche, semblable à l'obtention d'une adresse. En raison du domaine étendu recouvert par l'annuaire, on prévoit que ce cadre d'authentification sera très utilisé par toute une gamme d'applications.

1 Objectif et domaine d'application

1.1 La présente Recommandation:

- spécifie la forme sous laquelle l'information d'authentification est conservée par l'annuaire;
- décrit la façon dont on peut obtenir de l'annuaire cette information d'authentification;
- établit les hypothèses faites au sujet de la manière dont est constituée cette information d'authentification et de son emplacement dans l'annuaire;
- définit trois manières dont les applications peuvent employer cette information d'authentification pour effectuer des authentifications et explique comment d'autres services de sécurité peuvent être assurés par l'authentification.

1.2 La présente Recommandation contient les descriptions de deux niveaux d'authentification: l'authentification simple qui utilise un mot de passe pour la vérification de l'identité et l'authentification poussée qui fait intervenir des accréditations établies au moyen de techniques cryptographiques. Tandis que l'authentification simple n'offre qu'une protection limitée contre l'accès non autorisé, seule l'authentification poussée devra servir de base à la prestation de services sûrs.

1.3 On ne peut fournir d'authentification (et d'autres services de sécurité) que dans le contexte d'une politique de sécurité définie pour une application particulière. Il appartient à la ou aux normes de sécurité définissant cette application de définir leur propre politique de sécurité.

1.4 Il appartient aux normes de définir les applications qui utilisent le cadre d'authentification pour spécifier les échanges de protocole qu'il convient d'effectuer afin de parvenir à une authentification basée sur l'information d'authentification obtenue de l'annuaire. Le protocole utilisé par les applications pour obtenir des accréditations de l'annuaire est le protocole d'accès à l'annuaire (DAP) spécifié dans la Recommandation X.519.

1.5 La méthode d'authentification poussée spécifiée dans la présente Recommandation repose sur des systèmes cryptographiques à clé (de codage) publique. C'est le principal avantage de ces systèmes que les certificats d'utilisateur puissent être conservés dans l'annuaire et obtenus par les utilisateurs de l'annuaire de la même manière que d'autres informations de l'annuaire. On admet que les certificats d'utilisateur sont constitués par des moyens indépendants et sont mis en place dans l'annuaire par leur créateur. La génération de certificats d'utilisateur est effectuée par une autorité de certification autonome qui est complètement distincte des DSA de l'annuaire. En particulier, aucune exigence spéciale n'est prescrite aux fournisseurs de l'annuaire pour mémoriser ou communiquer les certificats d'utilisateur de façon sûre.

Une brève introduction à la cryptographie à clé publique est donnée dans l'annexe B.

1.6 En général, le cadre d'authentification ne dépend pas de l'utilisation d'un algorithme particulier pour autant qu'il possède les propriétés décrites au § 6.1. Il est possible en pratique d'utiliser un certain nombre d'algorithmes différents. Toutefois, deux utilisateurs qui désirent s'authentifier doivent utiliser le même algorithme cryptographique pour effectuer correctement l'authentification. De cette façon, dans le contexte d'un ensemble d'applications voisines, le choix d'un algorithme unique servira à élargir au maximum la communauté des utilisateurs capables de s'authentifier et de communiquer en sécurité. Un exemple d'algorithme cryptographique de clé publique est spécifié dans l'annexe C.

1.7 De même, deux utilisateurs qui désirent s'authentifier doivent utiliser la même fonction hachage [voir le § 3.3 f)] (utilisée pour former des accréditations et des jetons d'authentification). De nouveau, en principe, on peut utiliser un certain nombre de variantes de fonction hachage au prix d'un rétrécissement des communautés des utilisateurs capables de s'authentifier. Une brève introduction aux fonctions de hachage et un exemple de fonction de hachage sont donnés dans l'annexe D.

2 Références

2.1 ISO 7498-2: Systèmes de traitement de l'information - Interconnexion des systèmes ouverts - Architecture de sécurité.

3 Définitions

3.1 La présente Recommandation utilise les termes généraux relatifs à la sécurité définis dans la partie 2 du modèle de référence OSI sur la sécurité:

a) *asymétrique* (chiffrement);

b) *échange d'authentifications;*

c) *information d'authentification;*

d) *confidentialité;*

e) *accréditation;*

f) *cryptographie;*

g) *authentification de l'origine des données;*

h) *déchiffrement;*

i) *chiffrement;*

j) *clé;*

k) *mot de passe;*

l) *authentification d'entité homologue;*

m) *symétrique* (chiffrement).

3.2 Les termes suivants, utilisés dans la présente Recommandation, sont définis dans la Recommandation X.501:

a) *attribut;*

b) *base de données d'annuaire;*

c) *arbre d'information d'annuaire;*

d) *nom spécifique;*

e) *entrée;*

f) *objet;*

g) *racine.*

3.3 Les termes spécifiques suivants sont définis et utilisés dans la Recommandation:

a) *jeton d'authentification (jeton):* Information acheminée au cours d'un échange d'authentifications, qui peut être utilisée à authentifier son expéditeur;

b) *certificat:* Clé publique d'un utilisateur ainsi que certaines autres informations, rendue infalsifiable par chiffrement avec la clé secrète de l'autorité de certification qui l'a délivrée;

c) *autorité de certification:* Autorité chargée par un ou plusieurs utilisateurs de créer et d'attribuer leur clé publique et leur certificat. Cette autorité peut, facultativement, créer les clés d'utilisateur;

d) *itinéraire de certification:* Séquence ordonnée de certificats d'objets dans le DIT qui en plus de la clé publique de l'objet initial dans l'itinéraire, peut être traitée pour obtenir celle de l'objet final dans l'itinéraire;

e) *système cryptographique:* Recueil de transformations du texte en clair au texte chiffré et réciproquement, les transformations particulières à utiliser étant sélectionnées par des clés. Les transformations sont normalement définies par un algorithme mathématique;

f) *fonction hachage:* Fonction (mathématique) qui met en correspondance les valeurs d'un grand (et éventuellement très grand) domaine avec une gamme plus petite. Une "bonne" fonction de hachage est telle que les résultats de l'application de la fonction à un (grand) ensemble de valeurs du domaine seront régulièrement (et apparemment aléatoirement) répartis sur toute la gamme;

g) *fonction à une voie:* Fonction (mathématique) f qui est facile à calculer mais pour laquelle, à une valeur générale y de la gamme, il est difficile de faire correspondre par le calcul une valeur x du domaine telle que f(x) = y. Il peut exister un petit nombre de valeurs de y pour lesquelles la détermination de x n'est pas difficile à obtenir par le calcul;

h) *clé publique:* (dans un système cryptographique de clé publique) Clé d'une paire de clés d'utilisateur qui est publiquement connue;

i) *clé privée (clé secrète - déconseillée):* (dans un système cryptographique de clé publique) Clé d'une paire de clés d'utilisateur qui n'est connue que de cet utilisateur;

j) *authentification simple:* Authentification obtenue au moyen de simples arrangements de mot de passe;

k) *politique de sécurité:* Ensemble de règles établies par l'organisme de sécurité qui régit l'utilisation et la fourniture de services et de facilités de sécurité;

l) *authentification poussée:* Authentification obtenue au moyen d'accréditations déterminées par cryptographie;

m) *confiance:* Généralement, on peut dire qu'une entité se fie à une deuxième entité lorsqu'elle (la première entité) fait l'hypothèse que la deuxième entité se comportera exactement comme le prévoit la première entité. Cette confiance ne peut s'appliquer qu'à une certaine fonction particulière. Le rôle de confiance qui revient à la clé dans le cadre d'authentification consiste à décrire la relation entre une entité d'authentification et une autorité de certification; une entité d'authentification doit être certaine de pouvoir se fier à l'autorité de certification pour qu'elle crée uniquement des certificats valides et fiables;

n) *numéro de série d'utilisateur:* Valeur entière, unique pour la CA qui l'émet et associée sans ambiguïté au certificat émis par cette CA.

4 Notation et abréviations

4.1 La notation employée dans la Recommandation est définie dans le tableau 1/X.509.

Remarque - Dans l'introduction des notations, les symboles X, X_1, X_2, etc. apparaissent à la place des noms d'utilisateur tandis que le symbole I est mis à la place d'une information arbitraire.

4.2 Les abréviations suivantes sont utilisées dans la Recommandation.

CA Autorité de certification ("Certification Authority")

DIB Base de données d'annuaire ("Directory Information Base")

DIT Arbre d'information d'annuaire ("Directory Information Tree")

PKCS Système cryptographique à clé publique ("Public key cryptosystem").

TABLEAU 1/X.509

Notation

NOTATION	SIGNIFICATION
Xp	Clé publique d'un utilisateur X.
Xs	Clé secrète de X.
Xp[I]	Chiffrement d'une certaine information, I, au moyen de la clé publique de X.
Xs[I]	Chiffrement de [I] au moyen de la clé secrète de X.
X {I}	Signature de I par l'utilisateur X. Elle se compose de I avec un sommaire chiffré attaché.
CA(X)	Autorité de certification de l'utilisateur X.
$CA^n(X)$	(où n > 1): CA (CA (... n fois ...(X))).
$X_1\langle\langle X_2\rangle\rangle$	Certificat de l'utilisateur X_2 émis par l'autorité de certification X_1.
$X_1\langle\langle X_2\rangle\rangle X_2\langle\langle X_3\rangle\rangle$	Chaîne de certificats (pouvant être de longueur arbitraire) où chaque article est le certificat pour l'autorité de certification qui produit le texte. Il est fonctionnellement équivalent au certificat suivant $X_1\langle\langle X_{n+1}\rangle\rangle$. Par exemple, la possession A<<B>>B<<C>> fournit la même capacité que A<<C>>, à savoir la possibilité de découvrir Cp de Ap donné.
$X_{1p}\cdot X_1\langle\langle X_2\rangle\rangle$	Opération de dévoilement d'un certificat (ou d'une chaîne de certificats) pour en extraire une clé publique. C'est un opérateur d'infixe, dont l'opérande de gauche est la clé publique d'une autorité de certification, et dont l'opérande de droite est un certificat délivré par cette autorité de certification. Le résultat est la clé publique de l'utilisateur dont le certificat est l'opérande de droite. Par exemple: Ap·A<<B>>B<<C>> indique l'opération de l'utilisation de la clé publique de A pour obtenir la clé publique Bp de B, à partir de son certificat, suivi de l'utilisation de Bp pour dévoiler le certificat de C. Le résultat de l'opération est la clé publique Cp de C.
A → B	Itinéraire de certification de A en B composé d'une chaîne de certificats, débutant avec: $CA(A)\langle\langle CA^2(A)$ et finissant avec CA(B)<<B>>.

5 Procédure d'authentification simple

5.1 L'authentification simple vise à fournir une autorisation locale reposant sur un nom spécifique d'utilisateur, un mot de passe faisant l'objet (facultativement) d'un accord bilatéral et un accord bilatéral quant aux modalités d'emploi et de traitement de ce mot de passe dans un domaine donné. L'utilisation de l'authentification simple est essentiellement destinée à l'emploi local, c'est-à-dire pour authentification d'entités homologues entre un DUA et un DSA: l'authentification simple peut être réalisée de plusieurs manières:

a) transfert du nom spécifique de l'utilisateur et du mot de passe (facultatif) en clair (sans protection) au destinataire pour évaluation;

b) transfert du nom spécifique de l'utilisateur, du mot de passe et d'un numéro aléatoire et/ou d'une indication horaire, qui sont tous protégés par application d'une fonction à une voie;

c) transfert de l'information protégée décrite en b) ainsi qu'un numéro aléatoire et (ou) une indication horaire, qui sont tous protégés par application d'une fonction à une voie.

Remarque 1 - Il n'est pas exigé que les fonctions à une voie appliquées soient différentes.

Remarque 2 - La signalisation des procédures de protection des mots de passe pourra faire l'objet d'un complément à la présente Recommandation.

5.2 Quand les mots de passe ne sont pas protégés, un niveau de sécurité minimal est assuré pour empêcher un accès non autorisé. Il ne doit pas être considéré comme la base de services sûrs. La protection du nom spécifique de l'utilisateur et du mot de passe assure une sécurité plus grande. Les algorithmes à utiliser pour le mécanisme de protection sont en général des fonctions à une voie sans chiffrement qui sont très simples à mettre en oeuvre.

5.3 La figure 1/X.509 montre la procédure générale à appliquer pour obtenir une authentification simple.

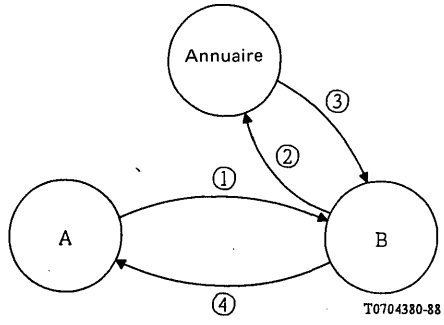


FIGURE 1/X.509

Procédure d'authentification simple sans protection

5.3.1 Les étapes de cette procédure sont les suivantes:

1) un utilisateur expéditeur A envoie son nom spécifique et son mot de passe à un utilisateur destinataire B;

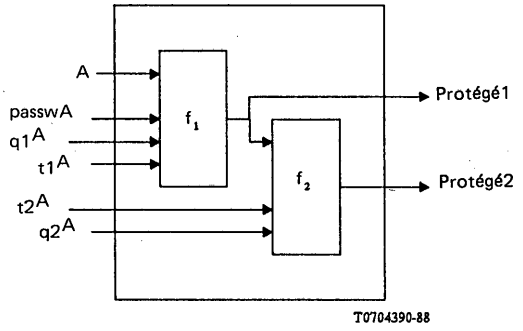
2) B envoie le nom spécifique visé et le mot de passe de A à l'annuaire, où le mot de passe est comparé avec celui qui détient l'attribut **mot de passe de l'utilisateur** dans l'entrée d'annuaire concernant A (en utilisant l'opération Compare de l'annuaire);

3) l'annuaire confirme (ou dément) à B que les accréditations sont valides;

4) le succès (ou l'échec) de l'authentification est communiqué à A.

5.3.2 La forme fondamentale d'authentification simple ne comporte que l'étape 1; elle peut aussi comporter l'étape 4 après que B a vérifié le nom spécifique et le mot de passe.

5.4 La figure 2/X.509 montre deux méthodes de production d'une information d'identification protégée. f_1 et f_2 sont des fonctions à une voie (identiques ou différentes) et les indications horaires et les numéros aléatoires sont facultatifs et dépendent d'accords bilatéraux.



A = nom spécifique de l'utilisateur
t^A = indications horaires
passwA = mot de passe de A
q^A = numéros aléatoires avec inclusion d'un compteur (facultatif)

FIGURE 2/X.509

Authentification simple protégée

5.4.1 La figure 3/X.509 montre la procédure d'authentification simple protégée.

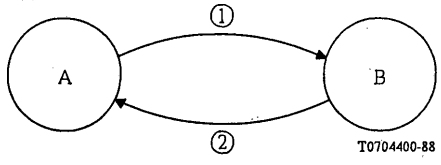


FIGURE 3/X.509

Procédure d'authentification simple protégée

Cette procédure comporte les étapes suivantes (avec, initialement, utilisation de f_1 seulement):

1) L'utilisateur d'origine (utilisateur A) envoie à l'utilisateur B son information d'identification protégée (Authentificator1). La protection est assurée par application de la fonction à une voie (f_1) de la figure 2/X.509, où l'indication horaire et (ou) le numéro aléatoire (s'il est utilisé) ont pour objet de réduire à un minimum les répétitions et de cacher le mot de passe.

   La protection du mot de passe de A a la forme:

   Protected1 = f_1 ($t1^A$, $q1^A$, A, passwA).

   L'information transmise à B prend la forme:

   Authentificator1 = $t1^A$, $q1^A$, A, Protected1.

B vérifie l'information d'identification protégée offerte par A en produisant une copie locale protégée du mot de passe de A (de la forme de Protected1) (au moyen de l'indication horaire, du nom spécifique et, facultativement, d'une indication horaire additionnelle et/ou d'un numéro aléatoire fourni par A, ainsi qu'une copie locale du mot de passe de A). B compare l'information d'identification visée (Protected1) avec la valeur produite localement, afin de s'assurer de leur égalité.

2) B confirme (ou dément) à A la vérification de l'information d'identification protégée.

5.4.2 La procédure du § 5.4.1 peut être modifiée de manière à assurer une plus grande protection (au moyen de f_1 et f_2).

Les principales différences sont les suivantes:

1) A envoie son information d'identification protégée (supplémentaire) (Authentificator2) à B. Une protection supplémentaire est obtenue en appliquant une autre fonction f_2 comme le montre la figure 2/X.509. La protection supplémentaire prend la forme:

Protected2 = f_2 ($t2^A$, $q2^A$, Protected1).

L'information transmise à B prend la forme:

Authentificator2 = $t1^A$, $t2^A$, $q1^A$, $q2^A$, A, Protected2.

Pour comparaison, B émet la valeur locale du mot de passe protégé additionnellement de A et le compare (pour en contrôler l'égalité) avec celle de Protected2 (le principe étant le même que pour l'étape 1 du § 5.4.1).

2) B confirme (ou dément) à A la vérification de l'information d'identification protégée.

Remarque - Les procédures définies dans le présent paragraphe sont spécifiées en fonction de A et B. Appliqué à l'annuaire (spécifié dans les Recommandations X.511 et X.518), A pourrait être un DUA lié à un DSA, B ou A pourrait être un DSA lié à un autre DSA, B.

5.5 Un type d'attribut de mot de passe d'utilisateur contient le mot de passe d'un objet. Une valeur d'attribut pour le mot de passe de l'utilisateur est une chaîne spécifiée par l'objet.

```
UserPassword ::=  ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
                OCTET STRING (SIZE (0..ub-user-password))
                MATCHES FOR EQUALITY
```

5.6 La macro ASN.1 suivante peut être utilisée pour définir le type de données provenant de l'application d'une fonction à une voie à un autre type de données donné.

```
PROTECTED MACRO ::= SIGNATURE
```

SECTION 2 - *Authentification poussée*

6 Bases de l'authentification poussée

6.1 La façon d'aborder une authentification poussée au sens de la présente Recommandation fait usage des propriétés de la famille des systèmes cryptographiques connus sous le nom de systèmes cryptographiques à clé publique (PKCS). Ces systèmes cryptographiques, également décrits comme asymétriques, font intervenir une paire de clés, l'une secrète et l'autre publique, plutôt qu'une seule clé comme dans les systèmes cryptographiques classiques. L'annexe B donne une brève introduction à ces systèmes cryptographiques et aux propriétés qui les rendent utiles pour l'authentification. Pour qu'un PKCS soit actuellement utilisable dans ce cadre d'authentification, il doit avoir la propriété de permettre l'usage des deux clés de la paire pour le chiffrement, la clé secrète étant utilisée pour le déchiffrement si c'est la clé publique qui a été utilisée, et la clé publique étant utilisée pour le déchiffrement si c'est la clé secrète qui a été utilisée. Autrement dit, $X_p \cdot X_s = Y_s \cdot Y_p$ si X_p/X_s sont des fonctions de chiffrement/déchiffrement utilisant les clés publique/secrète de l'utilisateur X.

Remarque - On pourra ultérieurement envisager d'autres types de PKCS, c'est-à-dire qui n'exigent pas la propriété de permutabilité et qui peuvent être mis en oeuvre sans grande modification de la présente Recommandation.

6.2 Le cadre d'authentification ne prescrit pas l'emploi d'un système cryptographique particulier. Il est prévu que ce cadre s'appliquera à tout système cryptographique à clé publique et qu'il suivra ainsi les modifications des méthodes utilisées à la suite des progrès à venir en crytographie, en techniques mathématiques ou en capacités de calcul. Toutefois deux utilisateurs qui désirent s'authentifier doivent utiliser le même algorithme cryptographique pour que les authentifications soient effectuées correctement. De cette façon, dans le contexte d'un ensemble d'applications voisines, le choix d'un algorithme unique servira à élargir au maximum la communauté des utilisateurs capables de s'authentifier et de communiquer en sécurité. Un exemple d'algorithme cryptographique est donné dans l'annexe C.

6.3 L'authentification repose sur la possession d'un nom spécifique unique pour chaque utilisateur. La responsabilité de l'attribution de noms spécifiques revient aux autorités de désignation. Chaque utilisateur doit donc se fier aux autorités d'appellation pour que les noms spécifiques ne soient pas émis en double.

6.4 Chaque utilisateur est identifié par la possession de sa clé secrète. Un deuxième utilisateur est capable de déterminer si un partenaire d'une communication est en possession de la clé secrète et peut utiliser ce renseignement pour confirmer que le partenaire de la communication est en fait l'utilisateur. La validité de cette confirmation dépend de la mesure dans laquelle la clé secrète demeure confidentielle au niveau de l'utilisateur.

6.5 Pour qu'un utilisateur puisse déterminer si un partenaire de communication est en possession de la clé secrète d'un autre utilisateur, il doit être lui-même en possession de la clé publique de cet utilisateur. S'il est simple d'obtenir la valeur de cette clé publique d'après l'inscription de l'utilisateur dans l'annuaire, il est plus problématique d'en vérifier l'exactitude. Il existe pour cela de nombreuses possibilités; le § 7 décrit un traitement au moyen duquel une clé publique d'utilisateur peut être contrôlée par référence à l'annuaire. Ce traitement ne peut remplir son rôle que s'il existe une chaîne continue de points de confiance dans l'annuaire entre les utilisateurs demandant à s'authentifier. Pour constituer cette chaîne on peut identifier un point de confiance commun. Ce point doit être relié à chaque utilisateur par une chaîne continue de points de confiance.

7 Obtention d'une clé publique d'usager

7.1 Pour qu'un utilisateur puisse avoir confiance en la procédure d'authentification, il doit obtenir la clé publique de l'autre utilisateur, d'une origine en laquelle il a confiance. Une telle origine, appelée autorité de certification (CA), utilise l'algorithme de clé publique pour certifier la clé publique en produisant un *certificat*. Ce certificat, dont la forme est spécifiée au § 7.2, possède les propriétés suivantes:

- tout utilisateur ayant accès à la clé publique de l'autorité de certification peut recouvrer la clé publique qui est certifiée;
- aucune partie autre que l'autorité de certification ne peut modifier le certificat sans que cela ne soit détecté (les certificats sont infalsifiables).

Du fait que les certificats sont infalsifiables, on peut les publier en les introduisant dans l'annuaire sans que ce dernier n'ait à prendre des dispositions particulières pour assurer leur protection.

Remarque - Bien que les CA soient définies sans ambiguïté par un nom spécifique dans le DIT, cela n'implique nullement une relation entre l'organisation des CA et le DIT.

7.2 Une autorité de certification produit les certificats de l'utilisateur en signant (voir le § 8) un recueil d'informations comprenant le nom spécifique d'utilisateur et la clé publique. Plus précisément, le certificat d'un utilisateur ayant A pour nom spécifique produit par l'autorité de certification CA prend la forme:

$$CA\langle\langle A\rangle\rangle = CA\ \{SN, AI, CA, A, Ap, T^A\}$$

où SN est le numéro de série du certificat, AI est l'identificateur de l'algorithme servant à signer le certificat: T^A indique la période de validité du certificat et comprend deux dates, correspondant au début et à la fin de cette validité. Comme on suppose que T^A est modifié par période d'au moins

24 heures, on prévoit que les systèmes utiliseront le temps universel coordonné comme base de temps de référence. Tout utilisateur ayant connaissance de CAP peut vérifier la validité de la signature du certificat. Le type de données ASN.1 suivant peut être utilisé pour représenter les certificats.

```
Certificate      ::=  SIGNED SEQUENCE{
     version                    [0]Version DEFAULT 1988,
     serialNumber               SerialNumber,
     signature                  Algorithmidentifier
     issuer                     Name
     validity                   Validity,
     subject                    Name,
     subjectPublicKeyInfo       SubjectPublicKeyInfo}

Version          ::=  INTEGER { 1988(0)}
SerialNumber     ::=  INTEGER

Validity ::=
     SEQUENCE{
          notBefore  UTCTime,
          notAfter   UTCTime}

SubjectPublicKeyInfo ::=
     SEQUENCE{
          algorithm      AlgorithmIdentifier
          subjectKey     BIT STRING}

AlgorithmIdentifier  ::=
     SEQUENCE{
          algorithm      OBJECT IDENTIFIER
          parameters     ANY DEFINED BY algorithm
                              OPTIONAL}
```

7.3 L'entrée d'annuaire de chaque utilisateur, A, qui participe à une authentification poussée contient le certificat de A. Ce certificat est produit par une autorité de certification de A, qui est une entité du DIT. L'autorité de certification de A, qui n'est pas forcément unique est indiquée par CA(A) ou simplement par CA si A est sous-entendu. La clé publique de A peut ainsi être découverte par tout utilisateur qui connaît la clé publique de CA. La découverte des clés publiques est ainsi récursive.

7.4 Si l'utilisateur A, qui essaie d'obtenir la clé publique de l'utilisateur B, a déjà obtenu la clé publique de CA(B), le processus est achevé. Pour permettre à A d'obtenir la clé publique de CA(B), l'entrée d'annuaire de chaque autorité de certification X contient plusieurs certificats, qui sont de deux types. D'abord les certificats vers l'avant de X produits par d'autres autorités de certification, ensuite les certificats inverses produits par X, qui sont les clés publiques certifiées d'autres autorités de certification. L'existence de ces certificats permet aux utilisateurs d'élaborer des itinéraires de certification entre deux points.

7.5 La liste des certificats nécessaires pour permettre à un utilisateur d'obtenir la clé publique d'un autre utilisateur est appelée *itinéraire de certification*, chaque élément de la liste étant un certificat de l'autorité de certification pour le suivant. Un itinéraire de certification de A en B (indiqué pour A → B):

- débute avec le certificat produit par CA(A), à savoir: $CA(A)\langle\langle X^1\rangle\rangle$ pour une entité X^1;
- continue avec d'autres certificats $X^i\langle\langle X^{i+1}\rangle\rangle$;
- finit avec le certificat de B.

Un itinéraire de certification forme logiquement une chaîne continue de points de confiance dans l'arbre d'information de l'annuaire entre deux utilisateurs qui désirent s'authentifier. La méthode précise utilisée par les utilisateurs A et B pour obtenir les itinéraires de certification A → B et B → A peut varier. Une façon d'y parvenir plus facilement consiste à établir une hiérarchie de CA pouvant coïncider ou non avec une partie, ou la totalité de la hiérarchie du DIT. L'avantage pour les utilisateurs qui ont des CA dans la hiérarchie est de pouvoir établir un itinéraire de certification entre eux au moyen de l'annuaire sans information préalable. Pour cela, chaque CA peut stocker un certificat et un certificat inverse désigné comme correspondant à son CA supérieure.

7.6 Les certificats sont conservés dans les entrées d'annuaire en tant qu'attributs des types **UserCertificate**, **CACertificate** et **CrossCertificatePair**. Ces types d'attribut sont connus de l'annuaire. On peut effectuer des opérations sur ces attributs en utilisant les mêmes opérations de protocole que pour d'autres attributs. La définition de ces types se trouve au § 3.3 de la présente Recommandation, leur spécification est la suivante:

```
UserCertificate          ::=    ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX Certificate
CACertificate            ::=    ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX Certificate

CrossCertificatePair     ::=    ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX CertificatePair

CertificatePair          ::=
    SEQUENCE{
        forward [0] Certificate OPTIONAL
        reverse [1] Certificate OPTIONAL
        -- at least one must be present --}
```

Un utilisateur peut obtenir une ou plusieurs certificats en provenance d'une ou plusieurs autorités de certification. Chaque certificat porte le nom de l'autorité de certification qui l'a émis.

7.7 Dans le cas général, avant que les utilisateurs puissent mutuellement s'authentifier, l'annuaire doit fournir les itinéraires complets de certification pour l'aller et le retour. Toutefois, dans la pratique, on peut réduire la quantité d'informations que doit fournir l'annuaire dans un cas donné d'authentification:

a) si les deux utilisateurs qui désirent s'authentifier sont desservis par la même autorité de certification, l'itinéraire de certification devient commun et les utilisateurs révèlent directement l'un à l'autre leur certificat;

b) si les CA des utilisateurs sont disposées par ordre hiérarchique, un utilisateur pourra mémoriser les clés publiques, les certificats et les certificats inverses de toutes les autorités de certification comprises entre les utilisateurs et la racine du DIT. Cela amène en général l'utilisateur à connaître les clés publiques et les certificats de trois ou quatre autorités de certification seulement. L'utilisateur dans ce cas n'aura besoin que d'obtenir les itinéraires de certification à partir du point commun de confiance;

c) si un utilisateur est fréquemment en communication avec d'autres utilisateurs certifiés par une autre CA donnée, il pourra connaître l'itinéraire de certification vers cette CA et l'itinéraire retour de certification provenant de cette CA, ne rendant ainsi nécessaire que l'obtention du certificat de l'autre utilisateur lui-même à partir de l'annuaire;

d) les autorités de certification peuvent se communiquer réciproquement leurs certificats par accord bilatéral, ce qui raccourcit l'itinéraire de certification;

e) si deux utilisateurs ont déjà communiqué et connaissent réciproquement leurs certificats, ils ont la possibilité de s'authentifier sans avoir recours à l'annuaire.

De toute façon, après avoir pris mutuellement connaissance de leurs certificats d'après l'itinéraire de certification, les utilisateurs doivent vérifier la validité des certificats reçus.

7.8 (Exemple). La figure 4/X.509 représente un exemple fictif de fragment de DIT, dans lequel les CA forment une hiérarchie. Outre l'information indiquée aux CA, on admet que chaque utilisateur a connaissance de la clé publique de son autorité de certification, ainsi que de ses propres clés publique et secrète.

7.8.1 Si les CA des utilisateurs sont disposées hiérarchiquement, A peut acquérir les certificats suivants en provenance de l'annuaire pour établir un itinéraire de certification de A vers B:

X<<W>>, W<<V>>, V<<Y>>, Y<<Z>>, Z<<B>>.

Lorsque A a obtenu ces certificats, il peut dévoiler l'itinéraire de certification en séquence pour livrer le contenu du certificat de B y compris Bp:

Bp = Xp · X<<W>> W<<V>> V<<Y>> Y<<Z>> Z<<B>>.

En général, A doit également acquérir les certificats suivants provenant de l'annuaire pour établir l'itinéraire retour de certification de B vers A:

Z<<Y>>, Y<<V>>, V<<W>>, W<<X>>, X<<A>>.

Lorsque B reçoit ces certificats de A, il peut dévoiler l'itinéraire retour de certification en séquence pour livrer le contenu du certificat de A y compris Ap:

Ap = Zp · Z<<Y>> Y<<V>> V<<W>> W<<X>> X<<A>>.

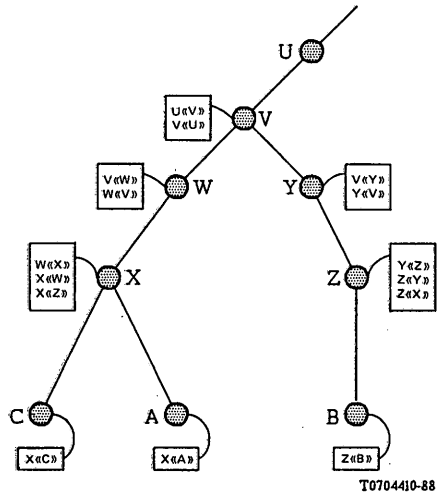


FIGURE 4/X.509

Hiérarchie des CA - Exemple fictif

7.8.2 En appliquant les optimisations du § 7.7:

a) Considérons A et C par exemple: tous les deux ont connaissance de Xp, de sorte qu'il ne reste à A qu'à acquérir directement le certificat de C. Le dévoilement de l'itinéraire de certification se réduit à:

Cp = Xp · X<<C>>

et le dévoilement de l'itinéraire retour de certification se réduit à:

Ap = Xp · X<<A>>.

b) En admettant que A ait connaissance de la même façon de W<<X>>, Wp, V<<W>>, Vp, U<<V>>, Up, etc., l'information que A doit obtenir de l'annuaire pour former l'itinéraire de certification, se réduit à:

V<<Y>>, Y<<Z>>, Z<<B>>

et l'information que A doit obtenir de l'annuaire pour former l'itinéraire retour de certification, se réduit à:

Z<<Y>>, Y<<V>>.

c) En admettant que A soit fréquemment en communication avec les utilisateurs certifiés par Z, il peut être au courant [en plus des clés publiques décrites à l'alinéa b) ci-dessus], de V<<Y>>, Y<<V>>, Y<<Z>> et Z<<Y>>. Pour communiquer avec B, il lui suffit donc d'obtenir seulement Z<<B>> de l'annuaire.

d) En admettant que les utilisateurs certifiés par X et Z soient fréquemment en communication, X<<Z>> sera conservé dans l'entrée d'annuaire pour X et réciproquement (ceci est représenté dans la figure 4/X.509). Si A désire procéder aux authentifications avec B, il lui suffit d'obtenir:

X<<Z>>, Z<<B>>

pour former l'itinéraire de certification, et

Z<<X>>

pour former l'itinéraire retour de certification.

e) En admettant que les utilisateurs A et C aient communiqué auparavant et aient été mis au courant réciproquement de leurs certificats, ils peuvent utiliser directement la clé publique de l'autre, c'est-à-dire:

Cp = Xp · X<<C>>

et

Ap = Xp · X<<A>>.

7.8.3 Dans le cas plus général, les autorités de certification n'ont pas entre elles de relations hiérarchiques. Dans l'exemple fictif de la figure 5/X.509, supposons que l'utilisateur D, certifié par U, désire s'authentifier avec l'utilisateur E certifié par W. L'entrée d'annuaire de l'utilisateur D contiendra le certificat U<<D>> et celle de l'utilisateur E contiendra le certificat W<<E>>.

Soit V une CA avec laquelle U et W ont auparavant échangé des clés publiques en confiance. Il s'ensuit que des certificats U<<V>>, V<<U>>, W<<V>> et V<<W>> ont été produits et mémorisés dans l'annuaire. En supposant que U<<V>> et W<<V>> sont stockés dans l'entrée de V, V<<U>> est stocké dans l'entrée de U et V<<W>> est stocké dans l'entrée de W.

L'utilisateur D doit trouver un itinéraire de certification vers E. Plusieurs méthodes peuvent être utilisées. L'une consiste à considérer les utilisateurs et les CA comme des noeuds et les certificats comme des arcs dans un graphique dirigé. Dans ces conditions, D doit rechercher dans le graphique un itinéraire de U vers E, qui pourra être U<<V>>, V<<W>>, W<<E>>. Une fois que cet itinéraire a été découvert, l'itinéraire inverse W<<V>>, V<<U>>, U<<D>> peut aussi être constitué.

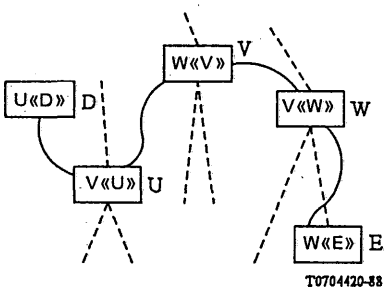


FIGURE 5/X.509

Exemple d'itinéaire de certification non hiérarchique

8 Signatures numériques

Ce paragraphe est destiné à spécifier non une norme traitant des signatures numériques en général, mais les moyens permettant de signer les jetons dans l'annuaire.

8.1 Pour signer une information (info) on lui ajoute un résumé chiffré de l'information. Ce résumé est produit au moyen d'une fonction hachage à une voie, tandis que le chiffrement est effectué au moyen de la clé secrète du signataire (voir figure 6/X.509) ainsi:

X{info} = Info,Xs[h (Info)]

Remarque - Le chiffrement utilisant la clé secrète garantit que la signature ne peut pas être falsifiée. La nature à une seule voie de la fonction hachage fait en sorte qu'une information fausse produite de façon à obtenir le même résultat de hachage (et ainsi la signature) ne puisse être substituée.

8.2 Le destinataire de l'information signée vérifie la signature en:

- appliquant la fonction hachage à une voie à l'information;
- comparant le résultat avec celui que l'on a obtenu par déchiffrement de la signature au moyen de la clé publique du signataire.

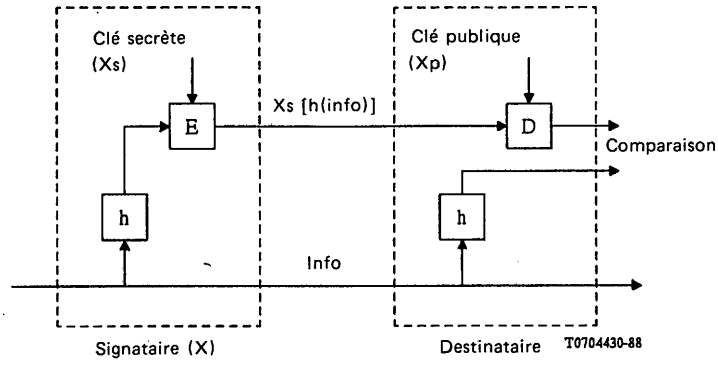


FIGURE 6/X.509

Signatures numériques

8.3 Ce cadre d'authentification ne prescrit pas d'utiliser pour la signature une fonction hachage unique à une voie. Il est prévu que ce cadre s'appliquera à toute fonction hachage appropriée et suivra ainsi les modifications des méthodes utilisées par suite des progrès futurs en cryptographie, dans les techniques mathématiques ou dans les capacités de calcul. Toutefois, deux utilisateurs qui désirent s'authentifier doivent utiliser la même fonction hachage pour que les authentifications soient effectuées correctement. De cette façon le contexte d'un ensemble d'applications voisines, le choix d'une fonction unique servira à élargir au maximum la communauté des utilisateurs capables de s'authentifier et de communiquer en sécurité. Une fonction hachage vraisemblablement appelée à être largement utilisée est spécifiée dans l'annexe D.

L'information signée comprend des indicateurs qui identifient l'algorithme de hachage et l'algorithme de chiffrement servant à établir la signature numérique.

8.4 Le chiffrement de certains éléments de données peut être décrit au moyen de la MACRO ASN.1 suivante:

```
ENCRYPTED MACRO     ::=
BEGIN

TYPE NOTATION       ::= type (ToBeEnciphered)
VALUE NOTATION      ::= value (VALUE BIT STRING)
END
```

La valeur de la chaîne de bits est produite ainsi: on prend les octets qui forment le codage complet (à l'aide des règles de codage de base ASN.1) de la valeur du **type à chiffrer** et on applique une procédure de chiffrement à ces octets.

Remarque 1 - La procédure de chiffrement exige un accord sur l'algorithme à employer et, le cas échéant, sur ses paramètres, comme les clés nécessaires, les valeurs d'initialisation et les instructions de remplissage. Il appartient aux procédures de chiffrement de spécifier les moyens qui permettent d'obtenir la synchronisation de l'émetteur et du récepteur des données, ce qui peut inclure une information dans les bits à transmettre.

Remarque 2 - La procédure de chiffrement doit prendre comme entrée une chaîne d'octets et engendrer une seule chaîne de bits en tant que résultat.

Remarque 3 - Les mécanismes d'obtention d'accord sur l'algorithme de chiffrement et ses paramètres par l'expéditeur et le destinataire des données n'entrent pas dans le cadre de la présente Recommandation.

8.5 Au cas où une signature doit être ajoutée à un type de données, la macro ASN.1 suivante peut être utilisée pour définir le type de données qui résulte de l'application d'une signature au type de données indiqué.

```
SIGNED MACRO    ::=
BEGIN

TYPE NOTATION   ::= type (ToBeSigned)

VALUE NOTATION  ::= value (VALUE

    SEQUENCE{
        ToBeSigned,
        AlgorithmIdentifier,
        -- of the algorithm used to compute
        -- the signature
        ENCRYPTED OCTET STRING
        -- where the octet string is the result
        -- of the hashing of the value of
        -- 'ToBeSigned' --}

END -- of SIGNED.    )
```

8.6 Au cas où seule la signature est exigée, la macro ASN.1 suivante peut être utilisée pour définir le type de données qui résulte de l'application d'une signature au type de données indiqué.

```
SIGNATURE MACRO    ::=
BEGIN

TYPE NOTATION      ::= type (OfSignature)

VALUE NOTATION     ::= value (VALUE

    SEQUENCE{
        AlgorithmIdentifier,
        -- of the algorithm used to compute
        -- the signature
        ENCRYPTED OCTET STRING
        -- where the octet string is a function (e.g. a
        -- compressed or hashed version) of the
        -- value 'OfSignature', which may include
        -- the identifier of the algorithm used to
        -- compute the signature --}

END -- of SIGNATURE.    )
```

8.7 Pour permettre la validation des types **SIGNED** et **SIGNATURE** dans un contexte réparti, un codage spécifique est nécessaire. Un tel codage d'une valeur de données **SIGNED** ou **SIGNATURE** est obtenu par application des règles de codage de base définies dans la Recommandation X.209, moyennant les restrictions suivantes:

a) on utilisera la forme définie de codage de longueur, codée dans un nombre minimal d'octets;

b) pour les types de chaîne, la forme construite de codage ne sera pas utilisée;

c) si la valeur d'un type est sa valeur par défaut, elle sera absente;

d) les éléments d'un type d'ensemble seront codés dans l'ordre ascendant de leur valeur d'étiquette;

e) les éléments d'un type d'ensemble seront codés par ordre ascendant de leur valeur d'octet;

f) si la valeur d'un type booléen est vraie, le codage aura son contenu d'octets mis sur $'FF'_{16}$;

g) tous les bits inutilisés du dernier octet lors du codage d'une valeur de chaîne de bits, sont, le cas échéant, mis sur zéro;

h) le codage du type réel sera tel que les bases 8, 10 et 16 ne seront pas utilisées et le facteur de proportionnalité binaire aura la valeur zéro.

9 Procédures d'authentification poussée

9.1 *Présentation générale*

9.1.1 La manière d'aborder fondamentalement l'authentification a été exposée ci-dessus, à savoir la confirmation de l'identité en faisant apparaître la possession d'une clef secrète. Or, il existe de nombreuses procédures possibles qui emploient cette approche. En général c'est la tâche d'une application particulière de déterminer les procédures appropriées de façon à répondre à la politique de sécurité pour l'application. Le § 9 contient la description de trois procédures particulières d'authentification qui peuvent s'avérer utiles au travers d'une gamme d'applications.

Remarque - La présente Recommandation ne spécifie pas les procédures en précisant les détails nécessaires à leur mise en oeuvre. Toutefois, des normes complémentaires pourraient être envisagées pour le faire soit dans le cas d'une application particulière, soit d'une façon qui viserait un objectif général.

9.1.2 Les trois procédures font intervenir différents nombres d'échanges d'information d'authentification et en conséquence offrent différents types d'assurance à leurs participants:

a) Une authentification à une voie, décrite au § 9.2, fait intervenir un transfert d'information unique d'un utilisateur (A) destiné à un autre utilisateur (B), et établit:

   - l'identité de A, et que le jeton d'authentification a été effectivement produit par A;
   - l'identité de B, et que le jeton d'authentification a été effectivement prévu pour être envoyé à B;
   - l'intégrité et "l'originalité" (la propriété de ne pas avoir été envoyé deux fois ou plus) du jeton d'authentification en cours de transfert.

   Ces dernières propriétés peuvent également être établies pour des données additionnelles arbitraires qui accompagnent le transfert.

b) Une authentification à deux voies, décrite au § 9.3, fait intervenir en plus une réponse de B en A. Elle établit en plus:

   - que le jeton d'authentification produit dans la réponse l'a effectivement été par B et qu'il est destiné à être envoyé en A;
   - l'intégrité et l'originalité du jeton d'authentification envoyé dans la réponse;
   - à titre d'option, le secret mutuel d'une partie des jetons.

c) Une authentification à trois voies, décrite au § 9.4, fait intervenir, en plus, un autre transfert de A en B. Elle établit les mêmes propriétés que l'authentification à deux voies mais le fait sans qu'il soit nécessaire d'associer une vérification de la date et de l'heure.

Dans chaque cas d'authentification poussée, A doit obtenir la clef publique de B et l'itinéraire retour de certification de B en A avant tout échange d'information. Ceci peut faire intervenir l'accès à l'annuaire comme décrit au § 7, mais cet accès, s'il existe, n'est pas mentionné de nouveau dans la description des procédures ci-dessous.

La vérification des dates et heures citée dans les paragraphes suivants, s'applique uniquement au cas où des horloges synchronisées sont employées dans un environnement local ou si des horloges sont synchronisées logiquement par des accords bilatéraux. Dans tous les cas, il est recommandé d'utiliser le temps universel coordonné.

Pour chacune des trois procédures d'authentification décrites ci-après, on admet que l'utilisateur A a vérifié la validité de tous les certificats sur le trajet de certification.

9.2 *Authentification à une voie*

Déroulement des opérations comme représenté à la figure 7/X.509.

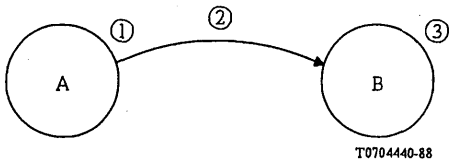


FIGURE 7/X.509

Authentification à une voie

1) A générer r^A, nombre non récurrent qui est utilisé pour détecter les attaques par redéfilement et pour empêcher les falsifications.

2) A envoie le message suivant en B:

   $B \rightarrow A,\ A\{t^A,r^A,B\}$

   où t^A représente la date et l'heure. t^A comprend une ou deux indications chronologiques, l'heure de production du jeton (facultatif) et la date d'expiration. Ou bien, si l'authentification de l'origine des données de "sgnData" doit être fournie par la signature numérique:

   $B \rightarrow A,\ A\{t^A,r^A,B,sgnData\}$

   Si l'information à transmettre est utilisée par la suite comme clé secrète (cette information est désignée par "encData"):

   $B \rightarrow A,\ A\{t^A,r^A,B,sgnData,Bp[encData]\}$

   L'emploi de "encData" comme clef secrète exige un soin particulier, par exemple qu'il s'agisse d'une clef solide pour tout système cryptographique utilisé comme indiqué dans le champ "sgnData" du jeton.

3) B effectue les opérations suivantes:

   a) obtient Ap à partir de $B \rightarrow A$, vérifiant que le certificat de A n'est pas périmé;

   b) vérifie la signature et ainsi l'intégrité de l'information signée;

   c) vérifie que B est bien le destinataire prévu;

   d) vérifie que la date et l'heure sont "à jour";

   e) facultativement, vérifie que r^A n'a pas été soumis à un redéfilement. Ceci peut par exemple s'obtenir en ayant introduit dans r^A une partie séquentielle qui est testée par une mise en oeuvre locale uniquement pour vérifier cette unicité de valeur.

r^A est valide jusqu'à la date d'expiration indiquée par t^A. r^A est toujours accompagné d'une partie séquentielle qui indique que A ne répétera pas le jeton pendant la durée t^A et ainsi, que la vérification de la valeur de r^A n'est pas nécessaire.

De toute façon, B a intérêt à stocker la partie séquentielle avec l'indication horaire t^A en clair et avec la partie hachée du jeton pendant la durée t^A.

9.3 *Authentification à deux voies*

Déroulement des opérations comme représenté à la figure 8/X.509.

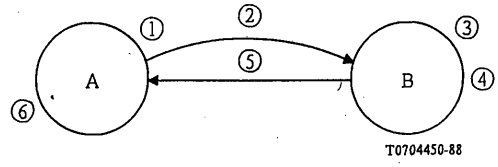


FIGURE 8/X.509

Authentification à deux voies

1) Comme pour le § 9.2.

2) Comme pour le § 9.2.

3) Comme pour le § 9.2.

4) B génère r^B, nombre non récurrent utilisé aux mêmes fins que r^A.

5) B envoie en A le jeton d'authentification suivant:

   B $\{t^B, r^B, A, r^A\}$

   où t^B est une indication horaire, définie comme t^A.

   Ou bien, si l'authentification d'origine des données de "sgnData" doit être fournie par la signature numérique:

   B $\{t^B, r^B, A, r^A, \text{sgnData}\}$

   Si l'information à transmettre est utilisée par la suite comme clef secrète (cette information est désignée par "encData"):

   B $\{t^B, r^B, A, r^A, \text{sgnData}, \text{Ap[encData]}\}$.

   L'emploi de "encData" comme clef secrète implique un choix approprié, par exemple pour obtenir une clef solide pour tout système cryptographique utilisé comme indiqué dans le champ "sgnData" du jeton.

6) A effectue les opérations suivantes:

   a) vérifie la signature et ainsi l'intégrité de l'information signée;

   b) vérifie que A est bien le destinataire prévu;

   c) vérifie que la date et l'heure t^B sont "à jour";

   d) facultativement, vérifie que r^B n'a pas été soumis à un redéfilement [voir au § 9.2 l'étape 3) e)].

9.4 *Authentification à trois voies*

Déroulement des opérations comme représenté à la figure 9/X.509.

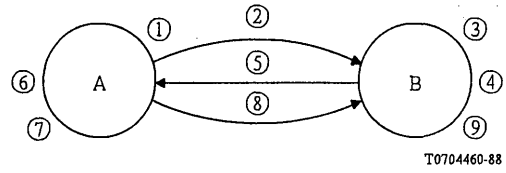


FIGURE 9/X.509

Authentification à trois voies

1) Comme pour le § 9.3.

2) Comme pour le § 9.3. L'indication horaire t^A peut avoir la valeur zéro.

3) Comme pour le § 9.3, excepté qu'il n'est pas nécessaire de vérifier la date ni l'heure.

4) Comme pour le § 9.3.

5) Comme pour le § 9.3. L'indication horaire t^B peut avoir la valeur zéro.

6) Comme pour le § 9.3 excepté qu'il n'est pas nécessaire de vérifier la date ni l'heure.

7) A vérifie que r^A reçu est identique au r^A émis.

8) A envoie en B le jeton d'authentification suivant:

   $A\{r^B\}$.

9) B effectue les opérations suivantes:

   a) vérifie la signature et ainsi vérifie l'intégrité de l'information signée;

   b) vérifie que le r^B reçu est identique au r^B émis par B.

10 Gestion des clefs et des certificats

10.1 *Génération de paires de clefs*

10.1.1 La politique de gestion de sécurité totale d'une mise en oeuvre définira le cycle de vie des paires de clefs et se trouve donc en dehors de l'objectif du cadre d'authentification. Cependant, il est capital pour la sécurité absolue que la connaissance des clefs secrètes reste limitée à l'utilisateur auquel ces clefs appartiennent.

Il n'est pas facile pour un utilisateur qui est un être humain de se souvenir des données des clefs, aussi il convient d'employer une méthode appropriée pour mémoriser ces données de manière à les transporter facilement. Un mécanisme réalisable consisterait à utiliser une "carte à mémoire". Elle conserverait les clefs secrète et (facultativement) publique de l'utilisateur, le certificat d'utilisateur et une copie de la clef publique de l'autorité de certification. L'emploi de cette carte devrait comporter une sécurité complémentaire, par exemple par l'utilisation d'un PIN ("Personal Identification Number": Numéro d'identification personnel), ce qui augmenterait la sécurité du système en exigeant de l'utilisateur qu'il possède la carte et qu'il sache comment y accéder. La méthode à choisir pour mémoriser ces données n'entre pas dans le cadre de la présente Recommandation.

10.1.2 Il y a trois façons de produire une paire de clefs comme indiqué aux § 10.1.2.1 à 10.1.2.3.

10.1.2.1 L'utilisateur génère sa propre paire de clefs. Cette méthode a l'avantage de ne jamais livrer une clef secrète d'utilisateur à une autre entité, mais elle nécessite un certain niveau de compétence de la part de l'utilisateur comme indiqué dans l'annexe C.

10.1.2.2 La paire de clefs est générée par un tiers. Celui-ci doit livrer la clef secrète à l'utilisateur de manière matérielle et sûre, puis détruire résolument toute information relative à la création de la paire de clefs, ainsi que les clefs elles-mêmes. Il conviendra de prendre les mesures de sécurité matérielles convenables, de manière que le tiers et les opérations sur les données soient exempts de toute fraude.

10.1.2.3 La paire de clefs est générée par la CA. C'est un cas particulier du § 10.1.2.2 et les mêmes considérations s'appliquent dans ce cas.

Remarque - L'autorité de certification présente déjà un potentiel de confiance auprès de l'utilisateur et sera soumise aux mesures nécessaires matérielles de sécurité. Cette méthode a l'avantage de ne pas nécessiter de transfert sur des données à la CA pour la certification.

10.1.2.4 Le système cryptographique en usage impose des contraintes (techniques) particulières à la génération des clefs.

10.2 *Gestion des certificats*

10.2.1 Un certificat associe la clef publique et le nom spécifique unique de l'utilisateur. De la sorte:

a) une autorité de certification doit s'assurer de l'identité d'un utilisateur avant de créer un certificat à son intention;

b) une autorité de certification ne doit pas délivrer de certificat pour deux utilisateurs ayant le même nom.

10.2.2 La production d'un certificat se présente comme une opération indépendante et ne doit pas être effectuée au moyen d'un mécanisme à question/réponse automatique. L'avantage de cette certification est que la clef secrète de l'autorité de certification, CAs, n'étant jamais connue si ce n'est de la CA isolée et matériellement sûre, le secret de la clef secrète ne peut être percé que par une attaque contre la CA elle-même, ce qui rend une compromission improbable.

10.2.3 Il importe que le transfert d'information à l'autorité de certification ne soit pas une source de compromission et il convient que des mesures matérielles appropriées de sécurité soient prises. A cet égard:

a) ce serait un grave manquement à la sécurité si la CA livrait un certificat à un utilisateur ayant une clef publique qui aurait été falsifiée;

b) si le moyen de générer des paires de clefs donné au § 10.1.2.3 est utilisé, aucun transfert sûr n'est nécessaire;

c) en cas d'emploi du moyen de production des paires de clefs du § 10.1.2.1 ou du § 10.1.2.2, l'utilisateur peut recourir à diverses méthodes (directes ou différées) pour communiquer sa clef publique à la CA en toute sécurité. Les méthodes "en ligne" sont parfois plus souples pour les opérations effectuées à distance entre l'utilisateur et la CA.

10.2.4 Un certificat est un élément d'information publiquement disponible et aucune mesure particulière de sécurité n'a besoin d'être prise en ce qui concerne son transport à l'annuaire. Etant donné qu'il est produit par une autorité de certification indépendante, au nom d'un utilisateur qui en recevra copie, il suffit à l'utilisateur de mémoriser cette information dans son entrée d'annuaire lors d'un accès subséquent à l'annuaire. Ou bien la CA pourra conserver le certificat pour l'utilisateur et dans ce cas il convient de donner à cet agent des droits d'accès appropriés.

10.2.5 Les certificats auront une durée de vie qui leur sera associée et à la fin de laquelle leur validité expirera. Pour assurer la continuité du service, la CA fera en sorte que des certificats de remplacement soient disponibles à temps pour remplacer les certificats dont la validité expire (ou a expiré). Les § 10.2.5.1 et 10.2.5.2 en montrent les différents aspects.

10.2.5.1 La validité des certificats peut être prévue de manière que chacun devienne valide au moment où expire la validité du précédent, mais un chevauchement des validités peut être autorisé. Dans ce dernier cas, cela épargne à la CA la nécessité d'installer et de distribuer un grand nombre de certificats qui peuvent ne plus être en vigueur à la date d'expiration.

10.2.5.2 Les certificats dont la validité a expiré sont en principe enlevés de l'annuaire. Il incombe à la CA de conserver les anciens certificats pendant quelque temps s'il existe un service de non-rejet des données.

10.2.6 Les certificats peuvent être annulés avant d'être périmés, par exemple si l'on suppose que la clef secrète de l'utilisateur a fait l'objet d'une compromission, ou si l'utilisateur ne doit plus être certifié par la CA, ou encore si l'on suppose que le certificat de la CA a donné lieu à une compromission. Les divers aspects en sont présentés aux § 10.2.6.1 à 10.2.6.4.

10.2.6.1 L'annulation d'un certificat d'utilisateur ou d'un certificat de CA sera communiquée par la CA et un nouveau certificat sera mis à disposition, si besoin est. La CA peut alors informer le titulaire du certificat que celui-ci est annulé en appliquant une procédure différée.

10.2.6.2 La CA conservera:

a) une liste datée des certificats qu'elle a émis et qui ont été annulés;

b) une liste datée des certificats annulés de toutes les clefs secrètes dont elle a connaissance, qu'elle a certifiés.

Ces deux listes certifiées doivent exister, même si elles sont vides.

10.2.6.3 Le maintien des entrées d'annuaire affectées par les listes d'annulation de la CA incombe à l'annuaire et à ses utilisateurs, qui agiront en conformité avec les dispositions de sécurité adoptées. Par exemple, l'utilisateur peut modifier son entrée d'objet en remplaçant l'ancien certificat par un nouveau. Ce dernier servira alors à authentifier l'utilisateur vis-à-vis de l'annuaire.

10.2.6.4 Les listes d'annulation ("listes noires") sont conservées dans les entrées comme des attributs des types **"Liste d'annulation de certificats"** et **"Liste d'annulation d'autorité"**. Ces attributs peuvent être exploités selon les mêmes opérations que les autres attributs. Ces types d'attribut sont définis ainsi:

```
CertificateRevocationList    ::=    ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX CertificateList

AuthorityRevocationList      ::=    ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX CertificateList

CertificateList              ::=    SIGNED SEQUENCE{
    signature AlgorithmIdentifier,
    issuer Name,
    lastUpdate UTCTime,
    revokedCertificates
        SIGNED SEQUENCE OF SEQUENCE{
            signature AlgorithmIdentifier,
            issuer Name, CertificateSerialNumber subject,
            revocationDate UTCTime}
                OPTIONAL}
```

Remarque 1 - La vérification de toute la liste des certificats est une question locale.

Remarque 2 - Si le service de non-rejet des données dépend des clefs fournies par la CA, le service doit s'assurer que toutes les clefs pertinentes de la CA (annulées ou périmées) et les listes d'annulation datées sont archivées et certifiées par une autorité actuelle.

ANNEXE A

(à la Recommandation X.509)

Exigences de sécurité

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

[On peut trouver le texte additionnel se rapportant à ce sujet dans le document ISO 7498 - Systèmes de traitement de l'information - Modèle de référence OSI - partie 2, Architecture de sécurité.]

De nombreuses applications OSI, des services définis par le CCITT et des services non définis par le CCITT comportent des exigences de sécurité. Ces exigences proviennent du besoin de protéger le transfert de l'information contre une gamme de dangers potentiels.

A.1 *Dangers*

Certains dangers sont bien connus:

a) *interception d'identité:* L'identité d'un ou de plusieurs des utilisateurs intervenant dans une communication est notée pour mauvaise utilisation;

b) *fausse identité:* Prétention d'un utilisateur à se faire passer pour un autre utilisateur afin d'avoir accès à l'information ou d'acquérir des privilèges supplémentaires;

c) *redéfilement:* Enregistrement et écoute ultérieure d'une communication;

d) *interception de données:* Observation de données d'utilisateur au cours d'une communication par un utilisateur non autorisé;

e) *manipulation:* Remplacement, insertion, suppression ou mise en désordre de données d'utilisateur au cours d'une communication par un utilisateur non autorisé;

f) *rejet:* Démenti d'un utilisateur d'avoir participé en partie ou pendant toute sa durée, à une communication;

g) *refus de service:* Empêchement ou interruption d'une communication ou encore retard d'opérations temporelles critiques;

Remarque - Ce danger pour la sécurité est très général et dépend de l'application sur le plan individuel ou de l'intention de la brusque interruption non autorisée; il ne fait donc pas explicitement partie de l'objectif du cadre d'authentification.

h) *acheminement erroné:* Acheminement erroné d'un itinéraire de communication prévu pour un usager vers un autre usager;

Remarque - L'acheminement erroné apparaîtra naturellement dans les couches OSI 1 à 3. L'acheminement erroné est donc en dehors de l'objectif du cadre d'authentification. Cependant, il peut être possible d'éviter les conséquences d'un acheminement erroné en utilisant les services de sécurité fournis dans le cadre d'authentification.

i) *analyse de trafic:* L'observation de l'information relative à une communication entre utilisateurs (c'est-à-dire absence/présence, fréquence, sens, séquence, type, volume, etc.).

Remarque - Les dangers provenant de l'analyse de trafic ne concernent naturellement pas exclusivement une couche OSI déterminée. Donc l'analyse du trafic est généralement en dehors de l'objectif du cadre d'authentification. Toutefois, on peut assurer partiellement une protection contre l'analyse de trafic par la production d'un trafic additionnel inintelligible (remplissage de trafic) en utilisant des données chiffrées ou aléatoires.

A.2 *Services de sécurité*

Afin d'assurer la protection contre les dangers perçus, il est nécessaire de prévoir divers services de sécurité. Les services de sécurité que fournit le cadre d'authentification sont assurés au moyen des mécanismes de sécurité décrits au § A.3 de la présente annexe.

a) *authentification des entités homologues:* Ce service donne la confirmation qu'un utilisateur dans un certain cas de communication est bien l'utilisateur demandé. Deux services différents d'authentification des entités homologues peuvent être demandés:

- *authentification d'entité unique* (soit authentification d'entité d'*expéditeur de données* soit authentification d'entité de *destinataire de données*);
- *authentification mutuelle* dans laquelle les deux utilisateurs en communication s'authentifient mutuellement.

Lorsqu'ils demandent un service d'authentification des entités homologues, les deux utilisateurs décident de concert si leurs identités seront protégées ou non.

Le service d'authentification des entités homologues est assuré par le cadre d'authentification. On peut l'utiliser pour la protection contre une fausse identité et un redéfilement en ce qui concerne l'identité des utilisateurs;

b) *commande d'accès:* On peut employer ce service contre l'utilisation non autorisée de ressources. Le service de commande d'accès est fourni par l'annuaire ou par une autre application et ne concerne donc pas le cadre d'authentification;

c) *confidentialité de données:* On peut employer ce service pour assurer la protection des données contre la divulgation non autorisée des données. Le service de confidentialité des données est assuré par le cadre d'authentification. On peut l'employer pour la protection contre l'interception de données;

d) *intégrité de données:* Ce service fournit la preuve de l'intégrité des données dans une communication. Le service d'intégrité de données est assuré par le cadre d'authentification. On peut l'employer pour déceler des manipulations et assurer une protection contre celles-ci;

e) *non-rejet:* Ce service fournit la preuve de l'intégrité et de l'origine de données - les deux dans une relation infalsifiable - qui peuvent être vérifiées par un tiers à tout moment.

A.3 *Mécanismes de sécurité*

Les mécanismes de sécurité indiqués dans ce paragraphe assurent les services de sécurité décrits au § A.2.

a) *échange d'authentifications:* Les mécanismes d'authentification fournis par le cadre d'authentification comportent deux niveaux:

- *authentification simple:* S'appuie sur la fourniture par l'expéditeur de son nom et de son mot de passe, qui sont vérifiés par le destinataire;
- *authentification poussée:* S'appuie sur l'utilisation des techniques cryptographiques pour protéger l'échange de validation d'information. Dans le cadre d'authentification, l'authentification poussée est basée sur un schéma asymétrique.

Le mécanisme d'échange d'authentifications est employé pour assurer le service d'authentification d'entité homologue;

b) *chiffrement:* Le cadre d'authentification couvre le chiffrement du transfert de données. On peut utiliser soit un schéma asymétrique soit un schéma symétrique. L'échange de clefs nécessaire pour chacun des cas est assuré soit au cours d'un échange d'authentifications antérieur, soit indépendamment à n'importe quel moment avant la communication prévue. Ce dernier cas est en dehors de l'objectif du cadre d'authentification. Le mécanisme de chiffrement assure le service de la confidentialité de données;

c) *intégrité de données:* Le mécanisme fait intervenir le chiffrement d'une chaîne comprimée des données pertinentes à transférer. En même temps que les données en clair, ce message est envoyé au destinataire. Le destinataire reproduit la compression et le chiffrement qui suit des données en clair et compare le résultat avec celles créées par l'expéditeur pour en vérifier l'intégrité.

Le mécanisme d'intégrité de données peut être fourni par le chiffrement des données en clair comprimées, soit par un schéma asymétrique, soit par un schéma symétrique (avec le schéma symétrique, la compression et le chiffrement de données peuvent être effectués simultanément). Le mécanisme n'est pas explicitement fourni par le cadre d'authentification. Cependant, il est totalement fourni en tant que partie du mécanisme de signature numérique (voir ci-dessous) comportant un schéma asymétrique.

Le mécanisme d'intégrité de données assure le service d'intégrité de données. Il assure aussi partiellement le service de non-rejet (ce service a besoin également du mécanisme de signature numérique pour que ses exigences soient entièrement satisfaites);

d) *signature numérique:* Ce mécanisme fait intervenir le chiffrement au moyen de la clef secrète de l'expéditeur, d'une chaîne comprimée des données pertinentes à transférer. La signature numérique de même que les données en clair sont envoyées au destinataire. De la même façon que dans le cas du mécanisme d'intégrité de données, ce message est traité par le destinataire pour en vérifier l'intégrité. Le mécanisme de signature numérique prouve également l'authenticité de l'expéditeur et les relations sans ambiguïté entre l'expéditeur et les données qui ont été transférées.

Le cadre d'authentification assure le mécanisme de signature numérique comportant un schéma asymétrique.

Le mécanisme de signature numérique assure le service d'intégrité de données et assure également le service de non-rejet.

A.4 *Dangers contre lesquels la protection est assurée par les services de sécurité*

Le tableau figurant à la fin de la présente annexe indique les dangers susceptibles de porter atteinte à la sécurité contre lesquels chaque service de sécurité peut assurer la protection. La présence d'un astérisque (*) indique qu'un service donné de sécurité assure la protection contre un danger donné.

A.5 *Négociation des services et des mécanismes de sécurité*

La fourniture des caractéristiques de sécurité au cours d'un cas de communication nécessite la négociation du contexte dans lequel les services de sécurité sont exigés. Ceci conduit à un accord sur le type de mécanismes de sécurité et les paramètres de sécurité qui sont nécessaires pour fournir ces services de sécurité. Les procédures requises pour ces négociations des mécanismes et des paramètres peuvent être appliquées en considérant soit qu'elles font corps avec la procédure normale d'établissement de communication, soit qu'elles constituent un traitement à part. Les détails précis de ces procédures de négociation ne sont pas spécifiés dans la présente annexe.

	SERVICES			
DANGERS	Authentification d'entité	Secret des données	Intégrité des données	Non-rejet
Interception d'identité	* (si nécessaire)			
Interception de données		*		
Fausse identité	*			
Redéfilement	* (identité)		* (données)	*
Manipulation			*	*
Rejet				*

ANNEXE B

(à la Recommandation X.509)

Introduction à la cryptographie de clef publique

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

Dans les systèmes cryptographiques classiques, la clef utilisée par l'expéditeur d'un message secret pour effectuer le chiffrement est la même que celle qui est utilisée par le destinataire légitime pour effectuer le déchiffrement.

Dans les systèmes cryptographiques à clef publique (PKCS), cependant, les clefs vont par paires, l'une des clefs étant utilisée pour le chiffrement et l'autre pour le déchiffrement. Chaque paire de clefs est associée à un utilisateur particulier X. L'une des clefs connue sous le nom de clef publique (Xp) est connue publiquement et peut être employée par n'importe quel utilisateur pour chiffrer les données. Seul X qui possède la clef secrète complémentaire (Xs) peut déchiffrer les données. (Ceci est représenté par la notation D = Xs[Xp[D]].) Il est impossible de découvrir par un calcul la clef secrète à partir de la connaissance de la clef publique. Tout utilisateur peut ainsi communiquer un élément d'information que seul X peut découvrir en la chiffrant au moyen de Xp. Par extension, deux utilisateurs peuvent communiquer en secret en utilisant l'un et l'autre la clef publique pour chiffrer les données comme indiqué dans la figure B-1/X.509.

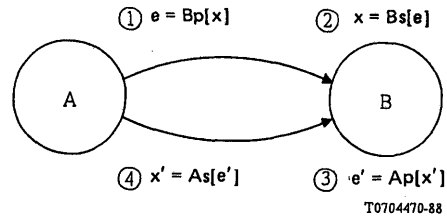


FIGURE B-1/X.509

Utilisation d'un PKCS pour l'échange d'information secrète

L'utilisateur A possède une clef publique Ap et une clef secrète As et l'utilisateur B possède un autre jeu de clefs Bp et Bs. A et B connaissent tous les deux les clefs publiques de l'un et de l'autre, mais ignorent la clef secrète de l'autre partie. A et B ont donc la possibilité d'échanger l'information secrète l'un avec l'autre en accomplissant les opérations ci-après (représentées dans la figure B-1/X.509):

1) A désire envoyer une certaine information secrète x à B. A chiffre donc x avec la clef de chiffrement de B et envoie l'information chiffrée e à B. Ceci est représenté par:

   e = Bp[x].

2) B peut maintenant déchiffrer cette information chiffrée e pour obtenir l'information x au moyen de la clef secrète de déchiffrement Bs. Il convient de noter que B est le seul possesseur de Bs et, étant donné que cette clef ne peut jamais être découverte ou envoyée, il est impossible à une autre partie d'obtenir l'information x. La possession de Bs détermine l'identité de B. L'opération de déchiffrement est représentée par:

   x = Bs[e] ou x = Bs[Bp[x]].

3) B peut maintenant envoyer de même une certaine information secrète x' à A avec la clef de chiffrement Ap de A:

   e' = Ap[x'].

4) A obtient x' par déchiffrement de e':

   x' = As[e'], ou x' = As[Ap[x']].

Par ce moyen, A et B ont échangé l'information secrète x et x'. Cette information ne peut être obtenue par personne d'autre que A et B du moment que leurs clefs secrètes ne sont pas révélées.

Alors qu'un tel échange transfère l'information secrète entre deux parties, il peut aussi servir à vérifier leurs identités. Plus précisément, A et B sont respectivement identifiés par la possession de leurs clefs secrètes de déchiffrement As et Bs. A peut déterminer si B est en possession de la clef secrète de déchiffrement Bs par l'obtention de la partie x de son message envoyé en retour dans le message x' de B. Cela indique à A que la communication est établie avec le possesseur de Bs. B peut de même vérifier l'identité de A.

Certains PKCS possèdent cette propriété que leurs opérations de déchiffrement et de chiffrement peuvent être inversées de façon à avoir D = Xp[Xs[D]]. Ainsi, un élément d'information qui pourrait avoir été expédié uniquement par X, soit lisible par tout autre utilisateur (qui soit en possession de Xp). Cela peut donc être utilisé pour certifier l'origine de l'information et sert de base aux signatures numériques. Seuls les PKCS qui possèdent cette propriété de permutabilité peuvent être utilisés dans le présent cadre d'authentification. Un algorithme de ce type est décrit dans l'annexe C.

Pour complément d'information, consulter:

DIFFIE, W. et HELLMAN, M. E. (novembre 1976) - News Directions in Cryptography, *IEEE Transactions on Information Theory*, IT-22 n° 6.

ANNEXE C

(à la Recommandation X.509)

Système cryptographique de clef publique (RSA)

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

Remarque - Le système de cryptographie spécifié dans la présente annexe inventé par R. L. Rivest, A. Shamir et L. Adleman est bien connu sous le sigle: "RSA".

C.1 *Objectif et domaine d'application*

Un exposé complet sur le système RSA dépasse l'objectif de ce document. Toutefois, on trouvera ci-après la description succincte de la méthode qui repose sur l'utilisation d'une élévation à une puissance d'un module.

C.2 *Références*

Pour un complément d'information, consulter:

1) Généralités

   RIVEST, R. L., SHAMIR, A. et ADLEMAN, L. (février 1978) - A Method for Obtaining Digital Signatures and Public-key Cryptosystems, *Communications of the ACM*, 21, 2, 120-126.

2) Production des clefs

   GORDON, J. - Strong RSA Keys, *Electronics Letters*, 20, 5, 514-516.

3) Déchiffrement

   QUISQUATER, J. J. et COUVREUR, C. (14 octobre 1982) - Fast Decipherment Algorithm for RSA Public-key Cryptosystems, *Electronics Letters*, 18, 21, 905-907.

C.3 *Définitions*

a) *clef publique:* Paire de paramètres qui se compose de l'exposant public et du module arithmétique;

   Remarque - L'élément de données **subjectPublicKey** ASN.1, défini comme **BIT STRING** (voir l'annexe G) doit être interprété, s'agissant de RSA, comme étant du type:

   SEQUENCE {INTEGER,INTEGER}

   où le premier nombre entier est le module arithmétique, le second étant l'exposant public. La séquence est représentée au moyen des règles de codage de base ASN.1.

b) *clef secrète:* Paire de paramètres qui se compose de l'exposant secret et du module arithmétique.

C.4 *Symboles et abréviations*

X,Y blocs de données qui sont arithmétiquement inférieurs au module

n module arithmétique

e exposant public

d exposant secret

p,q nombres premiers dont le produit forme le module arithmétique (n)

Remarque - Bien que les nombres premiers soient de préférence au nombre de deux, l'utilisation d'un module avec trois ou plusieurs facteurs premiers n'est pas à écarter.

mod n modulo arithmétique n.

C.5 *Description*

Cet algorithme asymétrique utilise la fonction puissance pour les transformations des blocs de données telles que:

$Y = X^e \bmod n$ avec $0 \leq X < n$

$X = Y^d \bmod n$ $0 \leq Y < n$

qui peuvent être satisfaites par exemple pour:

ed mod lcm(p-1,q-1)=1, ou

ed mod (p-1)(q-1)=1

Pour effectuer ce processus, un bloc de données doit être interprété comme un nombre entier. Pour cela, on considère que la totalité du bloc de données est une séquence de bits ordonnée (par exemple de longueur l). Le nombre entier est alors formé comme étant la somme des bits après avoir donné au premier bit le poids 2^{l-1} et divisé ce poids par 2 pour chaque bit suivant (le dernier bit aura le poids 1).

La longueur du bloc de données doit être le plus grand nombre d'octets contenant moins de bits que le module. Les blocs incomplets doivent être complétés de la façon que l'on désire. Un nombre de blocs de remplissage additionnel quelconque peut être ajouté.

C.6 *Exigences de sécurité*

C.6.1 *Longueurs de clef*

Il est reconnu que la longueur de clef acceptable est susceptible de changer avec le temps, en fonction du coût et de la disponibilité du matériel, du temps passé, des progrès techniques et du niveau de sécurité requis. Il est recommandé d'adopter initialement une valeur de 512 bits pour la longueur de n, mais sous réserve d'un *complément d'étude.*

C.6.2 *Génération de clefs*

La sécurité du système RCA repose sur la difficulté d'effectuer la factorisation de n. De nombreux algorithmes permettent d'effectuer cette opération et, afin de faire obstacle à l'emploi de toute technique actuellement connue, les valeurs p et q doivent être minutieusement choisies compte tenu des règles suivantes [par exemple voir la référence 2), § C.2]:

a) elles devront être choisies au hasard;

b) elles devront avoir être élevées;

c) elles devront être des nombres premiers;

d) |p-q| devra avoir une valeur élevée;

e) (p+1) devra posséder un facteur premier élevé;

f) (q+1) doit posséder un facteur premier élevé;

g) (p-1) doit posséder un facteur premier élevé, soit r;

h) (q-1) doit posséder un facteur premier élevé, soit s;

i) (r-1) doit posséder un facteur premier élevé;

j) (s-1) doit posséder un facteur premier élevé.

Après la génération des clefs publique et secrète, par exemple "Xp" et "Xs" définis aux § 3.3 et 4.1 de la présente Recommandation et qui sont constitués par d, e, et n, il serait préférable de détruire les valeurs p et q de même que toutes les autres données produites telles que le produit (p-1)(q-1) et les facteurs premiers de valeur élevée. Néanmoins la conservation de p et de q localement peut multiplier par 2 ou par 4 le débit du déchiffrement. La décision de conserver p et q est considérée comme un problème à résoudre à l'échelon local [référence 3)].

Il convient de faire en sorte que e soit > $\log_2(n)$ afin d'éviter une attaque préparée en prenant la racine e-ème de mod n pour découvrir le texte en clair.

C.7 *Exposant public*

L'exposant public (e) pourra être commun à tout l'environnement afin de minimiser la longueur de la partie de clef publique qui doit être effectivement diffusée et de réduire la capacité de transmission et la complexité de la transformation (voir la remarque 1).

L'exposant (e) devra être assez grand mais tel que l'élévation à une puissance puisse être effectuée efficacement en ce qui concerne la durée du traitement et la capacité de mémoire. Si l'on désire un exposant public fixe e, il y a grand avantage à utiliser le nombre de format F_4 (voir la remarque 2).

$F_4 = 2^{2^4} + 1$

= 65537 en numérotation décimale, et

= 1 0000 0000 0000 0001 en numérotation binaire.

Remarque 1 - Bien que le module n et l'exposant e soient tous les deux publics, le module ne doit pas être la partie commune à un groupe d'utilisateurs. La connaissance du module "n", de l'exposant public "e" et de l'exposant secret "d" suffit pour déterminer la factorisation de "n". Par conséquent, si le module est commun, chacun peut déduire ses facteurs et par là découvrir l'exposant secret de tous les autres.

Remarque 2 - L'exposant fixé devra avoir une valeur élevée et être un nombre premier, mais il devra également assurer un traitement efficace. Le nombre de format F_4 répond à ces exigences, par exemple l'authentification nécessite seulement 17 multiplications et est en moyenne 30 fois plus rapide que le déchiffrement.

C.8 *Conformité*

Si la présente annexe spécifie un algorithme pour les fonctions publiques et secrètes, il ne définit pas la méthode employée pour effectuer les calculs; il est donc possible qu'il y ait des produits différents qui satisfassent à cette annexe et qui soient mutuellement compatibles.

ANNEXE D

(à la Recommandation X.509)

Fonctions hachage

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

D.1 *Exigences pour les fonctions hachage*

Pour utiliser une fonction hachage comme fonction sûre à une voie, il ne doit pas être possible d'obtenir facilement le même résultat de hachage à partir de combinaisons différentes du message d'entrée.

Une fonction hachage à haut niveau de sécurité devra satisfaire aux exigences suivantes:

a) la fonction doit être à une voie, c'est-à-dire que quel que soit le résultat de hachage possible, il doit être impossible par calcul de construire un message d'entrée qui est réduit à ce résultat;

b) la fonction hachage doit être exempte de collision, c'est-à-dire qu'il doit être impossible par calcul de construire deux messages d'entrée distincts qui se réduisent au même résultat.

D.2 *Description d'une fonction hachage*

La fonction hachage suivante effectue la compression des données sur une base: bloc par bloc.

Ce hachage s'effectue en trois opérations principales:

1) la chaîne de données à hacher est divisée en blocs B d'égale longueur. Cette longueur est déterminée par les caractéristiques du système asymétrique de cryptographie utilisé pour la signature. Avec le système cryptographique RSA, cette longueur (en octets) est le plus grand nombre entier, l, tel qu'avec le module n, 16 l < $\log_2$ n;

2) pour des raisons d'impossibilité d'inversion, chaque octet du bloc est divisé en deux parties égales. Chacune de ces moitiés de bloc est précédée (remplie) de "uns" binaires. Grâce à cette répartition en zones, une inflexibilité ou redondance est introduite, qui augmente considérablement la qualité de l'impossibilité d'inversion que présente la fonction hachage. Chaque bloc généré au cours de l'opération 1) s'étend sur la longueur du module n;

3) chaque bloc résultant de l'opération 2) s'ajoute au bloc précédent modulo 2, élevé au carré et réduit modulo n jusqu'à ce que tous les m blocs soient traités.

Le résultat est la valeur H_m, où:

$H_0 = 0$

$H_i = (H_{i-1} \oplus B_i)^2 \bmod n$, pour $1 \leq i \leq m$

Si le dernier bloc est incomplet, il est rempli avec des "1".

ANNEXE E

(à la Recommandation X.509)

Dangers contre lesquels la protection est assurée par les services de sécurité

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

La méthode d'authentification poussée décrite dans la présente Recommandation assure une protection contre les dangers, comme cela est décrit dans la partie de l'annexe A relative à l'authentification poussée.

De plus, il existe une gamme de dangers potentiels qui tiennent à la méthode d'authentification poussée elle-même, à savoir:

Compromission de la clef secrète d'utilisateur - Un des principes de base de l'authentification poussée est que la clef secrète reste sûre. Plusieurs méthodes pratiques sont à la disposition de l'utilisateur pour qu'il conserve sa clef secrète de façon qu'elle assure une sécurité satisfaisante. Les conséquences de la compromission sont limitées à la déstabilisation de communication touchant l'utilisateur concerné.

Compromission de la clef secrète d'une CA - Un principe de base d'une authentification poussée est également que la clef secrète d'une CA reste sûre. Une sécurité matérielle et des méthodes de "nécessité d'accès" sont applicables. Les conséquences de la compromission sont limitées à la déstabilisation de communication touchant tout utilisateur certifié par cette CA.

Fait d'induire en erreur une CA par la délivrance d'un certificat non valide - Le fait que les CA soient indépendantes offre une certaine protection. La CA a la responsabilité de vérifier que les accréditations données comme sérieuses sont valides avant d'établir un certificat. Les conséquences de la compromission sont limitées à la déstabilisation de communication touchant l'utilisateur pour lequel le certificat a été établi et quiconque ayant été concerné par le certificat non valide.

Collusion entre une CA malhonnête et un utilisateur - Une telle attaque collusoire mettra la méthode en défaut. Cela constituera une trahison de la confiance témoignée à la CA. Les conséquences d'une CA malhonnête sont limitées à la déstabilisation de communication touchant n'importe quel utilisateur certifié par la CA.

Falsification d'un certificat - La méthode d'authentification poussée assure la protection contre la falsification d'un certificat, du fait que la CA doit le signer. La méthode dépend de la conservation du secret de la clef secrète de la CA.

Falsification d'un jeton - La méthode d'authentification poussée assure la protection contre la falsification d'un jeton, du fait que l'expéditeur doit le signer. La méthode dépend de la conservation du secret de la clé secrète de l'expéditeur.

Redéfilement d'un jeton - Les méthodes d'authentification à une ou deux voies assurent la protection contre le redéfilement d'un jeton par l'introduction de la date et de l'heure dans le jeton. La méthode à trois voies assure également la protection par vérification des nombres aléatoires.

Attaque du système cryptographique - Les possibilités d'une analyse cryptographique efficace du système dépendant des progrès réalisés dans la théorie du calcul des nombres et conduisant à la nécessité d'une clef de plus grande longueur peuvent valablement être annoncées à l'avance.

ANNEXE F

(à la Recommandation X.509)

Confidentialité des données

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

F.1 *Introduction*

Le traitement de la confidentialité des données peut être initialisé après que les clefs nécessaires au chiffrement ont été échangées. Celles-ci pourraient être fournies lors d'un échange d'authentifications précédent, comme décrit au § 9 ou par tout autre processus d'échange de clefs (ce dernier n'entre pas dans le cadre de la présente Recommandation).

La confidentialité des données peut être assurée en appliquant soit un schéma de chiffrement symétrique, soit un schéma de chiffrement asymétrique.

F.2 *Confidentialité des données par chiffrement asymétrique*

Dans ce cas, la confidentialité des données est assurée au moyen du chiffrement des données à transmettre par l'expéditeur qui utilise la clef publique du destinataire auquel les données sont destinées: le destinataire les déchiffrera alors en utilisant sa clef secrète.

F.3 *Confidentialité des données par chiffrement symétrique*

Dans ce cas la confidentialité des données s'obtient au moyen d'un algorithme de chiffrement symétrique. Ce choix est en dehors de l'objectif du cadre d'authentification.

Au cas où l'échange d'authentifications conformément au § 9 a été effectué par les deux parties concernées, une clef pour l'utilisation d'un algorithme symétrique peut être déterminée. Le choix des clefs secrètes dépend de la transformation à effectuer. Les parties doivent être sûres que ce sont des clefs à haut niveau de sécurité. La présente Recommandation ne spécifie pas la manière dont s'effectue ce choix bien qu'il soit manifestement nécessaire que cela ait l'accord des parties concernées ou que cela soit spécifié dans d'autres normes.

ANNEXE G

(à la Recommandation X.509)

Cadre d'authentification en ASN.1

La présente annexe fait partie intégrante de la présente Recommandation.

Cette annexe comprend la totalité des définitions ASN.1 des types de macros et des valeurs contenues dans la Recommandation sous la forme du module ASN.1 "**Cadre d'authentification**".

```
AuthenticationFramework {joint-iso-ccitt ds(5) modules(1)
                         authenticationFramework(7)}

DEFINITIONS ::=
BEGIN

EXPORTS AlgorithmIdentifier, AuthorityRevocationList, CACertificate, Certificate,
        Certificates, CertificationPath, CertificateRevocationList,
        UserCertificate, CrossCertificatePair, UserPassword, ALGORITHM,
        ENCRYPTED, PROTECTED, SIGNATURE, SIGNED;
```

```
IMPORTS
    informationFramework, selectedAttributeTypes, upperBounds
        FROM UsefulDefinitions  {joint-iso-ccitt ds(5)modules(1)
                                 usefulDefinitions(0)}
      Name, ATTRIBUTE,ATTRIBUTE-SYNTAX
        FROM InformationFramework informationFramework

ub-user-passwordFROM UpperBounds upperBounds;

-- types

Certificate      ::=  SIGNED SEQUENCE{
                          version                 [0] Version DEFAULT 1988,
                          serialNumber            SerialNumber,
                          signature               AlgorithmIdentifier,
                          issuer                  Name,
                          validity                Validity,
                          subject                 Name,
                          subjectPublicKeyInfo    SubjectPublicKeyInfo}

Version          ::=  INTEGER { 1988(0)}
SerialNumber     ::=  INTEGER

Validity         ::=  SEQUENCE{
                          notBefore               UTCTime
                          notAfter                UTCTime}

SubjectPublicKeyInfo ::=  SEQUENCE{
                            algorithm             AlgorithmIdentifier
                            subjectPublicKey      BIT STRING}

AlgorithmIdentifier  ::=  SEQUENCE{
                            algorithm             OBJECT IDENTIFIER,
                            parameters            ANY DEFINED BY algorithm OPTIONAL}

Certificates ::=        SEQUENCE{
                          certificate             Certificate,
                          certificationPath       ForwardCertificationPath OPTIONAL}

ForwardCertificationPath ::=  SEQUENCE OF CrossCertificates

CertificationPath  ::=        SEQUENCE{
                            userCertificate       Certificate,
                            theCACertificates     SEQUENCE OF CertificatePair
                                                        OPTIONAL}

CrossCertificates  ::= SET OF Certificate

CertificateList    ::= SIGNED SEQUENCE{
                          signature               AlgorithmIdentifier,
                          issuer                  Name,
                          lastUpdate              UTCTime,
                          revokedCertificates     SIGNEDSEQUENCE OF SEQUENCE{
                                signature  AlgorithmIdentifier,
                                issuer     Name,
                                userCertificate     SerialNumber,
                                revocationDate      UTCTime}
                                                        OPTIONAL}

CertificatePair    ::= SEQUENCE{
                          forward [0]             Certificate OPTIONAL,
                          reverse [1]             Certificate OPTIONAL
                          -- at least one of the pair must be present --}

-- attribute types

UserCertificate          ::=  ATTRIBUTE
                                  WITH ATTRIBUTE-SYNTAXCertificate

CACertificate            ::=  ATTRIBUTE
                                  WITH ATTRIBUTE-SYNTAXCertificate
```

```
CrossCertificatePair        ::=  ATTRIBUTE
                                    WITH ATTRIBUTE-SYNTAXCertificatePair

CertificateRevocationList   ::=  ATTRIBUTE
                                    WITH ATTRIBUTE-SYNTAXCertificateList

AuthorityRevocationList     ::=  ATTRIBUTE
                                    WITH ATTRIBUTE-SYNTAXCertificateList

UserPassword                ::=  ATTRIBUTE
                                    WITH ATTRIBUTE-SYNTAX
                                    OCTETSTRING(SIZE(0...ub-user-password))
                                    MATCHES FOR EQUALITY

-- macros

ALGORITHM MACRO     ::=
BEGIN
TYPE NOTATION       ::=  "PARAMETER" type
VALUE NOTATION      ::=  value(VALUE OBJECT IDENTIFIER)
END -- of ALGORITHM

ENCRYPTED MACRO     ::=
BEGIN

TYPE NOTATION       ::=  type (ToBeEnciphered)
VALUENOTATION       ::=  value (VALUE BIT STRING
     -- the value of the bit string is generated by
     -- taking the octets which form the complete
     -- encoding (using the ASN.1 Basic Encoding Rules)
     -- of the value of the ToBeEnciphered type and
     -- applying an encipherment procedure to those octets --

END

SIGNED MACRO        ::=
BEGIN
TYPE NOTATION       ::=  type (ToBeSigned)
VALUE NOTATION      ::=  value(VALUE

SEQUENCE{
     ToBeSigned,
     AlgorithmIdentifier, -- of the algorithm used to generate the signature
     ENCRYPTED OCTET STRING
     -- where the octet string is the result
     -- of the hashing of the value of
     -- "ToBeSigned" --}
                         )

END -- of SIGNED

SIGNATURE MACRO     ::=
BEGIN

TYPE NOTATION       ::=  type (OfSignature)

VALUE NOTATION      ::=  value(VALUE

     SEQUENCE{
          AlgorithmIdentifier,
          -- of the algorithm used to compute the signature
          ENCRYPTED OCTET STRING
          -- where the octet string is a function (e.g. a compressed or hashed version)
          -- of the value "OfSignature", which may include the identifier of the
          -- algorithm used to compute the signature --}
                                                  )

END -- of SIGNATURE

PROTECTED MACRO     ::= SIGNATURE

END -- of Authentication Framework Definitions
```

ANNEXE H

(à la Recommandation X.509)

Définition de référence des identificateurs d'objet d'algorithme

La présente annexe ne fait pas partie intégrante de la présente Recommandation.

Elle définit les identificateurs d'objet affectés aux algorithmes d'authentification et de chiffrement, en l'absence d'un enregistreur officiel. Il est prévu de recourir à un tel enregistreur quand il sera disponible. Les définitions prennent la forme du module ASN.1 **AlgorithmObjectIdentifiers.**

```
AlgorithmObjectIdentifiers      {joint-iso-ccitt ds(5) modules(1)
                                 algorithmObjectIdentifiers(8)}

DEFINITIONS ::=
BEGIN

EXPORTS
    encryptionAlgorithm, hashAlgorithm, signatureAlgorithm,
    rsa,squareMod-n,sqMod-nWithRSA;

IMPORTS
    algorithm,authenticationFramework
        FROM UsefulDefinitions  {joint-iso-ccitt ds(5)modules(1)
                                 usefulDefinitions(0)}

    ALGORITHM  FROM AuthenticationFramework authenticationFramework;


-- categories of object identifier

encryptionAlgorithm OBJECT IDENTIFIER ::=  {algorithm 1}

hashAlgorithm OBJECT IDENTIFIER       ::=  {algorithm 2}

signatureAlgorithm OBJECT IDENTIFIER  ::=  {algorithm 3}


-- algorithms

rsa  ALGORITHM
     PARAMETER KeySize
     ::=  {encryptionAlgorithm 1}

KeySize ::=      INTEGER

sqMod-n ALGORITHM
     PARAMETER BlockSize
     ::=  {hashAlgorithm 1}

BlockSize ::=    INTEGER

sqMod-nWithRSA ALGORITHM
     PARAMETER KeyAndBlockSize
     ::=  {signatureAlgorithm 1}

KeyAndBlockSize ::= INTEGER

END -- of Algorithm Object Identifier Definitions
```

L'ANNUAIRE : DEFINITION DU SERVICE ABSTRAIT [1)]

(Melbourne, 1988)

SOMMAIRE

---

1) La Recommandation X.511 et ISO 9594-3, Systèmes de traitement de l'information - Interconnexion des systèmes ouverts - Annuaire - Définition du service abstrait, ont été conçues en collaboration étroite et sont alignées du point de vue technique.

0 Introduction

0.1 Le présent document, avec les autres documents de la même série, a été établi pour faciliter l'interconnexion de systèmes de traitement de l'information afin de fournir des services d'annuaire. On peut considérer comme un tout, appelé *annuaire*, l'ensemble de ces systèmes et des données d'annuaire qu'ils contiennent. Les renseignements contenus dans l'annuaire, qui constituent collectivement la base de données d'annuaire (DIB), ont pour objet de faciliter la communication entre des objets, avec des objets ou concernant des objets tels qu'entités d'application, personnes, terminaux et listes de distribution.

0.2 L'annuaire joue un rôle significatif dans l'interconnexion des systèmes ouverts qui a pour objet de permettre, en n'exigeant que l'application d'un minimum d'accords techniques outre les normes d'interconnexion mêmes, l'interconnexion de systèmes de traitement de l'information:

- provenant de fabricants différents;
- placés sous des gestions différentes;
- de niveaux de complexité différents;
- d'âge différent.

0.3 La présente Recommandation spécifie les prestations que l'annuaire offre à ses utilisateurs.

0.4 L'annexe A décrit le module ASN.1 qui contient toutes les définitions relatives au service abstrait.

1 Objet et domaine d'application

1.1 La présente Recommandation décrit de façon abstraite le service extérieurement visible fourni par l'annuaire.

1.2 La présente Recommandation ne spécifie pas de mises en oeuvre ou de produits individuels.

SECTION 1 - *Considérations générales*

2 Références

Recommandation X.200 - Modèle de référence pour l'interconnexion des systèmes ouverts.

Recommandation X.208 - Spécification de la syntaxe abstraite - Notation Un (ASN.1).

Recommandation X.500 - Annuaire - Aperçu général des concepts, modèles et services.

Recommandation X.501 - Annuaire - Modèles.

Recommandation X.518 - Annuaire - Procédures de fonctionnement réparti.

Recommandation X.519 - Annuaire - Spécifications du protocole.

Recommandation X.520 - Annuaire - Types d'attribut.

Recommandation X.521 - Annuaire - Catégories d'objets.

Recommandation X.509 - L'annuaire - Cadre d'authentification.

Recommandation X.219 - Opérations à distance - Concepts - Modèle et définition des services.

Recommandation X.229 - Opérations à distance - Spécification du protocole.

Recommandation X.407 - Service abstrait, définition, conventions.

3 Définitions

3.1 *Définitions de base concernant l'annuaire*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.500:

a) *annuaire;*

b) *base de données d'annuaire (DIB);*

c) *utilisateur (d'annuaire).*

3.2 *Définitions du modèle d'annuaire*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.501:

a) *agent de système d'annuaire;*

b) *agent d'utilisateur d'annuaire.*

3.3 *Définitions relatives à la base de données d'annuaire*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.501:

a) *entrée pseudonyme;*

b) *arbre d'information de l'annuaire;*

c) *entrée (d'annuaire);*

d) *supérieur immédiat;*

e) *entrée/objet immédiatement supérieur;*

f) *objet;*

g) *catégorie d'objets;*

h) *entrée d'objet;*

i) *subordonné;*

j) *supérieur.*

3.4 *Définitions concernant les entrées d'annuaire*

La présente Recommandation utilise les termes suivants qui sont définis dans la Recommandation X.501:

a) *attribut;*

b) *type d'attribut;*

c) *valeur d'attribut;*

d) *assertion de valeur d'attribut.*

3.5 *Définitions relatives aux noms*

La présente Recommandation utilise les termes suivants qui sont définis dans la Recommandation X.501:

a) *pseudonyme, nom pseudonyme;*

b) *nom spécifique;*

c) *nom (d'annuaire);*

d) *nom visé;*

e) *nom spécifique relatif.*

3.6 *Définitions concernant les opérations réparties*

La présente Recommandation utilise les termes suivants qui sont définis dans la Recommandation X.518:

a) *chaînage;*

b) *renvoi.*

3.7 *Définitions concernant le service abstrait*

La présente Recommandation définit les termes suivants:

a) *filtre:* Assertion relative à la présente ou à la valeur de certains attributs d'une entrée afin de limiter la portée d'une recherche;

b) *commandes de services:* Paramètres transmis dans le cadre d'une opération abstraite qui limitent certains aspects de ses performances;

c) *expéditeur:* L'utilisateur qui est à l'origine d'une opération.

4 Abréviations

La présente Recommandation utilise les abréviations suivantes:

AVA Assertion de valeur d'attribut

DIB Base de données de l'annuaire

DIT Arbre d'information de l'annuaire

DMD Domaine de gestion d'annuaire

DSA Agent de système d'annuaire

DUA Agent d'utilisateur d'annuaire

RDN Nom spécifique relatif.

5 Conventions descriptives

La présente Recommandation utilise les conventions de définition du service abstrait définies dans la Recommandation X.407.

SECTION 2 - *Service abstrait*

6 Aperçu du service d'annuaire

6.1 Les services d'annuaire, décrits dans la Recommandation X.501, sont fournis au moyen de points d'accès aux DUA dont chacun agit au nom d'un utilisateur. La figure 1/X.511 illustre ces concepts.

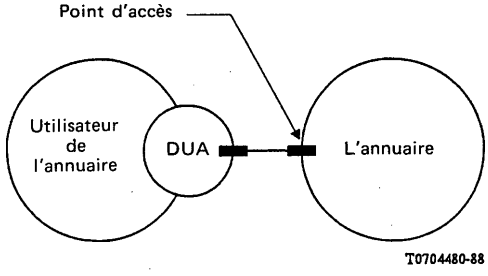


FIGURE 1/X.511

Accès à l'annuaire

6.2 En principe, les points d'accès à l'annuaire peuvent être de différents types et fournir des combinaisons de services différentes. On aura intérêt à considérer l'annuaire comme un *objet* sur lequel se trouvent plusieurs types de *ports*. Chacun des ports définit un type donné d'interaction dans le cadre de laquelle l'annuaire peut participer avec un DUA. Chacun des points d'accès correspond à une combinaison particulière de types de ports.

6.3 En utilisant la notation définie dans la Recommandation X.407, on peut définir l'annuaire de la façon suivante:

```
directory
    OBJECT
        PORTS {   readPort [S],
                  searchPort [S],
                  modifyPort [S]}
::= id-ot-directory
```

L'annuaire procède aux opérations au moyen de: Read Ports, qui permettent la lecture d'une information tirée d'une inscription de nom déterminé figurant dans la DIB; Search Ports, qui permettent une plus grande "exploration" de la DIB et Modify Ports qui permettent de modifier des inscriptions figurant dans la DIB.

Remarque - On prévoit pour l'avenir d'autres types de ports d'annuaire.

6.4 De même, un DUA peut (du point du vue de l'annuaire) être défini comme suit:

```
dua
    OBJECT
        PORTS {   readPort [C],
                  searchPort [C],
                  modifyPort [C]}
::= id-ot-dua
```

Le DUA est le consommateur des services fournis par l'annuaire.

6.5 Les ports cités aux § 6.2 à 6.4 peuvent se définir comme suit:

```
readPort
   PORT
     CONSUMER INVOKES {
          Read, Compare, Abandon}
::= id-pt-search

searchPort
   PORT
     CONSUMER INVOKES {
          List, Search}
::= id-pt-search

modifyPort
   PORT
   CONSUMER INVOKES {
          AddEntry, RemoveEntry,
          ModifyEntry, ModifyRDN}
   ::=    id-pt-modify
```

6.6 Les opérations passant par **readPort**, **searchPort** et le **modifyPort** sont respectivement définies aux § 9, 10 et 11.

6.7 Ces ports sont utilisés uniquement comme méthode de structuration de la description du service d'annuaire. La conformité avec les opérations d'annuaire est spécifiée dans la Recommandation X.519.

7 Types d'information

7.1 *Introduction*

7.1.1 Le présent paragraphe identifie et, dans quelques cas, définit un certain nombre de types d'information qui sont par la suite utilisés pour définir diverses opérations d'annuaire. Ces types d'information sont ceux qui sont communs à plusieurs opérations ou qui le seront probablement dans l'avenir, ou qui sont assez complexes ou assez autonomes pour justifier leur définition indépendamment de l'opération qui les utilise.

7.1.2 Plusieurs des types d'information utilisés pour définir les services d'annuaire se trouvent dans d'autres Recommandations. Ces types d'information sont identifiés au § 7.2 qui indique aussi la source de leur définition. Chacune des sous-sections restantes (§ 7.3 à 7.10) identifie et définit un type d'information.

7.2 *Types d'information définis dans d'autres Recommandations*

7.2.1 Les types d'information suivants sont définis dans la Recommandation X.501:

a) **attribut**;

b) **type d'attribut**;

c) **valeur d'attribut**;

d) **assertion de la valeur d'attribut**;

e) **nom spécifique**;

f) **nom**;

g) **nom spécifique relatif**.

7.2.2 Le type d'information suivant est défini dans la Recommandation X.520:

a) **adresse de présentation**.

7.2.3 Les types d'information suivants sont définis dans la Recommandation X.509:

a) **certificat**;

b) **SIGNED**;

c) **itinéraire de certification**.

7.2.4 Le type d'information suivant est défini dans la Recommandation X.219:

a) **InvokeID**.

7.2.5 Les types d'information sont définis dans la Recommandation X.518:

a) **OperationProgress**;

b) **ContinuationReference**.

7.3 *Arguments communs*

7.3.1 L'**information d'arguments** communs peut être présente pour accompagner l'appel de chaque opération que peut accomplir l'annuaire.

```
CommonArguments ::= SET {
        [30] ServiceControls DEFAULT { },
        [29] SecurityParameters DEFAULT { },
        requestor [28] DistinguishedName
                                OPTIONAL,
        [27] OperationProgress DEFAULT notStarted,
        aliasedRDNs [26] INTEGER OPTIONAL,
        extensions [25] SET OF EXTENSION OPTIONAL}

Extension ::= SET {
        identifier      [0]  INTEGER,
        critical        [1]  BOOLEAN DEFAULT FALSE,
        item            [2]  ANY DEFINED BY identifier}
```

7.3.2 Les différents composants ont le sens donné aux § 7.3.2.1 à 7.3.2.4.

7.3.2.1 Le composant **ServiceControls** est spécifié au § 7.5. Son absence est interprétée comme équivalant à un jeu de commandes vide.

7.3.2.2 Le composant **SecurityParameters** est spécifié au § 7.9. Son absence est interprétée comme équivalant à un jeu de paramètres de sécurité vide.

7.3.2.3 Le **requestor DistinguishedName** identifie le demandeur d'une opération abstraite donnée. Il détient le nom de l'utilisateur identifié au moment de la liaison avec l'annuaire. Il peut être nécessaire lorsque la demande est à signer (voir le § 7.10) et il doit contenir le nom de l'utilisateur qui est à l'origine de la demande.

7.3.2.4 L'**OperationProgress** définit le rôle que le DSA doit jouer dans l'évaluation répartie de la demande. La Recommandation X.518 en donne une définition plus précise.

7.3.2.5 Le composant **aliasedRDNs** indique au DSA que le composant objet de l'opération a été créé par une variation de référence d'un pseudonyme lors d'une précédente tentative d'opération. La valeur integer indique le nombre de RDN dans l'objet provenant du changement de référence du pseudonyme. (La valeur aura été fixée dans la réponse de référence de la précédente opération.)

7.3.2.6 Le composant **extensions** fournit un mécanisme pour exprimer des extensions normalisées à la forme de l'argument d'une opération abstraite d'annuaire.

Remarque - La forme du résultat de cette opération abstraite étendue est identique à celle de la version non étendue. (Néanmoins, le résultat d'une opération abstraite étendue donnée peut différer de son homologue non étendu.)

Les sous-composants sont ceux qui sont définis aux § 7.3.2.6.1 à 7.3.2.6.3.

7.3.2.6.1 **L'identificateur** sert à identifier une extension donnée. La valeur de ce composant lui sera assignée seulement par les versions futures de cette série de Recommandations.

7.3.2.6.2 Le sous-composant **critique** permet au demandeur de l'opération abstraite étendue d'indiquer que seule la performance de la forme étendue de l'opération abstraite est acceptable (c'est-à-dire que la forme non étendue n'est pas acceptable). En pareil cas, l'extension est une *extension critique*. Si l'annuaire, ou une partie de l'annuaire, est incapable d'effectuer une extension critique, il envoie une indication de **extension critique non disponible** (comme une **erreur de service** ou un **qualificateur de résultat partiel**). Si l'annuaire est incapable d'effectuer une extension qui ne soit pas critique, il ne tient pas compte de la présence de l'extension.

7.3.2.6.3 Le sous-composant **item** fournit l'information nécessaire à l'annuaire pour effectuer selon la forme étendue l'opération abstraite.

7.4 *Résultats communs*

7.4.1 L'information **Common Results** peut être présente pour qualifier le résultat de chaque opération de recherche que peut accomplir l'annuaire.

```
CommonResults ::=   SET {
    [30] SecurityParameters   OPTIONAL,
    Performer [29] DistinguishedName
                              OPTIONAL,
    aliasDereferenced    [28] BOOLEAN
                         DEFAULT FALSE}
```

7.4.2 Les différents composants ont le sens qui leur est donné aux § 7.4.2.1 à 7.4.2.3.

7.4.2.1 Le composant **SecurityParameters** est spécifié au § 7.9. Son absence est interprétée comme équivalant à un jeu de paramètres de sécurité vide.

7.4.2.2 Le **Performer DistinguishedName** identifie l'exécutant d'une opération donnée. Il peut être nécessaire lorsque le résultat est à signer (voir le § 7.10) et doit contenir le nom du DSA qui a signé le résultat.

7.4.2.3 Le composant **aliasDereferenced** est mis sur **TRUE** quand le nom prévu d'un objet ou d'un objet de base qui est la cible de l'opération comprend un pseudonyme déréférencé.

7.5 *Commandes de service*

7.5.1 Un paramètre **ServiceControls** contient les commandes, le cas échéant, qui doivent diriger ou limiter la fourniture du service.

```
ServiceControls      ::=   SET {
    options [0] BIT STRING {
        preferChaining(0)
        chainingProhibited (1),
        localScope (2),
        dontUseCopy (3),
        dontDereferenceAliases(4)}
        DEFAULT {},

priority [1] INTEGER {
    low (0),
    medium (1),
    high (2) } DEFAULT medium,
```

timeLimit [2] INTEGER OPTIONAL,
sizeLimit [3] INTEGER OPTIONAL,

scopeOfReferral [4] INTEGER {
dmd(0),
country(1)}
OPTIONAL }

7.5.2 Les différents composants ont le sens qui leur est donné aux § 7.5.2.1 à 7.5.2.5.

7.5.2.1 Le composant **options** contient des indications qui, si elles sont fixées, confirment la condition suggérée. Ainsi:

a) **preferChaining** indique que, pour fournir le service, le chaînage est utilisé de préférence aux références. L'annuaire n'est pas obligé de suivre cette préférence;

b) **chainingProhibited** indique que le chaînage et d'autres méthodes de répartition de la demande dans l'annuaire sont interdits;

c) **localScope** indique que l'opération doit être limitée à une portée locale. La définition de cette option est elle-même une question locale. Par exemple, dans un DSA ou DMD unique;

d) **dontUseCopy** indique que l'information copiée (définie dans la Recommandation X.518) ne doit pas être utilisée pour assurer le service;

e) **dontDereferenceAliases** indique que tout pseudonyme servant à identifier l'entrée affectée par une opération ne doit être déréférencé.

Remarque - Cela est nécessaire pour permettre une référence à une entrée de pseudonyme proprement dit plutôt qu'à l'entrée pseudonyme, par exemple, pour lire l'entrée de pseudonyme.

Si ce composant est omis, on admet ce qui suit: pas de préférence pour le chaînage, mais le chaînage n'est pas interdit, il n'y a pas de limite à la portée de l'opération, l'utilisation d'une copie est autorisée et les pseudonymes seront déréférencés (sauf s'il s'agit d'opérations modification, pour lesquelles les pseudonymes ne sont jamais déréférencés).

7.5.2.2 La **priorité** (faible, moyenne, haute) indique le rang selon lequel le service doit être fourni. On notera qu'il ne s'agit pas d'un service garanti, en ce sens que l'annuaire dans son ensemble ne mettra pas en oeuvre de files d'attente. Il n'y a pas de relation implicite avec l'emploi de "priorités" dans les couches sous-jacentes.

7.5.2.3 **TimeLimit** indique, en secondes, le laps de temps maximal qui peut s'écouler avant que le service ne doive être fourni. Si cette contrainte ne peut être respectée, une erreur est signalée. Si ce composant est omis, c'est qu'il n'y a pas de limite de temps. En cas de délai dépassé pour **List** ou **Search**, le résultat sera un choix arbitraire des résultats accumulés.

Remarque - Ce composant ne fixe pas le temps passé à traiter la demande pendant le laps de temps susmentionné: un nombre quelconque de DSA peuvent être engagés dans le traitement de la demande au cours de ce laps de temps.

7.5.2.4 **SizeLimit** s'applique seulement aux opérations **List** et **Search** et indique le nombre maximal d'objets à retourner. Si la taille de la liste est excessive, les résultats de **List** et de **Search** peuvent être un choix arbitraire des résultats accumulés, de nombre égal à la limite de taille. Les autres résultats éventuels sont mis au rebut.

7.5.2.5 **ScopeOfReferral** indique la portée d'une référence envoyée par un DSA. Selon qu'une valeur **dmd** ou **country** est choisie, seules les références à d'autres DSA dans la portée choisie seront envoyées.

Cela s'applique aux références dans **ReferralError** et le paramètre **unexplored** des résultats **List** et **Search**.

7.5.3 Certaines combinaisons de **priority**, **timeLimit** et **sizeLimit** peuvent être conflictuelles. Par exemple, une courte limite de temps peut être incompatible avec une faible priorité, une large limite de taille peut être incompatible avec une étroite limite de temps, etc.

7.6 *Sélection d'information d'entrée*

7.6.1 Un paramètre **EntryInformationSelection** indique l'information qui est demandée d'une entrée dans un service de recherche.

```
EntryInformationSelection ::=    SET {
    attributeTypes
        CHOICE  {
                allAttributes [0]  NULL,
                select [1]] SET OF AttributeType
                -- empty set implies no attributes
                -- are requested --}
        DEFAULT allAttributes NULL,

    InfoTypes [2]   INTEGER {
        attributeTypesOnly (0),
        attributeTypesAndValues (1) }
        DEFAULT attributeTypesAndValues }
```

7.6.2 Les différents composants ont le sens qui leur est donné aux § 7.6.2.1 et 7.6.2.2.

7.6.2.1 Le composant **attributeTypes** spécifie que le jeu d'attributs au sujet desquels une information est demandée:

a) si l'on choisit l'option **select**, les attributs correspondants sont indiqués dans une liste. Si la liste est vide, aucun attribut n'est renvoyé. L'information concernant un attribut choisi doit être envoyée si l'attribut est présent. Une **erreur d'attribut** avec le problème **noSuchAttribute** ne doit pas être renvoyée à moins qu'aucun des attributs choisis ne soit présent;

b) si l'on choisit l'option **allAttributes**, une information est demandée au sujet de tous les attributs de l'entrée.

L'information d'attribut n'est renvoyée que si les droits d'accès sont suffisants. Un **SecurityError** (avec un problème de **droits d'accès insuffisants**) sera envoyé seulement au cas où les droits d'accès interdisent la lecture de toutes les valeurs d'attribut demandées.

7.6.2.2 Le composant **infoTypes** spécifie si les informations relatives au type d'attribut et à la valeur d'attribut (le défaut) sont toutes deux demandées ou si la seule information demandée est le type d'attribut. Si le composant **attributeTypes** (voir le § 7.6.2.1) est tel qu'aucun attribut n'est demandé, ce composant n'est pas significatif.

7.7 *Information d'entrée*

7.7.1 Un paramètre **EntryInformation** transmet une sélection d'information tirée d'une entrée.

```
EntryInformation    ::=   SEQUENCE  {
    DistinguishedName,
    fromEntry BOOLEAN DEFAULT TRUE,
    SET OF CHOICE {
        AttributeType,
        Attribute} OPTIONAL }
```

7.7.2 Le **DistinguishedName** de l'entrée est toujours inclus.

7.7.3 Le paramètre **fromEntry** indique si l'information a été obtenue de l'entrée (**VRAI**) ou une copie de l'entrée (**FAUX**).

7.7.4 Un jeu d'**AttributeTypes** ou d'**Attributes** est inclus, s'il y a lieu; chacun d'entre eux peut être seul ou accompagné d'une ou plusieurs valeurs d'attribut.

7.8 *Filtre*

7.8.1 Un paramètre **Filtre** fait passer un test à une entrée, qui y satisfait ou non. Le filtre est exprimé par des assertions concernant la présence ou la valeur de certains attributs de l'entrée, et ce paramètre n'est satisfait que si, et seulement si, son évaluation est **TRUE**.

Remarque - Un filtre peut être **TRUE**, **FALSE** ou non défini.

```
Filter          ::=  CHOICE  {
    item             [0]  FilterItem,
    and              [1]  SET OF Filter,
    or               [2]  SET OF Filter,
    not              [3]  Filter }
```

```
FilterItem      ::=  CHOICE  {
    equality          [0]  AttributeValueAssertion,
    substrings        [1]  SEQUENCE {
        type                AttributeType,
        strings             SEQUENCE OF CHOICE {
                            Initial     [0]  AttributeValue,
                            any         [1]  AttributeValue,
                            final       [2]  AttributeValue}},
    greaterOrEqual        [2]  AttributeValueAssertion,
    lessOrEqual           [3]  AttributeValueAssertion,
    present               [4]  AttributeType,
    approximateMatch      [5]  AttributeValueAssertion }
```

7.8.2 Un **filtre** est soit un **FilterItem** (voir le § 7.8.3), soit une expression désignant des filtres plus simples combinés en utilisant les opérateurs logiques **et**, **ou** et **non**. Quand le **filtre** est non défini, s'il s'agit d'un **FilterItem** non défini, ou d'un ou de plusieurs **filtres** plus simples, qui sont tous non définis. Sinon, quand le **filtre** est:

a) un **item**, il est **VRAI** si, et seulement si, l'**articleFiltre** correspondant est **VRAI**;

b) un **et**, il est **VRAI** à condition qu'aucun des **filtres** emboîtés ne soit **FAUX**;

Remarque - En conséquence, s'il n'y a pas de **filtres** emboîtés, le **et** est évalué comme **VRAI**.

c) un **ou**, il est **FAUX**, à moins que l'un quelconque des **filtres** emboîtés ne soit **VRAI**;

Remarque - En conséquence, s'il n'y a pas de **filtres** emboîtés, le **ou** est évalué comme **FAUX**.

d) un **non**, il est **VRAI** si, et seulement si, le **filtre** emboîté est **FAUX**.

7.8.3 Un **FilterItem** est une assertion concernant la présence ou la ou les valeurs d'un attribut d'un type donné dans l'entrée soumise au test. Chacune de ces assertions correspond à **VRAI**, **FAUX**, ou **non défini**.

7.8.3.1 Chaque **FilterItem** comprend un **AttributeType** qui identifie l'attribut particulier concerné.

7.8.3.2 Toute assertion concernant la valeur d'un tel attribut n'est définie que si l'**AttributeType** est connu et que si la ou les **AttributeValue**(s) visées sont conformes à la syntaxe d'attributs définie pour ce type d'attribut.

Remarque 1 - Quand ces conditions ne sont pas satisfaites, le **FilterItem** est non défini.

Remarque 2 - Des restrictions à la commande d'accès peuvent exiger que le **FilterItem** soit considéré comme non défini.

7.8.3.3 Les assertions concernant la valeur d'un attribut sont évaluées sur la base des règles d'assortiment associées à la syntaxe d'attribut définie pour le type d'attribut. Une règle d'assortiment non définie pour une syntaxe d'attribut donnée ne peut être utilisée pour faire des assertions concernant cet attribut.

Remarque - Quand ces conditions ne sont pas satisfaites, le **FilterItem** est non défini.

7.8.3.4 Un **FilterItem** peut être non défini (comme indiqué aux § 7.8.3.2 et 7.8.3.3 ci-dessus). Sinon, quand l'assertion de **FilterItem** est:

a) **equality**, l'assertion est **VRAI** si, et seulement s'il existe une valeur de l'attribut égale à la valeur affirmée;

b) **substrings**, l'assertion est **VRAI** si, et seulement s'il existe une valeur de l'attribut dans laquelle la sous-chaîne spécifiée apparaît dans l'ordre donné. Les sous-chaînes ne doivent pas se chevaucher et peuvent (mais ne doivent pas nécessairement) être séparées des extrémités de la valeur d'attribut et les unes des autres par un zéro ou par d'autres éléments de chaîne.

Si **Initial** existe, la sous-chaîne correspondra à initiale de la valeur d'attribut; s'il est **final**, s'il existe, la sous-chaîne concordera avec la dernière sous-chaîne de la valeur d'attribut; **Any**, s'il existe, la sous-chaîne peut concorder avec toute sous-chaîne de la valeur d'attribut;

c) **greaterOrEqual**, l'assertion est **VRAI** si, et seulement si, l'ordre relatif (défini par l'algorithme d'ordre approprié) place la valeur fournie avant ou égale à toute valeur de l'attribut;

d) **lessOrEqual**, l'assertion est **VRAI** si, et seulement si, l'ordre relatif (défini par l'algorithme d'ordre approprié) place la valeur fournie après ou égale à toute valeur de l'attribut;

e) **present**, l'assertion est **VRAI** si, et seulement si, un tel attribut est présent dans l'inscription;

f) **approximateMatch**, l'assertion est **VRAI** si, et seulement s'il existe une valeur de l'attribut qui s'accorde avec celle qu'affirme un algorithme de concordance approximatif défini localement (par exemple variations d'orthographe, concordance phonétique). Il n'existe pas de directives spécifiques pour la concordance approximative dans la présente version de la Recommandation. Si la concordance approximative n'est pas assurée, ce **FilterItem** doit être traité comme une concordance pour l'**égalité**.

7.9 *Paramètres de sécurité*

7.9.1 Les **SecurityParameters** régissent le fonctionnement de différents dispositifs de sécurité associés à une opération d'annuaire.

Remarque - Ces paramètres sont acheminés de l'expéditeur au destinataire. Quand ils apparaissent dans l'argument d'une opération abstraite, le demandeur est l'expéditeur et l'exécutant est le destinataire. Dans un résultat, les rôles sont inversés.

```
SecurityParameters ::=                SET {
    certification-path                [0]
    CertificationPath       OPTIONAL,
    name                [1] DistinguishedName
                                OPTIONAL,
    time                [2] UTCTime OPTIONAL,
    random              [3] BIT STRING OPTIONAL,
    target              [4] ProtectionRequest OPTIONAL
                                }

ProtectionRequest   ::= INTEGER {
                            none(0),
                            signed (1)}
```

7.9.2 Les différents composants ont le sens qui leur est donné aux § 7.9.2.1 à 7.9.2.5.

7.9.2.1 Le composant **CertificationPath** se compose du certificat de l'expéditeur et, en option, d'une séquence de paires de certificats. Le certificat sert à associer le nom spécifique et la clé publique de l'expéditeur et peut servir à vérifier la signature sur l'argument ou résultat. Ce paramètre sera présent si l'argument ou résultat est signé. La séquence de paires de certificats comprend des contre-certificats d'autorité de certification. Il permet de valider le certificat de l'expéditeur. Il n'est pas exigé si le destinataire dépend de la même autorité de certification que l'expéditeur. Si le destinataire exige un jeu valide de paires de certificats et que ce paramètre n'est pas présent, la question de savoir si le destinataire refuse la signature sur l'argument ou résultat ou s'il tente d'établir l'itinéraire de certification est une question locale.

7.9.2.2 Le **name** est le nom spécifique du premier destinataire prévu de l'argument ou du résultat. Par exemple, si un DUA produit un argument signé, le name est le nom spécifique du DSA auquel l'opération est soumise.

7.9.2.3 Le **time** est la date d'expiration prévue pour la validité de la signature, quand des arguments signés sont utilisés. Il est utilisé conjointement avec le nombre aléatoire pour permettre la détection des attaques de répétition.

7.9.2.4 Le **random** number est un nombre qui doit différer pour chaque jeton encore valide. Il est utilisé conjointement avec le paramètre time pour permettre la détection d'attaques de répétition quand l'argument ou résultat a été signé.

7.9.2.5 La **cible ProtectionRequest** ne peut apparaître que dans la demande d'exécution d'une opération; elle indique la préférence du demandeur en ce qui concerne le degré de protection à assurer au résultat. Deux niveaux sont prévus: **aucune** (pas de demande de protection) ou **signée** (l'annuaire est prié de signer le résultat, le défaut). Le degré de protection effectivement appliqué au résultat est indiqué par la forme du résultat et peut être égal ou inférieur à celui qui est demandé, en fonction des restrictions de l'annuaire.

7.10 *OPTIONALLY-SIGNED*

7.10.1 Un type d'information **OPTIONALLY-SIGNED** est celui dont les valeurs, sur l'option du générateur, peuvent être accompagnées de leur signature numérique. Cette prestation est spécifiée au moyen de la macro suivante:

```
OPTIONALLY-SIGNED MACRO    ::=
BEGIN

TYPE NOTATION          ::=  type (Type)
VALUE NOTATION         ::=  value (VALUE
          CHOICE  {    Type, SIGNED Type})

END
```

7.10.2 La macro **SIGNED**, qui décrit la forme de la forme signée de l'information, est spécifiée dans la Recommandation X.509.

8 Opérations liaison et séparation

Les opérations **DirectoryBind** et **DirectoryUnbind**, respectivement définies aux § 8.1 et 8.2, sont utilisées par le DUA au début et à la fin d'une période donnée d'accès à l'annuaire.

8.1 *Liaison avec l'annuaire*

8.1.1 Une opération **DirectoryBind** est utilisée au début d'une période d'accès à l'annuaire.

```
DirectoryBind          ::=  ABSTRACT-BIND
     TO  { readPort, searchPort, modifyPort }
     BIND
     ARGUMENT   DirectoryBindArgument
     RESULT     DirectoryBindResult
     BIND-ERROR DirectoryBindError

DirectoryBindArgument  ::=  SET {
     credentials    [0]  Credentials     OPTIONAL,
     versions       [1]  Versions DEFAULT
                                  v1988}

Credentials      ::=  CHOICE {
     simple         [0]  SimpleCredentials,
     strong         [1]  StrongCredentials,
     externalProcedure   [2]  EXTERNAL }

SimpleCredentials      ::=  SEQUENCE {
     name           [0]  DistinguishedName,
     validity       [1]  SET {
          time1     [0]  UTCTime OPTIONAL,
          Time2     [1]  UTCTime OPTIONAL,
          random1   [2]  BIT STRING OPTIONAL,
          random2   [3]  BIT STRING OPTIONAL } OPTIONAL,
          -- in most instances the argument for
          -- time and random are relevant in
          -- dialogues employing protected password
          -- mechanisms and derive their meaning
          -- as per bilateral agreements

password  [2]          OCTET STRING OPTIONAL }
     -- the value could be an unprotected
     -- password or Protected1 or Protected2
     -- as specified in Recommendation X.509.

StrongCredentials      ::=  SET {
     certification-path[0]      CertificationPath
                                  OPTIONAL,
     bind-token     [1]  Token }
```

```
Token              ::=   SIGNED SEQUENCE {
     algorithm           [0]   AlgorithmIdentifier,
     name                [1]   DistinguishedName,
     time                [2]   UTCTime,
     random              [3]   BIT STRING }

Versions           ::=   BIT STRING {v1988(0)}

DirectoryBindResult ::=  DirectoryBindArgument

DirectoryBindError         ::=        SET {
     versions            [0]   Versions DEFAULT v1988,
     CHOICE {
          serviceError         [1]   ServiceProblem
          securityError        [2]   SecurityProblem
}}
```

8.1.2 Les différents arguments ont le sens qui leur est donné aux § 8.1.2.1 et 8.1.2.2.

8.1.2.1 Les **Credentials** de l'**argument DirectoryBind** permettent à l'annuaire d'établir l'identité de l'utilisateur. Il peut s'agir d'une procédure **simple**, **complexe** (comme celle décrite dans la Recommandation X.509) ou externe.

8.1.2.1.1 Les **SimpleCredentials** se composent d'un nom (il s'agit toujours du nom spécifique d'un objet) et (en option) d'un mot de passe, ce qui assure un niveau limité de sécurité. Si le mot de passe est protégé comme décrit au § 5 de la Recommandation X.509, **SimpleCredentials** comprend le nom, le mot de passe et (en option) l'indication horaire et (ou) des nombres aléatoires qui servent à détecter une répétition. Dans certains cas, un mot de passe protégé peut être vérifié par un objet qui connaît le mot de passe seulement après régénération locale de la protection à sa propre copie du mot de passe et calcul du résultat avec la valeur de l'argument de liaison (mot de passe). Dans d'autres cas, une comparaison directe est possible.

8.1.2.1.2 **StrongCredentials** se composent d'un jeton de liaison et, en option, d'un certificat et d'une séquence de contre-certificats de l'autorité de certification (définie dans la Recommandation X.509). Cela permet à l'annuaire d'authentifier l'identité du demandeur qui établit l'association et vice versa. Les arguments du jeton de liaison sont utilisés comme suit: **algorithm** est l'identificateur de l'algorithme utilisé pour signer l'information; **name** est le nom du destinataire prévu. Le paramètre **time** contient l'indication horaire d'expiration du jeton. Le nombre **aléatoire** est un nombre qui doit différer pour chaque jeton toujours valide; il peut être utilisé par le destinataire pour détecter les attaques de répétition.

8.1.2.1.3 Si l'**externalProcedure** est utilisée, la sémantique du schéma d'authentification utilisé n'entre pas dans le cadre du document d'annuaire.

8.1.2.2 L'argument **Versions** de l'**argument DirectoryBind** identifie les versions du service auxquelles le DUA est prêt à participer. Pour cette version du protocole, la valeur sera mise sur **v1988**(0).

8.1.2.3 L'évolution vers de futures versions de l'annuaire devra être facilitée ainsi:

a) tous éléments de l'**argument DirectoryBind** autres que ceux définis dans la présente Recommandation seront acceptés et ne seront pas pris en considération;

b) les options supplémentaires pour les bits nommés de l'**argument DirectoryBind** (par exemple **Versions**) non définies seront acceptées et ne sont pas prises en considération.

8.1.3 Si la demande de liaison réussit, un résultat sera retourné. Les paramètres du résultat ont le sens qui leur est donné aux § 8.1.3.1 et 8.1.3.2.

8.1.3.1 Les **Credentials** du **résultat DirectoryBind** permettent à l'utilisateur d'établir l'identité du DSA. Ils permettent d'envoyer au DUA l'information qui identifie le DSA (qui assure directement le service d'annuaire). Leur forme sera la même (c'est-à-dire **CHOICE**) que celle fournie par l'utilisateur.

8.1.3.2 Le paramètre **Versions** du **résultat DirectoryBind** indique la version du service demandé par le DUA qui va être effectivement fournie par le présent DSA.

8.1.4 Si la demande de liaison échoue, une erreur de liaison est envoyée comme défini aux § 8.1.4.1 et 8.1.4.2.

8.1.4.1 Le paramètre **Versions** de l'**erreur DirectoryBind** indique quelles versions sont acceptées par ce DSA.

8.1.4.2 Un **securityError** ou **serviceError** doit être fourni comme suit:

- **securityError** — **inappropriateAuthentication**, **invalidCredentials**
- **serviceError** — **unavailable.**

8.2 *Fin de liaison avec l'annuaire*

8.2.1 Une opération **DirectoryUnbind** est utilisée à la fin d'une période d'accès à l'annuaire.

```
DirectoryUnbind ::= ABSTRACT-UNBIND
    FROM {readPort, searchPort, modifyPort }
```

8.2.2 Il n'y a pas d'arguments dans **DirectoryUnbind.**

9 Opérations de lecture de l'annuaire

Il existe deux opérations de lecture, la **lecture** proprement dite (**Read**) et la **comparaison** (**compare**) définies aux § 9.1 et 9.2. L'opération **Abandon**, définie au § 9.3, a été jointe aux opérations "lecture" pour des raisons de commodité.

9.1 *Lecture*

9.1.1 L'opération **Read** sert à extraire une information d'une entrée explicitement identifiée. Elle peut aussi servir à vérifier un nom spécifique. Les arguments de l'opération peuvent en option être signés (voir le § 7.10) par le demandeur. Sur demande, l'annuaire peut signer le résultat.

```
Read ::= ABSTRACT-OPERATION
    ARGUMENT        ReadArgument
    RESULT          ReadResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, Abandoned,
        SecurityError }

ReadArgument ::= OPTIONALLY-SIGNED SET {
    object          [0] Name,
    selection       [1] Selection F[13] EntryInformationSelection
                        DEFAULT {}
    COMPONENTS OF CommonArguments }

ReadResult   ::= OPTIONALLY-SIGNED SET {
    entry           [0] EntryInformation,
    COMPONENTS OF CommonResults }
```

9.1.2 Les différents arguments ont le sens qui leur est donné aux § 9.1.2.1 à 9.1.2.3.

9.1.2.1 L'argument **d'objet** identifie l'entrée d'objet d'où l'on veut tirer une information. Si le nom est accompagné d'un ou plusieurs pseudonymes, ceux-ci sont déréférencés (à moins que les commandes de service pertinentes ne l'interdisent).

9.1.2.2 L'argument de **sélection** indique l'information que l'on veut tirer de l'entrée (voir le § 7.6).

9.1.2.3 Les **CommonArguments** (voir le § 7.3) comprennent la spécification des commandes de service applicables à la demande. Pour cette opération, le composant **sizeLimit** n'est pas pertinent et, s'il est fourni, il n'en est pas tenu compte.

9.1.3 Si la demande aboutit, le résultat sera envoyé. Les paramètres de résultat ont le sens qui leur est donné aux § 9.1.3.1 et 7.4.

9.1.3.1 Le paramètre de résultat de l'entrée contient l'information demandée (voir le § 7.7).

9.1.4 Si la demande n'aboutit pas, l'une des erreurs énumérées sera signalée. Si aucun des attributs explicitement énumérés dans la **sélection** ne peut être renvoyé, un **AttributeError** avec problème **noSuchAttribute** sera signalé. Les conditions dans lesquelles d'autres erreurs sont signalées sont définies au § 12.

9.2 *Comparaison*

9.2.1 L'opération **Compare** sert à comparer une valeur (fournie à titre d'argument de la demande) avec la ou les valeurs d'un type d'attribut particulier dans une entrée d'objet particulière. Les arguments de l'opération peuvent, en option, être signés (voir le § 7.10) par le demandeur. Sur demande, l'annuaire peut signer le résultat.

```
Compare ::= ABSTRACT-OPERATION
    ARGUMENT CompareArgument
    RESULT CompareResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, Abandoned,
        SecurityError }

CompareArgument ::= OPTIONALLY-SIGNED
SET {
    object          [0] Name,
    purported       [1] AttributeValueAssertion,
    COMPONENTS OF CommonArguments }

CompareResult ::= OPTIONALLY-SIGNED
SET {
    DistinguishedName OPTIONAL,
    matched         [0] BOOLEAN,
    from Entry      [1] BOOLEAN DEFAULT TRUE,
    COMPONENTS OF CommonResults }
```

9.2.2 Les différents arguments ont la valeur qui leur est donnée aux § 9.2.2.1 à 9.2.2.3.

9.2.2.1 L'argument d'**objet** est le nom de l'entrée d'objet dont il s'agit. Si le **nom** s'accompagne d'un ou plusieurs pseudonymes, ceux-ci sont déréférencés (à moins que les commandes pertinentes ne l'interdisent).

9.2.2.2 L'argument **visé** identifie le type d'attribut et la valeur à comparer avec celle de l'entrée.

9.2.2.3 Les **CommonArguments** (voir le § 7.3) spécifient les commandes de service applicables à la demande. Pour cette opération, le composant **sizeLimit** n'est pas pertinent et, s'il est fourni, il n'en est pas tenu compte.

9.2.3 Si la demande aboutit (c'est-à-dire si la comparaison a effectivement lieu), le résultat est envoyé. Les paramètres du résultat ont le sens qui leur est donné aux § 9.2.3.1 à 9.2.3.3 et 7.4.

9.2.3.1 Le **DistinguishedName** est présent si un pseudonyme a été déréférencé et représente le nom spécifique de l'objet lui-même.

9.2.3.2 Le paramètre du résultat **matched** contient les résultats de la comparaison. Ce paramètre prend la valeur **VRAI** si les valeurs ont été comparées et accordées, **FAUX** dans le cas contraire.

9.2.3.3 Si **fromEntry** est **VRAI**, l'information a été comparée avec l'entrée; si c'est **FAUX**, une partie de l'information est comparée avec une copie.

9.2.4 Si la demande n'aboutit pas, l'une des erreurs répertoriées sera signalée. Les conditions dans lesquelles les erreurs individuelles sont signalées sont définies au § 12.

9.3 *Abandon*

9.3.1 Les opérations d'interrogation d'annuaire peuvent être abandonnées au moyen de l'opération **Abandon** si l'utilisateur ne s'intéresse plus au résultat.

```
Abandon ::= ABSTRACT-OPERATION
    ARGUMENT AbandonArgument
    RESULT AbandonResult
    ERRORS {AbandonFailed}

AbandonArgument ::= SEQUENCE {
    InvokeID        [0] InvokeID}

AbandonResult ::= NULL
```

9.3.2 Un seul argument, **InvokeID**, identifie l'opération qui doit être abandonnée. La valeur de **invokeID** est le même **invokeID** qui a servi à appeler l'opération qui doit être abandonnée.

9.3.3 Si la demande aboutit, un résultat est envoyé, bien qu'il ne soit pas porteur d'information. L'opération originale n'aboutira pas avec une erreur **Abandon**.

9.3.4 Si la demande n'aboutit pas, l'erreur **AbandonFailed** sera signalée. Cette erreur est décrite au § 12.3.

9.3.5 L'**abandon** ne s'applique qu'aux opérations d'interrogation, c'est-à-dire **Read, Compare, List** et **Search.**

9.3.6 Un DSA peut abandonner une opération localement. Si le DSA a enchaîné ou multireporté l'opération à d'autres DSA, il peut leur demander d'abandonner l'opération. Un DSA peut choisir de ne pas abandonner l'opération et alors d'envoyer l'erreur **AbandonFailed.**

10 Opérations de recherche dans l'annuaire

Il y a deux opérations dites de recherche: **Listage** et **Recherche**, respectivement définies aux § 10.1 et 10.2.

10.1 *Listage*

10.1.1 Une opération de **listage** sert à dresser la liste des subordonnés immédiats d'une entrée explicitement identifiée. Dans certains cas, la liste retournée peut être incomplète. Les arguments de l'opération peuvent, sur option, être signés (voir le § 7.10) par le demandeur. Sur demande, l'annuaire peut signer le résultat.

```
List ::= ABSTRACT-OPERATION
    ARGUMENT ListArgument
    RESULT ListResult
    ERRORS {
                NameError
        ServiceError, Referral, Abandoned,
        SecurityError }

List Argument ::= OPTIONALLY-SIGNED SET {
    object [0] Name,
    COMPONENTS OF CommonArguments }

ListResult ::= OPTIONALLY-SIGNED
CHOICE {
    listInfo SET {
    DistinguishedName OPTIONAL,
    subordinates [1] SET OF SEQUENCE {
        RelativeDistinguishedName,
        aliasEntry [0] BOOLEAN DEFAULT FALSE
        fromEntry [1] BOOLEAN DEFAULT TRUE},
    partialOutcomeQualifier [2]
                PartialOutcomeQualifier OPTIONAL
    COMPONENTS OF CommonResults },
    uncorrelatedListInfo [0] SET OF
                ListResult }

PartialOutcomeQualifier ::= SET {
    limitProblem [0] LimitProblem
        OPTIONAL,
    unexplored [1] SET OF
        ContinuationReference OPTIONAL,
    unavailableCriticalExtensions [2] BOOLEAN DEFAULT FALSE }

LimitProblem ::= INTEGER {
    timeLimitExceeded (0),
    sizeLimitExceeded (1),
    administrativeLimitExceeded (2) }
```

10.1.2 Les différents arguments ont le sens qui leur est donné aux § 10.1.2.1 et 7.3.

10.1.2.1 L'argument **d'objet** identifie l'entrée d'objet (ou éventuellement la racine) dont la liste des subordonnés immédiats est à établir. Si le **nom** s'accompagne d'un ou de plusieurs pseudonymes, ceux-ci sont déréférencés (à moins que les commandes pertinentes ne l'interdisent).

10.1.3 La demande aboutit si l'objet est localisé, qu'il y ait ou non des informations subordonnées à envoyer. Les paramètres de résultat ont le sens défini aux § 10.1.3.1 à 10.1.3.4 et 7.4.

10.1.3.1 Le **DistinguishedName** est présent si un pseudonyme a été déréférencé. Il représente le nom spécifique de l'objet lui-même.

10.1.3.2 Le paramètre **subordonnés** transmet l'information concernant, le cas échéant, les subordonnés immédiats de l'entrée nommée. Les entrées subordonnées qui sont des pseudonymes ne sont pas déréférencées.

10.1.3.2.1 Le **Relative DistinguishedName** est celui du subordonné.

10.1.3.2.2 Le paramètre **fromEntry** indique si l'information a été obtenue de l'entrée (**VRAI**) ou d'une copie de l'entrée (**FAUX**).

10.1.3.2.3 Le paramètre **aliasEntry** indique si l'entrée subordonnée est une entrée pseudonyme (**VRAI**) ou non (**FAUX**).

10.1.3.3 Le **qualificatif PartialOutcome** se compose de trois sous-composants définis aux § 10.1.3.3.1 à 10.1.3.3.3. Ce paramètre est présent chaque fois que le résultat est incomplet.

10.1.3.3.1 Le paramètre **LimitProblem** indique si le délai, la taille limite ou une limite administrative a été dépassé. Les résultats incomplets, renvoyés, sont ceux qui étaient disponibles lorsque la limite a été atteinte.

10.1.3.3.2 Le paramètre **unexplored** est présent si des régions du DIT n'ont pas été explorées. Son information permet au DUA de continuer le traitement de l'opération **List** en contactant d'autres points d'accès s'il choisit de le faire. Ce paramètre se compose d'un jeu (éventuellement vide) de **références Continuation**, composées chacune du nom d'un objet de base à partir duquel l'opération peut progresser, d'une valeur appropriée d'**OperationProgress** et d'un jeu de points d'accès à partir desquels la demande peut encore progresser. Les **ContinuationReferences** renvoyées doivent être dans la portée de renvoi de la commande de service d'opération.

10.1.3.3.3 Le paramètre **unavailableCriticalExtensions** indique, s'il est présent, qu'une ou plusieurs extensions critiques ne sont pas disponibles dans une partie de l'annuaire.

10.1.3.4 Quand le DUA a demandé une demande de protection de **signed**, le paramètre **uncorrelatedListInfo** peut comprendre plusieurs jeux de paramètres de résultat provenant de et signés par différents composants de l'annuaire. Si aucun DSA de la chaîne ne peut corréler tous les résultats, le DUA doit assembler le résultat effectif à partir des différents éléments.

10.1.4 Si la demande n'aboutit pas, l'une des erreurs répertoriées sera signalée. Les conditions dans lesquelles les erreurs individuelles seront signalées sont définies au § 12.

10.2 *Recherche*

10.2.1 L'opération **recherche** sert à chercher dans une portion du DIT les entrées en cause et à retourner l'information sélectionnée en provenance de ces entrées. Les arguments de l'opération peuvent, en option, être signés (voir le § 7.10) par le demandeur. Sur demande, l'annuaire peut signer le résultat.

```
Search    ::=  ABSTRACT-OPERATION
     ARGUMENT       SearchArgument
     RESULT         SearchResult
     ERRORS {
               AttributeError, NameError,
               ServiceError, Referral, Abandoned,
               SecurityError }

SearchArgument     ::=  OPTIONALLY-SIGNED
SET {
     baseObject                [0]  Name,
     subset                    [1]  INTEGER {
          baseObject (0),
          oneLevel(1),
          wholeSubtree(2)}          DEFAULT baseObject,
     filter                    [2]  Filter DEFAULT and {}.
```

```
        searchAliases           [3]  BOOLEAN DEFAULT TRUE,
        selection               [4]  EntryInformationSelection DEFAULT {}
            COMPONENTS OF CommonArguments }

SearchResult    ::=  OPTIONALLY-SIGNED
        CHOICE  {
        searchInfo SET {
        DistinguishedName OPTIONAL,
        entries     [0]  SET OF EntryInformation,
        partialOutcomeQualifier
            [2] PartialOutcomeQualifier OPTIONAL,
COMPONENTS OF CommonResults },
uncorrelatedSearchInfo [0] SET OF
        SearchResult }
```

10.2.2 Les différents arguments ont le sens qui leur est donné aux § 10.2.2.1 à 10.2.2.3, 10.2.2.5 et 7.3.

10.2.2.1 L'argument **baseObject** identifie l'entrée d'objet (ou éventuellement la racine) au sujet de laquelle ont lieu les recherches.

10.2.2.2 L'argument **subset** indique si les recherches doivent s'appliquer:

a) au **baseObject** seulement;

b) uniquement aux subordonnés immédiats de l'objet de base (**un niveau**);

c) à l'objet de base et à tous ses subordonnés (**sous-arbre entier**).

10.2.2.3 L'argument **filter** sert à éliminer de l'espace de recherches les entrées qui n'entrent pas en ligne de compte. L'information ne sera envoyée qu'au sujet des entrées qu'admet le filtre (voir le § 7.8).

10.2.2.4 Les pseudonymes doivent être déréférencés au cours de la localisation de l'objet de base, à condition que soit fixée la commande **dontDereferenceAliasesServiceControl**. Les pseudonymes parmi les subordonnés de l'objet de base peuvent être déréférencés pendant la recherche, à condition que soit fixé le paramètre **searchAliases**. Si le paramètre **searchAliases** est **VRAI**, les pseudonymes sont déréférencés; s'il est **FAUX**, les pseudonymes ne doivent pas être déréférencés. Si le paramètre **searchAliases** est **VRAI**, la recherche continue dans le sous-arbre de l'objet désigné par un pseudonyme.

10.2.2.5 L'argument **sélection** indique quelle est l'information d'entrée demandée (voir le § 7.6).

10.2.3 La demande aboutit si l'objet de base est localisé, qu'il y ait ou non des subordonnés à envoyer.

Remarque - Comme corollaire, le résultat d'une **recherche** (non filtrée) appliquée à une seule entrée n'est pas forcément identique à une **lecture** cherchant à interroger le même jeu d'attributs de l'entrée. En effet, cette dernière envoie une **erreur d'attribut** si aucun des attributs choisis n'existe dans l'entrée.

Les paramètres du résultat ont le sens défini aux § 10.2.3.1 à 10.2.3.4 et 7.3.

10.2.3.1 Le **DistinguishedName** est présent si un pseudonyme a été déréférencé; il représente le nom spécifique de l'objet de base.

10.2.3.2 Le paramètre **entries** transmet l'information demandée en provenance de chaque entrée (zéro ou plus) admise par le filtre (voir le § 7.5).

10.2.3.3 Le **qualificatif PartialOutcome** se compose des deux sous-composants décrits pour l'opération Listage au § 10.1.3.4.

10.2.3.4 Le paramètre **uncorrelatedSearchInfo** est celui décrit pour **uncorrelatedListInfo** au § 10.1.3.4.

10.2.4 Si la demande n'aboutit pas, l'une des erreurs répertoriées sera signalée. Les conditions dans lesquelles les erreurs individuelles seront signalées sont définies au § 12.

11 Opérations de modification de l'annuaire

On compte quatre opérations de modification de l'annuaire: **AddEntry**, **RemoveEntry**, **ModifyEntry** et **ModifyRDN**, respectivement définies aux § 11.1 à 11.4.

Remarque 1 - Chacune de ces opérations abstraites identifie l'entrée-cible au moyen d'un nom *spécifique*.

Remarque 2 - Le succès des opérations **AddEntry**, **RemoveEntry**, **ModifyRDN** dépendra de la répartition physique de la DIB dans l'annuaire. L'échec sera signalé par un **UpdateError** et le problème **affecte MultipleDSA**. Voir la Recommandation X.518.

11.1 *Add Entry*

11.1.1 L'opération **AddEntry** sert à ajouter une entrée de feuille (une entrée d'objet ou de pseudonyme) du DIT. Les arguments de l'opération peuvent, en option, être signés (voir le § 7.10) par le demandeur.

```
AddEntry ::= ABSTRACT-OPERATION
    ARGUMENT        AddEntryArgument
    RESULT          AddEntryResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, SecurityError,
        UpdateError }

AddEntryArgument ::= OPTIONALLY-SIGNED
SET {
    object          [0] DistinguishedName,
    entry           [1] SET OF Attribute,
    COMPONENTS OF CommonArguments }

AddEntryResult   ::= NULL
```

11.1.2 Les différents arguments ont la valeur qui leur est donnée aux § 11.1.2.1 à 11.1.2.3.

11.1.2.1 L'argument **object** identifie l'entrée à ajouter. Le supérieur immédiat, qui doit déjà exister pour que l'opération réussisse, peut être déterminé par retrait du dernier composant RDN (qui appartient à l'entrée à créer).

11.1.2.2 L'argument **entry** contient l'information qui constitue l'entrée à créer. L'annuaire fera en sorte que l'entrée soit conforme au schéma d'annuaire. Lorsque l'entrée en cours de création est un pseudonyme, aucune vérification n'est effectuée pour s'assurer que l'attribut **aliasedObjectName** désigne une entrée valide.

11.1.2.3 Les **arguments communs** (voir le § 7.3) comprennent une spécification des commandes de service applicables à la demande. Pour cette opération, l'option **dontDereferenceAlias** et le composant **sizeLimit** ne sont pas significatifs et, s'ils sont fournis, il n'en est pas tenu compte. De plus, les pseudonymes ne sont jamais déréférencés par cette opération.

11.1.3 Si la demande aboutit, un résultat sera retourné, bien qu'il ne transmette aucune information.

11.1.4 Si la demande n'aboutit pas, l'une des erreurs répertoriées sera signalée. Les conditions dans lesquelles les erreurs individuelles seront signalées, sont définies au § 12.

11.2 *Remove Entry*

11.2.1 L'opération **RemoveEntry** sert à enlever une entrée de feuille (une entrée d'objet ou de pseudonyme) du DIT. Les arguments de l'opération peuvent, en option, être signés (voir le § 7.10) par le demandeur.

```
RemoveEntry ::= ABSTRACT-OPERATION
    ARGUMENT RemoveEntryArgument
    RESULT RemoveEntryResult
    ERRORS {
        NameError,
        ServiceError, Referral, SecurityError,
        UpdateError}
```

```
RemoveEntryArgument ::=     OPTIONALLY-SIGNED SET {
        object          [0]     DistinguishedName,
        COMPONENTS OF CommonArguments }

RemoveEntryResult   ::=     NULL
```

11.2.2 Les différents arguments ont le sens qui leur est donné aux § 11.2.2.1 et 11.2.2.2.

11.2.2.1 L'argument **object** identifie l'entrée à supprimer. Les pseudonymes du nom ne seront pas déréférencés.

11.2.2.2 Les **arguments Communs** (voir le § 7.3) comprennent une spécification des commandes de service applicables à la demande. Pour cette opération, l'option **dontDereferenceAlias** et le composant **sizeLimit** ne sont pas significatifs et, s'ils sont fournis, il n'en est pas tenu compte. Les pseudonymes ne sont jamais déréférencés par cette opération.

11.2.3 Si la demande aboutit, un résultat sera retourné, bien qu'il ne transmette aucune information.

11.2.4 Si la demande n'aboutit pas, l'une des erreurs répertoriées sera signalée. Les conditions dans lesquelles les erreurs individuelles seront signalées sont définies au § 12.

11.3 *Modify Entry*

11.3.1 L'opération **ModifyEntry** sert à apporter une série d'une ou plusieurs des modifications qui suivent à une entrée:

a) ajouter un nouvel attribut;

b) supprimer un attribut;

c) ajouter des valeurs d'attribut;

d) supprimer des valeurs d'attribut;

e) remplacer des valeurs d'attribut;

f) modifier un pseudonyme.

Les arguments de l'opération peuvent, en option, être signés par le demandeur (voir le § 7.10).

```
ModifyEntry ::=     ABSTRACT-OPERATION
    ARGUMENT    ModifyEntryArgument
    RESULT      ModifyEntryResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, SecurityError,
        UpdateError }

ModifyEntryArgument ::=     OPTIONALLY-SIGNED SET {
    object          [0]  DistinguishedName,
    changes         [1]  SEQUENCE OF EntryModification,
    COMPONENTS OF CommonArguments }

ModifyEntryResult   ::=  NULL

EntryModification   ::=  CHOICE {
    addAttribute    [0]      Attribute,
    removeAttribute          [1]  AttributeType,
    addValues                [2]  Attribute,
    removeValues    [3]      Attribute }
```

11.3.2 Les différents arguments ont le sens qui leur est donné aux § 11.3.2.1 et 11.3.2.2.

11.3.2.1 L'argument **object** identifie l'entrée à laquelle les modifications doivent être appliquées. Aucun pseudonyme éventuel du nom ne sera déréférencé.

11.3.2.2 L'argument **changes** définit une séquence de modifications qui sont appliquées dans l'ordre spécifié. En cas d'échec de l'une des modifications, une **erreur** est engendrée et l'entrée est laissée dans l'état où elle était avant l'opération. En d'autres termes, l'opération est insécable. Le résultat final de la séquence de modifications ne doit pas violer le schéma de l'annuaire. Il est toutefois possible - et parfois nécessaire - que les modifications **EntryModification** semblent le faire. Les types suivants de modifications peuvent se présenter:

a) **addAttribute**: Identifie un nouvel attribut, pleinement spécifié par l'argument, à ajouter à une entrée. Toute tentative d'adjonction d'un attribut déjà existant entraîne une **AttributeError**;

b) **removeAttribute**: L'argument identifie (par son type) un attribut à supprimer d'une entrée. Toute tentative de suppression d'un attribut qui n'existe pas se traduit par une **AttributeError**;

Remarque - Non autorisé si le type d'attribut est présent dans le RDN.

c) **addValues**: Identifie un attribut d'après le type d'attribut dans l'argument et spécifie une ou plusieurs valeurs à ajouter à l'attribut. Toute tentative d'adjonction d'une valeur déjà existante entraîne une erreur;

d) **removeValues**: Identifie un attribut d'après le type d'attribut dans l'argument et spécifie une ou plusieurs valeurs à enlever de l'attribut. Si les valeurs ne sont pas présentes dans l'attribut, le résultat est une **AttributeError**. S'il est fait une tentative de modifier l'attribut catégorie d'objet, une erreur de mise à jour est envoyée.

Remarque - Non autorisé si l'une des valeurs est présente dans le RDN.

Les valeurs peuvent être remplacées par une combinaison de **addValues** et de **removeValues** dans une seule opération **ModifyEntry**.

11.3.2.3 Les **arguments Communs** (voir le § 7.3) comprennent une spécification des commandes de service applicables à la demande. Pour les besoins de cette opération, l'option **dontDereferenceAlias** et le composant **sizeLimit** ne sont pas pertinents; s'ils sont fournis, il n'en est pas tenu compte. Les pseudonymes ne sont jamais déréférencés par cette opération.

11.3.3 Si la demande aboutit, un résultat sera retourné, bien qu'il ne transmette aucune information.

11.3.4 Si la demande échoue, l'une des erreurs répertoriées sera signalée. Les conditions dans lesquelles les erreurs individuelles seront signalées sont définies au § 12.

11.4 *Modify RDN*

11.4.1 L'opération **ModifyRDN** sert à modifier le nom spécifique relatif d'une entrée de feuille (entrée d'objet ou de pseudonyme) dans le DIT. Les arguments de l'opération peuvent en option être signés (voir le § 7.10) par le demandeur.

```
ModifyRDN ::= ABSTRACT-OPERATION
    ARGUMENT  ModifyRDNArgument
    RESULT    ModifyRDNResult
    ERRORS {
        NameError,
        ServiceError, Referral, SecurityError,
        UpdateError }

ModifyRDNArgument ::=  OPTIONALLY-SIGNED SET {
    object          [0]  DistinguishedName,
    newRDN          [1]  RelativeDistinguishedName,
    deleteOldRDN    [2]  BOOLEAN DEFAULT FALSE,
    COMPONENTS OF CommonArguments }

ModifyRDNResult   ::=  NULL
```

11.4.2 Les divers paramètres ont le sens qui est défini aux § 11.4.2.1 à 11.4.2.5.

11.4.2.1 L'argument **object** identifie l'entrée dont le nom spécifique relatif doit être modifié. Les pseudonymes dans le nom ne seront pas déréférencés. L'entrée du supérieur immédiat n'aura pas de référence subordonnée non spécifique (voir la Recommandation X.518).

11.4.2.2 L'argument **newRDN** spécifie le nouveau RDN de l'entrée.

11.4.2.3 Si une valeur d'attribut du nouveau RDN n'existe pas encore dans l'entrée (comme partie de l'ancien RDN ou comme valeur non spécifique) elle est ajoutée. Si elle ne peut pas être ajoutée, une erreur est envoyée.

11.4.2.4 Si le fanion **deleteOldRDN** est mis, toutes les valeurs d'attribut de l'ancien RDN qui ne sont pas dans le nouveau RDN sont supprimées. Si ce fanion n'est pas mis, les anciennes valeurs doivent rester dans l'entrée (non en tant que partie du RDN). Le fanion sera mis quand un attribut de valeur du RDN a sa valeur changée par l'opération. Si l'opération enlève la dernière valeur d'attribut d'un attribut, cet attribut doit être supprimé.

11.4.2.5 Les **CommonArguments** (voir le § 7.3) incluent une spécification des commandes de service applicables à la demande. Pour les besoins de cette opération, l'option **DontDereferenceAlias** et le composant **sizeLimit** ne sont pas pertinents et, s'ils sont fournis, il n'en est pas tenu compte. Les pseudonymes ne sont jamais déréférencés par cette opération.

11.4.3 Si la demande aboutit, un résultat sera envoyé, bien qu'il n'achemine aucune information.

11.4.4 Si la demande échoue, une des erreurs énumérées est signalée. Les conditions dans lesquelles les erreurs sont envoyées sont définies au § 12.

11.4.5 Telle qu'elle est définie dans la présente Recommandation, cette opération ne peut être utilisée que sur une entrée feuille.

12 Erreurs

12.1 *Priorité d'erreur*

12.1.1 L'annuaire ne poursuit pas une opération au-delà du point où il détermine qu'une erreur doit être signalée.

Remarque 1 - Il ressort de cette règle que la première erreur détectée peut différer lorsqu'il y a répétition d'une même demande, puisqu'il n'existe pas d'ordre logique spécifique de traitement d'une demande donnée. Par exemple, les DSA peuvent être recherchés dans des ordres différents.

Remarque 2 - La règle de la priorité de l'erreur ici spécifiée ne s'applique qu'au service abstrait fourni par l'annuaire dans son ensemble. Des règles différentes s'appliquent quand la structure interne de l'annuaire est prise en considération.

12.1.2 Si l'annuaire détecte simultanément plusieurs erreurs, la liste suivante détermine l'erreur qui est signalée. L'erreur a) a un rang de priorité plus élevé que l'erreur b), etc.; c'est donc la première des deux qui sera signalée.

a) **NameError**

b) **UpdateError**

c) **AttributeError**

d) **SecurityError**

e) **ServiceError.**

12.1.3 Il n'y a pas conflit de priorité entre les erreurs suivantes:

a) **AbandonFailed**, puisque l'erreur est propre à une seule opération (**Abandon**) au cours de laquelle ne peut se produire d'autres erreurs;

b) **Abandoned**, erreur qui n'est pas signalée si une opération **Abandon** est reçue en même temps que la détection d'une erreur. Dans ce cas, une erreur **AbandonFailed** signalant le problème **tooLate** est transmise avec l'annonce de l'erreur effective rencontrée;

c) **Referral**, qui n'est pas une erreur "réelle", mais qui indique seulement que l'annuaire a détecté que le DUA doit présenter sa demande à un autre point d'accès.

12.2 *Abandoned*

12.2.1 Cette issue peut être signalée pour toute opération de recherche en instance dans l'annuaire (c'est-à-dire **lecture, recherche, comparaison, listage**) si le DUA appelle une opération **Abandon** avec l'**InvokeID** approprié.

Abandoned ::= ABSTRACT-ERROR -- ***not literally an "error"***

12.2.2 Aucun paramètre n'est associé à cette erreur.

12.3 *Abandon Failed*

12.3.1 L'erreur **AbandonFailed** signale un problème rencontré pendant une tentative d'abandon d'une opération.

```
AbandonFailed ::= ABSTRACT-ERROR
    PARAMETER SET {
        problem   [0]  AbandonProblem,
        operation [1]  InvokeID}

AbandonProblem ::= INTEGER
                   noSuchOperation (1),
                   tooLate (2),
                   cannotAbandon (3) }
```

12.3.2 Les différents paramètres ont le sens qui leur est donné aux § 12.3.2.1 et 12.3.2.2.

12.3.2.1 Le **problème** rencontré est spécifié. L'un quelconque des problèmes suivants peut être indiqué:

a) **noSuchOperation**, quand l'annuaire n'a pas connaissance de l'opération qui est à abandonner (parce qu'il n'y a pas eu cet appel ou parce que l'annuaire l'a oublié);

b) **tooLate**, quand l'annuaire a déjà réagi à l'opération;

c) **cannotAbandon**, quand il y a eu tentative d'abandon d'une opération dont l'abandon est interdit (par exemple, modify), ou si l'abandon n'a pas pu être réalisé.

12.3.2.2 Identification de l'**opération** particulière (appel) à abandonner.

12.4 *Attribute Error*

12.4.1 Une **AttributeError** signale un problème d'attribut.

```
AttributeError ::= ABSTRACT-ERROR
    PARAMETER SET {
        object    [0]  Name,
        problems  [1]  SET OF SEQUENCE {
            problem   [0]  AttributeProblem,
            type      [1]  AttributeType
            value     [2]  AttributeValue
                                OPTIONAL }}

AttributeProblem    ::= INTEGER {
    noSuchAttributeOrValue (1),
    InvalidAttributeSyntax (2),
    undefinedAttributeType (3),
    InappropriateMatching (4),
    constraintViolation (5)
    attributeOrValueAlreadyExists (6) }
```

12.4.2 Les différents paramètres ont la valeur qui leur est donnée aux § 12.4.2.1 et 12.4.2.2.

12.4.2.1 Le paramètre **object** identifie l'entrée à laquelle s'appliquait l'opération quand l'erreur s'est produite.

12.4.2.2 Un ou plusieurs **problèmes** peuvent être spécifiés. Chaque **problème** (identifié ci-après) est accompagné par une indication du **type** d'attribut et, au besoin pour lever un doute, par la **valeur** de l'attribut qui a causé le problème:

a) **noSuchAttributeOrValue**: Dans l'entrée nommée manque l'un des attributs ou l'une des valeurs spécifiés comme arguments de l'opération;

b) **invalidAttributeSyntax**: Une valeur d'attribut visée, spécifiée à titre d'argument de l'opération, n'est pas conforme à la syntaxe d'attribut du type d'attribut;

c) **undefinedAttributeType**: Un type d'attribut indéfini a été fourni comme l'un des arguments de l'opération. Cette erreur ne peut se produire qu'à propos d'opération **Add**, **Remove**, **Modify** ou **ModifyRDN**;

d) **inappropriateMatching**: Une tentative a eu lieu, par exemple, dans un filtre, pour utiliser une règle d'appariement non définie pour le type d'attribut en cause;

e) **constraintViolation**: Un attribut, ou une valeur d'attribut appliquée à l'argument d'une opération abstraite n'est pas conforme aux limites imposées par la Recommandation X.501 ou par la définition d'attribut (par exemple, la valeur dépasse la taille maximale autorisée);

f) **attributeOrValueAlreadyExists**: Une tentative a eu lieu pour ajouter un attribut qui existe déjà dans l'entrée, ou une valeur qui existe déjà dans l'attribut.

12.5 *Name Error*

12.5.1 **NameError** signale un problème relatif au nom fourni à titre d'argument pour l'opération.

```
NameError ::= ABSTRACT-ERROR
    PARAMETER SET {
        problem   [0]  NameProblem,
        matched   [1]  Name}

NameProblem  ::=  INTEGER {
    noSuchObject (1),
    aliasProblem (2),
    invalidAttributeSyntax (3),
    aliasDereferencingProblem (4) }
```

12.5.2 Les différents paramètres ont le sens qui leur est donné aux § 12.5.2.1 et 12.5.2.2.

12.5.2.1 **Problème** particulier rencontré. L'un des problèmes suivants peut être indiqué:

a) **noSuchObject**: Le nom fourni ne concorde pas avec le nom d'un objet;

b) **aliasProblem**: Un pseudonyme déréférencé ne nomme aucun objet;

c) **invalidAttributeSyntax**: Un type d'attribut et la valeur d'attribut qui l'accompagne dans une AVA du nom sont incompatibles;

d) **aliasDereferencingProblem**: Un pseudonyme a été rencontré dans une circonstance où il n'est pas autorisé.

12.5.2.2 Le paramètre **concordant** contient le nom de l'entrée de dernier rang (objet ou pseudonyme) dans le DIT qui a été apparié et constitue une forme tronquée du nom fourni ou, si un pseudonyme a été déréférencé, du nom résultant.

Remarque - S'il se pose un problème au sujet du type et/ou de la valeur d'attribut dans le nom offert dans un argument d'opération d'annuaire, cela est signalé au moyen d'une **NameError** (avec problème **invalidAttributeSyntax**) plutôt qu'au moyen d'une **AttributeError** ou d'une **UpdateError**.

12.6 *Referral*

12.6.1 Un **Referral** dirige l'utilisateur du service vers un ou plusieurs points d'accès mieux équipés pour exécuter l'opération demandée.

```
Referral  ::=  ABSTRACT-ERROR  -- not literally an "error"
    PARAMETER SET {
        candidate          [0]  ContinuationReference }
```

12.6.2 L'erreur a un seul paramètre qui contient une **référence de continuation** qui peut servir à faire progresser l'opération (voir la Recommandation X.518).

12.7 *Security Error*

12.7.1 **SecurityError** signale un problème au cours d'une opération effectuée pour des raisons de sécurité.

```
SecurityError  ::=  ABSTRACT-ERROR
    PARAMETER SET {
        problem   [0]        SecurityProblem }

SecurityProblem      ::=  INTEGER {
    InappropriateAuthentication (1),
    InvalidCredentials (2),
    InsufficientAccessRights (3),
    InvalidSignature (4),
    protectionRequired (5),
    noInformation (6) }
```

12.7.2 L'erreur a un seul paramètre qui signale le **problème** rencontré. Les problèmes suivants peuvent être indiqués:

a) **inappropriateAuthentication**: Le niveau de sécurité associé aux pouvoirs du demandeur ne correspond pas au niveau de protection demandé, par exemple, des pouvoirs simples ont été fournis alors qu'il en aurait fallu de renforcés;

b) **invalidCredentials**: Les pouvoirs fournis ne sont pas valides;

c) **insufficientAccessRights**: Le demandeur n'a pas le droit de procéder à l'opération demandée;

d) **invalidSignature**: On a constaté que la signature de la demande n'était pas valide;

e) **protectionRequired**: L'annuaire a refusé d'effectuer l'opération demandée parce que l'argument n'était pas signé;

f) **noInformation**: L'opération demandée a produit une erreur de sécurité pour laquelle on ne dispose d'aucune information.

12.8 *Service Error*

12.8.1 **ServiceError** signale un problème lié à la prestation de service.

```
ServiceError    ::=   ABSTRACT-ERROR
    PARAMETER SET {
        problem   [0]   ServiceProblem }

ServiceProblem ::=   INTEGER {
        busy (1),
        unavailable (2),
        unwillingToPerform (3),
        chainingRequired (4),
        unableToProceed (5),
        invalidReference (6),
        timeLimitExceeded (7),
        administrativeLimitExceeded (8),
        loopDetected (9),
        unavailableCriticalExtension (10),
        outOfScope (11),
        ditError (12) }
```

12.8.2 L'erreur a un seul paramètre, qui signale le problème rencontré. Les problèmes suivants peuvent être indiqués:

a) **busy**: L'annuaire, ou l'une de ses parties, est trop occupé pour procéder à l'opération demandée, mais il pourra le faire à bref délai;

b) **unavailable**: L'annuaire, ou l'une de ses parties, est indisponible;

c) **unwillingToPerform**: L'annuaire, ou l'une de ses parties, n'est pas disposé à répondre à cette demande, parce qu'elle pourrait conduire à une consommation excessive de ressources, ou parce qu'elle est contraire à la politique de l'autorité administrative en cause;

d) **chainingRequired**: L'annuaire n'est pas en mesure de répondre à la demande, si ce n'est en procédant par chaînage, alors que le chaînage a été interdit par l'option de commande de service chainingProhibited;

e) **unableToProceed**: Le DSA qui renvoie cette erreur n'est pas habilité administrativement pour le contexte d'appellation approprié et il n'est donc pas en mesure de participer à la résolution du nom;

f) **invalidReference**: Le DSA n'était pas en mesure d'exécuter ce qui était demandé par le DUA (via **OperationProgress**). Cela peut être dû à l'utilisation d'un renvoi non valide;

g) **timeLimitExceeded**: L'annuaire a atteint le délai limite fixé par l'utilisateur dans une commande de service. Aucun résultat partiel n'est disponible pour être envoyé à l'utilisateur;

h) **administrativeLimitExceeded**: L'annuaire a atteint une limite fixée par une autorité administrative et aucun résultat partiel n'est disponible pour être envoyé à l'utilisateur;

i) **loopDetected**: L'annuaire n'est pas en mesure d'exécuter ce qui est demandé, par suite d'une boucle interne;

j) **unavailableCriticalExtension**: L'annuaire est incapable d'exécuter la demande du fait qu'une ou plusieurs extensions critiques ne sont pas disponibles;

k) **OutOfScope**: Il n'existe pas de renvois dans la portée demandée;

l) **ditError**: L'annuaire n'est pas en mesure d'accomplir ce qui est demandé, par suite d'un problème de cohérence de DIT.

12.9 *Update Error*

12.9.1 **UpdateError** signale les problèmes que posent des tentatives d'ajouter, supprimer ou modifier une information dans la DIB.

```
UpdateError ::= ABSTRACT-ERROR
    PARAMETER SET {
        problem [0] UpdateProblem }

UpdateProblem ::= INTEGER {
    namingViolation (1),
    objectClassViolation (2),
    notAllowedOnNonLeaf (3),
    notAllowedOnRDN (4),
    entryAlreadyExists (5),
    affectsMultipleDSAs (6),
    objectClassModificationProhibited (7) }
```

12.9.2 L'erreur a un seul paramètre de **problème**, qui signale le **problème** particulier rencontré. Les problèmes suivants peuvent être indiqués:

a) **namingViolation**: Une tentative d'adjonction ou de modification causerait une violation des règles de structure du DIT, telles qu'elles sont énoncées dans le schéma de l'annuaire et la Recommandation X.501. En effet, elle insérerait une entrée comme subordonnée d'une entrée pseudonyme ou dans une région du DIT non autorisée aux membres de sa catégorie d'objet, ou elle définirait un RDN pour une entrée incluant un type d'attribut interdit;

b) **objectClassViolation**: La tentative de mise à jour produirait une entrée incompatible avec la définition fournie par sa catégorie d'objet, ou avec les définitions de la Recommandation X.501 qui concernent les catégories d'objet;

c) **notAllowedOnNonLeaf**: L'opération tentée n'est autorisée que sur les entrées feuilles du DIT;

d) **notAllowedOnRDN**: L'opération tentée affecterait le RDN (par exemple, retrait d'un attribut qui fait partie du RDN);

e) **entryAlreadyExists**: Une tentative d'opération AddEntry nomme une entrée qui existe déjà;

f) **affectsMultipleDSAs**: Une tentative de mise à jour porterait nécessairement sur de multiples DSA, ce qui n'est pas autorisé;

g) **objectClassModificationProhibited**: Une tentative d'opération pour modifier l'attribut de la catégorie d'objet.

Remarque - **UpdateError** n'est pas utilisée pour signaler les problèmes concernant les types d'attribut, les valeurs d'attribut ou les violations de contrainte rencontrés dans une opération d'**adjonction**, de **suppression**, de **modification** ou de **modification du RDN**. Ces problèmes sont signalés au moyen d'**AttributeError**.

ANNEXE A

(à la Recommandation X.511)

Service abstrait en ASN.1

La présente annexe fait partie de la norme.

La présente annexe contient toutes les définitions de type, valeur et macro ASN.1 incluses dans la Recommandation sous la forme du module ASN.1, **DirectoryAbstractService**.

```
DirectoryAbstractService {joint-ISO-CCITT ds(5) modules(1)  directoryAbstractService(2)}
DEFINITIONS ::=
BEGIN

EXPORTS
     directory, readPort, searchPort, modifyPort,
     DirectoryBind, DirectoryBindArgument,
     DirectoryUnbind,
     Read, ReadArgument, ReadResult,
     Abandon, AbandonArgument, AbandonResult,
     Compare, CompareArgument, CompareResult,
     List, ListArgument, ListResult,
     Search, SearchArgument, SearchResult,
     AddEntry, AddEntryArgument, AddEntryResult,
     RemoveEntry, RemoveEntryArgument, RemoveEntryResult,
     ModifyEntry, ModifyEntryArgument, ModifyEntryResult,
     ModifyRDN, ModifyRDNArgument, ModifyRDNResult,
     Abandoned, AbandonFailed, AttributeError, NameError,
     Referral, SecurityError, ServiceError, UpdateError,
     SecurityParameters;

IMPORTS
     informationFramework, authenticationFramework,
               distributedOperations, directoryObjectIdentifiers
          FROM     UsefulDefinitions     {joint-iso-ccitt ds(5) modules(1)
                                         usefulDefinitions(0)}

OBJECT, PORT, ABSTRACT-BIND, ABSTRACT-UNBIND,
ABSTRACT-OPERATION, ABSTRACT-ERROR
     FROM     AbstractServiceNotation     {joint-iso-ccitt mhs-motis(6)
                                          asdc(2) modules(0) notation(1) }

Attribute, AttributeType, AttributeValue, AttributeValueAssertion,
DistinguishedName, Name, RelativeDistinguishedName
     FROM     InformationFramework InformationFramework

id-ot-directory, id-ot-dua, id-pt-read, id-pt-search, id-pt-modify
     FROM     DirectoryObjectIdentifiers  directoryObjectIdentifiers

ContinuationReference, OperationProgress
     FROM     DistributedOperations distributedOperations

Certificate, CertificationPath, SIGNED,
PROTECTED, AlgorithmIdentifier
     FROM     AuthenticationFramework authenticationFramework
InvokeID,
     FROM     Remote-Operations-Notation     {joint-iso-ccitt
                                             remoteOperations(4)  notation(0)};


-- macro for representing optional signing --

OPTIONALLY-SIGNED MACRO ::=
BEGIN
     TYPE NOTATION     ::=     type (Type)
     VALUE NOTATION    ::=     value (VALUE CHOICE { Type, SIGNED Type})
END


-- objects and ports --

directory
     OBJECT
          PORTS  { readPort [S].
                   searchPort [S],
                   modifyPort [S]}
::=  id-ot-directory
```

```
dua
     OBJECT
          PORTS  { readPort [C],
                   searchPort [C]
                   modifyPort [C]}
::=  id-ot-dua

readPort
     PORT
          CONSUMER INVOKES {
               Read, Compare, Abandon}
::=  id-pt-read

searchPort
     PORT
          CONSUMER INVOKES {
               List, Search }
::=  id-pt-search

modifyPort
     PORT
          CONSUMER INVOKES {
               AddEntry, RemoveEntry,
               ModifyEntry, ModifyRDN}
::=  id-pt-modify


-- bind and unbind --


DirectoryBind  ::=  ABSTRACT-BIND
     TO { readPort, searchPort, modifyPort }
     BIND
     ARGUMENT   DirectoryBindArgument
     RESULT     DirectoryBindResult
     BIND-ERROR DirectoryBindError

DirectoryBindArgument  ::=  SET {
     credentials    [0]  Credentials  OPTIONAL,
     versions       [1]  Versions DEFAULT v1988}

Credentials  ::=  CHOICE {
     simple     [0]  SimpleCredentials,
     strong     [1]  StrongCredentials,
     externalProcedure  [2]  EXTERNAL }

SimpleCredentials  ::=  SEQUENCE {
     name [0]  DistinguishedName,
     validity  [1]  SET {
          time1     [0]  UTCTime OPTIONAL,
          time2     [1]  UTCTime OPTIONAL,
          random1   [2]  BIT STRING OPTIONAL,
          random2   [3]  BIT STRING OPTIONAL }
                                        OPTIONAL,
          password  [2]  OCTET STRING OPTIONAL }

StrongCredentials  ::=  SET {
     certification-path        [0]  CertificationPath OPTIONAL,
     bind-token                [1]  Token }

Token  ::=  SIGNED SEQUENCE {
     algorithm  [0]  AlgorithmIdentifier
     name       [1]  DistinguishedName,
     time       [2]  UTCTime,
     random     [3]  BIT STRING }

Versions ::= BIT STRING (v1988(0))

DirectoryBindResult  ::= DirectoryBindArgument
```

```
DirectoryBindError  ::=  SET {
        versions [0] Versions DEFAULT v1988,
        CHOICE {
            serviceError    [1]  ServiceProblem,
            securityError   [2]  SecurityProblem }}

DirectoryUnbind     ::=  ABSTRACT-UNBIND
        FROM {readPort, searchPort, modifyPort }


-- operations, arguments, and results --

Read ::=  ABSTRACT-OPERATION
        ARGUMENT   ReadArgument
        RESULT     ReadResult
        ERRORS {
            AttributeError, NameError,
            ServiceError, Referral, Abandoned,
            SecurityError }

ReadArgument ::=  OPTIONALLY-SIGNED SET {
        object          [0]  Name,
        selection       [1]  EntryInformationSelection
                                 DEFAULT {},
        COMPONENTS OF CommonArguments }

ReadResult     ::=        OPTIONALLY-SIGNED SET {
        entry           [0]  EntryInformation,
        COMPONENTS OF CommonResults }

Compare  ::=  ABSTRACT-OPERATION
        ARGUMENT        CompareArgument
        RESULT  CompareResult
        ERRORS {
            AttributeError, NameError,
            ServiceError, Referral, Abandoned,
            SecurityError }

CompareArgument   ::=  OPTIONALLY-SIGNED SET {
        object          [0]  Name,
        purported       [1]  AttributeValueAssertion,
        COMPONENTS OF CommonArguments }

CompareResult ::=  OPTIONALLY-SIGNED SET {
        DistinguishedName OPTIONAL,
        matched [0]  BOOLEAN,
        fromEntry [1] BOOLEAN DEFAULT TRUE,
        COMPONENTS OF CommonResults }

Abandon  ::=  ABSTRACT-OPERATION
        ARGUMENT        AbandonArgument
        RESULT  AbandonResult
        ERRORS  {AbandonFailed}

AbandonArgument  ::=  SEQUENCE {
        InvokeID [0]  InvokeID}

AbandonResult ::=  NULL

List  ::=  ABSTRACT-OPERATION
        ARGUMENT        ListArgument
        RESULT          ListResult
        ERRORS {
            AttributeError, NameError,
            ServiceError, Referral, Abandoned,
            SecurityError }

ListArgument   ::=  OPTIONALLY-SIGNED SET {
        object     [0]  Name,
        COMPONENTS OF CommonArguments }
```

```
ListResult      ::=   OPTIONALLY-SIGNED CHOICE{
    listInfo SET    {
    DistinguishedName OPTIONAL
    subordinates [1] SET OF SEQUENCE {
        RelativeDistinguishedName,
        aliasEntry      [0]   BOOLEAN DEFAULT FALSE,
        fromEntry       [1]   BOOLEAN DEFAULT TRUE },
        partialOutcomeQualifier [2] PartialOutcomeQualifier
                                                    OPTIONAL,
        COMPONENTS OF CommonResults},
    uncorrelatedListInfo [0] SET OF
        ListResult }

PartialOutcomeQualifier   ::=   SET {
    limitProblem    [0]   LimitProblem OPTIONAL,
    unexplored      [1]   SET OF
        ContinuationReference OPTIONAL,
    unavailableCriticalExtensions [2] BOOLEAN DEFAULT FALSE }

LimitProblem   ::=   INTEGER {
    timeLimitExceeded   (0),
    sizeLimitExceeded   (1),
    administrativeLimitExceeded (2) }

Search    ::=   ABSTRACT-OPERATION
    ARGUMENT            SearchArgument
    RESULT              SearchResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, Abandoned,
        SecurityError }

SearchArgument ::=  OPTIONALLY-SIGNED SET {
    baseObject      [0]   Name,
    subset          [1]   INTEGER {
        baseObject(0),
        oneLevel(1),
        wholeSubtree(2)} DEFAULT baseObject,
    filter          [2]   Filter DEFAULT and {},
    searchAliases   [3]   BOOLEAN DEFAULT TRUE,
    selection       [4]   EntryInformationSelection DEFAULT {},
    COMPONENTS OF CommonArguments }

SearchResult    ::=   OPTIONALLY-SIGNED
    CHOICE {
    searchInfo SET {
    DistinguishedName OPTIONAL,
    entries  [0]  SET OF EntryInformation,
    partialOutcomeQualifier
        [2] partialOutcomeQualifier OPTIONAL,
    COMPONENTS OF CommonResults },
    uncorrelatedSearchInfo [0] SET OF
        SearchResult }

AddEntry ::=   ABSTRACT-OPERATION
    ARGUMENT        AddEntryArgument
    RESULT      AddEntryResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, SecurityError,
        UpdateError }

AddEntryArgument  ::=   OPTIONALLY-SIGNED SET {
        object      [0]         DistinguishedName,
        entry       [1]         SET OF Attribute,
        COMPONENTS OF CommonArguments}

AddEntryResult ::= NULL
```

```
RemoveEntry ::=          ABSTRACT-OPERATION
    ARGUMENT             RemoveEntryArgument
    RESULT               RemoveEntryResult
    ERRORS {
        NameError,
        ServiceError, Referral, SecurityError,
        UpdateError}

RemoveEntryArgument ::= OPTIONALLY-SIGNED SET {
        object      [0]  DistinguishedName,
        COMPONENTS OF CommonArguments }

RemoveEntryResult ::=   NULL

ModifyEntry ::=          ABSTRACT-OPERATION
    ARGUMENT             ModifyEntryArgument
    RESULT               ModifyEntryResult
    ERRORS {
        AttributeError, NameError,
        ServiceError, Referral, SecurityError,
        UpdateError}

ModifyEntryArgument ::= OPTIONALLY-SIGNED SET {
    object      [0]  DistinguishedName,
    changes     [1]  SEQUENCE OF EntryModification,
    COMPONENTS OF CommonArguments }

ModifyEntryResult ::=   NULL

EntryModification ::=   CHOICE {
    addAttribute     [0] Attribute,
    removeAttribute      [1]  AttributeType,
    addValues            [2]  Attribute,
    removeValues         [3]  Attribute}

ModifyRDN ::=           ABSTRACT-OPERATION
    ARGUMENT             ModifyRDNArgument
    RESULT               ModifyRDNResult
    ERRORS {
        NameError,
        ServiceError, Referral, SecurityError,
        UpdateError }

ModifyRDNArgument ::= OPTIONALLY-SIGNED SET {
    object          [0]      DistinguishedName,
    newRDN          [1]      RelativeDistinguishedName,
    deleteOldRDN    [2]      BOOLEAN DEFAULT FALSE,
    COMPONENTS OF CommonArguments }

ModifyRDNResult ::=          NULL


-- errors and parameters --


Abandoned ::= ABSTRACT-ERROR        -- not literally an "error"

AbandonFailed ::=  ABSTRACT-ERROR
    PARAMETER SET {
        problem          [0]  AbandonProblem,
        operation        [1]  InvokeID}

AbandonProblem ::= INTEGER {
                   noSuchOperation (1),
                   tooLate (2),
                   cannotAbandon (3)}
```

```
AttributeError ::=  ABSTRACT-ERROR
    PARAMETER SET {
        object          [0]  Name,
        problems        [1]  SET OF SEQUENCE {
            problem         [0]  AttributeProblem,
            type            [1]  AttributeType,
            value           [2]  AttributeValue OPTIONAL }}

AttributeProblem    ::=
    INTEGER {
        noSuchAttributeOrValue (1),
        invalidAttributeSyntax (2),
        undefinedAttributeType (3),
        inappropriateMatching (4),
        constraintViolation (5),
        attributeOrValueAlreadyExists (6) }

NameError ::=  ABSTRACT-ERROR
    PARAMETER SET {
        problem  [0]  NameProblem,
        matched  [1]  Name}

NameProblem  ::=  INTEGER {
        noSuchObject (1),
        aliasProblem (2),
        invalidAttributeSyntax (3),
        aliasDereferencingProblem (4)}

Referral  ::=  ABSTRACT-ERROR       -- not literally an "error"
    PARAMETER SET {
        candidate       [0]  ContinuationReference}

SecurityError  ::=  ABSTRACT-ERROR
    PARAMETER SET {
        problem  [0]  SecurityProblem }

SecurityProblem      ::=  INTEGER {
    inappropriateAuthentication (1),
    invalidCredentials (2),
    insufficientAccessRights (3),
    invalidSignature (4),
    protectionRequired (5),
    noInformation (6) }

ServiceError    ::=  ABSTRACT-ERROR
    PARAMETER SET {
        problem  [0]  ServiceProblem }

ServiceProblem ::=  INTEGER {
    busy (1),
    unavailable (2),
    unwillingToPerform (3),
    chainingRequired (4),
    unableToProceed (5),
    invalidReference (6),
    timeLimitExceeded (7),
    administrativeLimitExceeded (8),
    loopDetected (9),
    unavailableCriticalExtension (10),
    outOfScope (11),
    ditError (12) }

UpdateError    ::=  ABSTRACT-ERROR
    PARAMETER SET {
        problem  [0]  UpdateProblem }
```

```
UpdateProblem ::=   INTEGER {
        namingViolation (1),
        objectClassViolation (2),
        notAllowedOnNonLeaf (3),
        notAllowedOnRDN (4),
        entryAlreadyExists (5),
        affectsMultipleDSAs (6),
        objectClassModificationProhibited (7)}

-- common arguments/results --

CommonArguments  ::=   SET {
        [30]  ServiceControls DEFAULT {}
        [29]  SecurityParameters DEFAULT {},
        requestor [28] DistinguishedName OPTIONAL,
        [27]  OperationProgress DEFAULT notStarted,
        aliasedRDNs [26]  INTEGER OPTIONAL,
        extensions [25] SET OF Extension OPTIONAL }

Extension       ::=   SET {
        identifier      [0]   INTEGER,
        critical        [1]   BOOLEAN DEFAULT FALSE,
        item            [2]   ANY DEFINED BY identifier }

CommonResults ::=   SET {
        [30]   SecurityParameters  OPTIONAL,
        performer [29] DistinguishedName OPTIONAL,
        aliasDereferenced  [28] BOOLEAN DEFAULT FALSE}

-- common data types --

ServiceControls  ::=        SET {
        options [0]       BIT STRING {
        preferChaining (0),
        chainingProhibited (1),
        localScope (2),
        dontUseCopy (3),
        dontDereferenceAliases(4)}
              DEFAULT{},

        priority [1] INTEGER {
              low (0),
              medium (1),
              high (2) } DEFAULT medium,

timeLimit [2]   INTEGER OPTIONAL,

sizeLimit  [3]   INTEGER OPTIONAL,

scopeOfReferral [4]   INTEGER {
                    dmd(0),
                    country(1)}
                    OPTIONAL }

EntryInformationSelection       ::= SET {
        attributeTypes
              CHOICE {
                    allAttributes [0]  NULL,
                    select [1] SET OF AttributeType
                    -- empty set implies no attributes
                    -- are requested --}
              DEFAULT allAttributes NULL,

        infoTypes [2] INTEGER {
              attributeTypesOnly (0),
              attributeTypesAndValues (1) } DEFAULT
                          attributeTypesandValues }
```

```
EntryInformation ::=            SEQUENCE {
    DistinguishedName,
    fromEntry BOOLEAN DEFAULT TRUE,
    SET OF CHOICE {
        AttributeType,
        Attribute} OPTIONAL }

Filter      ::=  CHOICE {
    item        [0]  FilterItem,
    and         [1]  SET OF Filter,
    or          [2]  SET OF Filter,
    not         [3]  Filter }

FilterItem      ::=  CHOICE {
    equality            [0]  AttributeValueAssertion,
    substrings          [1]  SEQUENCE {
        type                AttributeType,
        strings             SEQUENCE OF CHOICE {
                            initial     [0]  AttributeValue,
                            any         [1]  AttributeValue,
                            final       [2]  AttributeValue}},
    greaterOrEqual      [2]  AttributeValueAssertion,
        lessOrEqual     [3]  AttributeValueAssertion,
        present         [4]  AttributeType,
    approximateMatch    [5]  AttributeValueAssertion }

SecurityParameters ::=      SET {
    certification-Path  [0]  CertificationPath OPTIONAL,
    name            [1]  DistinguishedName OPTIONAL,
    time            [2]  UTCTime OPTIONAL,
    random          [3]  BIT STRING OPTIONAL,
    target          [4]  ProtectionRequest OPTIONAL }

ProtectionRequest ::=       INTEGER {
                                none(0),
                                signed (1)}
```

ANNEXE B

(à la Recommandation X.511)

Identificateurs d'objet d'annuaire

Cette annexe fait partie de la présente Recommandation.

Elle comprend tous les identificateurs d'objet ASN.1 contenus dans la présente Recommandation sous la forme du module ASN.1 **DirectoryObjectIdentifiers**.

```
DirectoryObjectIdentifiers      {joint-ISO-CCITT ds(5) modules(1)
                                directoryObjectIdentifiers(9)}
DEFINITIONS ::=
BEGIN

EXPORTS
    id-ot-directory, id-ot-dua, id-pt-read, id-pt-search, id-pt-modify;

IMPORTS
    id-ot, id-pt
    FROM UsefulDefinitions  {joint-iso-ccitt ds(5) modules(1),
                            usefulDefinitions(0)}
```

```
-- Objects --

id-ot-directory      OBJECT IDENTIFIER ::=  {id-ot 1}

id-ot-dua            OBJECT IDENTIFIER ::=  {id-ot 2}

-- Port Types --

id-pt-read           OBJECT IDENTIFIER ::=  {id-pt 1}

id-pt-search         OBJECT IDENTIFIER ::=  {id-pt 2}

id-pt-modify         OBJECT IDENTIFIER ::=  {id-pt 3}

END
```

Recommandation X.518

L'ANNUAIRE - PROCEDURES DE FONCTIONNEMENT REPARTI [1)]

(Melbourne, 1988)

SOMMAIRE

---

1) La Recommandation X.518 et la norme ISO 9594-4, "Systèmes de traitement de l'information - Interconnexion des systèmes ouverts - L'annuaire - Procédures de fonctionnement réparti", ont été élaborées en étroite collaboration et sont alignées du point de vue technique.

SECTION 1 - *Introduction*

0 Introduction

0.1 Le présent document, ainsi que les autres de cette série, a été élaboré en vue de faciliter l'interconnexion de systèmes informatiques visant à assurer des services d'annuaire. L'ensemble de tous ces systèmes, avec les informations d'annuaire qu'ils détiennent, peut être considéré comme un tout intégré, appelé *annuaire*. Les informations de l'annuaire, désignées collectivement comme la Base de Données d'annuaire (DIB) sont normalement utilisées pour faciliter la communication entre, avec ou à propos d'objets tels que des entités d'application OSI, des personnes, des terminaux et des listes de diffusion.

0.2 L'annuaire joue un rôle significatif dans l'Interconnexion des Systèmes Ouverts (OSI) dont l'objectif est de permettre, au prix d'un minimum d'accords techniques en dehors des normes d'interconnexion proprement dites, d'interconnecter des équipements informatiques:

- de constructeurs différents;
- gérés de façons différentes;
- de niveaux de complexité différents;
- d'âges différents.

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

1 Objet et domaine d'application

1.1 La présente Recommandation spécifie le comportement des DSA qui participent à la mise en oeuvre répartie de l'annuaire. Le comportement autorisé a été conçu pour assurer un service cohérent, compte tenu d'une large répartition de la DIB entre de nombreux DSA.

1.2 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

Recommandation X.200 - Modèle de référence pour l'interconnexion des systèmes ouverts pour les applications du CCITT

Recommandation X.208 - Interconnexion des Systèmes Ouverts - Spécification de la notation 1 de la syntaxe abstraite (ASN.1)

Recommandation X.500 - L'annuaire - Aperçu des concepts, modèles et services

Recommandation X.501 - L'annuaire - Modèles

Recommandation X.511 - L'annuaire - Définition du service abstrait

Recommandation X.519 - L'annuaire - Spécifications du protocole

Recommandation X.520 - L'annuaire - Types d'attributs sélectionnés

Recommandation X.521 - L'annuaire - Catégories d'objets sélectionnées

Recommandation X.407 - Systèmes de messagerie - Conventions des définitions du service abstrait

3 Définitions

Les définitions de ce paragraphe utilisent les abréviations définies au § 4.

3.1 *Définitions relatives au modèle de référence OSI*

La présente Recommandation utilise le terme suivant défini dans la Recommandation X.200:

a) *appellation d'entité d'application.*

3.2 *Définitions de base relatives à l'annuaire*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.500:

a) *(l')annuaire*;

b) *Base de Données d'annuaire.*

3.3 *Définitions relatives au modèle de l'annuaire*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.501:

a) *point d'accès*;

b) *pseudonyme*;

c) *nom spécifique*;

d) *Arbre d'Information de l'annuaire*;

e) *Agent du Système de l'annuaire*;

f) *Agent d'Utilisateur de l'annuaire*;

g) *nom spécifique relatif.*

3.4 *Conventions des définitions du service abstrait*

La présente Recommandation utilise les termes suivants définis dans la Recommandation X.407:

a) *erreur abstraite*;

b) *opération abstraite*;

c) *résultat.*

3.5 *Définitions relatives aux Opérations Réparties*

La présente Recommandation utilise les termes suivants, tels que définis ci-après:

a) *chaînage:* mode d'interaction utilisé facultativement par un DSA qui ne peut pas effectuer une opération lui-même. Le DSA chaîne en appelant une opération d'un autre DSA, puis en retournant le résultat au demandeur initial;

b) *préfixe de contexte:* séquence de RDN allant de la Racine du DIT, jusqu'au noeud initial d'un contexte de dénomination, et correspondant au nom spécifique de ce noeud;

c) *référence croisée:* référence de connaissance contenant des informations sur le DSA qui détient une entrée. Elle est utilisée à des fins d'optimisation. L'entrée ne doit avoir ni relation supérieure ni relation subordonnée;

d) *fragment de DIB:* partie de la DIB détenue par un DSA, et comprenant un ou plusieurs contextes de dénomination;

e) *résolution répartie du nom:* processus de résolution du nom par plusieurs DSA;

f) *référence interne:* référence de connaissance contenant un pointeur interne vers une entrée contenue dans le même DSA;

g) *informations de connaissance:* informations qu'un DSA possède sur des entrées qu'il détient et sur la façon de localiser d'autres entrées dans l'annuaire;

h) *référence de connaissance:* connaissance qui associe, directement ou indirectement, une entrée du DIT au DSA où elle se trouve;

i) *arbre de connaissances:* modèle conceptuel des informations de connaissance que possède un DSA pour lui permettre d'effectuer la résolution répartie du nom;

j) *multireport:* mode d'interaction utilisable facultativement par un DSA qui ne peut pas exécuter une opération lui-même. Le DSA *multireporte* l'opération, c'est-à-dire lance la même opération auprès de plusieurs autres DSA (en série ou en parallèle) et communique un résultat approprié au demandeur initial;

k) *résolution du nom:* procédure permettant de localiser une entrée en comparant successivement chaque RDN d'un nom visé avec un noeud du DIT;

l) *contexte de dénomination:* sous-arbre partiel du DIT, commençant à un noeud et s'étendant aux niveaux inférieurs jusqu'à des noeuds feuilles ou non-feuilles. Ces noeuds constituent

la frontière du contexte de dénomination. Les noeuds non-feuilles et appartenant à la frontière indiquent le début d'autres contextes de dénomination;

m) *référence subordonnée non spécifique:* référence de connaissance contenant des informations sur le DSA qui détient une ou plusieurs entrées subordonnées non spécifiées;

n) *avancement de l'opération:* ensemble de valeurs qui précise dans quelle mesure la résolution du nom a été réalisée;

o) *chemin de référence:* séquence continue de références de connaissance;

p) *renvoi de référence:* résultat pouvant être retourné par un DSA qui ne peut pas effectuer une opération lui-même; ce résultat identifie un ou plusieurs autres DSA plus aptes à effectuer cette opération;

q) *décomposition de demande:* décomposition d'une demande en sous-demandes qui accomplissent chacune une partie de l'opération répartie;

r) *contexte de racine:* contexte de dénomination pour le noeud dont le nom comprend la séquence vide de RDN;

s) *référence subordonnée:* référence de connaissance contenant des informations sur le DSA qui détient une entrée subordonnée spécifique;

t) *sous-demande:* demande engendrée par une décomposition de demandes;

u) *référence supérieure:* référence de connaissance contenant des informations sur le DSA qui détient une entrée supérieure.

4 Abréviations

Les abréviations suivantes sont utilisées dans la présente Recommandation:

DIB Base de Données d'annuaire

DIT Arbre d'Information de l'annuaire

DSA Agent du Système de l'annuaire

DUA Agent d'Utilisateur de l'annuaire

RDN Nom Spécifique Relatif.

5 Notation

Les notations utilisées dans ce paragraphe sont définies comme suit:

a) la notation de syntaxe des données, le codage et la notation des macros sont définis dans la Recommandation X.208;

b) les notations concernant les modèles abstraits et les services abstraits sont définies dans la Recommandation X.407.

SECTION 2 - *Présentation générale*

6 Présentation générale

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 Recommandation X.511.

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 du Système de l'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 DSA et la coopération avec eux, en vue de fournir le service abstrait dans un environnement réparti.

La présente Recommandation spécifie l'affinage de l'objet abstrait annuaire, cet affinage étant exprimé en termes d'ensemble d'un ou de plusieurs objets DSA, qui constituent collectivement le service réparti d'annuaire. Ceci implique l'identification et la spécification des ports des DSA locales à l'objet annuaire. Pour chacun de ces ports, la présente Recommandation spécifie le service abstrait associé et ses procédures.

En outre, la présente Recommandation spécifie les modes de répartition autorisés de la DIB entre un ou plusieurs DSA. Dans le cas limite où la 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 ou de navigation sont spécifiés: ils garantissent que l'ensemble de la DIB est potentiellement accessible à tous les DSA qui comportent des entrées constitutives de l'annuaire.

De plus, des interactions de traitement des demandes sont spécifiées: elles permettent le contrôle, par les utilisateurs de l'annuaire, de 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 et concernant des informations contenues dans d'autres DSA, d'interroger directement d'autres DSA (chaînage/multireport) ou s'il doit fournir en réponse des informations sur d'autres DSA qui pourraient poursuivre le traitement de la demande (renvoi de référence).

En général, la décision d'un DSA d'opérer un chaînage, un multireport 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.

La présente Recommandation spécifie les procédures appropriées à exécuter par les DSA, en réponse à des demandes adressées à l'annuaire réparti, compte tenu du fait qu'en général l'annuaire sera effectivement réparti, et que les demandes qui lui sont adressées seront satisfaites par un nombre arbitraire de DSA coopérants, qui pourront opérer arbitrairement des chaînages, des multireports ou des renvois de référence, selon les critères ci-dessus. Ces procédures garantissent que les utilisateurs du service annuaire réparti en auront une perception conviviale et cohérente.

SECTION 3 - *Modèles de l'annuaire réparti*

7 Modèle du système de l'annuaire réparti

Le Service Abstrait d'annuaire, tel que défini dans la Recommandation X.511, modélise l'annuaire comme un objet qui fournit un ensemble de services d'annuaire à ses utilisateurs. Les services de l'annuaire sont modélisés en terme de ports, chaque port fournissant un ensemble particulier de services d'annuaire. Les utilisateurs de l'annuaire accèdent à ces services par des points d'accès. L'annuaire peut avoir un ou plusieurs points d'accès dont chacun est caractérisé par les services qu'il fournit et le mode d'interaction utilisé pour fournir ces services.

Ce paragraphe concerne la structure interne de l'objet annuaire: il définit ses objets constitutifs, leurs ports et facilite par là la spécification d'un service d'annuaire réparti.

La figure 1/X.518 représente le modèle d'annuaire réparti qui sera utilisé comme base de spécification des aspects répartis de l'annuaire. Elle représente l'objet annuaire comme constitué d'un ensemble d'un ou de plusieurs objets DSA.

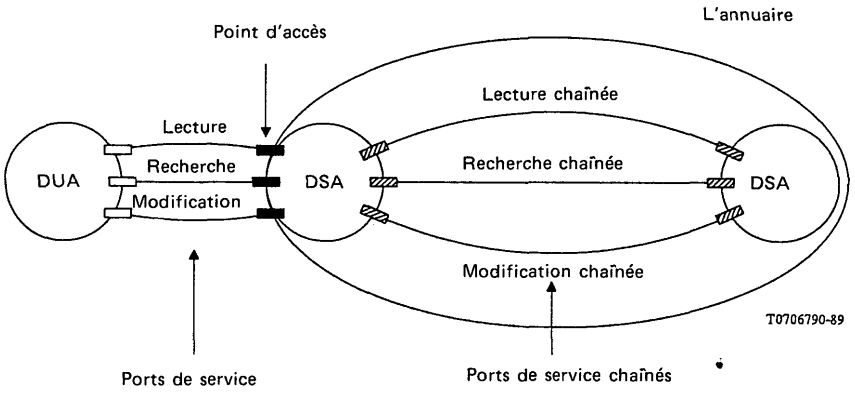


FIGURE 1/X.518

Objets du modèle d'annuaire réparti

Les objets DSA sont spécifiés en détail dans les paragraphes suivants de la présente Recommandation. Ce paragraphe établit simplement un certain nombre de leurs caractéristiques à titre d'introduction et pour préciser les relations entre la présente Recommandation et les autres Recommandations de cette série.

Les objets DSA sont définis en vue de permettre la répartition de la DIB et de permettre à plusieurs DSA répartis physiquement d'interagir, de façon prescrite et coopérative, pour fournir des services de l'annuaire à ses utilisateurs (DUA).

Les objets DSA sont, de même que l'objet annuaire, caractérisés par leurs ports visibles de l'extérieur. Les ports associés à un objet DSA sont de deux types: les ports de service et les ports de service chaînés.

Les ports de service d'un objet DSA sont identiques à ceux de l'objet annuaire, à savoir les ports de **Lecture**, **Recherche** et **Modification**. La figure 1/X.518 montre que les ports de service associés à un objet DSA constituent un point d'accès à travers lequel des services d'annuaire sont mis à disposition.

La spécification détaillée des ports de service de **lecture**, **recherche** et **modification** est donnée dans la Recommandation X.511. (Les spécifications du protocole concernant les éléments de service d'application OSI correspondants, qui dérivent des spécifications de ces ports, se trouvent dans la Recommandation X.519.)

En plus des ports de service de l'objet DSA qui permettent l'accès à l'objet annuaire, un second ensemble de ports est défini: les ports de service chaînés. Ils permettent une communication entre DSA, de manière à pouvoir réaliser le Service Abstrait d'annuaire dans un environnement réparti.

Les ports de service chaînés et les opérations assurées par leur intermédiaire sont en correspondance directe avec les ports de service de noms similaires: ce sont, respectivement, les ports de **lecture-Chaînée**, **recherche-Chaînée** et **modification-Chaînée**.

Le processus de spécification des objets constitutifs d'un objet abstrait est appelé "affinage". La spécification de l'affinage de l'objet annuaire en ses composants (les DSA) et la spécification du service abstrait fourni par chacun d'eux (le service abstrait d'un DSA) sont données dans la section 4 de la présente Recommandation. (La spécification du protocole des éléments de service d'application OSI correspondants, qui dérive de la définition des ports chaînés, se trouve dans la Recommandation X.519.)

8 Modèle des interactions entre les DSA

Une caractéristique de base de l'annuaire est qu'un utilisateur, étant donné une DIB répartie, doit être potentiellement capable d'obtenir satisfaction pour toute demande de service (compte tenu des politiques de sécurité, de contrôle d'accès et du gestionnaire), quel que soit le point d'accès auquel cette demande est présentée. Pour répondre à cet impératif, il est nécessaire que tous les DSA impliqués dans la réponse à une demande de service particulière, aient une certaine connaissance (telle que spécifiée dans le § 10 de la présente Recommandation) de l'endroit où l'information demandée est située, retournent cette connaissance au demandeur, ou essayent, pour le compte de celui-ci, de faire satisfaire sa demande. (Le demandeur peut être un DUA ou un autre DSA: dans le dernier cas, les deux DSA doivent avoir un port chaîné.)

Pour répondre à ces impératifs, trois modes d'interaction de DSA sont définis, à savoir le "chaînage", le "multireport" et le "renvoi de référence". Le "chaînage" et le "multireport" sont définis pour répondre au dernier de ces impératifs, alors que les "renvois de référence" concernent le premier.

8.1 *Chaînage*

Ce mode d'interaction illustré à la figure 2/X.518 peut être utilisé par un DSA, pour communiquer une demande à un autre DSA, quand le premier a connaissance des contextes de dénomination du second. Le chaînage peut être utilisé pour contacter un seul DSA désigné dans une référence croisée, une référence subordonnée ou une référence supérieure. Le multireport est une forme de chaînage, décrit au § 8.2.

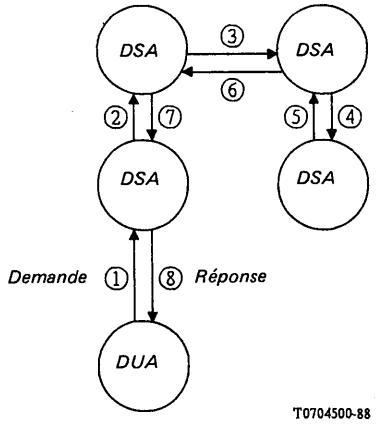


FIGURE 2/X.518

Mode chaînage

Remarque - Sur la figure 2/X.518, l'ordre des interactions est défini par les numéros associés aux flèches.

8.2 *Multireport*

Ce mode d'interaction (illustré aux figures 3a/X.518 et 3b/X.518) peut être utilisé par un DSA pour chaîner une demande identique, en parallèle (a) ou séquentiellement (b), à un ou à plusieurs autres DSA, quand le premier ne connaît pas les contextes de dénomination complets des autres DSA. Le multireport est uniquement utilisé par un DSA pour contacter d'autres DSA désignés dans une référence subordonnée non spécifique. Une demande identique est communiquée à chacun des DSA. Normalement, pendant la résolution du nom, seul un des DSA sera capable de continuer le traitement de l'opération distante, tous les autres retournant **l'erreur de service incapable de traiter**. Néanmoins, pendant la phase d'évaluation des opérations de recherche et de listage, tous les DSA d'une référence subordonnée non spécifique doivent pouvoir continuer à traiter la demande.

Remarque - Sur les figures 3a/X.518 et 3b/X.518, l'ordre des interactions est défini par les numéros associés aux flèches.

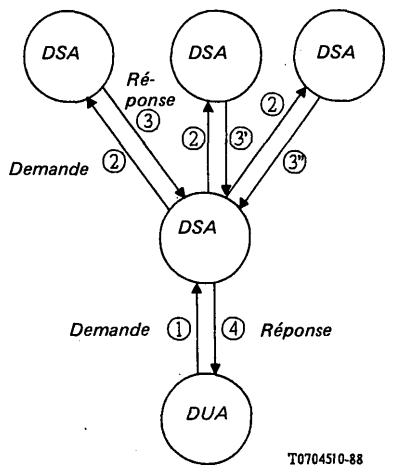


FIGURE 3a/X.518

Mode multireport

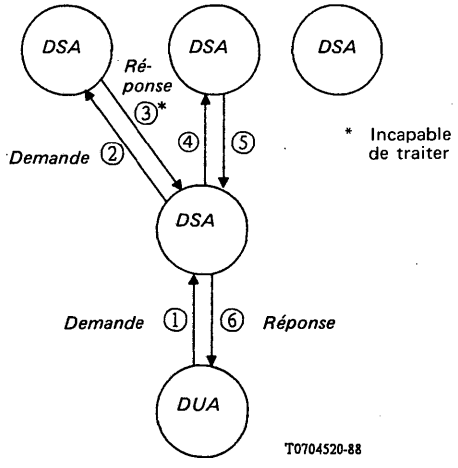


FIGURE 3b/X.518

Mode multireport

8.3 *Renvoi de référence*

Un renvoi de référence (illustré aux figures 4a/X.518 et 4b/X.518) est retourné par un DSA dans sa réponse à une demande qu'il a été prié d'effectuer, par un DUA ou par un autre DSA (auquel cas les deux DSA doivent avoir un port de service chaîné). 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, une référence supérieure, subordonnée, croisée, ou subordonnée non spécifique.

Le DSA (figure 4a/X.518) recevant le renvoi de référence peut utiliser la référence de connaissance qui y est contenue pour chaîner ou multireporter (selon le type de référence) l'opération 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, comme renvoi de référence. Un DUA (figure 4b/X.518) recevant un renvoi de référence peut l'utiliser pour contacter un ou plusieurs autres DSA afin de faire progresser la demande.

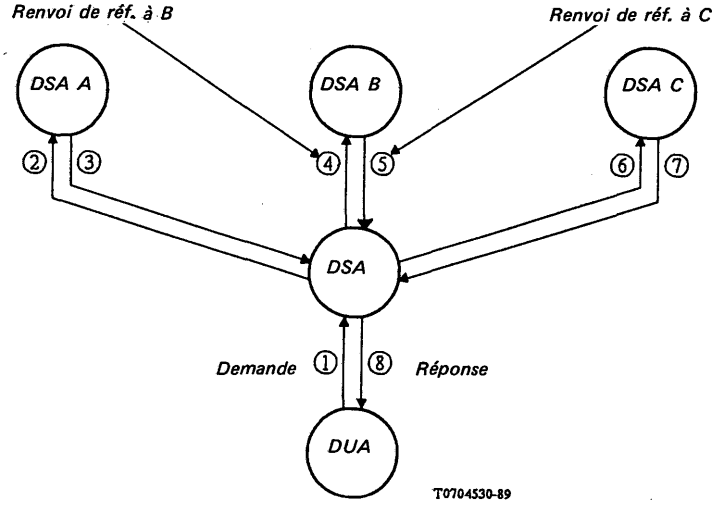


FIGURE 4a/X.518

Mode renvoi de référence - DSA avec ports chaînés

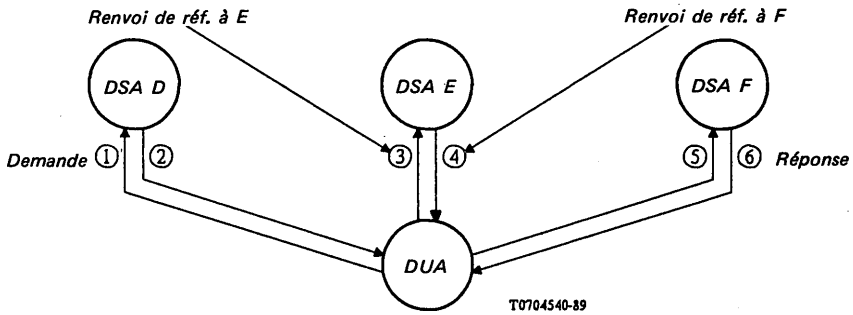


FIGURE 4b/X.518

Mode renvoi de référence - Demandes émises par un DUA - DSA sans port chaîné

Remarque - Sur les figures 4a/X.518 et 4b/X.518, l'ordre des interactions est défini par les numéros associés aux flèches.

8.4 *Détermination du mode*

Si un DSA ne peut pas satisfaire entièrement lui-même une demande, il doit chaîner ou multireporter 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 des commandes de service, auquel cas le DSA doit retourner un renvoi de référence ou une **erreur de service chaînageRequis** (au choix), ou

b) le DSA a des raisons d'ordre gestionnaire, opérationnelles ou techniques de préférer ne pas chaîner, auquel cas le DSA doit retourner un renvoi de référence.

Remarque 1 - Une "raison technique" interdisant le chaînage et le multireport est le fait que le DSA identifié dans la référence de connaissance n'a pas de port de service chaîné.

Remarque 2 - Si la commande de service **portée-Locale** est positionnée, alors le DSA (ou le DMD) doit soit résoudre la demande, soit retourner une erreur.

Remarque 3 - Si l'utilisateur préfère les renvois de référence, il doit positionner la commande de service **chaînage-Interdit**.

9 Répartition de l'annuaire

Ce paragraphe définit les principes selon lesquels la DIB peut être répartie.

Chaque entrée de la DIB est gérée par un Gestionnaire de DSA et un seul, qui est dit avoir autorité de gestion sur cette entrée. La tenue à jour et la gestion d'une entrée doivent s'exercer dans un DSA géré par l'autorité de gestion de cette entrée.

Bien que l'annuaire n'assure aucune prise en charge de la copie des entrées, il est néanmoins possible de réaliser des copies de deux façons:

- Des copies d'une entrée peuvent être enregistrées dans d'autres DSA dans le cadre d'un accord bilatéral. Les moyens par lesquels ces copies sont tenues à jour et gérées sont fonction de cet accord bilatéral et ne sont pas définis dans la présente Recommandation.
- Des copies d'une entrée peuvent être obtenues en enregistrant (localement et dynamiquement) une copie d'une entrée qui résulte d'une demande.

Remarque - L'acquisition d'entrées du type "cache" est soumise à la commande d'accès.

L'initiateur de la demande est informé (via **fromCopy**) du fait que les informations retournées en réponse à une demande proviennent d'une entrée copiée ou non. Une commande de service **dontUseCopy** est définie, qui permet à l'utilisateur d'interdire l'emploi d'entrées copiées.

Chaque DSA de l'annuaire détient un *fragment* de la DIB. Le fragment de DIB contenu par un DSA est décrit en termes de DIT et comprend un ou plusieurs contextes de dénomination. Un *contexte de dénomination* est un sous-arbre partiel du DIT défini comme commençant à un noeud et s'étendant aux

niveaux inférieurs jusqu'à des noeuds feuilles ou non-feuilles. De tels noeuds constituent la frontière du domaine de dénomination. Les subordonnées des noeuds non-feuilles appartenant à la frontière indiquent le début d'autres contextes de dénomination.

Il est possible pour un gestionnaire de DSA d'avoir autorité de gestion sur plusieurs contextes de dénomination disjoints. Pour chaque contexte de dénomination sur lequel il a autorité de gestion, un DSA doit logiquement détenir la séquence des RDN qui conduisent de la racine du DIT au noeud initial du sous-arbre comprenant le contexte de dénomination. Cette séquence de RDN est appelée *préfixe de contexte.*

Un gestionnaire de DSA peut déléguer à un autre DSA l'autorité de gestion sur tout subordonné immédiat d'une entrée détenue localement. Un DSA qui délègue son autorité est appelé *DSA supérieur*, et le contexte qui détient l'entrée supérieure de celle pour laquelle l'autorité de gestion a été déléguée, est appelé *contexte de dénomination supérieur.* La délégation d'autorité de gestion commence à la racine et descend dans le DIT: autrement dit, elle ne peut aller que d'une entrée vers ses subordonnées.

La figure 5/X.518 représente un DIT hypothétique, divisé logiquement en cinq contextes de dénomination (appelés A, B, C, D et E) qui sont physiquement répartis entre trois DSA (DSA1, DSA2 et DSA3).

On peut voir sur cet exemple que les contextes de dénomination de DSA particuliers peuvent être configurés pour répondre à une large gamme de caractéristiques opérationnelles. Certains DSA peuvent être configurés pour contenir les entrées qui représentent les domaines de dénomination de plus haut niveau de certaines parties logiques de la DIB, par exemple la structure organisationnelle d'une grande compagnie, mais pas nécessairement toutes les entrées subordonnées. Autre possibilité: des DSA peuvent être configurés pour contenir uniquement les contextes de dénomination représentant essentiellement des entrées feuilles.

D'après les définitions ci-dessus, le cas limite pour un contexte de dénomination peut être une entrée unique ou l'ensemble du DIT.

Bien qu'il soit possible de faire correspondre à la structure logique du DIT n'importe quelle configuration physique des DSA, la tâche de localisation et de gestion des informations sera simplifiée si les DSA sont configurés pour contenir un petit nombre de contextes de dénomination.

Pour qu'un DUA puisse commencer à traiter une demande, il doit contenir certaines informations, spécifiquement l'adresse de présentation, sur au moins un DSA qu'il peut contacter initialement. La façon dont il se procure et conserve cette information est une question locale.

Au cours du processus de modification d'entrée, il est possible que l'annuaire devienne incohérent. Cela risquera notamment de se produire si une modification implique des pseudonymes ou des objets pseudonymes qui peuvent être dans des DSA différents. L'incohérence doit être corrigée par une action spécifique du gestionnaire, par exemple pour supprimer des pseudonymes si les objets pseudonymes correspondants ont été supprimés. L'annuaire continue de fonctionner durant cette période d'incohérence.

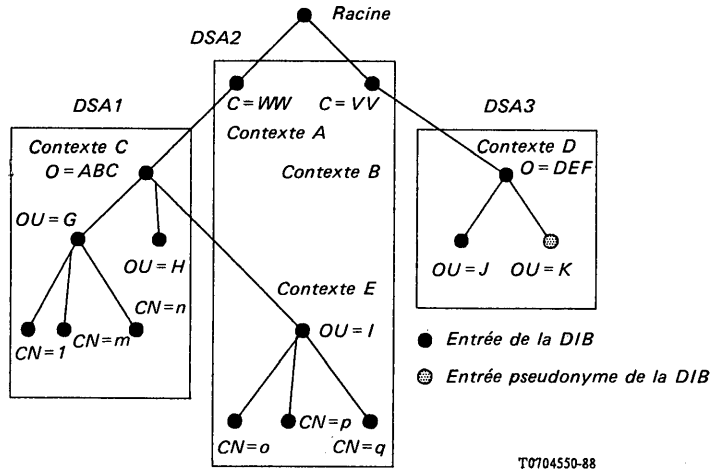


FIGURE 5/X.518

DIT hypothétique

Remarque - La Racine n'est détenue par aucun DSA, mais une certaine indication doit figurer au niveau local pour distinguer les noeuds (par exemple: C=VV, C=WW) qui sont des subordonnés immédiats de la Racine.

10 Connaissances

La DIB est potentiellement répartie entre plusieurs DSA dont chacun contient un fragment de DIB; les principes qui régissent la répartition de la DIB sont spécifiés au § 9 de la présente Recommandation.

Une caractéristique impérative de l'annuaire est que, pour certains modes d'interaction avec l'utilisateur, la répartition de cet annuaire est rendue transparente, donnant ainsi l'impression que l'ensemble de la DIB est à l'intérieur de chacun des DSA.

Pour assurer les caractéristiques opérationnelles décrites ci-dessus, il est nécessaire que chaque DSA contenant un fragment de la DIB soit capable d'identifier et, éventuellement, d'interagir avec d'autres fragments de cette DIB détenus par d'autres DSA.

Ce paragraphe définit la connaissance comme étant la base de la correspondance entre un nom et son emplacement dans un fragment du DIT.

Conceptuellement, les DSA contiennent deux types d'informations:

a) les Données d'annuaire;

b) les Informations de Connaissance.

Les *Données d'annuaire* sont constituées par l'ensemble des entrées des Contextes de Dénomination sur lesquels le Gestionnaire d'un DSA a Autorité de Gestion.

Les *Informations de Connaissance* comprennent le(s) contexte(s) de Dénomination d'un DSA particulier et indiquent comment il se situe dans la hiérarchie générale du DIT. La Résolution du Nom, processus de détermination du DSA qui a Autorité de Gestion sur une certaine entrée étant donné le nom de cette entrée, est fondée sur les informations de connaissance.

Un *Préfixe de Contexte* est la séquence de RDN menant de la Racine du DIT au premier noeud d'un contexte de dénomination; il correspond au nom spécifique de ce noeud.

Un *Contexte de Dénomination* comprend un ensemble de références de connaissance et un Préfixe de Contexte. Un Contexte de Dénomination doit contenir les références de connaissance suivantes et celles-là seulement:

- Toutes les références internes qui définissent la structure interne de la partie du DIT incluse dans le Contexte de Dénomination.
- Toutes les références subordonnées et subordonnées non spécifiques des autres Contextes de Dénomination.

10.1 *Références minimales de connaissance*

L'annuaire permet l'accès de chaque entrée quel que soit l'endroit où une demande est générée.

Pour ce faire, chaque DSA doit au moins tenir à jour les références de connaissance suivantes:

- des *références subordonnées* définies au § 10.3.2 et/ou des *références subordonnées non spécifiques* définies au § 10.3.5; et
- des *références supérieures* définies au § 10.3.3.

Il est alors possible de définir un *chemin de référence* en tant que séquence continue de références de connaissance, pour tous les contextes de dénomination de l'annuaire.

A titre facultatif, des références croisées, définies au § 10.3.4, peuvent faire partie d'un chemin de référence en vue d'optimiser les performances.

10.2 *Contexte racine*

Du fait de l'autonomie des différents pays ou organisations transnationales, il n'existe probablement pas de DSA "unique" contenant le contexte racine. Les fonctions d'un "DSA racine", en ce qui concerne le processus de résolution du nom, doivent être assurées par les DSA qui ont autorité de gestion sur les contextes de dénomination immédiatement subordonnés à la racine. Ces DSA sont appelés *DSA de Premier Niveau.* Chaque DSA de Premier Niveau doit être capable de simuler les fonctions du

"DSA racine". Ceci nécessite une connaissance intégrale du contexte de dénomination racine. Le contexte racine est reproduit dans chaque DSA de Premier Niveau, et doit donc être géré en commun par les autorités de gestion autonomes de premier niveau. Des procédures de gestion doivent être déterminées par des accords multilatéraux, qui n'entrent pas dans le cadre de la présente Recommandation.

- Chaque DSA de premier niveau doit contenir le contexte racine, ce qui implique un chemin de référence vers chaque autre DSA de premier niveau.
- Chaque DSA qui n'est pas de premier niveau doit avoir une référence supérieure, ce qui implique un chemin de référence vers tout DSA de premier niveau.

10.3 *Références de connaissance*

Les connaissances possédées par un DSA sont définies en termes d'ensemble d'une ou de plusieurs références de connaissance, où chaque référence associe, directement ou indirectement, des entrées de la DIB aux DSA qui contiennent ces entrées.

Pour satisfaire l'obligation d'être capable d'atteindre toutes les entrées de la DIB à partir de tous les DSA, chaque DSA est obligé d'avoir connaissance des entrées qu'il détient lui-même, ainsi que de leurs subordonnées et, éventuellement, de leurs supérieures. Ceci conduit à définir les types suivants de références de connaissance:

- références internes;
- références subordonnées;
- références supérieures;
- références subordonnées non spécifiques.

En outre, les types de références optionnels suivants sont définis à des fins d'optimisation:

- références croisées.

Dans le cas où l'ensemble des références de connaissance associées à un DSA particulier contient uniquement des références internes, ce DSA n'a pas connaissance d'autres DSA, et la DIB est donc centralisée.

10.3.1 *Références internes*

Une *référence interne* comprend:

- le RDN correspondant à une entrée de la DIB;
- un pointeur interne situant l'enregistrement de l'entrée dans la DIB locale. (La spécification de ce pointeur n'entre pas dans le cadre de la présente Recommandation.)

Toutes les entrées sur lesquelles un DSA a Autorité de Gestion sont représentées par des références internes dans les informations de connaissance de ce DSA.

10.3.2 *Références subordonnées*

Une *référence subordonnée* comprend:

- un RDN correspondant à une entrée de la DIB subordonnée immédiate;
- le Point d'Accès du DSA auquel l'Autorité de Gestion sur cette entrée a été déléguée.

Toutes les entrées subordonnées détenues par un autre DSA, auxquelles il a délégué son Autorité de Gestion, doivent être représentées par des références subordonnées (ou des références subordonnées non spécifiques, telles que décrites au § 10.3.5).

10.3.3 *Références supérieures*

Une *référence supérieure* comprend:

- le point d'accès d'un DSA.

Chaque DSA qui n'est pas de premier niveau, tient à jour une référence supérieure. La référence supérieure doit faire partie d'un chemin de référence vers la racine. A moins d'employer une méthode autre que la norme à cette fin, par exemple dans un DMD, on pourra effectuer cette opération en se rapportant à un DSA qui contient un contexte de dénomination dont le préfixe de contexte compte moins de RDN que le préfixe comportant le moins de RDN détenu par ce DSA.

Si un nouveau DSA qui n'est pas de premier niveau, est introduit, il doit avoir des connaissances initiales minimales, qui sont représentées par la référence supérieure. Toute

connaissance supplémentaire sera ajoutée par des références subordonnées ou des références croisées (comme décrit au § 10.3.4). Si un nouveau DSA de premier niveau est introduit, il doit acquérir le contexte racine et aviser tous les autres DSA de premier niveau. La façon dont ceci est réalisé n'entre pas dans le cadre de la présente Recommandation.

10.3.4 *Références croisées*

Une *référence croisée* comprend:

- un Préfixe de Contexte;
- le Point d'Accès d'un DSA qui a Autorité de Gestion sur ce Contexte de Dénomination.

Ce type de référence est facultatif et sert à optimiser la Résolution du Nom. Un DSA peut contenir n'importe quel nombre de références croisées (y compris zéro).

10.3.5 *Références subordonnées non spécifiques*

Une *référence subordonnée non spécifique* comprend:

- le Point d'Accès d'un DSA qui contient un ou plusieurs Contextes de Dénomination immédiatement subordonnés.

Ce type de référence est facultatif et permet de traiter le cas dans lequel on sait qu'un DSA contient certaines entrées subordonnées, mais que les RDN spécifiques de ces entrées ne sont pas connus.

Pour chacun de ces contextes de dénomination, un DSA peut contenir n'importe quel nombre (y compris zéro) de références subordonnées non spécifiques, qui seront évaluées si toutes les références internes et subordonnées spécifiques ont été utilisées. Les DSA auxquels donne accès une référence non spécifique doivent être capables de résoudre directement la demande (qu'il en résulte un succès ou un échec). Dans le cas d'un échec, une **erreur-Service** signalant un problème **incapable-Traiter** est retournée au demandeur.

10.4 *Gestion des connaissances*

Pour exploiter un annuaire largement réparti, avec un niveau acceptable de cohérence et de performances, des procédures sont nécessaires pour tenir à jour et étendre les connaissances détenues par chaque DSA. Les mêmes procédures sont appropriées pour la création des connaissances initiales.

Les connaissances peuvent être tenues à jour par:

a) Le DSA ou son autorité de gestion, en répercutant les modifications des connaissances aux DSA détenant tout type de références vers lui, chaque fois que des modifications de ce DSA rendent non valides ces références. C'est la seule façon dont des références supérieures, subordonnées et subordonnées non spécifiques peuvent être tenues à jour.

b) Des DSA demandant et obtenant des références croisées pour améliorer les performances via des opérations normales d'annuaire.

La présente Recommandation ne définit aucune procédure de répercussion des modifications des connaissances, telles que décrites en a). A cette fin, des accords bilatéraux doivent être conclus au niveau local.

10.4.1 *Demande de références croisées*

Pour améliorer les performances du Système de l'annuaire, l'ensemble local de références croisées peut être étendu par l'utilisation des Opérations normales d'annuaire. Si un DSA a un port chaîné, il peut demander à un autre DSA (qui doit également avoir un port chaîné) de retourner les références de connaissance qui contiennent des informations sur l'emplacement des contextes de dénomination relatifs au nom d'objet cible d'une opération normale d'annuaire.

Si le composant **références-Croisées-En-Retour** de l'**Argument OpérationsRéparties** est mis à **TRUE**, le composant **références-Croisées** du **Résultat OpérationsRéparties** peut figurer, et comprend une séquence d'éléments de références croisées.

Si un DSA n'est pas capable de chaîner une demande au DSA suivant, un renvoi de référence est retourné au DSA initial. Si le composant **référence-Croisée-En-Retour** de l'argument de chaînage 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 **préfixe-Contexte** ne figure pas si le renvoi de référence est fondé sur une référence subordonnée non spécifique. La référence croisée retournée par un renvoi de référence est uniquement fondée sur les connaissances détenues par le DSA qui génère le renvoi de référence.

Dans les deux cas (résultat du chaînage et renvoi de référence), une autorité de gestion peut, par le biais de son DSA, choisir d'ignorer la demande de retour de références croisées.

10.4.2 *Incohérences des connaissances*

L'annuaire doit assurer des mécanismes de vérification des connaissances, afin de garantir un certain niveau de cohérence des connaissances.

10.4.2.1 *Détection des incohérences des connaissances*

Les types d'incohérence et leur détection dépendent du type de référence de connaissance:

- *Références croisées et subordonnées:*

Ce type de référence est non valide si le DSA référencé n'a pas de contexte de dénomination local dont le préfixe est contenu dans la référence. Cette incohérence sera détectée durant la détermination du contexte de dénomination initial du processus de résolution du nom, d'après les composants avancement de l'opération et type de référence de l'**Argument OpérationsRéparties.**

- *Références subordonnées non spécifiques:*

Ce type de référence n'est pas valide si le DSA référencé n'a pas de contexte de dénomination local dont le préfixe de contexte immédiatement supérieur est contenu dans la référence, c'est-à-dire que la référence contient le préfixe du contexte local de ce DSA moins le dernier RDN. La vérification de cohérence est appliquée comme indiqué ci-dessus.

- *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 Recommandation.

Remarque - Il n'est pas toujours possible de détecter une référence supérieure non valide.

10.4.2.2 *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 **erreur-Service**, avec un problème de **référence-Non-Valide.**

Si un DSA retourne un renvoi de référence fondé sur une référence de connaissance non valide, une **erreur-Service** sera retournée au demandeur avec un problème de **référence-Non-Valide**, s'il utilise le renvoi de référence. La façon dont l'état d'erreur est répercuté vers le DSA qui enregistre la référence non valide n'entre pas dans le cadre de la présente Recommandation.

10.4.2.3 *Traitement des références de connaissances incohérentes*

Quand un DSA a détecté une référence non valide, il doit essayer de rétablir la cohérence des connaissances. Par exemple, ceci peut être réalisé en détectant simplement une référence croisée non valide, ou en la remplaçant par une référence croisée correcte, qui peut être obtenue en utilisant les mécanismes de **demande-Références-Croisées.**

La façon dont un DSA traite effectivement les références non valides est une question locale, qui n'entre pas dans le cadre de la présente Recommandation.

SECTION 4 - *Service abstrait des DSA*

11 Aperçu du service abstrait des DSA

11.1 Le service abstrait d'annuaire est entièrement décrit dans la Recommandation X.511. Quand un tel service est fourni dans un environnement réparti, selon le modèle décrit au § 7 de la présente Recommandation, il peut être considéré comme fourni au moyen d'un ensemble de DSA. Ceci est représenté sur la figure 1/X.518.

11.2 Pour décrire ce modèle, l'affinage de l'objet **annuaire** en ses composants objets **dsa** peut être exprimé comme suit:

```
DirectoryRefinement      ::=  REFINE directory AS
    dsa                       RECURRING
                              readPort              [S]  VISIBLE
                              searchPort            [S]  VISIBLE
                              modifyPort            [S]  VISIBLE
                              chainedReadPort       PAIRED with dsa
                              chainedSearchPort     PAIRED with dsa
                              chainedModifyPort     PAIRED with dsa
```

11.3 L'objet **dsa** lui-même peut être défini comme suit:

```
dsa OBJECT
    PORTS { readPort              [S],
            searchPort            [S],
            modifyPort            [S],
            chainedReadPort,
            chainedSearchPort,
            chainedModifyPort}
::= id-ot-dsa
```

Le DSA fournit les ports de Lecture, Recherche et Modification, rendant ainsi visibles ces services aux utilisateurs de l'objet annuaire, à savoir les DUA. En outre, un DSA fournit des versions "chaînées" de ces ports, à savoir Lecture Chaînée, Recherche Chaînée et Modification Chaînée, qui permettent aux DSA de propager les demandes concernant de tels services vers d'autres DSA.

11.4 Les ports mentionnés dans les § 11.2 et 11.3 (à l'exclusion de ceux définis dans la Recommandation X.511), sont définis comme suit:

```
chainedReadPort      PORT
    ABSTRACT OPERATIONS {
        ChainedRead, ChainedCompare,
        ChainedAbandon}
    ::=  id-pt-chained-read

chainedSearchPort    PORT
    ABSTRACT OPERATIONS {
        ChainedList, ChainedSearch}
    ::=  id-pt-chained-search

chainedModifyPort    PORT
    ABSTRACT OPERATIONS {
        ChainedAddEntry,
        ChainedRemoveEntry,
        ChainedModifyEntry,
        ChainedModifyRDN}
    ::=  id-pt-chained-modify
```

12 Types d'information

12.1 *Introduction*

12.1.1 Ce paragraphe identifie, et définit parfois, un certain nombre de types d'information qui sont utilisés par la suite dans la définition de diverses opérations du service abstrait des 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 dépendants pour mériter d'être définis séparément des opérations qui les utilisent.

12.1.2 Plusieurs des types d'information utilisés dans la définition du service abstrait des DSA sont en fait définis ailleurs. Le § 12.2 signale ces types et indique l'origine de leur définition. Chacun des paragraphes suivants (§ 12.3 à 12.9) identifie et définit un type d'information.

12.2 *Types d'information définis ailleurs*

12.2.1 Les types d'information suivants sont définis dans la Recommandation X.501:

a) **nom-Objet-Pseudonyme**;

b) **Nom-Spécifique;**

c) **Nom;**

d) **Nom-Spécifique-Relatif.**

12.2.2 Les types d'information suivants sont définis dans la Recommandation X.511:

(Liaison abstraite)

a) **Liaison-annuaire;**

(Opérations abstraites)

b) **Abandon;**

(Erreurs abstraites)

c) **abandonnée;**

d) **erreur-Attribut;**

e) **erreur-Nom;**

f) **erreur-Sécurité;**

g) **erreur-Service;**

h) **erreur-Mise-A-Jour;**

(Macro)

i) **OPTIONALLY-SIGNED;**

(Type de Données)

j) **Paramètres-Sécurité.**

12.2.3 Le type d'information suivant est défini dans la Recommandation X.520:

a) **Adresse-Présentation.**

12.3 *Arguments de chaînage*

12.3.1 Les **Arguments-Chaînage** figurent dans chaque opération abstraite 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 {notStarted},
    traceInformation    [3]  TraceInformation,
    aliasDereferenced   [4]  BOOLEAN DEFAULT FALSE,
    aliasedRDNs         [5]  INTEGER OPTIONAL,
        -- absent unless aliasDereferenced is TRUE
    returnCrossRefs     [6]  BOOLEAN DEFAULT FALSE,
    referenceType       [7]  ReferenceType DEFAULT superior,
    Info                [8]  DomainInfo OPTIONAL,
    timeLimit           [9]  UTCTime OPTIONAL,
                        [10] SecurityParameters DEFAULT {}}
```

12.3.2 Les divers composants ont les significations définies aux § 12.3.2.1 à 12.3.2.11.

12.3.2.1 Le composant **initiateur** véhicule le nom du (dernier) émetteur de la demande, à moins qu'il ne soit déjà spécifié dans les paramètres Sécurité. Si le **demandeur** est présent dans **ArgumentsCommuns**, cet argument peut être omis.

12.3.2.2 Le composant **objet-Cible** véhicule le nom de l'objet vers l'entrée d'annuaire duquel s'effectue l'acheminement. Le rôle de cet objet dépend de l'opération abstraite concernée: il peut s'agir de l'objet dont l'entrée doit être utilisée ou de l'objet de base pour une demande ou une sous-demande impliquant plusieurs objets (par exemple, **Listage-Chaînée** ou **Recherche-Chaînée**). Ce composant peut être omis seulement s'il a la même valeur que le paramètre d'objet de base dans **XArgument** (voir le § 14.3.1), auquel cas sa valeur implicite est cette valeur.

12.3.2.3 Le composant **avancement-Opération** est utilisé pour informer le DSA de l'avancement de l'opération et donc du rôle qu'il est supposé jouer dans son exécution totale. Les informations véhiculées dans ce composant sont spécifiées au § 12.5.

12.3.2.4 Le composant **informations-Traces** est utilisé pour éviter le bouclage entre des DSA quand le chaînage est utilisé. Un DSA ajoute un nouvel élément aux informations de trace avant de chaîner une opération à un autre DSA. Quand il lui est demandé d'exécuter une opération, un DSA vérifie, en examinant les informations de trace, que l'opération ne forme pas une boucle. Les informations véhiculées dans ce composant sont spécifiées au § 12.6.

12.3.2.5 Le composant **pseudonyme-Déréférencé** est une valeur booléenne utilisée pour indiquer si une ou plusieurs entrées pseudonymes ont déjà été rencontrées ou non, et déréférencées au cours de la résolution répartie du nom. La valeur par défaut **FALSE** indique qu'aucune entrée pseudonyme n'a été déréférencée.

12.3.2.6 Le composant **RDN-Pseudonymes** indique combien de RDN de l'**objet-Cible-Nom** ont été générés à partir des attributs **nom-Objet-Pseudonyme** d'une ou de plusieurs entrées pseudonymes. Sa valeur entière est positionnée chaque fois qu'une entrée pseudonyme est rencontrée et déréférencée. Ce composant doit figurer si et seulement si le composant **pseudonyme-Déréférencé** a la valeur **TRUE**.

12.3.2.7 Le composant **références-Croisées-En-Retour** est une valeur booléenne qui indique si des références de connaissance utilisées ou non au cours de l'exécution d'une opération répartie, font 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.

12.3.2.8 Le composant **type-Référence** indique au DSA auquel il est demandé d'exécuter l'opération abstraite, quel type de connaissance a été utilisé pour lui acheminer la demande. Le DSA peut ainsi être capable de détecter des erreurs dans les connaissances détenues par le demandeur. Si une telle erreur est détectée, elle doit être indiquée par une **Erreur-Service**, avec le problème **Référence-Non-Valide**. Le **Type-Référence** est décrit en détail au § 12.7.

Remarque - Si le **type-Référence** manque, il faut alors prendre la valeur **supérieur**.

12.3.2.9 Le composant **Info** est utilisé pour véhiculer des informations spécifiques à un DMD entre des DSA qui sont impliqués dans le traitement en commun d'une demande. Ce composant est du type **Info-Domaine**, qui est un type non restreint:

```
DomainInfo ::= ANY
```

12.3.2.10 Le composant **délai-Limite**, s'il est présent, indique le délai d'exécution de l'opération.

12.3.2.11 Le composant **Paramètres-Sécurité** est spécifié dans la Recommandation X.511. Lorsque ce paramètre est absent, on considère qu'il existe un ensemble vide de paramètres de sécurité.

12.4 *Résultats du chaînage*

12.4.1 Les **Résultats-Chaînage** figurent dans le résultat de chaque opération abstraite et fournissent des informations en retour au DSA qui a lancé cette opération abstraite.

```
ChainingResults     ::=  SET {
    Info                 [0]  DomainInfo OPTIONAL,
    crossReferences      [1]  SEQUENCE OF CrossReference OPTIONAL,
                         [2]  SecurityParameters DEFAULT {}}
```

12.4.2 Les divers composants ont les significations définies aux § 12.4.2.1 à 12.4.2.3.

12.4.2.1 Le composant **Info** est utilisé pour véhiculer des informations spécifiques à un DMD, entre des DSA qui sont impliqués dans le traitement en commun d'une demande. Ce composant est du type **Info-Domaine**, qui est non restreint.

12.4.2.2 Le composant **références-Croisées** ne figure pas dans les **Résultats-Chaînage**, sauf si le composant **références-Croisées-Retour** de la demande correspondante a la valeur **TRUE**. Ce composant comprend une séquence d'éléments **Référence-Croisée** dont chacun contient un **préfixe-Contexte** et un descripteur de **point-Accès** (voir le § 12.8).

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

Une **référence-Croisée** peut être ajoutée par un DSA quand il trouve une correspondance d'une partie de l'argument **objet-Cible** d'une opération abstraite avec un de ses préfixes de contexte. L'autorité de gestion d'un DSA peut avoir pour politique de ne pas retourner une telle connaissance: dans ce cas, elle n'ajoutera pas d'éléments à la séquence.

12.4.2.3 Le composant **Paramètres-Sécurité** est spécifié dans la Recommandation X.511. Lorsque ce paramètre est absent, on considère qu'il existe un ensemble vide de paramètres de sécurité.

12.5 *Avancement d'une opération*

12.5.1 Une valeur **Avancement-Opération** décrit l'état d'avancement de l'exécution d'une opération abstraite à laquelle doivent participer plusieurs DSA.

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

12.5.2 Les divers composants ont la signification définie aux § 12.5.2.1 et 12.5.2.2.

12.5.2.1 Le composant **phase-Résolution-Nom** indique la phase atteinte dans le traitement du nom **objet-Cible** d'une opération. Quand il indique que la résolution du nom est **non-Commencée**, ceci signale qu'il n'a pas encore été possible d'atteindre un DSA comportant un contexte de dénomination contenant le ou les RDN initiaux du nom. Si la résolution du nom est **en-Cours**, ceci indique que la partie initiale du nom a été reconnue, mais que le DSA contenant l'objet cible n'a pas encore été atteint. **Prochain RDN-A-Résoudre** indique quelle fraction du nom a déjà été reconnue (§ 12.5.2.2). Si la résolution du nom est **terminée**, le DSA contenant l'objet cible a été atteint, et l'exécution de l'opération proprement dite est en cours.

12.5.2.2 Le **prochain RDN-A-Résoudre** indique au DSA quel est le prochain des RDN du nom **objet-Cible** à résoudre. Il a la forme d'un entier de la plage 1 au nombre de RDN du nom. Ce composant n'est présent que si le composant **phase-Résolution-Nom** a la valeur **en-Cours**.

12.6 *Informations de trace*

12.6.1 Une valeur d'**Informations-Trace** 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, ces bouclages pouvant provenir de connaissances incohérentes ou de la présence de bouclages de pseudonymes dans le DIT.

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

12.6.2 Chaque DSA qui propage une opération vers un autre ajoute un nouvel élément aux informations de trace. Chacun de ces **éléments** contient:

a) le **nom** du dsa qui ajoute l'élément;

b) le **nom-objet-Cible** que le DSA ajoutant l'élément 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 l'**objet** ou **objet-Base** dans **X-Opération**) ou si sa valeur est la même que l'**objet-Cible** (réel ou implicite) dans l'**Argument-Opération** Répartie de la demande sortante;

c) l'**avancement-Opération** que le DSA ajoutant l'élément a reçu lors de la demande entrante.

12.7 *Type de référence*

12.7.1 Une valeur **Type-Référence** indique un des différents types de référence définis au § 10.

```
ReferenceType      ::=
    ENUMERATED {
        superior                (1),
        subordinate             (2),
        cross                   (3),
        nonSpecificSubordinate  (4)}
```

12.8 *Point d'accès*

12.8.1 Une valeur **Point-Accès** identifie un point particulier où peut s'effectuer un accès à l'annuaire, spécifiquement à un DSA. Le point d'accès a un **Nom**, celui du DSA concerné, et une **Adresse-Présentation**, à utiliser lors des communications OSI vers ce DSA.

```
AccessPoint ::= SET {
        ae-title    [0] Name,
        address     [1] PresentationAddress }
```

12.9 *Référence de continuation*

12.9.1 Une **référence-Continuation** décrit comment l'exécution de l'ensemble ou d'une partie d'une opération abstraite peut être continuée près d'un ou de plusieurs DSA différents. Elle est normalement retournée comme renvoi de référence quand le DSA concerné est incapable de répercuter la demande lui-même, ou ne le désire pas.

```
ContinuationReference ::= SET {
    targetObject      [0] Name,
    aliasedRDNs       [1] INTEGER OPTIONAL,
    operationProgress [2] OperationProgress,
    rdnsResolved      [3] INTEGER OPTIONAL,
    referenceType     [4] ReferenceType OPTIONAL,
    -- only present in the DSP
    accessPoints      [5] SET OF AccessPoint}
```

12.9.2 Les divers composants ont les significations définies aux § 12.9.2.1 à 12.9.2.6.

12.9.2.1 Le **nom-Objet-Cible** qui est proposé pour continuer l'opération. Cela pourrait différer du **nom-Objet-Cible** reçu lors de la demande entrante si, par exemple, un pseudonyme a été déréférencé ou si l'objet de base d'une recherche a été localisé.

12.9.2.2 Le composant **RDNpseudonymes** indique combien de RDN de la cible ont été - le cas échéant - produits en déréférençant un pseudonyme. Cet argument n'est présent que si un pseudonyme a été déréférencé.

12.9.2.3 L'**avancement-Opération** qui a été atteint, et qui déterminera la suite de l'exécution de l'opération abstraite par les DSA nommés, à condition que le DSA ou le DUA recevant la **Référence-Continuation** prenne la suite.

12.9.2.4 La valeur du composant **rdns-Résolus** (qui n'a besoin d'être présent que si certains des RDN du nom n'ont pas fait l'objet d'une résolution de nom complète, mais ont été supposés corrects d'après une référence croisée) indique combien de RDN ont été effectivement résolus, en utilisant uniquement des références internes.

12.9.2.5 Le composant **type-Référence**, qui figure uniquement dans le service abstrait des DSA, indique quel type de connaissance a été utilisé pour générer cette continuation.

12.9.2.6 Le composant **point-Accès** indique les points d'accès auxquels il faut aller ensuite pour réaliser cette continuation. Quand des Références Subordonnées Non Spécifiques sont impliquées, plusieurs **Point-Accès** peuvent être listés, vers chacun desquels il faut aller ensuite, par exemple, par multireport.

13 Liaison et séparation abstraites

Liaison-DSA et **Séparation-DSA** sont utilisés par un DSA, respectivement au début et à la fin d'une période d'accès à un autre DSA.

13.1 *Liaison de DSA*

13.1.1 Une opération de liaison abstraite **Liaison-DSA** est utilisée par un DSA pour lier ses ports **lecture-Chaînée**, **recherche-Chaînée** et **modification-Chaînée** à ceux d'un autre DSA.

```
DSABind ::= ABSTRACT-BIND
    TO      {chainedRead,
             chainedSearch,
             chainedModify}
    DirectoryBind
```

13.1.2 Les composants de la **Liaison-DSA** sont identiques à leurs contreparties de la **Liaison annuaire** (voir la Recommandation X.511) sauf sur les points suivants:

13.1.2.1 Les **Pouvoirs** de l'**Argument Liaison-annuaire** permettent d'envoyer au DSA qui répond l'information identifiant le Titre AE du DSA initiateur. Le Titre AE doit être sous forme d'un Nom Spécifique d'annuaire.

13.1.2.2 Les **Pouvoirs** du **Résultat Liaison d'annuaire** permettent d'envoyer au DSA appelant l'information identifiant le Titre AE du DSA initiateur. Le Titre AE doit être sous la forme d'un Nom Spécifique.

13.2 *Séparation de DSA*

13.2.1 Une opération **séparation-DSA** est utilisée pour séparer les ports de Lecture-Chaînée, Recherche-Chaînée et Modification-Chaînée d'un couple de DSA.

```
DSAUnbind ::= ABSTRACT-UNBIND
   FROM       {chainedRead,
               chainedSearch,
               chainedModify}
```

13.2.2 Elle ne comporte aucun argument, résultat ni erreur.

14 Opérations abstraites chaînées

14.1 A chacun des ports du service abstrait d'annuaire, correspond un port du DSA qui permet d'assurer ce service par la coopération de DSA. Les opérations abstraites des ports correspondants sont également en correspondance biunivoque avec ces ports. Les noms des ports et les opérations abstraites ont été choisis pour refléter cette correspondance: le port ou l'opération abstraite du service abstrait des DSA est formé d'après ceux du service abstrait d'annuaire, en ajoutant comme suffixe l'adjectif "Chaîné". Les ports et opérations abstraites résultants se correspondent comme suit:

ChainedReadPort:	**ChainedRead,** **ChainedCompare,** **ChainedAbandon**
ChainedSearchPort:	**ChainedList,** **ChainedSearch**
ChainedModifyPort:	**ChainedAddEntry,** **ChainedRemoveEntry,** **ChainedModifyEntry,** **ChainedModifyRDN**

14.2 Les arguments, résultats et erreurs des opérations abstraites chaînées sont, à une exception près, formés systématiquement à partir des arguments, résultats et erreurs des opérations abstraites correspondantes du service abstrait d'annuaire (tel que décrit au § 14.3). Cette exception est l'opération abstraite **Abandon-Chaîné**, qui est syntaxiquement équivalente à sa contrepartie du service abstrait d'annuaire (décrit au § 14.4).

14.3 On utilise une opération abstraite **X-Chaîné** pour propager entre des DSA une demande, qui est (normalement) émise quand un DUA lance une opération abstraite **X** vers un DSA, ce DSA ayant choisi de la chaîner. Les arguments de l'opération abstraite peuvent être, facultativement, signés par le demandeur et, si ceci est demandé, le DSA exécutant peut signer les résultats.

14.3.1 La dérivation systématique d'une opération abstraite chaînée **X-Chaîné** de sa contrepartie **X** est définie comme suit:

Etant donné:

```
X    ::=
     ABSTRACT-OPERATION
          ARGUMENT   XArgument
          RESULT     XResult

          ERRORS {..., Referral,...}
```

L'opération abstraite chaînée est dérivée comme suit:

```
ChainedX ::=
    ABSTRACT-OPERATION
        ARGUMENT OPTIONALLY-SIGNED SET{
                ChainingArgument,
                [0]  XArgument}
    RESULT  OPTIONALLY-SIGNED SET{
                ChainingResult,
                [0]  XResult}
    ERRORS {...,DsaReferral,...}
```

Remarque - La spécification définitive du service abstrait de DSA de l'annexe A s'applique à cette dérivation pour l'intégralité des opérations abstraites chaînées.

14.3.2 Les arguments de l'opération abstraite dérivée ont la signification décrite aux § 14.3.2.1 et 14.3.2.2.

14.3.2.1 L'**Argument-Chaînage** contient les informations, situées au-dessus et au-dessous des arguments initiaux fournis par le DUA, et nécessaires au DSA exécutant pour effectuer l'opération. Ce type d'information est défini au § 12.3.

14.3.2.2 L'**Argument-X** contient les arguments initiaux fournis par le DUA, tel que spécifié dans le paragraphe approprié de la Recommandation X.511.

14.3.3 Si la demande peut être traitée avec succès, le résultat sera retourné. Les paramètres du résultat ont la signification décrite aux § 14.3.3.1 et 14.3.3.2.

14.3.3.1 Le **Résultat-Chaînage** contient les informations, situées au-dessus et au-dessous de celles à fournir au DUA demandeur, et qui peuvent être nécessaires aux DSA précédents de la chaîne. Ce type d'information est défini au § 12.4.

14.3.3.2 Le **Résultat-X** contient le résultat qui est retourné par l'exécutant de cette opération abstraite et qui est destiné à être communiqué dans le résultat au DUA demandeur. Cette information est telle que spécifiée dans le paragraphe approprié de la Recommandation X.511.

14.3.4 Si la demande échoue, une des erreurs indiquées sera retournée. L'ensemble des erreurs qui peuvent être signalées est tel que décrit dans l'opération abstraite correspondante de la Recommandation X.511, sauf que **Renvoi-Référence-DSA** est retourné à la place de **Renvoi-Référence**. Les diverses erreurs sont définies ou référencées dans le § 15.

14.4 Une opération abstraite **Abandon-Chaîné** est utilisée par un DSA pour indiquer à un autre qu'il n'est plus intéressé par l'exécution d'une opération chaînée qu'il a demandée antérieurement. Un certain nombre de raisons peuvent être invoquées, dont en voici des exemples:

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

b) le DSA a obtenu les informations nécessaires d'une autre façon, par exemple de la part d'un DSA impliqué dans un multireport qui lui a répondu plus rapidement.

Un DSA n'est jamais obligé d'émettre un **Abandon-Chaîné**, ni d'abandonner effectivement une opération, s'il lui est demandé de le faire.

Si l'**Abandon-Chaîné** réussit effectivement à interrompre l'exécution d'une opération, un résultat sera alors retourné et l'opération concernée retournera une erreur abstraite **abandonnée**. Si l'**Abandon-Chaîné** ne réussit pas à interrompre l'opération, il retourne alors lui-même une erreur **échec-Abandon**.

15 Erreurs abstraites chaînées

15.1 *Introduction*

15.1.1 Pour la plupart, les erreurs abstraites qui peuvent être retournées dans le service abstrait des DSA sont les mêmes que celles qui peuvent l'être dans le service abstrait d'annuaire. Les seules exceptions sont "l'erreur" **Renvoi-Référence-DSA**, qui est retournée à la place de **Renvoi-Référence** (voir le § 15.2) et les problèmes de service suivants qui ont la même syntaxe abstraite mais une sémantique différente:

a) **invalidReference** - Le DSA n'était pas en mesure de répondre à la demande comme indiqué par le DUA (via l'Avancement-Opération). Cela peut avoir été dû à l'utilisation d'un renvoi de référence non valide.

b) **loopDetected** - L'annuaire n'est pas en mesure de répondrre à la demande en raison d'une boucle interne.

15.1.2 La précédence des erreurs abstraites qui peuvent se produire est la même que leur précédence dans le service abstrait d'annuaire, tel que spécifié dans la Recommandation X.511.

15.2 *Renvoi de référence de DSA*

15.2.1 L'erreur abstraite **Renvoi-Référence-DSA** est générée par un DSA quand, pour une raison ou une autre, il ne désire pas continuer l'exécution d'une opération abstraite en chaînant ou en multireportant l'opération abstraite à un ou à plusieurs autres DSA. Les circonstances dans lesquelles il peut retourner un renvoi de référence sont décrites au § 8.4.

```
DSAReferral          ::=
    ABSTRACT-ERROR
        PARAMETER SET{
            [0]  ContinuationReference,
            contextPrefix [1] DistinguishedName OPTIONAL }
```

15.2.2 Les divers paramètres ont les significations décrites aux § 15.2.2.1 et 15.2.2.2.

15.2.2.1 La **Référence-Continuation** contient les informations nécessaires au demandeur pour propager éventuellement une demande ultérieure appropriée vers un autre DSA. Ce type d'information est spécifié au § 12.9.

15.2.2.2 Si le composant **références-Croisées-Retour** des **Arguments-Chaînage** de cette opération abstraite a la valeur **TRUE** et que le renvoi de référence est fondé sur une référence subordonnée ou croisée, le paramètre **préfixe-Contexte** peut être facultativement inclus. L'autorité de gestion de tout DSA décidera quelles références de connaissance peuvent, le cas échéant, être retournées de cette façon (les autres peuvent être, par exemple, confidentielles et réservées à ce DSA).

SECTION 5 - *Procédures des opérations réparties*

16 Introduction

16.1 *Objectif et limites*

Ce paragraphe spécifie les procédures des opérations réparties 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 produit l'intégralité des services fournis aux utilisateurs par l'annuaire.

La description des procédures relatives à un seul DSA est fondée sur les modèles des § 7 à 10 de la présente Recommandation.

A noter que le modèle et les procédures sont présentés à titre de directives et ne sont pas destinés à imposer des contraintes quant à la réalisation d'un DSA effectif, ni à régir cette réalisation.

Ce paragraphe est divisé en trois sous-paragraphes: la présente introduction, un modèle conceptuel pour décrire le comportement de l'annuaire et une introduction aux modèles point de vue-DSA et point de vue-Opérations des opérations DSA.

16.2 *Modèle conceptuel*

La complexité du fonctionnement réparti de l'annuaire rend nécessaire une modélisation conceptuelle utilisant à la fois des techniques de descriptions narratives et graphiques. Toutefois, aucun de ces schémas narratifs ou graphiques ne saurait être interprété comme une description formelle du fonctionnement réparti de l'annuaire.

16.3 *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 du DSA. Dans cette optique, l'ensemble des procédures qui prennent en charge 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 contrôle. Le § 18 décrit les procédures des DSA du point de vue du DSA.

b) Point de vue de l'opération. Le point de vue axé sur le DSA donne une description détaillée complète mais rend difficile à comprendre la structure des opérations individuelles, qui peuvent impliquer un traitement par plusieurs DSA. En conséquence, le § 17 adopte un point de vue essentiellement axé sur l'opération pour présenter les phases de traitement concernant chacune des opérations.

Afin de prendre en charge le fonctionnement réparti de l'annuaire, chaque DSA doit effectuer les actions nécessaires pour réaliser l'objet de chaque opération et des actions supplémentaires, nécessaires pour répartir cette réalisation entre plusieurs DSA. Le § 17 explore les différences entre ces deux types d'actions. Les deux types d'actions sont spécifiés en détail dans le § 18.

17 Comportement réparti de l'annuaire

17.1 *Exécution coopérative des opérations*

Chaque DSA est doté de 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 lancée par un DUA ou par un autre DSA; le DSA effectue un traitement sur l'objet, puis le dirige vers un autre DSA pour traitement ultérieur.

Un autre point de vue possible serait de considérer le traitement total vécu par une opération au cours de son exécution par plusieurs DSA coopérants. Ce point de vue met en évidence les phases de traitement communes, qui concernent toutes les opérations.

17.2 *Phases du 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 situer 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, lecture) est réellement effectuée;

c) la phase de Fusion des Résultats - au cours de laquelle les résultats d'une opération spécifiée sont retournés au DUA demandeur. Si un mode d'interaction de chaînage a été choisi, la phase de Fusion des Résultats peut impliquer plusieurs DSA, dont chacun a chaîné la demande ou sous-demande initiale (définie au § 17.3.1: Décomposition de la Demande) à un autre DSA durant l'une ou l'autre des phases précédentes, ou durant ces deux phases.

Dans le cas des opérations **Lecture, Comparaison, Listage, Recherche** et **Modification d'Entrée,** la résolution du nom concerne le nom d'objet fourni dans l'argument de l'opération. Dans le cas des opérations **Ajout d'Entrée, Suppression d'Entrée** et **Modification de RDN,** la résolution du nom concerne le nom de l'objet immédiatement supérieur (dérivé en supprimant le RDN final du nom fourni dans l'argument de l'opération).

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 en ses trois phases.

17.2.1 *Phase de résolution du nom*

La Résolution du Nom est le processus de comparaison séquentielle de chacun des RDN d'un nom présenté à un arc (ou noeud) 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 être uniquement capable d'exécuter une fraction du processus de résolution du nom. Un DSA donné effectue sa part du processus de Résolution du Nom en parcourant ses connaissances locales. Ce

processus est décrit au § 18.6 et par les diagrammes associés (figures 11/X.518 à 13/X.518). Quand un DSA atteint la frontière de son contexte de dénomination, il peut savoir d'après les informations de connaissances qui y sont contenues, si la résolution peut être continuée par un autre DSA ou si le nom est erroné.

17.2.2 *Phase d'évaluation*

Quand la phase de résolution du nom est terminée, l'opération requise proprement dite (par exemple, Lecture ou Recherche) est effectuée.

Les opérations qui impliquent une seule entrée - **Lecture, Comparaison, Ajout d'Entrée, Suppression d'Entrée, Modification de RDN** et **Modification d'Entrée** - peuvent être exécutées entièrement à l'intérieur du DSA dans lequel cette entrée a été située. **Ajout d'Entrée, Suppression d'Entrée** et **Modification de RDN** peuvent affecter les connaissances dans plusieurs DSA (voir le § 18.7.1).

Les opérations qui impliquent plusieurs entrées - **Listage** et **Recherche** - ont besoin de situer des subordonnées de la cible, qui peuvent ou non résider dans le même DSA. Si elles ne résident pas toutes dans le même DSA, les opérations doivent être dirigées vers les DSA spécifiés dans les références subordonnées pour exécuter le processus d'évaluation.

17.2.3 *Phase de fusionnement des résultats*

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

Dans les cas où l'opération concerne une seule entrée, le résultat de cette opération peut simplement être retourné au DUA demandeur. Dans les cas où l'opération a affecté plusieurs entrées de plusieurs DSA, les résultats doivent être combinés.

Les réponses permises retournées à un demandeur après fusion 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 (s'applique uniquement à **Listage** et **Recherche**). Un tel *résultat partiel* peut incorporer des références de continuation concernant des parties inexplorées du DIT;

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

d) et si le demandeur était un DSA, un Résultat-Chaînage.

17.3 *Gestion des opérations réparties*

Des informations sont incluses dans l'argument de chaque opération abstraite dont l'exécution peut être demandée à un DSA: elles indiquent l'avancement de chaque opération au cours de sa traversée de divers DSA de l'annuaire. Ceci 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 au DSA pour répartir physiquement les opérations et répondre à d'autres besoins résultant de leur répartition.

17.3.1 *Décomposition de la demande*

La décomposition de la demande est un processus exécuté de façon interne par un DSA, avant la communication avec un ou plusieurs autres DSA. Une demande est décomposée en plusieurs sous-demandes, de sorte que chacune de ces dernières accomplit une partie de la tâche initiale. La décomposition de la demande peut être utilisée, par exemple, dans l'opération Recherche, après avoir trouvé l'objet de base. Après la décomposition, chacune des sous-demandes peut alors être chaînée ou multireportée vers d'autres DSA pour continuer l'exécution de la tâche.

17.3.2 *Réponse à une demande émanant d'un DSA*

Un DSA qui reçoit une demande peut déterminer l'avancement de son traitement à l'aide du paramètre Avancement-Opération. C'est ainsi qu'il détermine 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 essayer de satisfaire. Si le DSA ne peut pas satisfaire entièrement la demande, il doit communiquer l'opération à un ou à plusieurs DSA qui peuvent aider à satisfaire la demande (par chaînage ou multireport), retourner un renvoi de référence à un autre DSA ou terminer la demande par une erreur.

17.3.3 *Exécution des opérations*

Chaque DSA qui a lancé une opération ou répercuté une opération sur un ou plusieurs autres DSA doit garder trace de l'existence de cette opération jusqu'à ce que chacun des autres DSA ait retourné un résultat ou une erreur, ou que le délai limite 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 vers d'autres DSA dans l'annuaire.

17.4 *Autres considérations relatives aux opérations réparties*

17.4.1 *Validation de demande*

A la réception d'une opération d'annuaire, un DSA doit, dans un premier temps, valider l'opération pour s'assurer qu'elle peut être poursuivie. Des situations telles que les bouclages dans le DIT, provoqués par l'utilisation inadéquate de pseudonymes ou l'utilisation de connaissances erronées, peuvent entraîner l'envoi, à des DSA, d'opérations qui ne peuvent pas être traitées.

Dans les cas simples, ces états d'erreur sont traités de façon adéquate par des procédures de résolution du nom telles que décrites au § 18, mais quand les circonstances ont provoqué des bouclages d'opération (tels que décrits au § 17.4.3), la seule résolution du nom est insuffisante.

Les actions de validation de la demande garantissent la détection d'un bouclage avant toute tentative de poursuite d'une opération à travers des données erronées produites par ce bouclage. Le processus de détection est effectué selon la procédure de détection de bouclage spécifiée au § 18.5.1.

Quand des procédures de sécurité sont en vigueur, la validation de la demande vérifie également l'identité du DSA ou du DUA demandeur, ainsi que la validité de la demande.

17.4.2 *Informations d'état et de trace*

La progression d'une opération à l'intérieur de l'annuaire et la présence d'une situation de bouclage sont déterminées par un "état" de l'opération, cet état étant défini par:

- le nom du DSA traitant actuellement l'opération;
- le nom de **l'objet-Cible**, tel que contenu dans l'argument de l'opération;
- **l'avancement-Opération**, tel que contenu dans l'argument de l'opération et défini au § 12.5.

En plus de l'état courant d'une opération, un DSA a également besoin de connaître tous les états antérieurs de cette opération. Ils sont enregistrés dans l'argument **informations-Traces** et véhiculés avec l'opération.

L'argument **informations-Traces** constitue la base des stratégies de prévention/détection de bouclage, telles que spécifiées au § 17.4.3.

17.4.3 *Bouclages*

Dans le contexte d'une opération d'annuaire particulière, un bouclage se produit si, à un moment donné, une opération revient à un état antérieur (tel que défini ci-dessus). Les bouclages sont gérés à l'aide de l'argument **informations-Traces**. Deux stratégies sont définies pour traiter les bouclages. Dans la stratégie de *détection de bouclage*, un DSA détermine si un bouclage s'est éventuellement produit dans une opération entrante et, dans l'affirmative, retourne une erreur. Dans la stratégie de *prévention de bouclage*, un DSA détermine si la propagation d'une opération risque de produire un bouclage.

17.4.4 *Commandes de service*

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

a) **Interdiction-Chaînage**: Un DSA consulte cette commande de service, pour déterminer le mode de propagation d'une opération. Si elle est positionnée, le DSA utilise alors toujours le mode de 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) **Délai-Limite**: Un DSA doit tenir compte de cette commande de service pour garantir que le délai limite n'est pas dépassé à son niveau. Un DSA, auquel un DUA demande d'exécuter une opération, note en premier lieu le délai limite exprimé par le DSA, en secondes, pour l'exécution de cette opération. Si un chaînage est demandé, le *délai-Limite* est inclus

dans l'argument de chaînage à communiquer aux DSA suivants. Dans ce cas, la même valeur de la 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 un argument de chaînage où est spécifié un **délai-Limite**, il doit respecter cette limite.

c) **Taille-Limite**: 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 ou multireportée. 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 résultats multiples et applique la limite à leur ensemble pour garantir que seul le nombre demandé est retourné. Si la limite a été dépassée, ceci est indiqué dans la réponse.

d) **Priorité**: 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 existe.

e) **Portée-Locale**: L'opération est limitée à une portée définie au niveau local et ne peut être propagée par aucun des modes.

f) **Renvoi-Référence-Portée**: Si le DSA retourne un renvoi de référence ou un résultat partiel à une opération **Listage** ou **Recherche**, les **Références-Continuation** incluses doivent faire partie de la portée requise.

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

17.4.5 *Extensions*

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

Remarque - Une autorité administrative peut juger approprié d'envoyer une **erreur de service** avec le problème **refus d'exécuter** si elle ne souhaite pas propager une extension.

17.4.5.2 Si un DSA rencontre une extension lors de la phase exécution du traitement, il existe deux possibilités. Si l'extension n'est pas critique, le DSA ne tient pas compte de l'extension. Si l'extension est critique, le DSA envoie une **erreur de service** avec le problème **Extension critique non disponible.**

Une extension critique vers une opération d'objet multiple peut se traduire par des résultats et des erreurs de ce type. Un DSA qui regroupe ces résultats et ces erreurs met au rebut ces erreurs de service et emploie le composant **Extension critique non disponible** du **qualificatif Résultat Partiel** décrit au § 10.1.1 de la Recommandation X.511.

17.4.6 *Déréférençage de pseudonyme*

Le déréférençage de pseudonyme est le processus qui consiste à créer un nouveau nom objet cible, en remplaçant la partie nom spécifique d'entrée pseudonyme du nom objet cible initial par la valeur d'attribut du nom objet pseudonyme de l'entrée pseudonyme. Dans l'opération, le nom objet n'est pas affecté par le déréférençage de pseudonyme.

17.5 *Authentification des opérations réparties*

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

Deux ensembles de procédures d'authentification sont proposés pour permettre de satisfaire collectivement une gamme d'impératifs d'authentification. Un de ces ensembles se compose des procédures assurées par la Liaison: elles facilitent l'authentification opérée entre deux entités d'application de l'annuaire à des fins d'établissement d'association. Les procédures de l'ensemble Liaison permettent toute une gamme d'échanges à des fins d'authentification, depuis le simple échange d'identités jusqu'à l'authentification poussée.

En plus de l'authentification de l'entité homologue d'une association, comme celle assurée par la Liaison, 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 du demandeur, qui concernent l'identification par un DSA, de l'auteur de la demande de service initiale. Le second ensemble facilite des services d'authentification des résultats, qui concernent l'authentification, par le demandeur, de tous les résultats qui lui sont retournés.

Deux procédures sont définies pour l'authentification du demandeur, l'une fondée sur un simple échange d'identités, appelée authentification fondée sur l'identité, et l'autre fondée sur des techniques de signatures numériques, appelée authentification fondée sur la signature. 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 du résultat, une procédure unique d'authentification des résultats est définie, fondée sur des techniques de signatures numériques; du fait de la nature généralement complexe du collationnement des résultats, aucune procédure simple, fondée sur l'identité, n'est définie.

L'authentification des réponses d'erreur n'est pas prise en charge par ces procédures.

Les services décrits ci-dessus doivent être considérés comme étendant ceux fournis par le service de Liaison; les procédures de Liaison sont supposées avoir été exécutées avec succès avant l'authentification des opérations d'annuaire.

Les procédures à effectuer par un DSA pour authentifier le demandeur et les résultats d'authentification sont spécifiées au § 18.9.

18 Comportement d'un DSA

18.1 *Introduction*

A chaque opération lancée par un demandeur (par exemple, un DUA ou un DSA), le DSA exécutant doit se comporter conformément à des procédures bien définies, de sorte qu'une réponse appropriée soit renvoyée de façon parfaitement déterministe. Ce paragraphe spécifie le comportement autorisé d'un DSA, en le modélisant en termes de processus mettant en oeuvre un ensemble particulier de procédures. Il est essentiel de réaliser qu'un DSA doit uniquement se conformer au comportement visible de l'extérieur impliqué par ces procédures et non aux procédures elles-mêmes.

18.2 *Aperçu du comportement d'un DSA*

Le comportement global de l'annuaire réparti est la somme des comportements de ses DSA coopérants. Chacun de ces DSA peut être considéré comme un processus, pris en charge localement par un ensemble de procédures.

La figure 6/X.518 présente le comportement d'un DSA, vu de l'intérieur.

L'Aiguilleur d'Opérations est la principale procédure de contrôle d'un DSA. Il guide chaque opération au cours des trois phases de traitement décrites au § 17.2.

Les procédures qui prennent en charge l'Aiguilleur d'Opérations sont: la Résolution du Nom, la Recherche du Contexte de Dénomination, la Résolution Locale du Nom, L'Evaluation, l'Evaluation d'un Objet Unique, l'Evaluation de Plusieurs Objets et le Fusionnement des Résultats. Les relations entre ces procédures sont présentées schématiquement sur la figure 6/X.518.

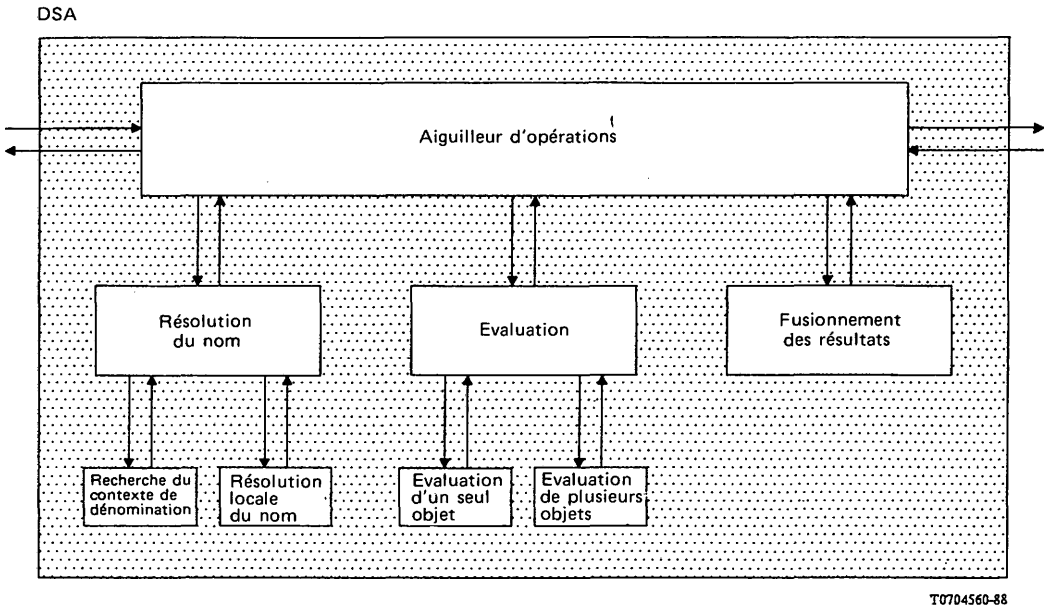


FIGURE 6/X.518

Comportement d'un DSA - Vu de l'intérieur

18.2.1 *L'Aiguilleur d'opérations*

A la réception d'une opération, l'Aiguilleur d'Opérations commence par la valider, en vérifiant l'absence de bouclage ou d'erreur d'authentification. Si aucun bouclage ni aucune erreur n'est trouvé, il appelle la Résolution du Nom, qui renvoie une indication Trouvé, une Référence ou une indication d'erreur. Les références sont traitées par une action de Renvoi de référence, de Chaînage ou de Multireport. Les indications Trouvé sont traitées en appelant la procédure d'Evaluation, qui exécute effectivement l'opération prévue. Une fois renvoyés, les résultats internes ou externes sont fusionnés par le Fusionnement des Résultats, et en l'absence d'erreurs, renvoyés au DUA ou au DSA demandeur.

18.2.2 *Résolution du nom*

La Résolution du Nom appelle la Recherche du Contexte de Dénomination. Si le contexte renvoyé est local, alors la Résolution Locale du nom est appelée, autrement la Résolution du Nom retourne un renvoi de référence ou une erreur et termine son intervention. Si la Résolution du Nom rencontre un pseudonyme, il est déréférencé (s'il est renvoyé) et la Résolution du Nom répète l'analyse depuis le début. Autrement, la Résolution Locale du Nom retourne une indication Trouvé, une erreur ou un Renvoi de référence, qui est renvoyé à l'Aiguilleur d'opérations.

18.2.3 *Recherche du contexte de dénomination*

La Recherche du Contexte de Dénomination compare le Nom visé avec des Préfixes de Contexte. Si aucun ne correspond, la Recherche du Contexte de Dénomination essaye alors de déterminer une référence croisée ou supérieure. Si un préfixe de contexte correspond, la Recherche du Contexte de Dénomination peut, soit renvoyer une référence croisée associée à des niveaux inférieurs du DIT, soit renvoyer une indication selon laquelle un contexte de dénomination approprié a été trouvé localement et positionner Phase-Résolution-Nom à "en-Cours".

18.2.4 *Résolution Locale du Nom*

La procédure de Résolution Locale du Nom procède localement à des comparaisons des RDN du Nom visé, jusqu'à ce qu'elle puisse renvoyer une indication Trouvé. Si elle est incapable de trouver, au niveau interne, une correspondance pour tous les RDN, elle essaye d'identifier les références, d'abord spécifiques, puis subordonnées non spécifiques et les renvoie à la Résolution du Nom. Si un pseudonyme est rencontré, et que le déréférençage est autorisé par les commandes de service, une indication de pseudonyme déréférencé est renvoyée. Si le déréférençage n'est pas autorisé, une indication Trouvé est renvoyée si, et seulement si, tous les RDN avaient trouvé une correspondance au moment où le pseudonyme a été rencontré, sinon une **erreur-Nom** est renvoyée.

18.2.5 *Evaluation*

La procédure d'Evaluation exécute effectivement l'opération d'annuaire concernant l'objet cible. Selon le type d'opération, une Evaluation d'Objet Unique ou une Evaluation de Plusieurs Objets est lancée.

18.2.6 *Evaluation d'objet unique*

L'évaluation d'objet unique est lancée pour les opérations **Lecture, Comparaison, Ajout d'Entrée, Suppression d'Entrée, Modification d'Entrée** et **Modification de RDN**. C'est au cours de cette procédure que les attributs sont effectivement recherchés, vérifiés ou modifiés.

18.2.7 *Evaluation de plusieurs objets*

La procédure d'Evaluation de Plusieurs Objets est lancée pour les opérations de **Recherche** et de **Listage** afin de vérifier les filtres, rechercher les résultats et, éventuellement, aiguiller les sous-demandes.

18.2.8 *Fusionnement des résultats*

La procédure de Fusionnement des Résultats fusionne les résultats ou les erreurs reçus d'autres DSA avec les résultats récupérés localement.

18.3 *Opérations spécifiques*

Les opérations relèvent de trois catégories (dans chaque cas, l'opération et sa contrepartie Chaînée appartiennent toutes deux à la même catégorie), concernant respectivement:

a) les Opérations sur un Seul Objet: **Lecture, Comparaison, Ajout d'Entrée, Modification d'Entrée, Modification de RDN, Suppression d'Entrée;**

b) les Opérations sur Plusieurs Objets: **Listage, Recherche;**

c) l'Opération d'Abandon: **Abandon.**

Les traitements respectifs de ces catégories d'opérations sont décrits aux § 18.3.1 à 18.3.3. Comme il existe une similitude considérable entre la façon dont un DSA se comporte pour exécuter une opération d'un port de service, et pour exécuter sa contrepartie chaînée d'un port de service chaîné, une description unique est fournie pour ces deux familles d'opérations, les exceptions à cette règle étant signalées.

18.3.1 *Opérations sur un Seul Objet*

Les opérations sur un seul objet sont celles qui affectent une seule entrée, et qui peuvent donc être entièrement effectuées à l'intérieur du DSA qui contient l'entrée sur laquelle l'opération doit être effectuée. Ces opérations sont en général décrites par la séquence d'événements suivante:

1) activation de l'Aiguilleur d'Opérations;

2) exécution de la Résolution du Nom pour situer l'objet dont le nom a été spécifié comme argument de l'opération;

3) exécution de la procédure d'évaluation d'un seul objet;

4) les commandes de service, telles que le délai limite, doivent être consultées au cours de l'opération, de façon à respecter les contraintes spécifiées par l'utilisateur;

5) renvoi des résultats au DUA ou au DSA qui a transmis la demande.

18.3.2 *Opérations sur plusieurs objets*

Les opérations sur plusieurs objets sont celles qui affectent plusieurs entrées qui peuvent ou non être situées dans le même DSA. Ces opérations peuvent ainsi impliquer un effort de coopération de la part de plusieurs DSA pour situer et traiter toutes les entrées affectées par l'opération demandée. Le comportement commun de telles opérations peut être résumé comme suit:

1) activation de l'Aiguilleur d'Opérations;

2) exécution des procédures de Résolution du Nom pour situer l'objet dont le nom a été spécifié comme argument de l'opération;

3) quand l'objet cible de l'opération a été situé, exécution des procédures d'évaluation de plusieurs objets;

4) si une décomposition de la demande a eu lieu au cours d'une des procédures d'évaluation portant sur plusieurs objets, et que des sous-demandes ont été chaînées ou multireportées, l'Aiguilleur d'Opérations met à jour les résultats locaux courants, attend les réponses chaînées et active le Fusionnement des Résultats;

5) les Commandes de service, telles que le délai limite et la taille limite, doivent être consultées au cours de l'opération, de façon à respecter les contraintes spécifiées dans l'argument commun;

6) renvoi des résultats ou des erreurs au DUA ou DSA qui a transmis la demande.

18.3.3 *Abandon d'opération*

A la réception d'une opération d'abandon, un DSA détermine s'il est capable d'abandonner l'opération spécifiée; dans l'affirmative, il l'abandonne et retourne un résultat (l'opération qui a été abandonnée retourne une erreur **Abandonné**). S'il ne peut abandonner l'opération spécifiée, il retourne une erreur **Echec d'Abandon**.

La procédure spécifique à une opération d'**Abandon** est spécifiée ci-après:

1) localisation de l'opération dont l'identificateur de lancement est spécifié comme argument de l'opération **Abandon**;

2) composition, facultative, de la ou des demandes avec l'identificateur de lancement, afin d'abandonner toute opération en cours, chaînée ou multireportée vers d'autres DSA;

3) à titre facultatif, l'opération d'abandon est exécutée localement comme défini dans la Recommandation X.511;

4) renvoi du résultat ou d'une erreur au DUA ou DSA qui a transmis la demande.

18.4 *L'Aiguilleur d'opérations*

18.4.1 *Introduction*

L'Aiguilleur d'Opérations utilise la Résolution du Nom décrite au § 18.6 de la présente Recommandation, ainsi que toutes les interactions (c'est-à-dire de DSA à DSA ou de DUA à DSA) qui sont nécessaires pour localiser les entrées cibles dans un environnement d'annuaire réparti. La figure 7/X.518 présente un diagramme détaillé décrivant l'Aiguilleur d'Opérations. L'algorithme est résumé ci-après.

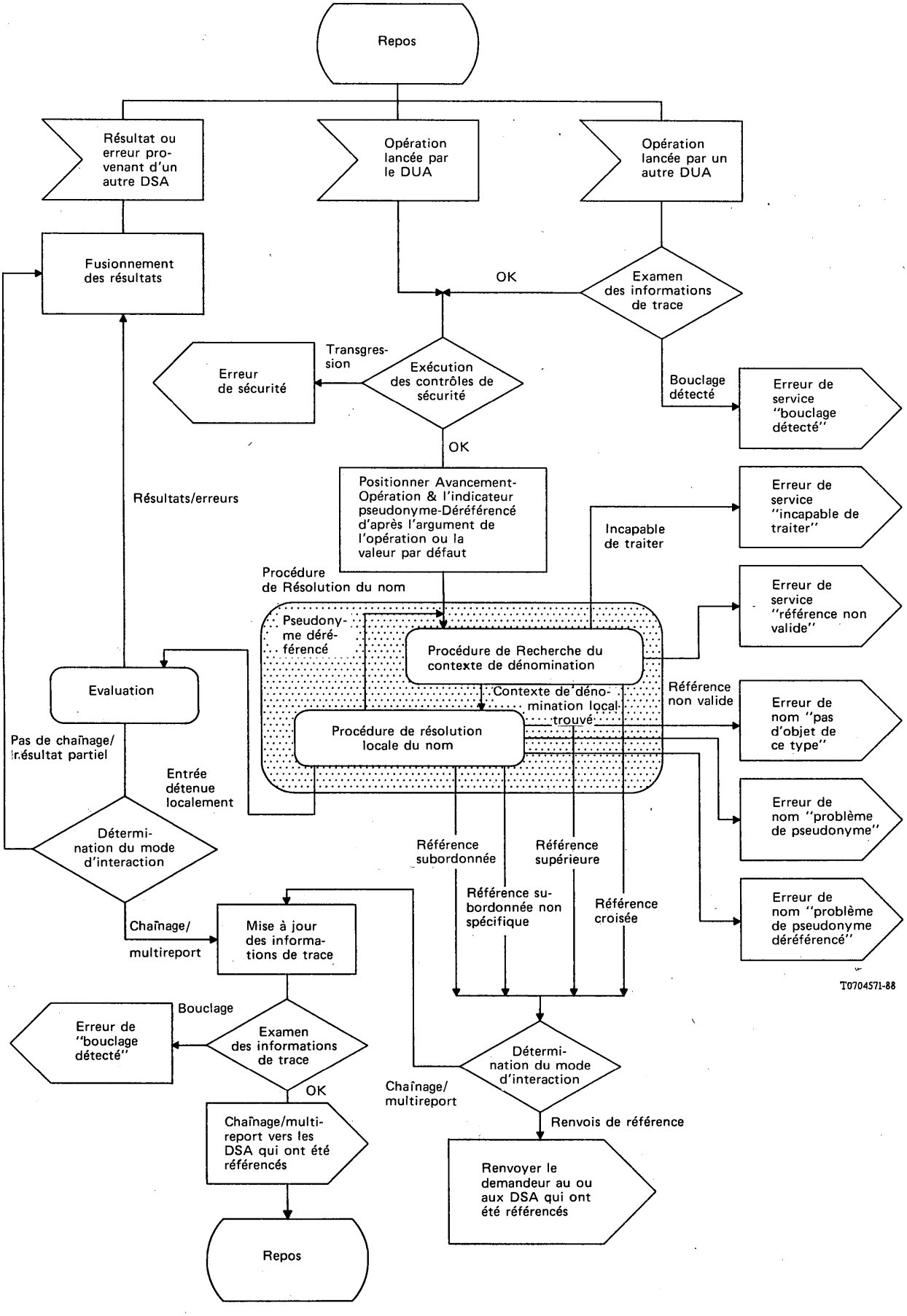


FIGURE 7/X.518

L'Aiguilleur d'opérations

18.4.2 *Actions implicites*

18.4.2.1 *Sécurité*

A noter que, bien que la vérification des signatures ne soit pas incluse explicitement dans cet algorithme, cette action vient toujours en premier quand un DSA reçoit une opération signée, un résultat signé ou une erreur signée.

Remarque - Cela n'englobe pas les signatures incluses.

Si la signature n'est pas valide, ou absente dans un cas où elle devrait exister, une **erreur-Sécurité** est renvoyée. Il est mis fin à tout traitement de l'opération et l'aiguilleur d'opérations passe en état de repos.

La signature du résultat d'une opération, le cas échéant, est, de même, la dernière étape implicite avant son renvoi.

18.4.2.2 *Commandes de service*

Bien que les **Commandes-Service** ne soient pas explicitement mentionnées, elles sont respectées. Par exemple, la vérification du **délai-Limite** à la réception d'une opération et la vérification de la **taille-Limite** avant le renvoi d'un résultat, sont considérées comme obligatoires. Ceci est traité au § 17.4.4.

18.4.2.3 *Informations de trace*

Les **Informations-Traces** sont toujours mises à jour d'après l'état auquel est parvenu le DSA, avant qu'y soient inclus les **arguments-Chaînage**. Ceci n'est pas explicitement mentionné dans le texte qui suit.

18.4.3 *Arguments*

Les Arguments-Chaînage de ladite opération.

18.4.4 *Résultats*

Le Résultat-Chaînage de ladite opération.

18.4.5 *Erreurs*

Toute erreur définie dans la présente Recommandation.

18.4.6 *Algorithme*

1) Recevoir l'opération.

   Si l'opération provient d'un autre DSA, elle comprendra les arguments de chaînage suivants: **Opération-Avancement, pseudonyyme-Déréférencé, RDN pseudonymes, objet-Cible-Nom** et **informations-Traces** ainsi que les paramètres contenus dans l'opération initiale.

   Si l'opération provient d'un DUA, elle ne contiendra pas l'indication **pseudonyme-Déréférencé**: adopter alors la valeur **FALSE**. L'argument n'inclut pas non plus d'**Informations-Traces**, aucune vérification de bouclage ne doit donc être effectuée. Positionner **l'objet-Cible-Nom** au nom de l'objet cible de l'opération (voir le § 17.2). Les autres arguments de chaînage sont positionnés conformément au paramètre de l'opération DAP. Le **demandeur** est positionné au nom de l'utilisateur.

2) Si l'opération provient d'un DSA, rechercher dans les opérations de trace la présence éventuelle d'un bouclage (activer la Détection de Bouclage). Si un bouclage est détecté, renvoyer une **erreur-Service**, avec un problème **bouclage-Détecté** et mettre fin au traitement.

3) Faire les contrôles de sécurité pour l'opération (provenant soit d'un DUA, soit d'un DSA). En cas de transgression, une **erreur de sécurité** est envoyée. Autrement, positionner **opération-Avancement** et **pseudonyme déréférencé** selon l'argument d'opération ou par défaut.

4) Exécuter la Procédure de Résolution du Nom.

   La Procédure de Résolution du Nom renverra une indication Trouvé, une référence distante ou une indication d'erreur.

5) Une des erreurs suivantes peut se produire:

   Erreur-Service (Incapable-De-Poursuivre) - Si un DSA détermine qu'il lui a été transmis une opération concernant des informations qu'il ne détient pas.

Erreur-service (Référence Non Valide) - Si un DSA détermine qu'une référence de connaissance non valide a été utilisée.

Erreur-Nom (absence d'un Tel Objet) - S'il est déterminé que le nom visé qui est spécifié dans la demande d'opération est non valide.

Erreur-Nom (problème de Pseudonyme) - Si un pseudonyme a été déréférencé qui ne nomme aucun objet.

Erreur-Nom (problème de Pseudonyme Déréférencé) - Si un pseudonyme a été rencontré dans une situation où il n'est pas autorisé.

A la réception de l'une de ces erreurs, l'Aiguilleur d'Opérations met fin au traitement et une erreur est renvoyée au DSA ou au DUA qui a lancé l'opération répartie.

6) Si Trouvé est renvoyé, activer la Procédure d'Evaluation.

7) Si une référence distante est renvoyée (à l'issue d'une Résolution ou d'une Evaluation du Nom), elle peut être l'une des suivantes: une référence croisée, une référence subordonnée, une référence supérieure ou une référence subordonnée non spécifique.

Si une de ces références est renvoyée, cela signifie que la Résolution ou l'Evaluation du Nom ne peut pas être effectuée dans ce DSA mais doit impliquer le DSA identifié dans la référence.

L'Aiguilleur d'Opérations vérifie alors le mode: renvoi de référence ou chaînage.

8) Si le mode de renvoi de référence ou d'interaction a été choisi, il s'ensuit, sous réserve du **Renvoi-Référence-Portée**, que les informations contenues dans la référence renvoyée seront renvoyées au DUA ou au DSA initial comme renvoi de référence ou qu'une **Erreur-Service Hors-De-Portée** sera renvoyée. Il est alors mis fin au traitement de cette opération.

Remarque - Si **retour-RéfCroisée** a la valeur TRUE et que la référence est une référence subordonnée non spécifique ou une référence supérieure, et qu'en outre l'autorité de gestion est disposée à fournir des connaissances, le préfixe de contexte du renvoi de référence peut alors être positionné.

9) Si le mode d'interaction de chaînage a été choisi, l'opération est transmise au DSA spécifié dans la référence. Dans le cas d'une référence subordonnée non spécifique, l'opération doit être transmise à chaque DSA dont le nom a été atteint comme partie d'une référence subordonnée non spécifique. Une telle transmission peut être réalisée par multireport ou par chaînage séquentiel de l'opération.

10) Appliquer la procédure de Prévention de bouclage à chaque opération à renvoyer. Si la prévention s'avère inapplicable ou qu'aucun bouclage n'est détecté, affecter des valeurs aux arguments de chaînage, y compris une version mise à jour des informations-Traces et envoyer les opérations.

Si aucune opération n'a été envoyée (du fait de problèmes de bouclage), renvoyer une erreur-Service (avec un problème bouclage-Détecté) et mettre fin au traitement de cette opération.

Remarque - Si une opération décomposée a été interrompue prématurément au cours de cette étape, par application de la procédure de Prévention de bouclage, la décision de renvoyer un résultat partiel ou de mettre éventuellement fin à toute l'opération et de renvoyer une erreur est d'ordre local. Si cette dernière solution est retenue, renvoyer alors une **Erreur-Service** avec **Bouclage-Détecté** et mettre fin au traitement.

11) Attendre les réponses, puis exécuter la procédure de fusionnement des résultats.

18.5 *Bouclage*

Dans le contexte d'une opération d'annuaire particulière, un bouclage se produit si, à un moment donné, l'opération revient à un état antérieur (comme défini au § 17.4.2). Ceci ne veut pas dire qu'une opération ne peut pas être traitée plusieurs fois par un même DSA, mais signifie que ce DSA ne doit pas traiter plusieurs fois la même opération dans le même état.

Les bouclages sont gérés à l'aide de l'argument informations-Traces, comme défini au § 12.6. Deux stratégies sont définies pour parer aux bouclages: la détection de bouclage et la prévention de bouclage, décrites respectivement aux § 18.5.1 et 18.5.2.

18.5.1 *Détection de bouclage*

La détection de bouclage nécessite qu'un DSA détermine, quand il reçoit une opération entrante, si l'état courant de l'opération figure dans la séquence des états antérieurs enregistrés dans l'argument **informations-Traces** de cette opération. Si c'est le cas, l'opération est en train de boucler et une **erreur-Service** (avec un problème de **bouclage-Détecté**) sera renvoyée. Autrement, le DSA continue de traiter l'opération selon les procédures spécifiées au § 18.4.

18.5.2 *Prévention des bouclages*

La prévention des bouclages nécessite qu'un DSA, immédiatement avant de transmettre une opération à un autre DSA (dans le cadre d'une procédure de chaînage, de multireport ou de décomposition de demande), détermine si l'état de l'opération résultant des traitements du DSA (s'il est connu) apparaît dans la séquence des états antérieurs enregistrés dans l'argument informations-traces de l'opération entrante initiale. L'état de l'opération résultant des traitements du DSA est la valeur d'**Elément-Trace** ajouté à **Informations-Trace** par le DSA récepteur.

Dans le cas où l'opération entrante initiale se trouvait à un port de service (et non à un port de service chaîné), il n'y a pas d'informations de trace, et la procédure de prévention de bouclage n'est pas applicable.

Si l'état de l'opération résultant des traitements du DSA est connu et ne figure pas dans les **informations-Traces**, cette opération pourra, si elle est lancée, provoquer un bouclage. Dans ce cas, la réponse appropriée à l'opération initiale est une **erreur-Service** (avec un problème de **bouclage-Détecté**).

18.6 *Procédure de résolution du nom*

Ce paragraphe décrit en détail la procédure de Résolution du Nom, ses paramètres d'entrée et de sortie et ses éventuels états d'erreur. La figure 7/X.518 présente l'ensemble de la procédure sous la forme d'un diagramme. La procédure de Résolution du Nom appelle deux procédures composantes:

1) Recherche du Contexte de Dénomination (figure 8/X.518);

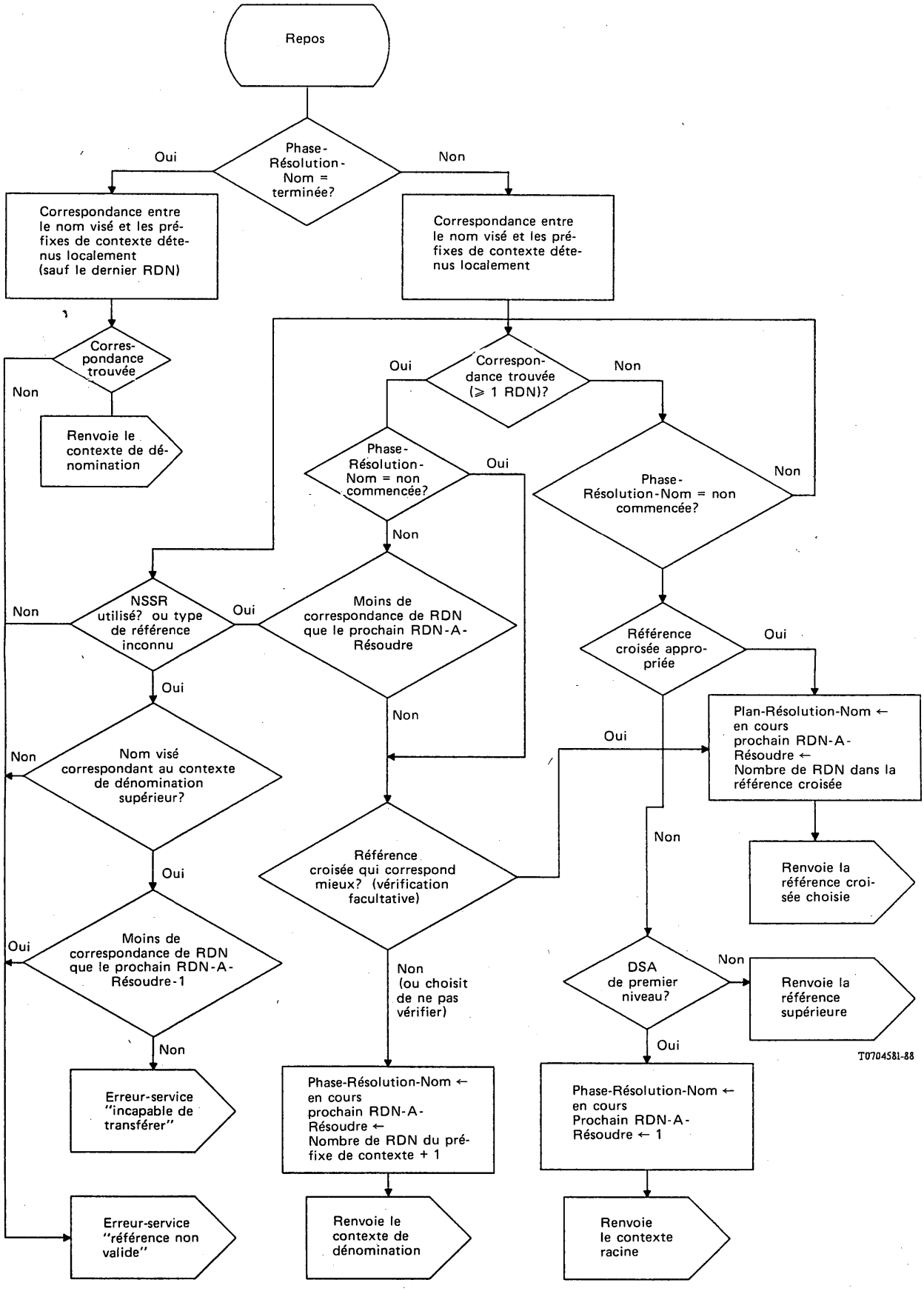


FIGURE 8/X.518

Recherche du contexte de dénomination

2) Résolution Locale du Nom (figure 9/X.518).

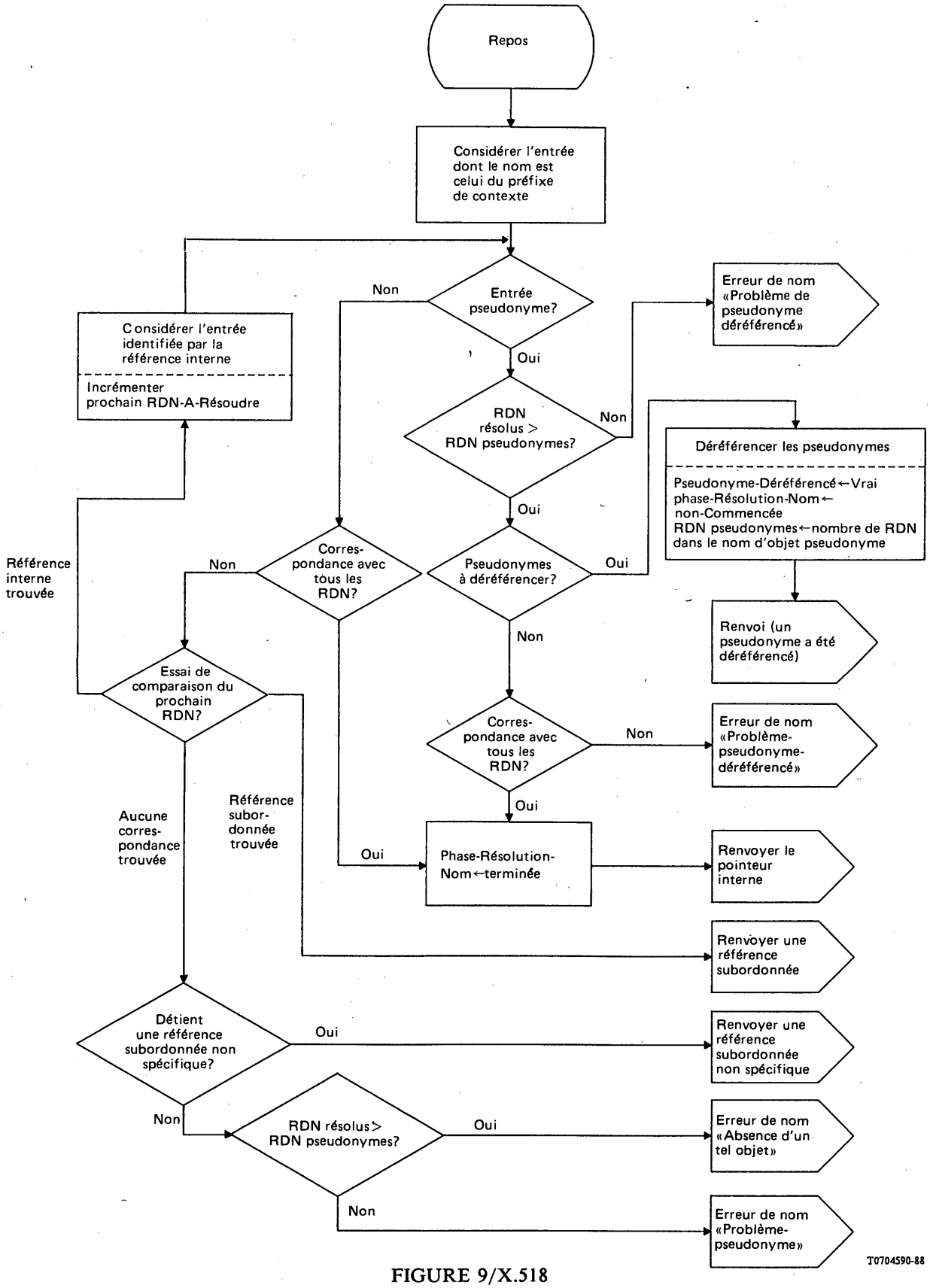


FIGURE 9/X.518

Résolution locale du nom

La procédure de Résolution du Nom renvoie à l'Aiguilleur d'Opérations les résultats des procédures composantes mentionnées ci-dessus, sauf dans les deux cas suivants. Le premier cas est celui où la procédure de Recherche du Contexte de Dénomination identifie un contexte approprié dont il faut poursuivre l'examen, et renvoie le contexte de dénomination local. Le second cas est celui où la procédure de Résolution Locale du Nom indique qu'elle a déréférencé un pseudonyme. Dans le premier cas, la procédure de Résolution du Nom appelle la procédure de Résolution locale du Nom. Dans le second cas, la procédure de Résolution du Nom est réactivée avec le nouveau nom d'objet cible.

18.6.1 *Arguments*

La procédure utilise les arguments suivants:

- le nom de l'objet cible (le nom visé);
- l'avancement de l'opération;
- la valeur de la commande de service **Ne pas déréférencer de pseudonymes**;
- la valeur du paramètre **RDNpseudonymes**;
- la valeur du paramètre **pseudonymeDéréférencé**.

18.6.2 *Résultats*

Deux cas de succès peuvent se présenter:

Dans le premier cas, la procédure renvoie:

- une référence;
- l'avancement de l'opération (mis à jour de façon appropriée);
- une indication de **pseudonyme déréférencé** et, facultativement, des **RDNpseudonymes**.

Dans le second cas, elle renvoie:

- une indication que le contexte de dénomination a été trouvé (avec le pointeur local vers l'entrée);
- l'avancement de l'opération (mis à jour de façon appropriée);
- une indication de **pseudonyme déréférencé** et, facultativement, des **RDNpseudonymes**.

18.6.3 *Erreurs*

L'une des erreurs suivantes peut être renvoyée:

- **erreur de service (incapable de traiter)**;
- **erreur de service (référence non valide)**;
- **erreur de nom (problème de pseudonyme, absence d'un tel objet** ou **problème de déréférençage de pseudonyme)**.

18.6.4 *Procédures*

1) Activer la procédure de Recherche du Contexte de Dénomination.
2) Attendre la réponse de la procédure de Recherche du Contexte de Dénomination.
3) Recevoir les résultats ou l'erreur retournés, c'est-à-dire: Contexte de Dénomination Local trouvé, Référence Distante, Erreur Incapable de Traiter, Erreur de Nom ou Référence non valide.
4) Exécuter les fonctions correspondant au résultat ou à l'erreur renvoyé.
   a) Si le contexte de dénomination local a été trouvé, activer la procédure de Résolution Locale du Nom. Cette procédure peut renvoyer une indication de Référence Interne Trouvée, une Référence Distante, une indication de Pseudonyme Déréférencé ou une Erreur de Nom. Chacune de ces réponses met fin à la Résolution du Nom avec le résultat renvoyé, sauf si un pseudonyme a été déréférencé, auquel cas la procédure est relancée à l'étape 1.
   b) Tout autre résultat est renvoyé à l'Aiguilleur d'Opérations.

18.6.5 *Procédures de Recherche du Contexte de Dénomination*

18.6.5.1 *Introduction*

La figure 8/X.518 représente cette procédure sous la forme d'un diagramme. Une description textuelle est donnée ci-après. Il y est supposé que la valeur courante de l'Avancement de l'Opération est toujours renvoyée au terme de la procédure.

18.6.5.2 *Arguments*

La procédure utilise les arguments suivants:

- le nom d'objet cible (le nom visé);
- l'avancement de l'opération.

18.6.5.3 *Résultats*

Deux cas de succès peuvent se présenter.

Dans le premier cas, la procédure renvoie:

- une référence;
- l'avancement de l'opération (mis à jour de façon appropriée).

Dans le second cas, elle renvoie:

- une indication qu'un contexte de dénomination approprié a été trouvé localement;
- l'avancement de l'opération (mis à jour de façon appropriée).

18.6.5.4 *Erreurs*

L'une des erreurs suivantes peut être renvoyée:

- **erreur de service (incapable de traiter);**
- **erreur de service (référence non valide).**

18.6.5.5 *Procédure*

1) Si la **phase-Résolution-Nom** est positionnée sur **terminée** à l'entrée, essayer de trouver une correspondance entre le nom visé et les préfixes de contexte des contextes de dénomination supérieurs de tous les contextes de dénomination détenus localement. Si une correspondance est trouvée, renvoyer tous les contextes de dénomination appropriés détenus localement. Si aucune correspondance n'est trouvée, renvoyer une **Erreur-Service Référence-Non valide.**

2) Si la **phase-Résolution-Nom** n'est pas positionnée sur **terminée,** essayer de trouver une correspondance entre des préfixes de contexte et une séquence d'un ou de plusieurs RDN de la partie initiale du nom visé. Pour qu'une correspondance soit trouvée, il faut que tous les RDN d'un préfixe de contexte se trouvent en correspondance. Les préfixes de contexte utilisés sont ceux des Contextes de Dénomination sur lesquels ce DSA a autorité de gestion. Dans le cas de correspondances multiples, celle comportant le nombre maximal de RDN en correspondance est choisie.

   Si une correspondance est trouvée, exécuter 3).

   Si aucune correspondance n'est trouvée, exécuter 5).

3) Si la **phase-Résolution-Nom** n'est **pas lancée,** exécuter 4). Si le nombre de RDN dans la partie initiale du nom visé, mis en correspondance comme indiqué en 2) ci-dessus, est supérieur ou égal au composant **prochain RDN-A-Résoudre** d'**Opération-Avancement,** exécuter 4), sinon exécuter 9).

   Si la mise en correspondance est couronnée de succès, exécuter 3). Autrement, renvoyer une erreur de service "incapable de traiter".

4) Le **prochain RDN-A-Résoudre** est mis au nombre de RDN en correspondance plus un et la **phase-Résolution-Nom** est mise à **en-cours.** Le contexte est renvoyé et il est mis fin à cette procédure.

   Pour améliorer la performance, le DSA peut, en option, faire correspondre le nom visé avec les références croisées détenues par le DSA. S'il y a davantage de RDN en correspondance avec une référence croisée qu'avec les préfixes de contexte détenus localement, exécuter l'étape 7).

Remarque - Dans le cas de ce résultat, la procédure de Résolution du Nom appelle la procédure de Résolution Locale du Nom.

5) Si aucune correspondance n'a été trouvée, la valeur de la **phase-Résolution-Nom** est vérifiée. Si la **phase-Résolution-Nom** est **non commencée**, exécuter 6).

   Si la **phase-Résolution-Nom** est **en-cours**, ou **terminée**, exécuter 9).

6) Essayer, en utilisant les préfixes de contexte de Référence Croisée, de trouver une correspondance pour une séquence d'un ou de plusieurs RDN de la partie initiale du nom visé. Dans le cas de correspondances multiples, celle comportant le nombre maximal de RDN en correspondance est choisie.

7) Si une correspondance a été trouvée pour une référence croisée, mettre le **prochain RDN-A-Résoudre** au nombre de RDN de la référence croisée choisie. La référence croisée est renvoyée et il est mis fin à cette procédure.

8) Si aucune correspondance n'est trouvée avec une référence croisée, déterminer si le DSA est un DSA de premier niveau. Si ce n'est pas le cas, il aura une référence supérieure. Retourner cela et mettre fin à la procédure.

   Si le DSA est un DSA de premier niveau, positionner sur un **prochain RDN-A-Résoudre** et **phase-Résolution-Nom** sur **en cours**. Retourner le contexte de dénomination de racine et mettre fin à la procédure.

9) Vérifier la valeur du composant **type-Référence** des **arguments de chaînage**. Si une référence subordonnée non spécifique a été utilisée ou si la demande émanait d'un DUA, exécuter 10); sinon, retourner un problème **erreur de service Référence-non valide** et mettre fin à la procédure.

10) Comparer la portion initiale du nom visé avec les préfixes du contexte (moins leur dernier RDN) des contextes de dénomination détenus localement. Il s'agit effectivement d'une comparaison avec certains contextes de dénomination du supérieur immédiat auprès de ce DSA.

    En l'absence de correspondance, retourner **Erreur de Service Référence-non valide** et mettre fin à la procédure.

    Si on trouve une correspondance, et si le nombre de RDN correspondants est moins élevé que dans **prochain RDN-A-Résoudre-1**, retourner une **Erreur de Service Référence-non valide**; sinon, retourner **Erreur de Service Incapable de traiter**. Mettre fin à la procédure.

18.6.6 *Résolution Locale du Nom*

18.6.6.1 *Introduction*

La Résolution Locale du Nom fait correspondre des RDN du nom visé avec des références internes de connaissances. Elle renvoie une indication Trouvé, Référence Distante, Pseudonyme Déréférencé ou Erreur.

La figure 9/X.518 présente cette procédure sous la forme d'un diagramme. Une description textuelle est donnée ci-après.

18.6.6.2 *Arguments*

La procédure utilise les arguments suivants:

- la référence interne du contexte de dénomination (avec un pointeur vers l'entrée dont le nom est identique au préfixe du contexte);
- le nom de l'objet cible (le nom visé);
- l'avancement de l'opération;
- la valeur de la commande de service "ne pas déréférencer de pseudonymes";
- la valeur du paramètre "RDNpseudonymes";
- la valeur du paramètre "pseudonyme déréférencé".

18.6.6.3 *Résultats*

Trois cas de succès peuvent se présenter.

Dans le premier cas, la procédure renvoie:

- une référence;

- l'avancement de l'opération (mis à jour de façon appropriée).

Dans le second cas, elle renvoie:

- une indication que l'entrée a été trouvée localement;
- l'avancement de l'opération (mis à jour de façon appropriée).

Dans le troisième cas, elle renvoie:

- une indication qu'un pseudonyme a été déréférencé;
- l'avancement de l'opération (remis à "non commencé").

18.6.6.4 *Erreurs*

L'une des erreurs suivantes peut être renvoyée:

- erreur de nom.

18.6.6.5 *Procédure*

Le contexte de dénomination renvoyé par la Recherche du Contexte de Dénomination désignera l'entrée de la racine du sous-arbre. Dans le cas du contexte racine, l'entrée n'est qu'une entrée nulle.

1) Si la référence interne concerne une entrée de pseudonyme, exécuter l'étape 7); autrement, exécuter l'étape 2).

2) Si tous les RDN du nom visé ont été mis en correspondance, l'entrée cible a alors été trouvée. Positionner **phase-Résolution-Nom** sur **terminée**. Un pointeur interne est renvoyé et il est mis fin à la procédure.

   Autrement, l'étape 3) doit être exécutée.

   Remarque - La correspondance peut être réalisée avec le seul préfixe de contexte, ou avec le préfixe de contexte plus des RDN successifs contenus dans des références internes de l'arbre de connaissances.

3) S'il a été trouvé une référence interne subordonnée à l'entrée courante de l'arbre de connaissances (qui correspond au prochain RDN du nom visé), incrémenter alors le **prochain RDN-A-Résoudre**, mettre l'entrée courante sur entrée subordonnée et exécuter à nouveau l'étape 1) de cette procédure.

4) Si l'entrée courante a une référence subordonnée dont le RDN correspond au RDN suivant dans le nom visé, la retourner et mettre fin à la procédure.

5) S'il existe des références subordonnées non spécifiques, subordonnées à l'entrée courante dans l'arbre de connaissances, les retourner comme des références et mettre fin à la procédure.

6) S'il n'est pas trouvé de référence interne, de référence subordonnée ou de référence subordonnée non spécifique, vérifier le nombre de RDN dans le nom visé qui ont été mis en correspondance. S'il a été mis en correspondance davantage de RDN que dans le composant **RDNpseudonyme** de l'**argument de Chaînage**, retourner **Erreur-Nom, absence d'un Tel Objet**. Si moins de RDN ont été mis en correspondance, retourner **Erreur-Nom, problème-Pseudonyme**.

7) Si le nombre de RDN dans le nom visé qui ont été mis en correspondance est inférieur ou égal au composant **RNSpseudonymes** de l'**argument de Chaînage** (le cas échéant), le précédent pseudonyme qui a été déréférencé (le cas échéant) désigne un autre pseudonyme. En pareil cas, retourner **Erreur-Nom, problème-Déréférençage-Pseudonyme**.

8) Si le composant **RDNpseudonymes** manque, ou si le nombre de RDN mis en correspondance est plus grand que le composant **RDNpseudonymes** de l'**argument de Chaînage**, vérifier la commande de service **ne pas DéréférencerPseudonyme**. Si les pseudonymes peuvent être déréférencés, exécuter l'étape 9), sinon l'étape 10).

9) Déréférencer le pseudonyme. Mettre la **phase-Résolution-Nom d'Avancement-Opération** sur **non commencé**. Mettre le composant **pseudonyme-Déréférencé** de l'**argument de Chaînage** sur **TRUE** et **RDNpseudonymes** sur le nombre de RDN de l'attribut **Nom-Objet pseudonyme** de l'entrée de pseudonyme. Mettre l'objet cible sur le nouveau nom. Mettre fin à la procédure (le processus de Résolution-Nom sera recommencé).

10) Si tous les RDN du nom visé ont été mis en correspondance, exécuter l'étape 2). Sinon, retourner **Erreur-Nom, problème-pseudonyme-Déréférencé.**

18.7 *Procédures d'évaluation d'objet*

Les procédures d'évaluation d'objet spécifiées relèvent de deux catégories:

a) les procédures d'évaluation d'un seul objet;

b) les procédures d'évaluation de plusieurs objets.

La figure 10/X.518 représente les procédures d'évaluation d'objet.

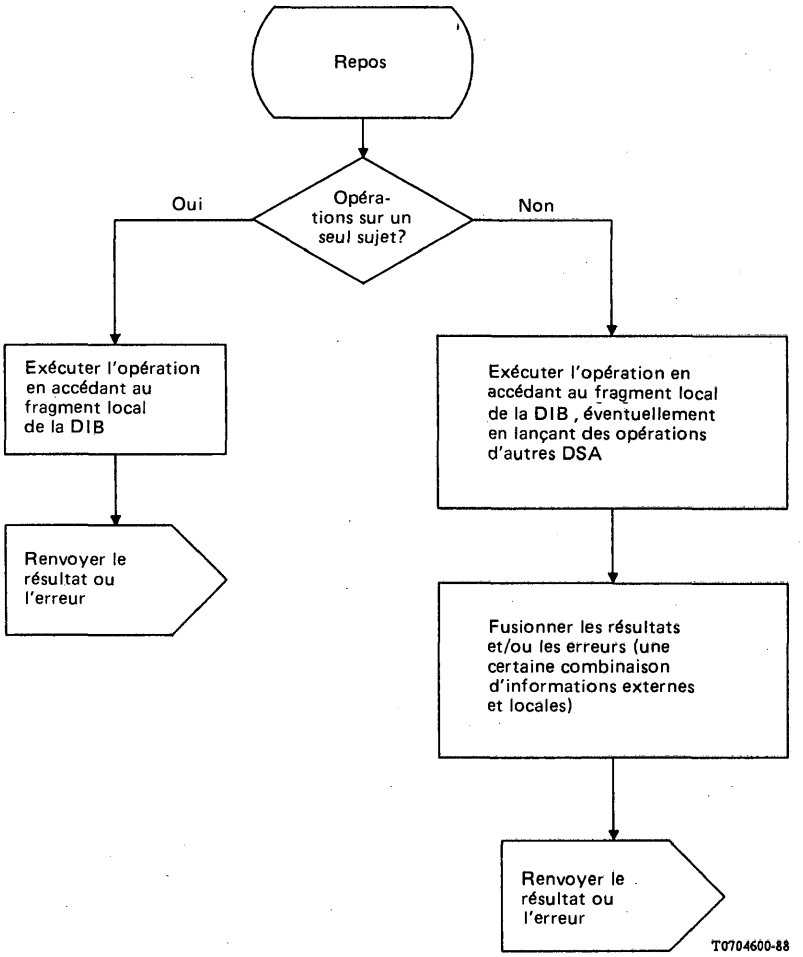


FIGURE 10/X.518

Evaluation et fusionnement des résultats

18.7.1 *Procédures d'évaluation d'un seul objet*

Les procédures d'évaluation d'un seul objet, qui sont communes à la classe des opérations concernées par l'accès à un seul objet, sont exécutées directement, le résultat ou l'erreur étant renvoyé au demandeur.

Ces opérations comprennent la **Lecture**, la **Comparaison**, l'**Ajout d'Entrée**, la **Suppression d'Entrée**, la **Modification d'Entrée** et la **Modification de RDN**, ainsi que leurs contreparties Chaînées.

Les mesures qu'il convient d'appliquer à l'entrée sont décrites au paragraphe approprié de la Recommandation X.511.

Les opérations **Ajout d'Entrée, Suppression d'Entrée** et **Modification de RDN** affectent les connaissances. Si le supérieur immédiat de l'entrée se trouve dans un DSA différent, des références de connaissances externes correctes doivent être maintenues. Il n'appartient pas à la présente Recommandation d'indiquer les moyens d'y parvenir.

La présente Recommandation n'a pas pour objet d'indiquer la façon dont on choisit le DSA qui contient l'entrée créée par l'**Ajout d'Entrée**.

Si le supérieur immédiat d'une entrée devant être créée par l'**Ajout d'Entrée** ou être modifiée par la **Modification de RDN** comporte des références subordonnées non spécifiques, il faut suivre des procédures ne relevant pas de la présente Recommandation pour veiller à ce que deux entrées n'aient pas le même nom spécifique.

Les demandes qui ne peuvent être satisfaites dans ces conditions aboutiront à un échec avec une **Erreur de Mise à Jour** dont le problème **affecte les DSA multiples.**

18.7.2 *Procédures d'évaluation de plusieurs objets*

Les procédures d'évaluation de plusieurs objets, qui sont communes à la classe des opérations concernées par l'accès à plusieurs objets, sont spécifiées dans les paragraphes suivants.

Ces opérations comprennent le **Listage**, la **Recherche** et leurs contreparties Chaînées.

18.7.2.1 *Listage*

Ce paragraphe spécifie la procédure d'évaluation spécifique au **Listage** et au **Listage Chaîné**. (Dans ce qui suit, le terme "Listage" concerne les deux opérations.)

18.7.2.1.1 *Procédure de Listage (I)*

Cette procédure s'applique quand la demande de Listage a un composant de la **Phase-Résolution-Nom** de l'**Avancement-Opération** mis sur **non commencée** ou **en cours** et quand le DSA, après exécution de la Résolution du Nom trouve qu'il détient l'objet de base.

L'objet de base sera noté "e".

1) Prendre chaque subordonnée immédiate de "e" détenue localement, pour former un ensemble local de résultats. Mettre **Entrée-de pseudonyme** et **De l'entrée** dans **Résultat-Listage**, selon le cas.

2) Prendre l'ensemble de références subordonnées non spécifiques et de références subordonnées aux DSA qui détiennent les subordonnées immédiates de "e".

3) Communiquer la sous-demande avec un objet de base = e et **Avancement-Opération** mis sur **terminé** à l'Aiguilleur d'Opérations, qui la transmet ensuite à chaque DSA qui détient des subordonnées immédiates de e.

Remarque - Si le DSA détient des références subordonnées indiquant si l'entrée subordonnée est ou non un pseudonyme et "si **ne pas Utiliser de Copie**" est **FALSE**, cette étape peut être omise pour ces entrées. L'information relative aux subordonnées est directement disponible.

18.7.2.1.2 *Procédure de Listage (II)*

Cette procédure s'applique à une demande de Listage dont le composant **Phase-Résolution-Nom** d'**Avancement-Opération** est mis sur **terminé.**

L'objet de base sera noté "e".

1) Prendre chaque subordonnée immédiate de e détenue localement, pour former un ensemble local de résultats. Mettre **Entrée de pseudonyme** et **De l'entrée** dans **Résultat-Listage**, selon le cas.

2) Communiquer les résultats à l'Aiguilleur d'Opérations qui les transmet au DUA ou au DSA demandeur.

18.7.2.2 *Recherche*

Ce paragraphe spécifie la procédure d'évaluation spécifique à la **Recherche** et à la **Recherche Chaînée**. (Dans ce qui suit, le terme "Recherche" s'applique aux deux.)

A noter que deux situations exigent deux procédures distinctes. La première procédure (§ 18.7.2.2.1) s'applique quand le DSA exécutant la Recherche contient l'**Objet-cible** comme entrée locale. La seconde procédure (§ 18.7.2.2.2) s'applique quand le DSA exécutant la Recherche ne détient pas l'**Objet-cible** mais seulement des subordonnées de l'**Objet-cible.**

18.7.2.2.1 *Procédure de Recherche (I)*

Cette procédure s'applique à une demande de Recherche dont le composant **Phase-Résolution-Nom** d'**Avancement-Opération** est mis sur **non commencée** ou **en cours** et quand le DSA, après exécution de la Résolution du Nom, détermine qu'il détient l'objet-cible.

L'objet de base sera noté "e".

1) Si l'argument de **sous-ensemble** est **Objet de base** ou **Sous-Arbre entier**, appliquer l'argument de filtre spécifié dans la Recherche à l'entrée e, pour former un ensemble de résultats locaux. Retourner les résultats pour Fusionnement des Résultats. Si l'argument de sous-ensemble est **Objet de base**, mettre fin à la procédure, sinon continuer à 2).

2) Si l'argument de sous-ensemble est **unNiveau** ou **Sous-Arbre-entier**, former un ensemble E à partir des subordonnées immédiates de e détenues localement, sauf que:

   Si des pseudonymes doivent être déréférencés, c'est-à-dire que le paramètre Pseudonyme-Recherche est TRUE, les entrées de pseudonyme qui sont trouvées sont traitées au § 5) ci-après et ne contribuent pas à ces résultats.

   Appliquer les arguments de filtre à E en vue de fournir un sous-ensemble filtré E' de E; retourner cet ensemble E' de résultats locaux pour Fusionnement de Résultats.

3) D'autres subordonnées de e peuvent résider dans d'autres DSA; en pareil cas, elles seront référencées comme références subordonnées ou subordonnées non spécifiques. Pour chaque DSA ainsi référencé, préparer une nouvelle Recherche avec Objet-cible = e, et avec la **phase-Résolution-Nom** d'**Avancement-Opération** mis sur **terminé**. Retourner chaque sous-demande Recherche à l'Aiguilleur d'Opérations qui les transmet. Si un résultat d'erreur est retourné d'une sous-demande, il n'en est pas tenu compte comme si aucune demande n'avait été envoyée.

4) Si l'argument de **sous-ensemble** est **unNiveau**, la Recherche est à présent achevée, mettre donc fin à la procédure.

   Si l'argument de **sous-ensemble** est **Sous-Arbre-entier**:

   si l'ensemble E du § 2) est vide, le sous-arbre entier détenu dans ce DSA a été recherché; mettre donc fin à la procédure;

   sinon continuer comme suit le traitement:

   que chaque entrée qui était dans l'ensemble E soit désignée e. Répéter la procédure de Recherche du § 2), pour chaque entrée e.

5) Si des pseudonymes doivent être déréférencés, les entrées de pseudonyme trouvées à l'étape 2) sont placées dans l'ensemble D. Pour chaque entrée d de D, déréférencer le pseudonyme, et formuler une nouvelle Recherche avec la **phase-Résolution-Nom** mise sur **non commencée**, et l'**Objet-cible** créé à partir de l'attribut **Nom-Objet-pseudonyme** et de l'ancien nom **Objet-cible**.

   Si l'argument de **sous-ensemble** est **unNiveau**, le mettre sur **Objet-base** dans la nouvelle sous-demande, sinon, le mettre sur **Sous-Arbre-entier**.

   Si un résultat d'erreur est retourné de la sous-demande, il n'en est pas tenu compte, comme si aucune demande n'avait été faite.

18.7.2.2.2 *Procédures de Recherche (II)*

Cette procédure s'applique à une demande de Recherche dont le composant **Phase-Résolution-Nom** de l'**Avancement-Opérations** est mis sur **terminé**.

L'objet-cible sera noté "e".

Pour chaque subordonnée immédiate e' de e détenue localement, formuler une nouvelle demande avec Objet-cible = e'. Si l'argument de **sous-ensemble** est **unNiveau**, le mettre sur **Objet de base**, sinon le laisser comme **Sous-Arbre-entier**. Exécuter ensuite la procédure définie aux étapes 1) à 5) au § 18.7.2.2.1. S'il n'existe pas de telles subordonnées, retourner l'**erreur de service incapable De Traiter**.

18.8 *Procédure de fusionnement des résultats*

Cette procédure est appelée quand il existe des résultats et/ou des erreurs externes. Il peut également exister un résultat interne. Tous les résultats et erreurs sont supposés être conservés dans le DSA jusqu'à la fin de la procédure.

Les informations externes peuvent provenir d'un chaînage, d'un multireport ou d'une décomposition de la demande.

Dans le cas d'un chaînage, il y aura un résultat ou une erreur unique. Dans le cas d'un multireport, il peut n'y avoir aucun résultat, un résultat ou plusieurs résultats identiques. De plus, il peut y avoir quelques erreurs. S'il y a plusieurs résultats, ils sont tous, sauf un, arbitrairement mis au rebut. Un résultat est toujours renvoyé de préférence à une erreur. En l'absence de résultat, une erreur est renvoyée, avec les exceptions suivantes:

i) Si une **Référence non valide** a été renvoyée, la référence est notée en tant que telle et le DSA peut soit utiliser une autre référence externe appropriée pour poursuivre la demande, soit renvoyer une **ditError** au demandeur. (Le traitement des références externes non valables dépasse le cadre de la présente Recommandation.)

ii) Dans le cas d'un multireport, les erreurs **incapable De Traiter** ne doivent pas être prises en considération, à moins que toutes les réponses soient de ce type, auquel cas il faut renvoyer une **Erreur de Nom absence d'un Tel Objet** au répondeur. Si un résultat au moins est renvoyé, on peut alors faire abstraction de toutes les erreurs.

iii) Dans le cas de renvois de référence, ceux-ci ne doivent pas être traités comme des erreurs et une suite peut leur être donnée.

Si le fusionnement est requis à la suite d'une décomposition de demande, il revient à constituer la réunion logique des résultats.

Dans le cas d'une décomposition où il y a à la fois des résultats et des erreurs à fusionner, un résultat incomplet est retourné au demandeur.

Un DSA peut, à ce stade, choisir d'extraire des renvois de référence des résultats et des erreurs entrants qui doivent être fusionnés. Il peut alors décider de pousser l'exploration d'une partie ou de la totalité de ces derniers, auquel cas des opérations sont chaînées. L'ancien résultat devra être sauvegardé et fusionné ultérieurement avec les résultats ou les erreurs produits par les chaînages.

Le traitement des signatures qui peuvent figurer avec les résultats renvoyés, est spécifié au § 18.9.2 ci-après.

18.9 *Procédures d'authentification répartie*

Ce 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 du demandeur, qui est assurée sous une forme non protégée (fondée sur la simple identité) ou protégée (fondée sur des signatures numériques); et
- authentification des résultats, qui est protégée de la même façon (fondée également sur des signatures numériques).

18.9.1 *Authentification du demandeur*

18.9.1.1 *Authentification fondée sur l'identité*

Le service d'authentification fondée sur l'identité permet aux DSA d'authentifier le demandeur initial d'informations à des fins d'exécution de contrôles d'accès locaux. Les DSA qui désirent exploiter ce service doivent adopter la procédure suivante:

- DSA demandant l'authentification d'une demande du DAP: le DSA se procure le nom spécifique du demandeur via les procédures de Liaison au moment de l'établissement d'une association par le DUA (DUA ou DSA). Le succès de ces procédures ne préjuge en aucune façon du niveau d'authentification qui peut être par la suite requis pour des opérations de traitement utilisant cette association.
- Le DSA avec lequel le DUA établit l'association doit insérer le nom spécifique du demandeur dans le champ demandeur de l'Argument-Chaînage de toutes les opérations subséquentes 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 déterminés localement (par un mécanisme défini au niveau local). Si le résultat n'est pas satisfaisant, une **Erreur-Sécurité** peut être renvoyée avec un **Problème-Sécurité** mis sur **droits-Accès-Insuffisants.**

18.9.1.2 *Authentification du demandeur fondée sur la signature*

Le service d'authentification du demandeur fondée sur la signature permet à un DSA d'authentifier (de manière protégée) l'auteur d'une demande de service particulière. Les procédures à exécuter par un DSA pour réaliser ce service sont décrites dans ce paragraphe.

Le service d'authentification fondée sur la signature est lancé par un DUA utilisant la variante **SIGNEE** d'une demande de service avec option de signature.

Un DSA, à la réception d'une demande signée de la part d'un autre DSA, détruit la signature de ce DSA avant de traiter l'opération. En supposant que le résultat de toute vérification de signature s'avère satisfaisant, le DSA continue le traitement de l'opération. Si, durant ce traitement, le DSA a besoin d'effectuer un chaînage, un multireport ou une décomposition de la demande, l'ensemble d'arguments de chaque opération chaînée associée sera construit comme suit:

- le DSA forme un ensemble d'arguments qui peuvent être facultativement signés; l'ensemble d'arguments comprend l'ensemble d'arguments signés entrant, complété d'un argument de **chaînage modifié**.

Dans le cas où le DSA est capable de contribuer aux informations de la réponse, l'authentification du demandeur, 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 qui sont uniquement communiquées sous réserve d'authentification du demandeur, une **erreur-Sécurité** est renvoyée avec un **Problème-Sécurité** mis sur **Protection-Demandée**.

18.9.2 *Authentification des résultats*

Ce service est fourni pour permettre aux demandeurs d'opérations d'annuaire (des DUA ou des DSA) de vérifier (de manière protégée, en utilisant des techniques de signatures numériques), la source des résultats. Le service d'authentification des résultats peut être demandé indépendamment de l'utilisation de l'authentification du demandeur.

Le service d'authentification des résultats est lancé en utilisant la valeur **signée** du composant **demande-Protection**, contenu dans l'ensemble d'arguments des opérations d'annuaire; un DSA recevant une opération pour laquelle cette option a été choisie peut alors signer tous les résultats subséquents. L'option **signée** de la **demande de protection** sert à indiquer au DSA la préférence des demandeurs; le DSA peut alors signer ou non tout résultat subséquent.

Dans le cas où un DSA effectue un chaînage, un multireport ou une décomposition d'une telle demande, 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 réponses partielles non fusionnées (signées ou non signées), ou plus; zéro ou plusieurs réponses de cet ensemble peuvent être signées et zéro ou une non signée. Dans le cas où figure un résultat partiel non signé, ce peut être en fait l'assemblage d'une ou de plusieurs réponses partielles non signées, reçues d'autres DSA, fournies par ce DSA ou provenant de ces deux sources.

ANNEXE A

(à la Recommandation X.518)

Définitions ASN.1 des opérations réparties

Cette annexe fait partie de la Recommandation.

Elle comprend toutes les définitions concernant le type, la valeur et la macro ASN.1 contenues dans cette Recommandation, sous la forme du module ASN.1 **DistributedOperations**.

```
DistributedOperations {joint-iso-ccitt ds(5) modules(1) distributedOperations(3)}
DEFINITIONS ::=
BEGIN
```

```
EXPORTS
        DirectoryRefinement, chainedReadPort, chainedSearchPort, chainedModifyPort,
        DSABind, DSABindArgument,
        DSAUnbind,
        ChainedRead, ChainedCompare, ChainedAbandon,
        ChainedList, ChainedSearch,
        ChainedAddEntry, ChainedRemoveEntry,
        ChainedModifyEntry, ChainedModifyRDN,
        DsaReferral, ContinuationReference;

IMPORTS
        InformationFramework, abstractService, distributedOperations,
        directoryObjectIdentifiers, selectedAttributeTypes
            FROM    UsefulDefinitions {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0)}

        DistinguishedName, Name, RelativeDistinguishedName
            FROM    InformationFramework informationFramework

        id-ot-dsa, id-pt-chained-read, id-pt-chained-search, id-pt-chained-modify,
            FROM    DistributedDirectoryObjectIdentifiers, distributedDirectoryObjectIdentifiers

        PresentationAddress
            FROM    SelectedAttributeTypes selectedAttributeTypes

        directory, readPort, searchPort, modifyPort
        DirectoryBind,
        ReadArgument, ReadResult,
        CompareArgument, CompareResult,
        Abandon
        ListArgument, ListResult,
        SearchArgument, SearchResult,
        AddEntryArgument, AddEntryResult,
        RemoveEntryArgument, RemoveEntryResult,
        ModifyEntryArgument, ModifyEntryResult,
        ModifyRDNArgument, ModifyRDNResult,
        Abandoned, AttributeError, NameError, ServiceError, SecurityError, UpdateError
        OPTIONALLY-SIGNED, SecurityParameters
            FROM    DirectoryAbstractService directoryAbstractService

-- objects and ports --

DirectoryRefinement ::= REFINE directory AS
        dsa RECURRING
            readPort            [S]  VISIBLE
            searchPort          [S]  VISIBLE
            modifyPort          [S]  VISIBLE
            chainedReadPort          PAIRED WITH dsa
            chainedSearchPort        PAIRED WITH dsa
            chainedModifyPort        PAIRED WITH dsa

dsa OBJECT
            PORTS { readPort            [S],
                    searchPort          [S],
                    modifyPort          [S],
                    chainedReadPort,
                    chainedSearchPort,
                    chainedModifyPort}
        ::= id-ot-dsa

chainedReadPort PORT
        ABSTRACT OPERATIONS {
            ChainedRead, ChainedCompare,
            ChainedAbandon}
        ::= id-pt-chained-read

chainedSearchPort PORT
        ABSTRACT OPERATIONS {
            ChainedList, ChainedSearch}
        ::= id-pt-chained-search
```

```
chainedModifyPort PORT
        ABSTRACT OPERATIONS {
                ChainedAddEntry, ChainedRemoveEntry,
                ChainedModifyEntry, chainedModifyRDN}
        ::=     id-pt-chained-modify

DSABind     ::=     ABSTRACT-BIND
        TO      {chainedRead,
                chainedSearch,
                chainedModify}
        DirectoryBind

DSAUnbind ::=   UNBIND
        FROM            {chainedRead,
                        chainedSearch,
                        chainedModify}

-- operations, arguments and results --

ChainedRead     ::=
        ABSTRACT-OPERATION
                ARGUMENT    OPTIONALLY-SIGNED       SET{
                                                    ChainingArgument,
                                                    [0]   ReadArgument}
                RESULT      OPTIONALLY-SIGNED       SET{
                                                    ChainingResult,
                                                    [0]   ReadResult}
                ERRORS {
                        DsaReferral, Abandoned, AttributeError, NameError,
                        ServiceError, SecurityError}

ChainedCompare      ::=
        ABSTRACT-OPERATION
                ARGUMENT    OPTIONALLY-SIGNED       SET{
                                                    ChainingArgument,
                                                    [0]   CompareArgument}
                RESULT      OPTIONALLY-SIGNED       SET{
                                                    ChainingResult,
                                                    [0]   CompareResult}
                ERRORS {
                        DsaReferral, Abandoned, AttributeError, NameError,
                        ServiceError, SecurityError}

ChainedAbandon      ::=     Abandon

ChainedList     ::=
        ABSTRACT-OPERATION
                ARGUMENT    OPTIONALLY-SIGNED       SET{
                                                    ChainingArgument,
                                                    [0]   ListArgument}
                RESULT      OPTIONALLY-SIGNED       SET{
                                                    ChainingResult,
                                                    [0]   ListResult}
                ERRORS {
                        DsaReferral, Abandoned, AttributeError, NameError,
                        ServiceError, SecurityError}

ChainedSearch   ::=
        ABSTRACT-OPERATION
                ARGUMENT    OPTIONALLY-SIGNED       SET{
                                                    ChainingArgument,
                                                    [0]   SearchArgument}
                RESULT      OPTIONALLY-SIGNED       SET{
                                                    ChainingResult,
                                                    [0]   SearchResult}
                ERRORS {
                        DsaReferral, Abandoned, AttributeError, NameError,
                        ServiceError, SecurityError}
```

```
ChainedAddEntry ::=
    ABSTRACT-OPERATION
        ARGUMENT   OPTIONALLY-SIGNED      SET{
                                          ChainingArgument,
                                          [0]   AddEntryArgument}
        RESULT     OPTIONALLY-SIGNED      SET{
                                          ChainingResult,
                                          [0]   AddEntryResult}
        ERRORS {
            DsaReferral, Abandoned, AttributeError, NameError,
            ServiceError, SecurityError, UpdateError}

ChainedRemoveEntry ::=
    ABSTRACT-OPERATION
        ARGUMENT   OPTIONALLY-SIGNED      SET{
                                          ChainingArgument,
                                          [0]   RemoveEntryArgument}
        RESULT     OPTIONALLY-SIGNED      SET{
                                          ChainingResult,
                                          [0]   RemoveEntryResult}
        ERRORS {
            DsaReferral, Abandoned, NameError,
            ServiceError, SecurityError, UpdateError}

ChainedModifyEntry ::=
    ABSTRACT-OPERATION
        ARGUMENT   OPTIONALLY-SIGNED      SET{
                                          ChainingArgument,
                                          [0]   ModifyEntryArgument}
        RESULT     OPTIONALLY-SIGNED      SET{
                                          ChainingResult,
                                          [0]   ModifyEntryResult}
        ERRORS {
            DsaReferral, Abandoned, AttributeError, NameError,
            ServiceError, SecurityError, UpdateError}

ChainedModifyRDN ::=
    ABSTRACT-OPERATION
        ARGUMENT   OPTIONALLY-SIGNED      SET{
                                          ChainingArgument,
                                          [0]   ModifyRDNArgument}
        RESULT     OPTIONALLY-SIGNED      SET{
                                          ChainingResult,
                                          [0]   ModifyRDNResult}
        ERRORS {
            DsaReferral, Abandoned, NameError,
            ServiceError, SecurityError, UpdateError}

-- errors and parameters --

DSAReferral ::=
    ABSTRACT-ERROR
        PARAMETER SET {
            [0]  ContinuationReference,
            contextPrefix  [1]  DistinguishedName OPTIONAL}

-- common arguments/results --

ChainingArguments ::= SET {
    originator          [0]  DistinguishedName OPTIONAL,
    targetObject        [1]  DistinguishedName OPTIONAL,
    operationProgress   [2]  OperationProgress DEFAULT {not Started},
    traceInformation    [3]  TraceInformation,
    aliasDereferenced   [4]  BOOLEAN DEFAULT FALSE,
    aliasedRDNs         [5]  INTEGER OPTIONAL,
        -- absent unless aliasDereferenced is TRUE
```

```
        returnCrossRefs     [6]  BOOLEAN DEFAULT FALSE,
        referenceType       [7]  ReferenceType DEFAULT superior,
        info                [8]  DomainInfo OPTIONAL,
        timeLimit           [9]  UTCTime OPTIONAL,
                            [10] SecurityParameters DEFAULT { }}

ChainingResults     ::=  SET {
        info                [0]  DomainInfo OPTIONAL,
        crossReferences     [1]  SEQUENCE OF CrossReference OPTIONAL,
                            [2]  SecurityParameters DEFAULT { }}

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

ReferenceType       ::=  ENUMERATED {
                superior                    (1),
                subordinate                 (2),
                cross                       (3),
                nonSpecificSubordinate      (4)}

TraceInformation    ::=  SEQUENCE OF
        SEQUENCE {
                targetObject    Name,
                dsa Name,
                OperationProgress}

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

DomainInfo          ::=  ANY

ContinuationReference    ::=  SET {
        targetObject        [0]  Name,
        aliasedRDNs         [1]  INTEGER OPTIONAL,
        operationProgress   [2]  OperationProgress,
        rdnsResolved        [3]  INTEGER OPTIONAL,
        referenceType       [4]  ReferenceType OPTIONAL,
                            -- only present in the DSP --
        accessPoints        [5]  SET OF AccessPoint }

AccessPoint         ::=  SET {
        ae-title            [0]  Name,
        address             [1]  PresentationAddress }
```

ANNEXE B

(à la Recommandation X.518)

Modélisation des connaissances

Cette annexe ne fait pas partie de la Recommandation.

B.1 *Exemple de modélisation de connaissances*

L'exemple suivant représente les informations de connaissance qui devraient être tenues à jour par le DSA représenté sur la figure 5/X.518 (§ 9). La figure 5/X.518 représente un DIT hypothétique divisé logiquement en cinq Contextes de Dénomination (A, B, C, D et E) et physiquement répartis entre trois DSA (DSA1, DSA2, DSA3). Dans cet exemple, DSA1 contient le contexte C, DSA2 les contextes A, B et E, et DSA3 le contexte D.

Les abréviations suivantes ont été utilisées dans les figures B-1/X.518 à B-3/X.518:

SUPR: référence supérieure

SUBR: référence subordonnée

INTR: référence interne

NSSR: référence subordonnée non spécifique

CROSSR: référence croisée

DSAn: Nom Spécifique du DSAn

PS: Adresse de Présentation

CP: préfixe de contexte

RDN: nom spécifique Relatif

DSA: nom spécifique d'un DSA

PTR: Pointeur

AON: Nom d'Objet Pseudonyme

Remarque - Les figures suivantes sont uniquement destinées à fournir une représentation graphique des contextes définis dans cette annexe. La façon dont les informations de connaissance sont effectivement enregistrées et gérées dans une réalisation particulière de DSA est une question locale qui n'entre pas dans le cadre de la présente Recommandation.

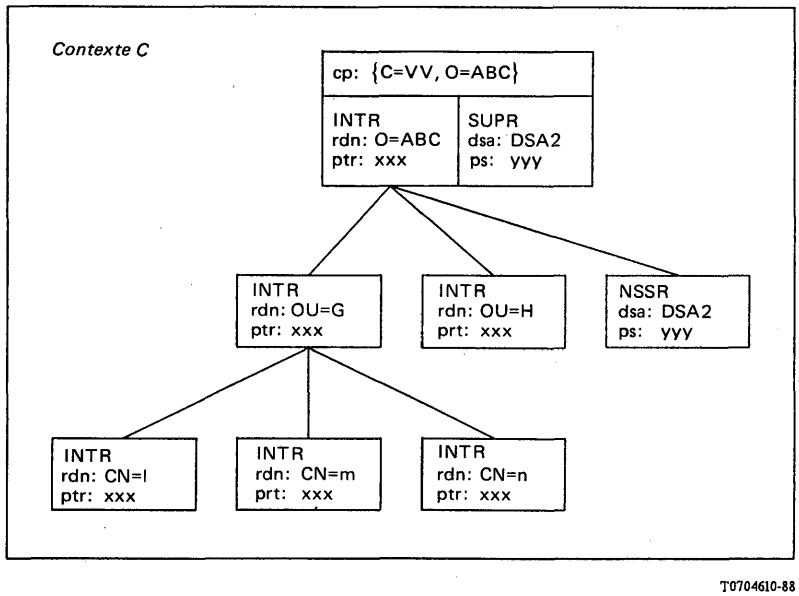


FIGURE B-1/X.518

Informations de connaissance de DSA1

La figure B-1/X.518 représente les informations de connaissance qui doivent être détenues par DSA1. Elles doivent comporter les préfixes de contexte et l'ensemble de références suivants:

Préfixes de contexte: {C=WW, O=ABC}, contexte C.

Références croisées: { }

Références supérieures: {DSA2, adresse de présentation de DSA2}

Références internes du Contexte C: {C=WW, O=ABC},
{OU=G}, {OU=H},
{OU=G, CN=l},
{OU=G, CN=m},
{OU=G, CN=n}.

Références subordonnées: { }

Références subordonnées
non spécifiques: {DSA2, adresse de présentation de DSA2}

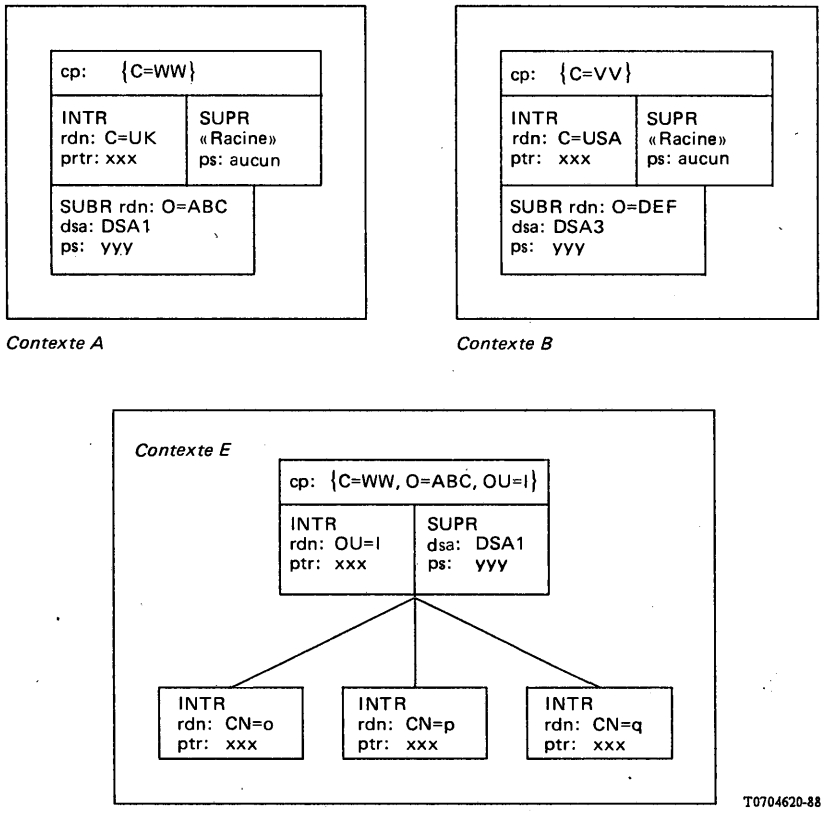


FIGURE B-2/X.518

Informations de connaissance de DSA2

La figure B-2/X.518 représente les informations de connaissance qui doivent être détenues par DSA2. Elles doivent comprendre les préfixes de contexte et l'ensemble de références suivants:

Préfixes de contexte: {C=WW}, contexte A
{C=VV}, contexte B
{C=WW, O=ABC, OU=I}, contexte E.

Références croisées: { }

Références supérieures: { }

Références internes
du contexte A: {C=WW}

Références internes
du contexte B: {C=VV}

Références internes
du contexte E: {C=WW, O=ABC, OU=I},
{CN=o},
{CN=p},
{CN=q}.

Références subordonnées
du contexte A: {C=WW, O=ABC}

Références subordonnées
du contexte B: {C=VV, O=DEF}

Références subordonnées
non spécifiques: { }

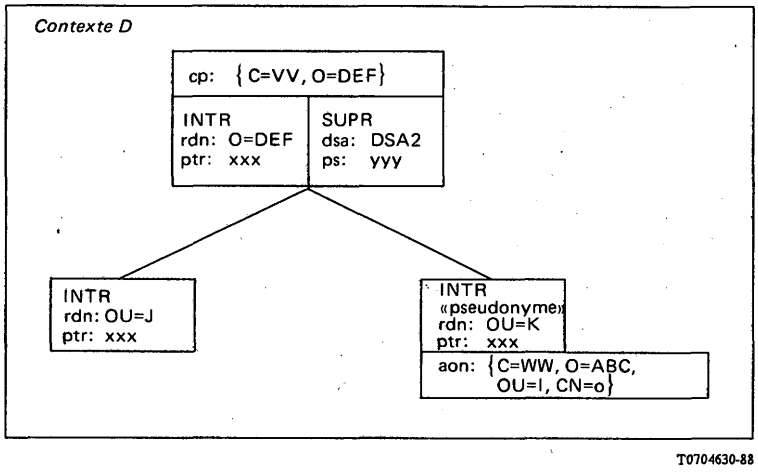


FIGURE B-3/X.518

Informations de connaissance de DSA3

La figure B-3/X.518 représente les informations de connaissance qui doivent être détenues par DSA3. Elles doivent inclure les préfixes de contexte et l'ensemble de références suivants:

Préfixes de contexte:	{C=VV, O=DEF}, contexte D
Références croisées:	{{C=WW, O=ABC, OU=H}, DSA1, adresse de présentation de DSA1} (non représenté sur la figure ci-dessus)
Références supérieures:	{DSA2, adresse de présentation de DSA2}
Références internes du contexte D:	{DSA1, adresse de présentation de DSA1} {C=VV, O=DEF}, {OU=J}, {OU=K} pseudonyme de {C=WW, O=ABC, OU=I, CN=o} (les informations de pseudonyme ne font pas partie des connaissances)
Références subordonnées:	{ }
Références subordonnées non spécifiques:	{ }

B.2 *Exemple de résolution répartie du nom*

Voici un exemple montrant comment la Résolution Répartie du Nom est utilisée en vue de traiter différentes demandes adressées à l'annuaire. L'exemple est fondé sur le DIT hypothétique représenté sur la figure 5/X.518 (§ 9) et la ou les configurations correspondantes de DSA représentées sur les figures B-1/X.518 à B-3/X.518 (annexe B).

En supposant un mode de propagation de chaînage, les demandes suivantes adressées au DSA1 seraient traitées comme suit:

1) Demande avec un nom spécifique {C=WW, O=ABC, OU=G, CN=l}.

   - Correspondra au contexte de préfixe {C=WW, O=ABC} du contexte C sur lequel DSA1 a autorité de gestion. La résolution du nom commencera donc dans DSA1 par le contexte C.
   - La résolution du nom se poursuivra vers le bas dans le contexte C par la mise en correspondance de chaque RDN restant, jusqu'à avoir situé CN=l.

2) Une demande avec un nom spécifique {C=WW, O=JPR}.

- Ne correspondra à aucun préfixe de contexte détenu par DSA1, qui utilisera donc sa référence supérieure pour transmettre la demande à son DSA supérieur, DSA2.
- Dans DSA2, la demande correspondra au préfixe de contexte {C=WW} et la résolution du nom y commencera par le contexte A.
- La résolution du nom ne trouvera pas de subordonné de C=WW à faire correspondre au RDN O=JPR; la demande aboutira donc à un échec et le nom sera déterminé comme nom non valide (c'est-à-dire une référence à un objet non existant).

3) Une question avec un nom spécifique {C=VV, O=DEF, OU=K}.

- Ne correspondra à aucun préfixe de contexte détenu par DSA1.
- DSA1 transmettra donc la demande à son DSA supérieur, DSA2.
- La demande correspondra au contexte de préfixe {C=VV} du contexte B détenu par DSA2. La résolution du nom commencera donc dans DSA2 par le contexte B.
- En essayant de trouver une correspondance pour O=DEF, la résolution du nom trouvera une référence subordonnée indiquant que {C=VV, O=DEF} est le point de départ d'un nouveau contexte détenu dans DSA3.
- La résolution du nom continuera dans DSA3 jusqu'à ce que {C=VV, O=DEF, CN=K} ait été situé.
- En supposant que les pseudonymes doivent être déréférencés, un nouveau nom sera construit en utilisant le nom de pseudonyme contenu dans l'entrée {C=VV, O=DEF, CN=K}. Le nouveau nom résultant sera: {C=WW, O=ABC, OU=I, CN=o}.
- DSA3 reprendra le traitement de la demande en utilisant le nouveau nom obtenu par déréférençage.

ANNEXE C

(à la Recommandation X.518)

Mise en oeuvre répartie de l'authentification

Cette annexe ne fait pas partie de la Recommandation.

C.1 *Résumé*

Le modèle de Sécurité est défini au § 10 de la Recommandation X.501. Les principales caractéristiques du modèle sont résumées ci-après.

- L'authentification simple de l'initiateur de l'opération n'est pas assurée dans le DSP.
- L'authentification poussée, par signature de la demande et du résultat, est assurée dans le DSP.
- Le chiffrement de la demande ou du résultat n'est pas assuré dans le DSP.
- L'authentification des erreurs, y compris des renvois de référence, n'est pas assurée dans le DSP.

La présente annexe décrit la mise en oeuvre du DSA "b" dans l'annuaire réparti. Elle utilise la terminologie et la notation de la Recommandation X.509.

C.2 *Authentification simple*

L'authentification du DUA fait partie intégrante de l'Opération Liaison du DAP. Seul le nom du DUA sera donc véhiculé dans le DSP, dans le champ "initiateur" de l'Argument de Chaînage.

C.3 *Modèle de l'authentification répartie*

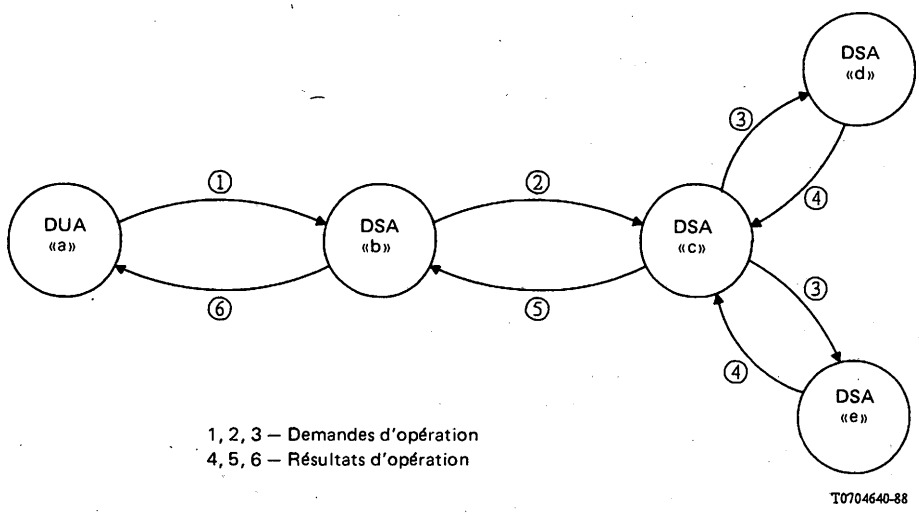


FIGURE C-1/X.518

Modèle d'authentification répartie

La figure C-1/X.518 représente le modèle à utiliser pour spécifier les procédures d'authentification répartie. Le modèle détermine la séquence des flux d'informations, dans le cas général d'une opération de listage ou de recherche. L'opération est considérée comme provenant du DUA "a", citant un objet cible qui réside dans le DSA "c"; l'exécution de l'opération implique les DSA "b", "c", "d" et "e".

Le DUA "a" contacte initialement un DSA quelconque (le DSA "b") qui ne détient pas l'objet cible, mais qui est capable de naviguer, via un chaînage, vers le DSA (DSA "c") détenant cet objet cible. Si tous les DSA fonctionnent en mode de renvoi de référence, le modèle est alors notablement simplifié, et chaque échange DUA/DSA est équivalent, en termes d'authentification, à l'interaction entre le DUA "a" et le DSA "b".

C.4 *DUA à DSA*

L'authentification du demandeur est réalisée comme conséquence de l'échange (1) de la figure C-1/X.518; la procédure d'authentification est la suivante:

Soit:

OA = l'Argument de l'Opération, c'est-à-dire Recherche, Lecture, Comparaison, etc. L'Argument étant tel que défini dans la partie 3; et

a{OA} = l'Argument de l'Opération signé par le DUA "a".

L'authentification sera déterminée par la vérification de la signature.

C.5 *Transfert du DAP au DSP*

Cette procédure est effectuée par le DSA "b" de la figure C-1/X.518 et représente le transfert de l'identité signée de l'initiateur du DSA au DSP.

Le DSA "b" formule l'Argument de Chaînage approprié, tel que décrit au § 12.3 de la présente Recommandation et le combine avec l'Argument d'Opération du DAP, formant ainsi une Opération Chaînée, par exemple: Lecture Chaînée, Recherche Chaînée, Listage Chaîné, etc., du DSP. L'Opération Chaînée ainsi formée sera signée avant d'être communiquée à d'autres DSA (le DSA "c" sur la figure C-1/X.518). La structure de données peut être représentée comme suit:

b{ChA, a{OA}} = l'Opération Chaînée signée par le DSA "b"

où:

ChA = Argument de Chaînage.

Les informations d'authentification véhiculées dans le DSP entre deux DSA [libellées échange (2) sur la figure C-1/X.518] comprennent donc deux parties:

- l'Argument d'Opération signé par l'initiateur, qui permet l'authentification de cet initiateur;
- l'Opération Chaînée, signée par le DSA expéditeur, qui permet l'authentification de ce DSA expéditeur.

C.6 *Chaînage au travers de DSA intermédiaires*

Cette procédure serait effectuée par le DSA "c" dans le modèle décrit à la figure C-1/X.518. Le DSA "c" abandonne la signature fournie par le DSA expéditeur (DSA "b" sur la figure C-1/X.518) et modifie l'Argument de Chaînage, comme décrit au § 12.3 de la présente Recommandation. Le DSA "c" combinera alors l'Argument de Chaînage modifié avec l'Argument de l'Opération signé et signera le résultat pour créer une Opération Chaînée signée, modifiée. Ceci peut être représenté par:

c{ChA',a{OA}} = l'Opération Chaînée signée par le DSA "c"

où:

ChA' = l'Argument de Chaînage modifié.

L'Opération Chaînée modifiée est représentée sur la figure C-1/X.518 par l'échange (3). Selon la nature de l'opération et selon le type de connaissances détenues, le DSA "c" peut effectuer la décomposition de la demande avant de chaîner ou de multireporter toute(s) opération(s) résultante(s). Ceci a été représenté sur la figure C-1/X.518 par le DSA "c", qui envoie des opérations aux DSA "d" et "e" ; dans chaque cas, la procédure d'identification est identique.

C.7 *Authentification des résultats*

Le service d'authentification des résultats est demandé par un initiateur d'une opération d'annuaire, à l'aide de l'option **signée** dans le **Paramètre de Sécurité demande-Protection**. Pour fournir une réponse à une telle demande, un DSA peut, facultativement, décider de signer n'importe quel résultat ou la totalité d'entre eux: le service d'authentification des résultats n'assure pas l'authentification des réponses d'erreur.

Dans le contexte d'un DSA particulier, traitant les résultats d'un nombre arbitraire de DSA (dont chacun est associé à une demande de service particulière), les cas suivants sont possibles:

- Le DSA fournit un ensemble complet de résultats d'une opération, sans nécessiter l'exécution d'aucune fonction de fusionnement (représenté par les DSA "d" et "e" sur la figure C-1/X.518).
- Le DSA collationne les résultats locaux (originaires de ce DSA), avec les résultats d'un ou de plusieurs autres DSA (représenté par le DSA "c" sur la figure C-1/X.518).
- Le DSA chaîne un résultat d'un DSA vers un autre DSA ou DUA et, de ce fait, ne contribue pas à l'ensemble des résultats (représenté par le DSA "b" sur la figure C-1/X.518).

C.7.1 *Résultats des DSA - Aucun fusionnement*

Ce paragraphe concerne le rôle d'un DSA qui est l'unique source de résultats suite à une demande d'opération particulière, c'est-à-dire qui n'a aucune fonction de fusionnement à effectuer. On envisage le cas du DSP et celui du DAP.

C.7.1.1 *DSP*

Le DSA peut choisir d'exécuter l'une des procédures suivantes:

- retourner le résultat non signé, ce qui peut être représenté par:

  ChR,OR = Résultat de l'Opération Chaînée (non signé)

  où:

  ChR = Résultats du Chaînage

  OR = Résultat de l'Opération;

- signer uniquement le Résultat de l'Opération, ce qui peut être représenté par:

  ChR,d{OR} = Résultat de l'Opération signé par le DSA "d";

- signer uniquement le Résultat de l'Opération Chaînée, ce qui peut être représenté par:

d{ChR, OR} = Résultat de l'Opération Chaînée signé par le DSA "d";

- signer le Résultat de l'Opération et le Résultat de l'Opération Chaînée, ce qui peut être représenté par:

  d{ChR, d{OR}} = Résultat de l'Opération et Résultat de l'Opération Chaînée, signés par le DSA "d".

Remarque - Dans le cas où le Résultat de l'Opération est signé, il sera retourné par l'initiateur; dans le cas où le Résultat de l'Opération chaînée a été signé, le DSA destinataire devra détruire la signature pour pouvoir modifier l'argument Résultat du Chaînage avant de transmettre le Résultat de l'Opération Chaînée.

C.7.1.2 *DAP*

La procédure étant intégralement décrite dans la Recommandation X.511, ce qui suit est un résumé, fourni afin de disposer d'un texte complet.

Le DSA peut choisir de retourner les résultats non signés, ce qui peut être représenté par:

OR = Résultat de l'Opération

ou signés, ce qui peut être représenté par:

d{OR} = Résultat de l'Opération signé par le DSA "d".

C.7.2 *Résultats de DSA - Avec fusionnement*

Ce paragraphe décrit le rôle d'un DSA lors du retour du résultat d'une demande de service particulière, lorsque le fusionnement et l'intégration des résultats provenant d'autres résultats sont un préalable nécessaire. On considère le cas du DSP et celui du DAP.

C.7.2.1 *DSP*

Considérant que zéro ou plusieurs résultats reçus d'autres DSA peuvent être signés, cette procédure permet à un DSA de collationner et d'intégrer les résultats, de signer zéro ou plusieurs parties constitutives du résultat composite et, facultativement, de signer le résultat composite comme un tout.

C.7.2.1.1 *Production de l'argument des résultats du chaînage*

Cette procédure nécessite qu'un DSA (représenté par le DSA "c" de la figure C-1/X.518) supprime toutes les signatures de Résultat d'Opération Chaînée des résultats reçus des DSA extérieurs (les DSA "d" et "e" de la figure C-1/X.518). Le DSA "c" possède alors un ensemble de Résultats de Chaînage non signés, un ensemble de Résultats d'Opérations signés et un ensemble de Résultats d'Opérations non signés.

Tous les Résultats du Chaînage sont manipulés comme décrit au § 12.4 de la présente Recommandation, pour créer un Résultat de Chaînage unique modifié, noté:

i) ChR' = Résultats du chaînage modifiés.

C.7.2.1.2 *Résultats non signés produits localement*

Si le DSA ne désire pas signer les résultats générés localement, l'ensemble des Résultats d'Opération non signés est fusionné avec le résultat local, pour former un ensemble modifié de Résultats d'Opération, noté:

OR' = Résultat de l'Opération Fusionné

L'ensemble complet des Résultats d'Opération est alors la réunion de l'ensemble des Résultats d'Opération signés ultérieurement, notés:

d{OR}, e{OR} ...

et le Résultat d'Opération Fusionné, noté collectivement:

ii) OR', d{OR}, e{OR} ... = Résultat de l'Opération.

C.7.2.1.3 *Résultat signé produit localement*

Si le DSA désire signer les résultats générés localement, l'ensemble généré extérieurement des Résultats d'Opération non signés est d'abord fusionné. L'ensemble complet des Résultats d'Opération est alors la réunion de l'ensemble des Résultats d'Opération signés localement, noté C{OR},

l'ensemble fusionné des Résultats d'Opération non signés extérieurement, noté OR", et l'ensemble des Résultats d'Opération signés extérieurement, noté:

d{OR}, e{OR}, ..., qui sont collectivement notés:

iii) c{OR}, OR", d{OR}, e{OR}, ... = Résultat de l'Opération.

C.7.2.1.4 *Résultat non signé d'une opération chaînée*

Si le DSA ne désire pas signer le Résultat d'une Opération Chaînée, alors ce dernier comprendra les Résultats du Chaînage [spécifiés en i) ci-dessus] complétés par les Résultats d'Opération spécifiés ci-dessus en ii) ou iii), notés collectivement:

ChR', OR', d{OR}, e{OR}, ... = Résultat de l'Opération Chaînée (non signé)

ou,

ChR', c{OR}, OR", d{OR}, e{OR}, ... = Résultat de l'Opération Chaînée (non signé) et Résultat de l'Opération signé par le DSA "c".

C.7.2.1.5 *Résultat signé d'une opération chaînée*

Si le DSA désire signer le Résultat d'Opération Chaînée, alors ce résultat comprendra les Résultats du Chaînage [spécifiés en i) ci-dessus] complétés par le Résultat de l'Opération [spécifié ci-dessus en ii) ou iii)], notés collectivement:

c{ChR', OR', d{OR}, e{OR}, ...} = Résultat d'Opération Chaînée signé par le DSA "c"

ou,

c{ChR', c{OR}, OR", d{OR}, e{OR}, ...} = Résultat d'Opération Chaînée et Résultat d'Opération signés par le DSA "c".

C.7.2.2 *Dans le DAP*

La procédure est très similaire à celle décrite au § C.7.2.1 de la présente Recommandation, sauf que l'argument Résultats du chaînage n'est pas communiqué dans le DAP.

C.7.3 *Résultats d'opérations chaînées de DSA*

Ce paragraphe concerne les procédures à effectuer par un DSA pour chaîner un résultat d'opération en retour vers le DSA ou le DUA demandeur, respectivement dans le DSP et le DAP.

C.7.3.1 *DSP*

Le DSA commence par supprimer la signature (s'il en existe une) du Résultat d'Opération Chaînée. Puis il manipule l'argument Résultat du Chaînage, comme décrit dans cette Recommandation, pour produire un argument Résultats du Chaînage modifiés. Ce dernier est alors fusionné avec l'argument Résultat d'Opération, pour produire un Résultat d'Opération Chaînée modifié. En dernier lieu, le DSA peut, facultativement, signer le Résultat d'Opération Chaînée, avant de le communiquer au DSA suivant de la chaîne.

C.7.3.2 *DAP*

Un DSA (représenté par le DSA "b" de la figure C-1/X.518) supprime d'abord la signature (s'il en existe une) du Résultat d'Opération Chaînée. Puis il analyse et détruit l'argument Résultats du Chaînage et enfin signe, facultativement, l'argument Résultat d'Opération restant, avant de communiquer le résultat au DUA.

ANNEXE D

(à la Recommandation X.518)

Identificateurs d'objets répartis de l'annuaire

Cette annexe fait partie de la Recommandation.

Elle comprend tous les identificateurs d'objets ASN.1 contenus dans cette Recommandation sous la forme du module ASN.1 **DistributedDirectoryObjectIdentifiers.**

```
DistributedDirectoryObjectIdentifiers    {joint-iso-ccitt ds(5) modules(1)
                                          distributedDirectoryObjectIdentifiers(13)}

DEFINITIONS ::=
BEGIN

EXPORTS
        id-ot-dsa, id-pt-chainedRead, id-pt-chainedSearch, id-pt-chainedModify;

IMPORTS
        id-ot, id-pt
            FROM    UsefulDefinitions    {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0)};

-- objects --

id-ot-dsa    OBJECT IDENTIFIER    ::=    {id-ot 3}

-- part types --

id-pt-chainedRead     OBJECT IDENTIFIER    ::=    {id-pt 4}
id-pt-chainedSearch   OBJECT IDENTIFIER    ::=    {id-pt 5}
id-pt-chainedModify   OBJECT IDENTIFIER    ::=    {id-pt 6}

END
```

Recommandation X.519

L'ANNUAIRE - SPECIFICATIONS DU PROTOCOLE [1)]

(Melbourne, 1988)

SOMMAIRE

---

1) La Recommandation X.519 et la norme ISO 9594-5, L'annuaire - Spécifications du protocole, ont été élaborées en étroite collaboration et sont alignées du point de vue technique.

0 Introduction

0.1 Le présent document, ainsi que les autres de cette série, a été élaboré en vue de faciliter l'interconnexion de systèmes informatiques visant à assurer des services d'annuaire. L'ensemble de tous ces systèmes, avec les informations d'annuaire qu'ils détiennent, peut être considéré comme un tout intégré, appelé *annuaire*. Les informations de l'annuaire, désignées collectivement comme la Base de Données d'annuaire (DIB) sont normalement utilisées pour faciliter la communication entre, avec ou à propos d'objets tels que des entités d'application OSI, des personnes, des terminaux et des listes de diffusion.

0.2 L'annuaire joue un rôle significatif dans l'Interconnexion des Systèmes Ouverts (OSI) dont l'objectif est de permettre, au prix d'un minimum d'accords techniques en dehors des normes d'interconnexion proprement dites, d'interconnecter des équipements informatiques:

- de constructeurs différents;
- gérés de façons différentes;
- de niveaux de complexité différents;
- d'âges différents.

0.3 La présente Recommandation spécifie les éléments des services d'application et les contextes d'application pour deux protocoles - le Protocole d'Accès à l'annuaire (DAP) et le Protocole du Système d'annuaire (DSP). Le DAP assure l'accès à l'annuaire pour rechercher ou modifier l'information qu'il contient. Le DSP assure le chaînage des demandes de recherche ou de modification d'information d'annuaire avec d'autres parties du Système d'annuaire réparti où peut se trouver l'information.

1 Objet et domaine d'application

La présente Recommandation spécifie le Protocole d'Accès à l'annuaire et le Protocole du Système d'annuaire, répondant aux services abstraits spécifiés dans les Recommandations X.511 et X.518.

2 Références

Recommandation X.200 - Modèle de référence pour l'interconnexion des systèmes ouverts pour les applications du CCITT

Recommandation X.208 - Interconnexion des Systèmes Ouverts - Spécification de la notation 1 de la syntaxe abstraite (ASN.1)

Recommandation X.209 - Interconnexion des Systèmes Ouverts - Spécification des règles de codage de base pour l'ASN.1

Recommandation X.500 - L'Annuaire - Aperçu des concepts, modèles et services

Recommandation X.501 - L'Annuaire - Modèles

Recommandation X.511 - L'Annuaire - Définition du service abstrait

Recommandation X.518 - L'Annuaire - Procédures de fonctionnement réparti

Recommandation X.520 - L'Annuaire - Types d'attributs sélectionnés

Recommandation X.521 - L'Annuaire - Catégories d'objets sélectionnées

Recommandation X.219 - Exploitation à distance - Modèle, notation et définition des services

Recommandation X.229 - Exploitation à distance - Spécifications du protocole

Recommandation X.217 - Interconnexion des Systèmes Ouverts - Commande d'association: définition du service

Recommandation X.227 - Interconnexion des Systèmes Ouverts - Commande d'association: spécification du protocole

Recommandation X.216 - Interconnexion des Systèmes Ouverts - Définition du service de la couche Présentation.

3 Définitions

Dans le présent paragraphe, on utilise les abréviations définies au § 4.

3.1 *Définitions relatives au modèle de référence OSI*

La présente Recommandation est fondée sur les concepts développés dans la Recommandation X.200. Elle utilise les expressions suivantes, qui y sont définies:

a) *élément de service d'application*;

b) *information de commande de protocole d'application*;

c) *unité de données de protocole d'application*;

d) *contexte d'application*;

e) *entité d'application*;

f) *syntaxe abstraite.*

3.2 *Définitions de base relatives à l'annuaire*

La présente Recommandation utilise les termes et expressions suivants, définis dans la Recommandation X.501:

a) *l'annuaire*;

b) *utilisateur (de l'annuaire)*;

c) *Agent du Système de l'annuaire (DSA)*;

d) *Agent d'Utilisateur de l'annuaire (DUA).*

3.3 *Définitions relatives aux opérations réparties*

La présente Recommandation utilise les termes suivants, définis dans la Recommandation X.518:

a) *chaînage*;

b) *renvoi de référence.*

4 Abréviations

Les abréviations suivantes sont utilisées dans la présente Recommandation:

AC	Contexte d'Application
ACSE	Elément de Service de Commande d'Association
AE	Entité d'Application
APCI	Information de Commande de Protocole d'Application
APDU	Unité de Données de Protocole d'Application
ASE	Elément de Service d'Application
DAP	Protocole d'Accès à l'annuaire
DSA	Agent du Système de l'annuaire
DSP	Protocole du Système d'annuaire
DUA	Agent d'Utilisateur de l'annuaire
ROSE	Elément de Service d'Opérations Distantes

5 Conventions

La Recommandation utilise les conventions suivantes:

a) les définitions de syntaxe abstraite du § 7 sont établies au moyen de la notation de syntaxe abstraite définie dans la Recommandation X.208,

b) les macros d'opération distante (notation-RO) et les macros élément de service d'application et contexte d'application sont définies dans la Recommandation X.219,

c) les mots des expressions définies et les noms et valeurs des paramètres de service et des champs de protocole commencent, sauf pour les noms propres, par une lettre minuscule et sont reliés par un tiret, par exemple terme-défini. Les noms propres commencent par une majuscule et ne sont pas reliés par un tiret, par exemple, Nom Propre.

6 Aperçu du protocole

6.1 *Modèle de Protocole d'annuaire*

La Recommandation X.511 définit le service abstrait entre un DUA et l'annuaire qui permet à un utilisateur d'avoir accès aux services d'annuaire. L'annuaire est, de plus, modélisé pour être représenté par un DSA qui fonctionne au point d'accès concerné. La Recommandation X.518 définit les interactions entre une paire de DSA situés dans l'annuaire pour satisfaire à des demandes chaînées émanant de l'utilisateur. La figure 1/X.519 illustre ces concepts.

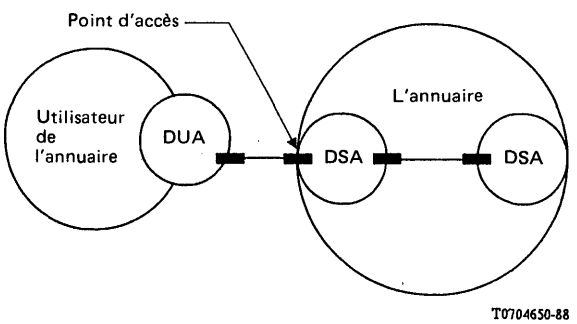


FIGURE 1/X.519

Interactions d'annuaire

Quand un DUA se trouve dans un autre système ouvert qu'un DSA avec lequel il interagit, les interactions suivent le Protocole d'Accès à l'annuaire (DAP), qui est un protocole de couche d'application OSI. De même, quand deux DSA interactifs se trouvent dans des systèmes ouverts différents, les interactions suivent le Protocole de Système d'annuaire (DSP) qui est également situé dans la couche d'application.

Le DAP et le DSP sont des protocoles chargés d'assurer une communication entre une paire de processus d'application. Dans le contexte OSI, cela est représenté comme une communication entre une paire d'entités-d'application (AE) utilisant le service de présentation. La fonction d'une AE est fournie par un ensemble d'éléments-de-service-d'application (ASE). L'interaction entre AE est décrite en fonction de leur utilisation des services assurés par les ASE. Les deux ASE communs aux deux protocoles d'annuaire sont résumés dans le présent paragraphe.

L'Elément de Service d'Opérations Distantes (ROSE) satisfait au paradigme de demande/réponse de l'opération abstraite qui se produit aux ports du modèle abstrait. Les ASE d'annuaire assurent la fonction de mise en correspondance de la notation de syntaxe abstraite du service abstrait d'annuaire avec les services fournis par le ROSE.

L'Elément de Service de Commande d'Association (ACSE) assure l'établissement et la libération d'une association-application entre une paire d'AE. Les associations entre un DUA et un DSA peuvent être établies uniquement par le DUA. Seul l'initiateur d'une association établie peut la libérer.

6.2 *Protocole d'Accès à l'annuaire*

Le Protocole d'Accès à l'annuaire (DAP) sert à réaliser le Service Abstrait d'annuaire. Il comprend trois ASE spécifiques d'annuaire, outre le ROSE et l'ACSE, à savoir **readASE**, **searchASE** et **modifyASE**. Ils correspondent aux **readPort**, **searchPort** et **modifyPort** du service abstrait. Le contexte d'application **directoryAccessAC** identifie la combinaison de: **readASE**, **searchASE** et **modifyASE**, **aCSE**, **rOSE**.

6.3 *Protocole de Système d'annuaire*

Le Protocole de Système d'annuaire (DSP) sert à réaliser le caractère fonctionnel de l'opération répartie décrite dans la Recommandation X.518. Il comprend trois ASE spécifiques d'annuaire, outre ROSE et ACSE, à savoir **chainedReadASE**, **chainedSearchASE** et **chainedModifyASE**. Ils correspondent aux **chainedReadPort**, **chainedSearchPort** et **chainedModifyPort** du service abstrait. Le contexte d'application du **systèmed'annuaireAC** identifie la combinaison de **chainedReadASE**, **chainedSearchASE** et **chainedModifyASE**, **aCSE**, **rOSE**.

6.4 *Utilisation des Services Sous-Jacents*

Les protocoles DAP et DSP utilisent les services sous-jacents comme indiqué ci-dessous.

6.4.1 *Utilisation des services ROSE*

L'Elément de Service d'Opérations Distantes (ROSE) est défini dans la Recommandation X.219.

Le ROSE contribue au paradigme demande/réponse des opérations distantes.

Les ASE d'annuaire sont les utilisateurs des services RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U et RO-REJECT-P de ROSE.

Les opérations distantes du DAP et du DSP sont des opérations (asynchrones) de Catégorie 2. A noter que le DUA étant un consommateur du DAP, il peut décider de fonctionner d'une manière synchrone.

Le DAP utilise la Catégorie 1 d'Association. Cela signifie que le DSA ne peut pas lancer d'opérations sur le DUA. Le DSP utilise la Catégorie 3 d'Association, ce qui signifie que le DSA qui répond peut lancer des opérations sur le DSA initiateur et vice versa.

6.4.2 *Utilisation des services ACSE*

L'Elément de Service de Commande d'Association (ACSE) est défini dans la Recommandation X.217.

L'ACSE assure la commande (établissement, libération, abandon) des associations-applications entre AE.

Directory Bind et Directory Unbind (ou DSA Bind et DSA Unbind) sont les seuls utilisateurs des services A-ASSOCIATE et A-RELEASE de l'ACSE en mode normal. Le processus-application est l'utilisateur des services A-ABORT et A-P-ABORT de l'ACSE.

6.4.3 *Utilisation du Service Présentation*

Le service-présentation est défini dans la Recommandation X.216.

La Couche Présentation coordonne la représentation (syntaxe) de la sémantique de Couche Application qui doit être échangée.

En mode normal, un contexte-présentation différent est utilisé pour chaque syntaxe-abstraite incluse dans le contexte-application.

L'ACSE est le seul utilisateur des services P-CONNECT, P-RELEASE, P-U-ABORT ET P-P-ABORT du service-présentation.

Le ROSE est un utilisateur du service P-DATA du service-présentation.

6.4.4 *Utilisation des Services de Couche Inférieure*

Le service-session est défini dans la Recommandation X.215. La Couche Session structure le dialogue du flux d'informations entre les systèmes-terminaux.

Les unités fonctionnelles Kernel et Duplex du service-session sont utilisées par la Couche Présentation.

Le service-transport est défini dans la Recommandation X.214. La Couche Transport assure le transfert transparent des données de bout en bout sur la connexion de réseau sous-jacente.

Le choix de la catégorie de service-transport utilisé par la Couche Session dépend des besoins de multiplexage et de retour au fonctionnement normal. L'acceptation de la Catégorie 0 de Transport (sans multiplexage) est obligatoire. Le Service Transport Exprès n'est pas utilisé.

L'acceptation d'autres catégories est facultative. Une catégorie de multiplexage peut être utilisée pour multiplexer le DAP ou le DSP et d'autres protocoles sur la même connexion de réseau. Une catégorie de retour au fonctionnement normal peut être choisie sur une connexion de réseau avec un taux d'erreurs résiduelles inacceptable.

Un réseau sous-jacent assurant le service-réseau OSI défini dans la Recommandation X.213 est supposé.

Une adresse-réseau est telle que définie dans les Recommandations X.121, E.163/E.164 ou X.200 (adresse NSAP OSI).

7 Syntaxe abstraite du Protocole d'annuaire

7.1 *Syntaxes abstraites*

Les ASE d'annuaire spécifiés aux § 7.2.1, 7.2.3 et 7.2.5 utilisent en commun une seule syntaxe abstraite, **id-as-directory-AccessAS**. Ceux qui sont spécifiés aux § 7.2.2, 7.2.4 et 7.2.6 utilisent aussi en commun une seule syntaxe abstraite, **id-as-directorySystemsAS**. Dans chaque cas, cela définit l'information-commande-protocole-application (APCI) qui, quand elle est utilisée conjointement avec le ROSE, définit un ensemble d'APDU. Les APDU d'annuaire sont définies par la syntaxe abstraite des ASE et ROSE d'annuaire. Ceux-ci, plus la syntaxe abstraite d'ACSE, forment la définition complète des APDU utilisées pendant une association d'annuaire.

La syntaxe abstraite ACSE **id-as-acse** est nécessaire pour établir les associations.

Ces syntaxes abstraites doivent (au moins) être codées conformément aux Règles Fondamentales de Codage ASN.1

7.2 *Eléments de service d'application d'annuaire*

Le présent paragraphe spécifie les ASE qui sont utilisés comme modules dans la construction des divers contextes d'application d'annuaire du § 7.3.

Remarque - Ces ASE servent à la construction des contextes d'application définis dans la présente Recommandation. Ils ne sont pas destinés à permettre des affirmations de conformité avec des ASE individuels ou d'autres combinaisons d'ASE.

7.2.1 *Read ASE*

ReadASE contribue aux opérations abstraites de **readPort**, à savoir **Read**, **Compare** et **Abandon**, définies dans la Recommandation X.511.

```
readASE
    APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES
            {read, compare, abandon}
    ::=  id-ase-readASE

read        Read       ::= 1

compare     Compare    ::= 2

abandon     Abandon    ::= 3
```

7.2.2 *Chained Read ASE*

ChainedReadASE contribue aux opérations abstraites de **ChainedReadPort**, à savoir **ChainedRead**, **ChainedCompare** et **ChainedAbandon**, définies dans la Recommandation X.518.

```
chainedReadASE
    APPLICATION-SERVICE-ELEMENT
        OPERATIONS {
                chainedRead,
                chainedCompare,
                chainedAbandon}
    ::= id-ase-chainedReadASE

chainedRead         ChainedRead      ::= 1

chainedCompare      ChainedCompare   ::= 2

chainedAbandon      ChainedAbandon   ::= 3
```

7.2.3 *Search ASE*

SearchASE contribue aux opérations abstraites de **SearchPort**, à savoir **List** et **Search** définies dans la Recommandation X.511.

```
searchASE
    APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {list, search}
    ::=  id-ase-searchASE

list        List     ::= 4

search      Search   ::= 5
```

7.2.4 *Chained Search ASE*

ChainedSearchASE contribue aux opérations abstraites de **ChainedSearchPort**, à savoir **ChainedList** et **ChainedSearch**, définies dans la Recommandation X.518.

```
chainedSearchASE
    APPLICATION-SERVICE-ELEMENT
        OPERATIONS {
                chainedList, chainedSearch}
    ::=  id-ase-chainedSearchASE

chainedList         ChainedList      ::= 4

chainedSearch       ChainedSearch    ::= 5
```

7.2.5 *Modify ASE*

ModifyASE contribue aux opérations abstraites de **ModifyPort**, à savoir **AddEntry**, **RemoveEntry**, **ModifyEntry** et **ModifyRDN** définies dans la Recommandation X.511.

```
modifyASE
    APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES
            {addEntry, removeEntry,
            modifyEntry, modifyRDN}
    ::=  id-ase-modifyASE

addEntry        AddEntry        ::= 6
removeEntry     RemoveEntry     ::= 7
modifyEntry     ModifyEntry     ::= 8
modifyRDN       ModifyRDN       ::= 9
```

7.2.6 *Chained Modify ASE*

ChainedModifyASE contribue aux opérations abstraites de **ChainedModifyPort**, à savoir **ChainedAddEntry**, **ChainedRemoveEntry**, **ChainedModifyEntry** et **ChainedModifyRDN** définies dans la Recommandation X.518.

```
chainedModifyASE
    APPLICATION-SERVICE-ELEMENT
        OPERATIONS
            {chainedAddEntry,
            chainedRemoveEntry,
            chainedModifyEntry,
            chainedModifyRDN}
    ::=  id-ase-chainedModifyASE

chainedAddEntry       ChainedAddEntry       ::= 6
chainedRemoveEntry    ChainedRemoveEntry    ::= 7
chainedModifyEntry    ChainedModifyEntry    ::= 8
chainedModifyRDN      ChainedModifyRDN      ::= 9
```

7.3 *Contextes d'application d'annuaire*

7.3.1 *Contexte d'application d'accès d'annuaire*

Le **directoryAccessAC** permet au DUA d'accéder aux opérations des ASE suivants: **readASE**, **searchASE** et **modifyASE**.

```
directoryAccessAC
    APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS
            {aCSE}
        BIND DirectoryBind
        UNBIND DirectoryUnbind
        REMOTE OPERATIONS {rOSE}
        INITIATOR CONSUMER OF {
            readASE,
            searchASE,
            modifyASE}
        ABSTRACT SYNTAXES {
            id-as-acse,
            id-as-directoryAccessAS}
    ::=  id-ac-directoryAccessAC
```

7.3.2 *Contexte d'application de système d'annuaire*

Le **directorySystemAC** permet aux DSA de communiquer pour le chaînage des opérations.

```
directorySystemAC
    APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS
            {aCSE}
        BIND DSABind
        UNBIND DSAUnbind
```

```
        REMOTE OPERATIONS {rOSE}
        OPERATIONS OF
            {chainedReadASE,
            chainedSearchASE,
            chainedModifyASE}
        ABSTRACT SYNTAXES {
            id-as-acse,
            id-as-directorySystemAS}
    ::= id-ac-directorySystemAC
```

7.4 *Erreurs*

A chaque erreur abstraite définie dans le Service Abstrait correspond une valeur d'erreur qui peut être acheminée par le protocole. Les assignations sont les suivantes:

abandoned	**Abandoned**	::= 5
attributeError	**AttributeError**	::= 1
nameError	**NameError**	::= 2
referral	**Referral**	::= 4
securityError	**SecurityError**	::= 6
serviceError	**ServiceError**	::= 3
updateError	**UpdateError**	::= 8
dSAReferral	**DSAReferral**	::= 9
abandonFailed	**AbandonFailed**	::= 7

8 Mise en correspondance avec les services utilisés

Ce paragraphe définit la mise en correspondance des DAP et DSP avec les services utilisés.

8.1 *Mise en correspondance avec ACSE*

Ce paragraphe définit la mise en correspondance des services abstract-bind (**DirectoryBind** ou **DSABind**) et abstract-unbind (**DirectoryUnbind** ou **DSAUnbind**) avec les services de l'ACSE. Celui-ci est défini dans la Recommandation X.217.

8.1.1 *Abstract-bind avec A-ASSOCIATE*

Le service abstract-bind est mis en correspondance avec le service A-ASSOCIATE de l'ACSE. L'utilisation des paramètres du service A-ASSOCIATE est précisée aux sous-paragraphes suivants:

8.1.1.1 *Mode*

Ce paramètre sera fourni par l'initiateur de l'association avec la primitive de demande A-ASSOCIATE et aura la valeur "mode normal".

8.1.1.2 *Nom de Contexte d'Application*

L'initiateur de l'association propose l'un des deux contextes suivants: **directoryAccessAC** ou **directorySystemAC**.

8.1.1.3 *Information d'Utilisateur*

La mise en correspondance de l'opération-liaison du service abstract-bind avec les paramètres d'Information d'Utilisateur de la primitive de demande A-ASSOCIATE est définie dans la Recommandation X.219.

8.1.1.4 *Liste de Définition de Contexte de Présentation*

L'initiateur de l'association fournira la Liste de Définition de Contexte de Présentation dans la primitive de demande A-ASSOCIATE qui contiendra la syntaxe abstraite ACSE (**id-as-acse**) et la syntaxe abstraite DAP (**id-as-directoryAccessAS**) ou DSP (**id-as-directorySystemAS**).

8.1.1.5 *Qualité de service*

Ce paramètre sera fourni par l'initiateur de l'association dans la primitive de demande A-ASSOCIATE et par le répondeur de l'association dans la primitive de réponse A-ASSOCIATE. Les paramètres "Extended Control" et "Optimized Dialogue Transfer" seront mis à "fonction non désirée". Les paramètres restants seront tels que des valeurs par défaut seront utilisées.

8.1.1.6 *Besoins de session*

Ce paramètre sera mis par l'initiateur de l'association dans la primitive de demande A-ASSOCIATE et par le répondeur de l'association dans la primitive de réponse A-ASSOCIATE. Ce paramètre sera mis pour spécifier les unités fonctionnelles suivantes:

a) Kernel,

b) Duplex.

8.1.1.7 *Titre d'Entité d'Application et Adresse de Présentation*

Ces paramètres seront fournis par l'initiateur et par le répondeur de l'association (un Titre d'Entité d'Application est fourni à titre facultatif).

Lorsqu'un DUA établit une association pour une demande initiale, ces paramètres sont obtenus à partir des informations détenues localement.

Lorsqu'un DUA (ou un DSA) établit une association avec un DSA auquel il a été renvoyé, ces paramètres s'obtiennent à partir de la valeur de **PointAccès** d'une **RéférenceContinuation**. Lorsqu'un DSA établit une association, ce paramètre s'obtient à partir de ses Informations de Connaissance, c'est-à-dire une référence externe.

8.1.2 *Abstract-Unbind avec A-RELEASE*

Le service abstract-unbind est mis en correspondance avec le service A-RELEASE de l'ACSE. L'utilisation des paramètres du service A-RELEASE est précisée dans le sous-paragraphe suivant.

8.1.2.1 *Résultat*

Ce paramètre aura la valeur "affirmatif".

8.1.3 *Utilisation des services A-ABORT et A-P-ABORT*

Le processus-application est l'utilisateur des services A-ABORT et A-P-ABORT de l'ACSE.

8.2 *Mise en correspondance avec ROSE*

Les services ASE d'annuaire sont mis en correspondance avec les services RO-INVOKE, RO-RESULT, RO-ERROR, RO-REJECT-U et RO-REJECT-P de ROSE. La mise en correspondance de la notation de syntaxe abstraite des ASE d'annuaire avec les services ROSE est définie dans la Recommandation X.219.

9 Conformité

Ce paragraphe définit les conditions de conformité à la présente Recommandation.

9.1 *Conformité par DUA*

Pour être conforme à la présente Recommandation, un DUA mis en service doit répondre aux conditions spécifiées aux § 9.1.1 à 9.1.3.

9.1.1 *Besoins de déclaration*

Il sera déclaré ce qui suit:

a) les opérations du contexte-application **directoryAccessAC** que le DUA est capable de lancer et pour lesquelles la conformité est annoncée; et

b) le ou les niveaux de sécurité pour lesquels la conformité est annoncée (aucune, simple, poussée).

9.1.2 *Besoins statiques*

Un DUA doit:

a) être à même de contribuer au contexte-application **directoryAccessAC** qui est défini par sa syntaxe abstraite au § 7.

9.1.3 *Besoins dynamiques*

Un DUA doit:

a) se conformer à la mise en correspondance avec les services utilisés définie au § 8.

9.2 *Conformité par DSA*

Une mise en oeuvre de DSA annonçant une conformité avec la présente Recommandation doit satisfaire aux conditions spécifiées aux § 9.2.1 à 9.2.3.

9.2.1 *Besoins de déclaration*

Il sera déclaré ce qui suit:

a) les contextes-application pour lesquels la conformité est annoncée: **directoryAccessAC**, **directorySystemAC** ou les deux. Si un DSA est tel que sa connaissance a été diffusée, causant des références de connaissance pour le DSA devant être détenu par un ou d'autres DSA en dehors de son DMD, il annoncera sa conformité avec le **directorySystemAC**;

Remarque - Un contexte d'application ne doit pas être divisé, sauf comme indiqué ici: en particulier, la conformité ne peut pas être annoncée avec des ports ou des opérations particuliers.

b) si le DSA est capable ou non d'agir en tant que DSA de premier niveau, comme défini dans la Recommandation X.518;

c) si la conformité est annoncée avec le contexte-application **directorySystemAC**, qu'il s'agisse ou non d'un mode d'opération chaîné, tel que défini dans la Recommandation X.518;

d) le ou les niveaux de sécurité pour lesquels la conformité est annoncée (aucune, simple, poussée);

e) les types d'attributs sélectionnés et définis dans la Recommandation X.520 et tous autres types d'attributs pour lesquels la conformité est annoncée; et

f) les catégories d'objets sélectionnées, définies dans la Recommandation X.521 et toutes autres catégories d'objets pour lesquelles la conformité est annoncée.

9.2.2 *Besoins statiques*

Un DSA doit:

a) être à même de contribuer aux contextes-application pour lesquels la conformité est annoncée, tels que définis par leur syntaxe abstraite au § 7;

b) être à même de contribuer au cadre d'information défini par sa syntaxe abstraite dans la Recommandation X.501;

c) se conformer aux besoins minimaux de connaissance définis dans la Recommandation X.518;

d) si la conformité est annoncée comme un DSA de premier niveau, se conformer aux besoins d'appui du contexte de racine, défini dans la Recommandation X.518;

e) être à même de contribuer aux types d'attributs pour lesquels la conformité est annoncée, tels que définis par leur syntaxe abstraite; et

f) être à même de contribuer aux catégories d'objets pour lesquelles la conformité est annoncée, telles que définies par leur syntaxe abstraite.

9.2.3 *Besoins dynamiques*

Un DSA doit:

a) se conformer à la mise en correspondance avec les services utilisés, définie au § 8 de la présente Recommandation;

b) se conformer aux procédures applicables à l'opération répartie de l'annuaire vis-à-vis des renvois de référence, comme défini dans la Recommandation X.518;

c) si la conformité est annoncée avec le contexte-application **directoryAccessAC**, se conformer aux procédures de la Recommandation X.518 en ce qui concerne le mode de renvoi de référence du DAP;

d) si la conformité est annoncée avec le contexte-application **directorySystemAC**, se conformer au mode de renvoi de référence d'interaction, défini dans la Recommandation X.518;

e) si la conformité est annoncée avec le mode chaîné d'interaction, se conformer au mode chaîné d'interaction, défini dans la Recommandation X.518.

Remarque - C'est le seul cas où il faut qu'un DSA soit capable de lancer des opérations au moyen de **directorySystemAC**.

ANNEXE A

(à la Recommandation X.519)

DAP en ASN.1

Cette annexe fait partie de la Recommandation.

Elle comprend toutes les définitions concernant le type et la valeur ASN.1 contenues dans la présente Recommandation sous la forme du module ASN.1 **DirectoryAccessProtocol.**

```
DirectoryAccessProtocol {joint-iso-ccitt ds(5) modules(1) dap(11)}
DEFINITIONS ::=
BEGIN

EXPORTS
    directoryAccessAC, readASE, searchASE, modifyASE;

IMPORTS
    abstractService
    FROM    UsefulDefinitions
            {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0)}

    APPLICATION-SERVICE-ELEMENT, APPLICATION-CONTEXT, aCSE
    FROM    Remote-Operations-Notation-extension
            {joint-iso-ccitt remoteOperations(4) notation-extension(2)}

    id-ac-directoryAccessAC, id-ase-readASE, id-ase-searchASE,
    id-ase-modifyASE, id-as-directoryAccessAS, id-as-acse
    FROM    ProtocolObjectIdentifiers
            {joint-iso-ccitt  ds(5) modules(1)
            protocolObjectIdentifiers(4)}

    DirectoryBind, DirectoryUnbind, Read, Compare, Abandon, List,
    Search, AddEntry, RemoveEntry, ModifyEntry, ModifyRDN, Abandoned, AbandonFailed,
    AttributeError, NameError, Referral, SecurityError, ServiceError,
    UpdateError
    FROM    DirectoryAbstractService
            directoryAbstractService;

-- Application Contexts --

directoryAccessAC
    APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS {aCSE}
        BIND DirectoryBind
```

```
        UNBIND DirectoryUnbind
        REMOTE OPERATIONS {rOSE}
            INITIATOR CONSUMER OF {readASE, searchASE, modifyASE}
        ABSTRACT SYNTAXES {
            id-as-acse, id-as-directoryAccessAS}
    ::= id-ac-directoryAccessAC

-- Read ASE --

readASE
    APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {read, compare, abandon}
    ::= id-ase-readASE

-- Search ASE --

searchASE
    APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES {list, search}
    ::= id-ase-searchASE

-- Modify ASE --

modifyASE
    APPLICATION-SERVICE-ELEMENT
        CONSUMER INVOKES
            {addEntry, removeEntry,
            modifyEntry, modifyRDN}
    ::= id-ase-modifyASE

-- Remote Operations --

read            Read            ::= 1
compare         Compare         ::= 2
abandon         Abandon         ::= 3
list            List            ::= 4
search          Search          ::= 5
addEntry        AddEntry        ::= 6
removeEntry     RemoveEntry     ::= 7
modifyEntry     ModifyEntry     ::= 8
modifyRDN       ModifyRDN       ::= 9

-- Remote Errors --

attributeError  AttributeError  ::= 1
nameError       NameError       ::= 2
serviceError    ServiceError    ::= 3
referral        Referral        ::= 4
abandoned       Abandoned       ::= 5
securityError   SecurityError   ::= 6
abandonFailed   AbandonFailed   ::= 7
updateError     UpdateError     ::= 8
END
```

ANNEXE B

(à la Recommandation X.519)

DSP en ASN.1

Cette annexe fait partie de la Recommandation.

Elle comprend toutes les définitions concernant le type et la valeur ASN.1 contenues dans la présente Recommandation sous la forme du module ASN.1 **DirectorySystemProtocol**.

```
DirectorySystemProtocol {joint-iso-ccitt ds(5) modules(1) dsp(12)}
DEFINITIONS ::=
BEGIN

EXPORTS
    directorySystemAC, chainedReadASE, chainedSearchASE,
    chainedModifyASE;

IMPORTS
    distributedOperations, directoryAbstractService
    FROM    UsefulDefinitions
            {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0)}

    APPLICATION-SERVICE-ELEMENT, APPLICATION-CONTEXT, aCSE
    FROM    Remote-Operations-Notation-extension
            {joint-iso-ccitt remoteOperations(4) notation-extension(2)}

    id-ac-directorySystemAC, id-ase-chainedReadASE,
    id-ase-chainedSearchASE, id-ase-chainedModifyASE,
    id-as-directorySystemAS, id-as-acse;
    FROM    ProtocolObjectIdentifiers
            {joint-iso-ccitt  ds(5) modules(1)
            protocolObjectIdentifiers(4)}

    Abandoned, AttributeError, AbandonFailed,
    NameError, DSAReferral, SecurityError, ServiceError, UpdateError
    FROM    DirectoryAbstractService directoryAbstractService

    DSABind, DSAUnbind,
    ChainedRead, ChainedCompare, ChainedAbandon,
    ChainedList, ChainedSearch,
    ChainedAddEntry, ChainedRemoveEntry, ChainedModifyEntry,
    ChainedModifyRDN, DSAReferral
    FROM    DistributedOperations
            distributedOperations;

-- Application Contexts --

directorySystemAC
    APPLICATION-CONTEXT
        APPLICATION SERVICE ELEMENTS {aCSE}
        BIND DSABind
        UNBIND DSAUnbind
        REMOTE OPERATIONS {rOSE}
            OPERATIONS OF {
                chainedReadASE, chainedSearchASE, chainedModifyASE}
        ABSTRACT SYNTAXES {
            id-as-acse, id-as-directorySystemAS}
    ::= {id-ac-directorySystemAC}

-- Chained Read ASE --

chainedReadASE
    APPLICATION-SERVICE-ELEMENT
        OPERATIONS          {chainedRead, chainedCompare, chainedAbandon}
    ::= id-ase-chainedReadASE
```

```
-- Chained Search ASE --

chainedSearchASE
    APPLICATION-SERVICE-ELEMENT
        OPERATIONS          {chainedList, chainedSearch}
    ::= id-ase-chainedSearchASE

-- Chained Modify ASE --

chainedModifyASE
    APPLICATION-SERVICE-ELEMENT
        OPERATIONS
            {chainedAddEntry, chainedRemoveEntry,
            chainedModifyEntry, chainedModifyRDN}
    ::= id-ase-chainedModifyASE

-- Remote Operations --

chainedRead             ChainedRead             ::= 1
chainedCompare          ChainedCompare          ::= 2
chainedAbandon          ChainedAbandon          ::= 3
chainedlist             ChainedList             ::= 4
chainedSearch           ChainedSearch           ::= 5
chainedAddEntry         ChainedAddEntry         ::= 6
chainedRemoveEntry      ChainedRemoveEntry      ::= 7
chainedModifyEntry      ChainedModifyEntry      ::= 8
chainedModifyRDN        ChainedModifyRDN        ::= 9

-- Remote Errors --

attributeError      AttributeError      ::= 1
nameError           NameError           ::= 2
serviceError        ServiceError        ::= 3
abandoned           Abandoned           ::= 5
securityError       SecurityError       ::= 6
abandonFailed       AbandonFailed       ::= 7
updateError         UpdateError         ::= 8
dsaReferral         DSAReferral         ::= 9

END
```

ANNEXE C

(à la Recommandation X.519)

Définition de référence des identificateurs d'objets de protocole

Cette annexe fait partie de la Recommandation.

Elle comprend tous les Identificateurs d'objets ASN.1 assignés dans la présente Recommandation sous la forme du module **ProtocolObjectIdentifiers.**

```
ProtocolObjectIdentifiers {joint-iso-ccitt ds(5) modules(1) protocolObjectIdentifiers(4)}
DEFINITIONS ::=
BEGIN
```

```
EXPORTS
    id-ac-directoryAccessAC, id-ac-directorySystemAC, id-ase-readASE, id-ase-searchASE,
    id-ase-modifyASE, id-ase-chainedReadASE,
    id-ase-chainedSearchASE, id-ase-chainedModifyASE, id-as-acse,
    id-as-directoryAccessAS, id-as-directorySystemsAS;

IMPORTS
    id-ac, id-ase, id-as
    FROM    UsefulDefinitions
            {joint-iso-ccitt ds(5) modules(1) usefulDefinitions(0)};

-- Contextos de aplicación --

id-ac-directoryAccessAC   OBJECT IDENTIFIER ::= {id-ac 1}
id-ac-directorySystemAC   OBJECT IDENTIFIER ::= {id-ac 2}

-- ESA (ASEs) --

id-ase-readASE            OBJECT IDENTIFIER ::= {id-ase 1}
id-ase-searchASE          OBJECT IDENTIFIER ::= {id-ase 2}
id-ase-modifyASE          OBJECT IDENTIFIER ::= {id-ase 3}
id-ase-chainedReadASE     OBJECT IDENTIFIER ::= {id-ase 4}
id-ase-chainedSearchASE   OBJECT IDENTIFIER ::= {id-ase 5}
id-ase-chainedModifyASE OBJECT IDENTIFIER ::= {id-ase 6}

-- SA (ASs) --

id-as-directoryAccessAS   OBJECT IDENTIFIER    ::= {id-as 1}
id-as-directorySystemAS   OBJECT IDENTIFIER    ::= {id-as 2}
id-as-acse                OBJECT IDENTIFIER    ::=
                          {joint-iso-ccitt association-control(2) abstract-syntax(1) apdus(0)
                          version1(1)}

END
```

Recommandation X.520

L'ANNUAIRE: TYPES D'ATTRIBUTS SELECTIONNES [1)]

(Melbourne, 1988)

TABLE DES MATIERES

---

1) La Recommandation X.520 et la norme ISO 9594-6 - Systèmes de traitement de l'information - Interconnexion des systèmes ouverts - L'annuaire - Types d'attributs sélectionnés - ont été élaborées en étroite collaboration et sont alignées du point de vue technique.

0 Introduction

0.1 Le présent document, ainsi que les autres de cette série, a été élaboré en vue de faciliter l'interconnexion de systèmes informatiques visant à assurer des services d'annuaire. L'ensemble de tous ces systèmes, avec les informations d'annuaire qu'ils détiennent, peut être considéré comme un tout intégré, appelé *annuaire*. Les informations de l'annuaire, désignées collectivement comme la Base de Données d'annuaire (DIB) sont normalement utilisées pour faciliter la communication entre, avec ou à propos d'objets tels que des entités d'application OSI, des personnes, des terminaux et des listes de diffusion.

0.2 L'annuaire joue un rôle significatif dans l'Interconnexion des Systèmes Ouverts (OSI) dont l'objectif est de permettre, au prix d'un minimum d'accords techniques en dehors des normes d'interconnexion proprement dites, d'interconnecter des équipements informatiques:

- de constructeurs différents;
- gérés de façons différentes;
- de niveaux de complexité différents;
- d'âges différents.

0.3 La présente Recommandation définit un certain nombre de types d'attributs qui peuvent être utiles dans toute une gamme d'applications de l'annuaire. Un emploi particulier à de nombreux attributs ainsi définis est la formation de noms, notamment pour les catégories d'objets définies dans la Recommandation X.521. La présente Recommandation définit aussi un certain nombre de syntaxes d'attributs normalisées.

0.4 L'annexe A, qui fait partie de la présente Recommandation, indique la notation ASN.1 du module complet qui définit les attributs et les syntaxes d'attributs.

0.5 Pour plus de commodité, l'annexe B, qui ne fait pas partie de la présente Recommandation, contient un index alphabétique des types d'attributs.

1 Objet et domaine d'application

1.1 La présente Recommandation définit un certain nombre de types d'attributs qui peuvent se révéler utiles dans bien des applications de l'annuaire.

1.2 Les types (et syntaxes) d'attributs se répartissent en trois catégories, décrites aux § 1.2.1 à 1.2.3.

1.2.1 Certains types (syntaxes) d'attributs sont utilisés dans une grande variété d'applications ou sont compris et/ou utilisés par l'annuaire même.

Remarque - Quand un type (syntaxe) d'attributs défini dans le présent document convient à l'application envisagée, il est recommandé de l'utiliser plutôt que d'en créer un nouveau.

1.2.2 Certains types (syntaxes) d'attributs normalisés à l'échelon international sont spécifiques de telle ou telle application. Ils sont, dans ce cas, définis dans les normes associées à l'application concernée.

1.2.3 Toute autorité administrative peut définir à toutes fins utiles ses propres types (syntaxes) d'attributs. Ceux-ci n'étant pas normalisés à l'échelon international, les autres autorités administratives ne peuvent les utiliser qu'aux termes d'accords bilatéraux.

2 Références

ISO 3166 - Codes pour la représentation des noms de pays

Recommandation X.121 - Plan de numérotage international pour les réseaux publics pour données

Recommandation X.208 - Interconnexion des Systèmes Ouverts - Spécification de la notation 1 de la syntaxe abstraite (ASN.1) (voir aussi ISO 8824)

Recommandation X.501 - L'annuaire - Modèles (voir aussi ISO 9594-2)

Recommandation X.521 - L'annuaire - Catégories d'objets sélectionnées (voir aussi ISO 9594-7)

Recommandation E.123 - Notation pour les numéros de téléphone nationaux et internationaux

3 Définitions

La présente Recommandation utilise les définitions suivantes, tirées de la Recommandation X.501:

a) *type d'attribut*;

b) *syntaxe d'attribut*;

c) *catégorie d'objet*.

4 Notation

Les types et les syntaxes d'attributs sont définis dans le présent document au moyen de la notation spéciale de macros ASN.1 de la Recommandation X.501. Il existe deux de ces macros: **ATTRIBUTE** et **ATTRIBUTE-SYNTAX**.

Deux identificateurs d'objet "génériques" (**attributeType** et **attributeSyntax**) sont utilisés pour définir les identificateurs d'objet respectivement affectés aux types d'attributs et aux syntaxes d'attributs. On trouvera leur définition dans l'annexe B de la Recommandation X.501.

Des exemples d'emploi des types d'attributs sont décrits à l'aide d'une notation informelle où les types d'attributs et les paires de valeurs sont représentés par un acronyme pour le type d'attribut, suivi du signe égal ("="), lui-même suivi de la valeur exemple pour l'attribut.

SECTION 1 - *Types d'attributs sélectionnés*

5 Définition des types d'attributs sélectionnés

La présente Recommandation définit plusieurs types d'attributs qui peuvent se révéler utiles pour toute une gamme d'applications de l'annuaire.

5.1 *Types d'attributs de système*

Ces types d'attributs concernent les informations relatives aux objets connus de l'annuaire.

5.1.1 *Catégorie d'objet*

Le type d'attribut *Object Class*, connu de l'annuaire est spécifié, sauf dans le cas de l'attribution d'un identificateur d'objet, dans la Recommandation X.501.

objectClass ObjectClass ::= {attributeType 0}

5.1.2 *Nom d'objet pseudonyme*

Ce type d'attribut est défini, sauf pour l'attribution d'un identificateur d'objet, dans la Recommandation X.501.

aliasedObjectName AliasedObjectName ::= {attributeType 1}

5.1.3 *Information de connaissance*

Le type d'attribut *Knowledge Information* spécifie une description accumulée, accessible en lecture par l'homme, de la connaissance maîtrisée par un DSA donné.

```
knowledgeInformation ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
    ::= {attributeType 2}
```

5.2 *Types d'attributs avec étiquetage*

Ces types d'attributs concernent l'information relative à des objets, qui a été explicitement associée aux objets par un processus d'étiquetage.

5.2.1 *Common Name*

Le type d'attribut *Common Name* spécifie l'identificateur d'un objet. Le Common Name n'est pas un nom d'annuaire, mais le nom (parfois ambigu) sous lequel l'objet est généralement connu dans un cercle assez restreint (par exemple, une organisation) et il est conforme aux conventions d'appellation du pays ou de la culture auxquels il est associé.

Pour Common Name, une valeur d'attributs est une chaîne choisie par la personne ou l'organisation qu'elle décrit ou par l'organisation responsable de l'objet décrit pour des dispositifs et des entités d'application. Par exemple, le nom typique d'une personne se trouvant dans un pays anglophone comprend un titre personnel (par exemple, Mr, Mrs, Dr, Professor, Sir, Lord), un prénom, un ou plusieurs autres prénoms, un nom de famille, une indication de génération (par exemple, le cas échéant, Jr.), les titres et les décorations (par exemple, le cas échéant, QC).

Exemples:

CN = "Mr. Robin, Lachlan, McLeod, BSc(Hons) CEng MIEE"

CN = "Divisional Coordination Committee"

CN = "High Speed Modem"

Toutes les variantes doivent être associées à l'objet nommé sous la forme de valeurs d'attributs séparées et secondaires.

D'autres variantes communes doivent aussi être admises, par exemple, l'emploi du second prénom au lieu du premier, le remplacement de "William" par "Bill", etc.

```
commonName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-common-name))
    ::= {attributeType 3}
```

5.2.2 *Surname*

Le type d'attribut *Surname* spécifie la construction linguistique dont une personne hérite en général de ses parents ou qu'elle prend lors de son mariage et sous laquelle elle est généralement connue.

Pour Surname, une valeur d'attribut est une chaîne, par exemple "McLeod".

surName ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
caseIgnoreStringSyntax
(SIZE(1..ub-surname))
::= {attributeType 4}

5.2.3 *Serial Number*

Le type d'attribut *Serial Number* spécifie un identificateur, le numéro de série d'un dispositif.

Pour Serial Number, une valeur d'attribut est une chaîne imprimable.

serialNumber ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
printableStringSyntax
(SIZE(1..ub-serial-number))
::= {attributeType 5}

5.3 *Types d'attributs géographiques*

Ces types d'attributs concernent les positions géographiques ou les régions auxquelles des objets sont associés.

5.3.1 *Country Name*

Le type d'attribut *Country Name* spécifie un pays. Utilisé comme composant d'un nom d'annuaire, il identifie le pays dans lequel l'objet nommé est situé physiquement ou auquel cet objet est associé d'une autre manière pertinente.

Pour Country Name, une valeur d'attribut est une chaîne choisie dans la norme ISO 3166.

countryName ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
PrintableString (SIZE(2)) - *IS 3166 codes only*
MATCHES FOR EQUALITY
SINGLE VALUE
::= {attributeType 6}

La règle de correspondance applicable aux valeurs de ce type est la même que pour **caseIgnoreStringSyntax**.

5.3.2 *Locality Name*

Le type d'attribut *Locality Name* spécifie une localité. Utilisé comme composant d'un nom d'annuaire, il identifie une zone géographique ou une localité dans laquelle l'objet nommé est situé physiquement ou à laquelle cet objet est associé d'une autre manière pertinente.

Pour Locality Name, une valeur d'attribut est une chaîne, par exemple,

localityName ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
caseIgnoreStringSyntax
(SIZE(1..ub-locality-name))
::= {attributeType 7}

5.3.3 *State or Province Name*

Le type d'attribut *State or Province Name* spécifie un Etat ou une province. Utilisé comme composant d'un nom d'annuaire, il identifie une subdivision géographique dans laquelle l'objet nommé est situé physiquement ou à laquelle cet objet est associé d'une autre manière pertinente.

Pour State or Province Name, une valeur d'attribut est une chaîne, par exemple, S = "Ohio".

stateOrProvinceName ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
caseIgnoreStringSyntax
(SIZE(1..ub-state-name))
::= {attributeType 8}

5.3.4 *Street Address*

Le type d'attribut *Street Address* spécifie un emplacement pour la distribution locale et la remise physique à une adresse postale, c'est-à-dire le nom de la rue, de la place, ou de l'avenue et le numéro de la maison. Utilisé comme composant d'un nom d'annuaire, il identifie l'adresse de la rue à laquelle l'objet nommé est situé ou à laquelle cet objet est associé d'une autre manière pertinente.

Pour Street Address, une valeur d'attribut est une chaîne, par exemple, "Arnulfstraße 60".

```
streetAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-street-address))
    ::= {attributeType 9}
```

5.4 *Types d'attributs d'organisation*

Ces types d'attributs concernent les organisations et peuvent servir à décrire des objets d'après les organisations auxquelles ils sont associés.

5.4.1 *OrganizationName*

Le type d'attribut *OrganizationName* spécifie une organisation. Utilisé comme composant d'un nom d'annuaire, il identifie une organisation à laquelle l'objet nommé est affilié.

Pour OrganizationName, une valeur d'attribut est une chaîne choisie par l'organisation (par exemple, 0 = "Scottish Telecommunications plc"). Toute variante doit être associée à l'Organisation nommée sous la forme de valeurs d'attribut séparées et secondaires.

```
organizationName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-organization-name))
    ::= {attributeType 10}
```

5.4.2 *Organizational Unit Name*

Le type d'attribut *Organizational Unit Name* spécifie une unité d'organisation. Utilisé comme composant d'un nom d'annuaire, il identifie une unité d'organisation à laquelle l'objet nommé est affilié.

On considère que l'unité d'organisation désignée est une partie d'une organisation désignée par un attribut OrganizationalName.

En conséquence, si un attribut Organizational Unit Name est utilisé dans un nom d'annuaire, il doit être associé à un attribut OrganizationName.

Pour Organizational Unit Name, une valeur d'attribut est une chaîne choisie par l'organisation dont l'unité fait partie (par exemple, OU = "Technology Division"). On notera que l'abréviation "TD" communément utilisée serait une valeur d'attribut séparée et secondaire.

Exemples:

0 = "Scottel", OU = "TD"

```
organizationalUnitName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-organizational-unit-name))
    ::= {attributeType 11}
```

5.4.3 *Title*

Le type d'attribut *Title* spécifie la position ou la fonction désignées de l'objet dans une organisation.

Pour Title, une valeur d'attribut est une chaîne.

Exemple:

T = "Manager, Distributed Applications"

```
title ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-title))
    ::= {attributeType 12}
```

5.5 *Types d'attributs explicatifs*

Ces types d'attributs expliquent (par exemple, dans un langage naturel) quelque chose qui concerne un objet.

5.5.1 *Description*

Le type d'attribut *Description* spécifie un texte qui décrit l'objet associé.

Par exemple, à l'objet "Participation aux normes" peut être associée la description "liste de distribution pour l'échange d'information concernant l'élaboration des normes dans la compagnie".

Pour Description, une valeur d'attribut est une chaîne.

```
description ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-description))
    ::= {attributeType 13}
```

5.5.2 *Search Guide*

Le type d'attribut *Search Guide* spécifie l'information des critères de recherche suggérés qui peuvent être inclus dans certaines entrées devant être un objet de base commode pour l'opération de recherche, par exemple un pays ou une organisation.

Les critères de recherche comprennent un identificateur facultatif pour la catégorie d'objet recherché et des combinaisons de types d'attributs et d'opérateurs logiques à utiliser pour la construction d'un filtre. Il est possible de spécifier, pour chaque élément de critère de recherche, le niveau de correspondance, par exemple une correspondance approximative.

L'attribut *Search Guide* peut se répéter pour tenir compte des divers types de demande, par exemple, une recherche d'une Residential Person ou d'une Organizional Person, qui peuvent être accomplis à partir de l'objet de base donné où le Search Guide est lu.

```
searchGuide ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        Guide
    ::= {attributeType 14}

Guide ::= SET {
    objectClass             [0]  OBJECT-CLASS OPTIONAL,
    criteria                [1]  Criteria}

Criteria ::=
    CHOICE {
        Type                [0]  CriteriaItem,
        and                 [1]  SET OF Criteria,
        or                  [2]  SET OF Criteria,
        not                 [3]  Criteria}

CriteriaItem
    CHOICE {
        equality            [0]  AttributeType,
        substrings          [1]  AttributeType,
        greaterOrEqual      [2]  AttributeType,
        lessOrEqual         [3]  AttributeType,
        approximateMatch    [4]  AttributeType}
```

Exemple: On trouvera ci-après une valeur potentielle de l'attribut Search Guide qui peut être stocké dans des entrées de catégorie d'objet Locality pour indiquer comment il est possible de trouver des entrées de la catégorie d'objet Residential Person.

```
residential-person-guide      Guide ::= {
      objectClass residentialPerson,
      criteria and {
            type substrings commonName,
            type substrings streetAddress }}
```

La construction d'un Filtre à partir de cette valeur de Guide est simple.

L'étape (1) produit la valeur de Filtre intermédiaire:

```
intermediate-filter    Filter ::=    and {
      item substrings  {
            type commonName,
            strings {any T61String "Dubois" }}, - value supplied for CommonName
      item substrings {
            type streetAddress
            strings {any T61String "Hugo" }}}   - value supplied for StreetAddress
```

L'étape (2) produit un filtre pour mettre en correspondance les entrées Residential Person dans le sous-arbre:

```
residential-person-filter      Filter  ::=   {
  and {
      item equality {
        objectClass,
        OBJECT-CLASS residentialPerson },
      intermediate-filter }}
```

5.5.3 *Business Category*

L'attribut *Business Category* spécifie l'information relative à l'occupation de certains objets communs, tels que des personnes. Par exemple, cet attribut permet d'interroger l'annuaire au sujet de personnes partageant la même occupation.

```
businessCategory ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX
            caseIgnoreStringSyntax
                  (SIZE(1..ub-business-category))
      ::=   {attributeType 15}
```

5.6 *Types d'attributs d'adressage postal*

Ces types d'attributs concernent l'information nécessaire pour la remise postale physique à un objet.

5.6.1 *Postal Address*

Le type d'attribut *Postal Address* spécifie l'information d'adresse nécessaire pour la remise physique des messages postaux par l'autorité postale à l'objet nommé.

Pour Postal Address, une valeur d'attribut sera généralement composée d'attributs choisis à partir de la MHS Unformatted Postal O/R Address version 1 conformément à la Recommandation F.401 et limitée à 6 lignes de 30 caractères comprenant un Nom de Code Postal. Normalement, l'information contenue dans une telle adresse peut comprendre un nom de destinataire, nom et numéro de la rue, ville, Etat ou province, code postal et éventuellement un numéro de Boîte Postale, en fonction des besoins propres à l'objet nommé.

```
postalAddress ATTRIBUTE
      WITH ATTRIBUTE-SYNTAX PostalAddress
      MATCHES FOR EQUALITY
      ::=   {attributeType 16}

PostalAddress ::= SEQUENCE SIZE(1..ub-postal-line) OF CHOICE {
      T61String (SIZE(1..ub-postal-string)),
      PrintableString (SIZE(1..ub-postal-string))}
```

La règle de mise en correspondance applicable aux valeurs de ce type est la même que pour **caseIgnoreListSyntax**.

5.6.2 *Postal Code*

Le type d'attribut *Postal Code* spécifie le code postal de l'objet nommé. Si cette valeur d'attribut est présente, elle fait partie de l'adresse postale de l'objet.

Pour Postal Code, une valeur d'attribut est une chaîne.

```
postalCode ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-postal-code))
    ::= {attributeType 17}
```

5.6.3 *Post Office Box*

Le type d'attribut *Post Office Box* spécifie la Boîte Postale par laquelle l'objet recevra la remise postale physique. Si elle est présente, la valeur d'attribut fait partie de l'adresse postale de l'objet.

```
postOfficeBox ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-post-office-box))
    ::= {attributeType 18}
```

5.6.4 *Physical Delivery Office Name*

Le type d'attribut *Physical Delivery Office Name* spécifie le nom de la ville, du village, etc., où se trouve un bureau de remise physique.

Pour Physical Delivery Office Name, une valeur d'attribut est une chaîne.

```
physicalDeliveryOfficeName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-physical-office-name))
    ::= {attributeType 19}
```

5.7 *Types d'attributs d'adressage pour télécommunication*

Ces types d'attributs concernent l'information d'adressage nécessaire pour communiquer avec l'objet en utilisant des moyens de télécommunication.

5.7.1 *Telephone Number*

Le type d'attribut *Telephone Number* spécifie un numéro de téléphone associé à un objet.

Pour Telephone Number, une valeur d'attribut est une chaîne qui correspond au format international normalisé pour représenter les numéros de téléphone internationaux (Recommandation E.123), par exemple "+ 44 582 10101".

```
telephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        telephoneNumberSyntax
    ::= {attributeType 20}
```

5.7.2 *Telex Number*

Le type d'attribut *Telex Number* spécifie le numéro télex, l'indicatif de pays et l'indicatif d'un terminal télex associé à un objet.

```
telexNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX TelexNumber
    ::= {attributeType 21}
```

```
TelexNumber ::=          SEQUENCE{
    telexNumber                  PrintableString,
        (SIZE(1..ub-telex-number)),
    countryCode                  PrintableString,
        (SIZE(1..ub-country-code)),
    answerback PrintableString,
        (SIZE(1..ub-answerback))}
```

5.7.3 *Teletex Terminal Identifier*

Le type d'attribut *Teletex Terminal Identifier* spécifie l'identificateur (et, facultativement, les paramètres) du terminal télétex associé à un objet.

Pour Teletex Terminal Identifier, une valeur d'attribut est une chaîne qui satisfait aux spécifications de la Recommandation F.200 du CCITT et un ensemble facultatif dont les composants sont conformes à la Recommandation T.62.

```
teletexTerminalIdentifier ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        TeletexTerminalIdentifier
    ::=  {attributeType 22}

TeletexTerminalIdentifier ::=  SEQUENCE {
    teletexTerminal PrintableString
        (SIZE(1..ub-teletex-terminal-id)),
    parameters TeletexNonBasicParameters
        OPTIONAL}
```

5.7.4 *Facsimile Telephone Number*

Le type d'attribut *Facsimile Telephone Number* spécifie un numéro de téléphone pour un terminal de télécopie (et, facultativement, ses paramètres) associé à un objet.

Pour Fascimile Telephone Number, une valeur d'attribut est une chaîne qui satisfait au format international convenu pour représenter les numéros de téléphone internationaux (Recommandation E.123), par exemple "+81 3 347 7418" et une chaîne de bits facultative (formatée selon la Recommandation T.30).

```
facsimileTelephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
            FacsimileTelephoneNumber
    ::= {attributeType 23}

FacsimileTelephoneNumber ::= SEQUENCE{
    telephoneNumber PrintableString
        (SIZE(1..ub-telephone-number)),
    parameters G3FacsimileNonBasicParameters
        OPTIONAL}
```

5.7.5 *X.121 Address*

Le type d'attribut *X.121 Address* spécifie une adresse, telle que définie dans la Recommandation X.121 du CCITT et associée à un objet.

```
x121Address ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        NumericString
            (SIZE(1..ub-x121-address))
    MATCHES FOR EQUALITY SUBSTRINGS
    ::= {attributeType 24}
```

Les règles de correspondance applicables aux valeurs de ce type sont les mêmes que pour **numericStringSyntax**.

5.7.6 *International ISDN Number*

Le type d'attribut *International ISDN Number* spécifie un numéro RNIS international associé à un objet.

Pour International ISDN Number, une valeur d'attribut est une chaîne qui satisfait au format international convenu pour les adresses RNIS et présenté dans la Recommandation E.164 du CCITT.

```
internationalISDNNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        NumericString
            (SIZE(1..ub-isdn-address))
    ::= {attributeType 25}
```

La règle de correspondance applicable aux valeurs de ce type est la même que pour **numericStringSyntax.**

5.7.7 *Registered Address*

Le type d'attribut *Registered Address* spécifie un mnémonique pour une adresse associée à un objet à l'emplacement d'une ville donnée. Le mnémonique est enregistré dans le pays où se trouve la ville et il est utilisé pour la fourniture du service public des télégrammes (selon la Recommandation F.1).

```
registeredAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX PostalAddress
    ::= {attributeType 26}
```

5.7.8 *Destination Indicator*

Le type d'attribut *Destination Indicator* spécifie (selon les Recommandations F.1 et F.3) le pays et la ville associés à l'objet (le destinataire) et nécessaires pour fournir le service public des télégrammes.

Pour Destination Indicator, une valeur d'attribut est une chaîne.

```
destinationIndicator ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        PrintableString
            (SIZE(1..ub-destination-indicator))
    - alphabetical characters only
    MATCHES FOR EQUALITY SUBSTRINGS
    ::= {attributeType 27}
```

Les règles de correspondance applicables aux valeurs de ce type sont les mêmes que pour **caseIgnoreStringSyntax.**

5.8 *Types d'attributs de préférence*

Ces types d'attributs concernent les préférences d'un objet.

5.8.1 *Preferred Delivery Method*

Le type d'attribut *Preferred Delivery Method* spécifie l'ordre de priorité de l'objet au sujet de la méthode à utiliser pour communiquer avec lui.

```
preferredDeliveryMethod ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        SEQUENCE OF INTEGER {
            any-delivery-method     (0),
            mhs-delivery            (1),
            physical-delivery       (2),
            telex-delivery          (3),
            teletex-delivery        (4),
            g3-facsimile-delivery   (5),
            g4-facsimile-delivery   (6),
            ia5-terminal-delivery   (7),
            videotex-delivery       (8),
            telephone-delivery      (9)}
    SINGLE VALUE
    ::= {attributeType 28}
```

5.9 *Types d'attributs d'application OSI*

Ces types d'attributs concernent l'information relative aux objets dans la Couche Application OSI.

5.9.1 *Presentation Address*

Le type d'attribut *Presentation Address* spécifie une adresse de présentation associée à un objet représentant une entité d'application OSI.

Pour Presentation Address, une valeur d'attribut est une adresse de présentation telle que définie dans la Recommandation X.200.

```
presentationAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        PresentationAddress
    MATCHES FOR EQUALITY
    SINGLE VALUE
    ::= {attributeType 29}

PresentationAddress ::=          SEQUENCE{
    pSelector       [0]          OCTET STRING OPTIONAL,
    sSelector       [1]          OCTET STRING OPTIONAL,
    tSelector       [2]          OCTET STRING OPTIONAL,
    nAddresses      [3]          SET SIZE (1..MAX) OF OCTET STRING}
```

La règle de correspondance applicable aux valeurs de ce type est la suivante: une adresse de présentation proposée correspond à une adresse enregistrée si, et seulement si, les sélecteurs sont égaux et si les nAddresses présentées sont un sous-ensemble des adresses enregistrées.

5.9.2 *Supported Application Context*

Le type d'attribut *Supported Application Context* spécifie l'identificateur/les identificateurs d'objet d'un/de contexte(s) d'application que l'objet (une entité d'application OSI) admet.

```
supportedApplicationContext ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        objectIdentifierSyntax
    ::= {attributeType 30}
```

5.10 *Types d'attributs relationnels*

Ces types d'attributs concernent une information relative aux objets qui, d'une certaine façon, sont associés à un objet particulier.

5.10.1 *Member*

Le type d'attribut *Member* spécifie un groupe de noms associés à l'objet.

Pour Member, une valeur d'attribut est un nom spécifique.

```
member ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        distinguishedNameSyntax
    ::= {attributeType 31}
```

5.10.2 *Owner*

Le type d'attribut *Owner* spécifie le nom d'un objet qui a quelque responsabilité à l'égard de l'objet associé.

Pour Owner, une valeur d'attribut est un nom spécifique (qui peut représenter un groupe de noms); elle peut se répéter.

```
owner ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        distinguishedNameSyntax
    ::= {attributeType 32}
```

5.10.3 *Role Occupant*

Le type d'attribut *Role Occupant* spécifie le nom d'un objet qui assume un rôle dans l'organisation.

Pour Role Occupant, une valeur d'attribut est un nom spécifique.

roleOccupant ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
distinguishedNameSyntax
::= {attributeType 33}

5.10.4 *See Also*

Le type d'attribut *See Also* spécifie les noms d'autres objets d'annuaire qui peuvent être d'autres aspects (dans un certain sens) du même objet concret.

Pour See Also, une valeur d'attribut est un nom spécifique.

seeAlso ATTRIBUTE
WITH ATTRIBUTE-SYNTAX
distinguishedNameSyntax
::= {attributeType 34}

5.11 *Types d'attributs de sécurité*

Ces types d'attributs concernent la sécurité ou les privilèges de sécurité d'un objet. Ils sont spécifiés, sauf pour l'attribution d'un identificateur d'objet, dans la Recommandation X.509.

5.11.1 *User Password*

userPassword UserPassword
::= {attributeType 35}

5.11.2 *User Certificate*

userCertificate UserCertificate
::= {attributeType 36}

5.11.3 *CA Certificate*

cACertificate CACertificate
::= {attributeType 37}

5.11.4 *Authority Revocation List*

authorityRevocationList AuthorityRevocationList
::= {attributeType 38}

5.11.5 *Certificate Revocation List*

certificateRevocationList CertificateRevocationList
::= {attributeType 39}

5.11.6 *Cross Certificate Pair*

crossCertificatePair CrossCertificatePair
::= {attributeType 40}

SECTION 2 - *Syntaxes d'attributs*

6 Définition des syntaxes d'attributs

6.1 *Syntaxes d'attributs utilisées par l'annuaire*

6.1.1 *Undefined*

La syntaxe d'attribut *Undefined* est destinée aux attributs dont les valeurs ne seront sans doute pas comparées par l'annuaire.

Spécifier cette syntaxe d'attribut pour un attribut équivaut à spécifier le type de données **ANY** et aucune règle de correspondance dans la macro **ATTRIBUTE** pour l'attribut.

undefined ATTRIBUTE-SYNTAX
ANY
::= {attributeSyntax 0}

6.1.2 *Distinguished Name*

La syntaxe d'attribut *Distinguished Name* est destinée aux attributs dont les valeurs sont des noms spécifiques. Elle est définie, sauf pour l'attribution d'un identificateur d'objet, dans la Recommandation X.501.

distinguishedNameSyntax DistinguishedNameSyntax
::= {attributeSyntax 1}

6.1.3 *Object Identifier*

La syntaxe d'attribut *Object Identifier* est destinée aux attributs dont la valeur est un identificateur d'objet. Elle est définie, sauf pour l'attribution d'un identificateur d'objet, dans la Recommandation X.501.

objectIdentifierSyntax ObjectIdentifierSyntax
::= {attributeSyntax 2}

6.2 *Syntaxes d'attributs de chaîne*

Dans les syntaxes spécifiées aux § 6.2.1 à 6.2.4, les espaces suivants sont considérés comme non significatifs:

- les espaces précédant le premier caractère imprimé;
- les espaces suivant le dernier caractère imprimé;
- les espaces internes multiples consécutifs (considérés comme équivalant à un seul caractère espace).

Les attributs conformes à ces syntaxes seront mis en correspondance sous une forme omettant les espaces qui ne sont pas significatifs en vertu de ces règles.

6.2.1 *Case Exact String*

La syntaxe d'attribut *Case Exact String* est destinée aux attributs dont les valeurs sont des chaînes (Chaînes T.61 ou Chaînes Imprimables), mais quand la casse (majuscules/minuscules) est significative aux fins de comparaison (par exemple, "Dundee" ne correspond pas à "DUNDEE").

caseExactString ATTRIBUTE-SYNTAX
CHOICE {T61String, PrintableString}
MATCHES FOR EQUALITY SUBSTRINGS
::= {attributeSyntax 3}

Pour deux chaînes ayant cette syntaxe à faire correspondre pour l'égalité, les chaînes doivent avoir la même longueur et les caractères correspondants doivent être identiques. Une Chaîne Imprimable peut être comparée avec une Chaîne T.61; quand les caractères correspondants se trouvent tous deux dans le jeu de caractères de Chaîne Imprimable, la comparaison se poursuit normalement. En revanche, si le caractère de la Chaîne T.61 ne se trouve pas dans le jeu de caractères de la Chaîne Imprimable, la mise en correspondance échoue.

6.2.2 *Case Ignore String*

La syntaxe d'attribut *Case Ignore String* est destinée aux attributs dont les valeurs sont des chaînes (Chaînes T.61 ou Chaînes Imprimables), mais quand la casse (minuscules ou majuscules) n'est pas significative aux fins de comparaison (par exemple, "Dundee" concorde avec "DUNDEE").

caseIgnoreStringSyntax ATTRIBUTE-SYNTAX
CHOICE {T61String, PrintableString}
MATCHES FOR EQUALITY SUBSTRINGS
::= {attributeSyntax 4}

Les règles de correspondance sont les mêmes que pour la syntaxe d'attribut Case Exact String, si ce n'est que les caractères qui diffèrent seulement par la casse sont considérés comme identiques.

6.2.3 *Printable String*

La syntaxe d'attribut *Printable String* est destinée aux attributs dont les valeurs sont des Chaînes Imprimables.

printableStringSyntax ATTRIBUTE-SYNTAX
PrintableString
MATCHES FOR EQUALITY SUBSTRINGS
::= {attributeSyntax 5}

Les règles de correspondance sont les mêmes que pour la syntaxe d'attribut Case Exact String.

6.2.4 *Numeric String*

La syntaxe d'attribut *Numeric String* est destinée aux attributs dont les valeurs sont des Chaînes Numériques.

numericStringSyntax ATTRIBUTE-SYNTAX
NumericString
MATCHES FOR EQUALITY SUBSTRINGS
::= {attributeSyntax 6}

Les règles de correspondance sont les mêmes que pour la syntaxe d'attribut Case Exact String, si ce n'est que tous les caractères espace sont omis pendant la comparaison.

6.2.5 *Case Ignore List*

La syntaxe d'attribut *Case Ignore List* est destinée aux attributs dont les valeurs sont des séquences de chaînes (Chaînes T.61 ou Chaînes Imprimables), mais quand la casse (minuscules/majuscules) n'est pas significative aux fins de comparaison.

caseIgnoreListSyntax ATTRIBUTE-SYNTAX
SEQUENCE OF
CHOICE {T61String, PrintableString}
MATCHES FOR EQUALITY SUBSTRINGS
::= {attributeSyntax 7}

Deux Case Ignore Lists correspondent pour l'égalité si, et seulement si, le nombre de chaînes de chacun d'eux est le même et si les chaînes correspondantes concordent. Cette dernière concordance est la même que pour la syntaxe d'attribut Case Ignore String (§ 6.1.3).

6.3 *Syntaxes d'attributs diverses*

6.3.1 *Boolean*

La syntaxe d'attribut *Boolean* est destinée aux attributs dont les valeurs sont booléennes (c'est-à-dire représentent vrai ou faux).

booleanSyntax ATTRIBUTE-SYNTAX
BOOLEAN
MATCHES FOR EQUALITY
::= {attributeSyntax 8}

Deux valeurs d'attribut de cette syntaxe concordent pour l'égalité si elles sont toutes deux vraies ou toutes deux fausses.

6.3.2 *Integer*

La syntaxe d'attribut *Integer* est destinée aux attributs dont les valeurs sont des nombres entiers.

integerSyntax ATTRIBUTE-SYNTAX
INTEGER
MATCHES FOR EQUALITY ORDERING
::= {attributeSyntax 9}

Deux valeurs d'attribut de cette syntaxe concordent pour l'égalité si les nombres entiers sont les mêmes. Les règles d'ordre pour les nombres entiers s'appliquent.

6.3.3 *Octet String*

La syntaxe d'attribut *Octet String* est destinée aux attributs dont les valeurs sont des Chaînes d'Octets.

octetStringSyntax ATTRIBUTE-SYNTAX
OCTET STRING
MATCHES FOR EQUALITY SUBSTRINGS ORDERING
::= {attributeSyntax 10}

Pour deux chaînes ayant cette syntaxe d'attribut à mettre en correspondance, les chaînes doivent avoir la même longueur et les octets correspondants doivent être identiques. L'ordre est déterminé par la relation d'ordre entre les premiers octets devant différer lors de la comparaison des chaînes depuis le début.

6.3.4 *UTC Time*

La syntaxe d'attribut *UTC Time* est destinée aux attributs dont les valeurs représentent une heure absolue.

uTCTimeSyntax ATTRIBUTE-SYNTAX
UTCTime
MATCHES FOR EQUALITY ORDERING
::= {attributeSyntax 11}

Deux valeurs d'attribut de cette syntaxe concordent pour l'égalité si elles représentent la même heure. Une heure antérieure est considérée comme "inférieure" à une heure postérieure.

6.3.5 *Telephone Number*

La syntaxe d'attribut *Telephone Number* est destinée aux attributs dont les valeurs sont des numéros de téléphone.

telephoneNumberSyntax ATTRIBUTE-SYNTAX
PrintableString
(SIZE(1..ub-telephone-number))
MATCHES FOR EQUALITY SUBSTRINGS
::= {attributeSyntax 12}

Les règles de correspondance sont les mêmes que pour la syntaxe d'attribut Case Exact, si ce n'est que tous les caractères espace et "-" sont omis pendant la comparaison.

ANNEXE A

(à la Recommandation X.520)

Types d'attributs sélectionnés en ASN.1

Cette annexe fait partie de la Recommandation.

Elle comprend toutes les définitions concernant le type et la valeur ASN.1 contenues dans la présente Recommandation sous la forme du module ASN.1 **SelectedAttributeTypes**.

```
SelectedAttributeTypes      {joint-iso-ccitt ds(5) modules(1)
                             selectedAttributeTypes(5)}
DEFINITIONS ::=
BEGIN

-- Exports everything --

IMPORTS
    informationFramework, authenticationFramework, attributeType,
    upperBounds
        FROM    UsefulDefinitions    {joint-ISO-CCITT ds(5) modules(1)
                                      usefulDefinitions(0)},
    ATTRIBUTE, ATTRIBUTE-SYNTAX, AttributeType, OBJECT-CLASS,
    ObjectClass, AliasedObjectName,
    DistinguishedNameSyntax, ObjectIdentifierSyntax
        FROM    InformationFramework informationFramework
    G3FacsimileNonBasicParameters,
    TeletexNonBasicParameters
        FROM    MTSAbstractService  {joint-ISO-CCITT mhs-motis(6)
                                     mts(3) modules(0) mts-abstract-service(1)}
    UserCertificate, CACertificate, CrossCertificatePair,
      CertificateRevocationList,
    AuthorityRevocationList, UserPassword
        FROM    AuthenticationFramework, authenticationFramework
        ub-answerback,
        ub-common-name, ub-surname, ub-serial-number,
        ub-locality-name, ub-state-name,
        ub-street-address, ub-organization-name,
        ub-organizational-unit-name, ub-title,
        ub-description, ub-business-category, ub-postal-line,
        ub-postal-string, ub-postal-code, ub-post-office-box,
        ub-physical-office-name, ub-telex-number,
        ub-country-code, ub-teletex-terminal-id,
        ub-telephone-number, ub-x121-address,
        ub-international-isdn-number, ub-destination-indicator,
        ub-user-password
            FROM  UpperBounds upperBounds;

-- attribute types --

objectClass   ObjectClass ::= {attributeType 0}

aliasedObjectName   AliasedObjectName ::= {attributeType 1}

knowledgeInformation ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX caseIgnoreStringSyntax
    ::= {attributeType 2}

commonName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-common-name))
    ::= {attributeType 3}

surname ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-surname))
    ::= {attributeType 4}

serialNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        printableStringSyntax
            (SIZE(1..ub-serial-number))
    ::= {attributeType 5}
```

```
countryName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                PrintableString (SIZE(2)) - IS 3166 codes only
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= {attributeType 6}
localityName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-locality-name))
        ::= {attributeType 7}
stateOrProvinceName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-state-name))
        ::= {attributeType 8}
streetAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-street-address))
        ::= {attributeType 9}
organizationName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-organization-name))
        ::= {attributeType 10}
organizationalUnitName ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-organizational-unit-name))
        ::= {attributeType 11}
title ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-title))
        ::= {attributeType 12}
description ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                caseIgnoreStringSyntax
                        (SIZE(1..ub-description))
        ::= {attributeType 13}
searchGuide ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                Criteria
        ::= {attributeType 14}
Guide ::= SET {
        objectClass [0] OBJECT-CLASS OPTIONAL,
                criteria [1] Criteria }
Criteria ::=
        CHOICE {
                type [0]  CriteriaItem,
                and  [1]  SET OF Criteria
                or   [2]  SET OF Criteria
                not  [3]  Criteria}
CriteriaItem ::=
        CHOICE {
                equality            [0]  AttributeType
                substrings          [1]  AttributeType
                greaterOrEqual      [2]  AttributeType
                lessOrEqual         [3]  AttributeType
                approximateMatch    [4]  AttributeType}
```

```
businessCategory ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-business-category))
    ::= {attributeType 15}

postalAddress ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX PostalAddress
    MATCHES FOR EQUALITY
    ::= {attributeType 16}

PostalAddress ::= SEQUENCE SIZE(1..ub-postal-line) OF
    CHOICE {
        T61String (SIZE(1..ub-postal-string)),
        PrintableString (SIZE(1..ub-postal-string))}

postalCode ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-postal-code))
    ::= {attributeType 17}

postOfficeBox ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-post-office-box))
    ::= {attributeType 18}

physicalDeliveryOfficeName ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        caseIgnoreStringSyntax
            (SIZE(1..ub-physical-office-name))
    ::= {attributeType 19}

telephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        telephoneNumberSyntax
    ::= {attributeType 20}

telexNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX TelexNumber
    ::= {attributeType 21}

TelexNumber ::= SEQUENCE {
    telexNumber         PrintableString
        (SIZE(1..ub-telex-number)),
    countryCode         PrintableString,
        (SIZE(1..ub-country-code)),
    answerback PrintableString
        (SIZE(1..ub-answerback))}

teletexTerminalIdentifier ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        TeletexTerminalIdentifier
    ::= {attributeType 22}

TeletexTerminalIdentifier ::= SEQUENCE {
    teletexTerminalPrintableString
        (SIZE(1..ub-teletex-terminal-id)),
    parameters TeletexNonBasicParameters
        OPTIONAL}

facsimileTelephoneNumber ATTRIBUTE
    WITH ATTRIBUTE-SYNTAX
        FacsimileTelephoneNumber
    ::= {attributeType 23}

FacsimileTelephoneNumber ::= SEQUENCE {
    telephoneNumber PrintableString
        (SIZE(1..ub-telephone-number)),
    parameters G3FacsimileNonBasicParameters OPTIONAL}
```

```
x121Address ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                NumericString
                        (SIZE(1..ub-x121-address))
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeType 24}

internationalISDNNumber ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                NumericString
                        (SIZE(1..ub-isdn-address))
        ::= {attributeType 25}

registeredAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX PostalAddress
        ::= {attributeType 26}

destinationIndicator ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                PrintableString
                        (SIZE(1..ub-destination-indicator))
        - alphabetical characters only
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeType 27}

preferredDeliveryMethod ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                SEQUENCE OF INTEGER {
                        any-delivery-method     (0),
                        mhs-delivery            (1),
                        physical-delivery       (2),
                        telex-delivery          (3),
                        teletex-delivery        (4),
                        g3-facsimile-delivery   (5),
                        g4-facsimile-delivery   (6),
                        ia5-terminal-delivery   (7),
                        videotex-delivery       (8),
                        telephone-delivery      (9)}
        SINGLE VALUE
        ::= {attributeType 28}

presentationAddress ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                PresentationAddress
        MATCHES FOR EQUALITY
        SINGLE VALUE
        ::= {attributeType 29}

PresentationAddress ::= SEQUENCE {
        pSelector       [0]   OCTET STRING OPTIONAL,
        sSelector       [1]   OCTET STRING OPTIONAL,
        tSelector       [2]   OCTET STRING OPTIONAL,
        nAddresses      [3]   SET SIZE (1..MAX) OF OCTECT STRING}

supportedApplicationContext ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                objectIdentifierSyntax
        ::= {attributeType 30}

member ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                distinguishedNameSyntax
        ::= {attributeType 31}

owner ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                distinguishedNameSyntax
        ::= {attributeType 32}
```

```
roleOccupant ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                distinguishedNameSyntax
        ::= {attributeType 33}

seeAlso ATTRIBUTE
        WITH ATTRIBUTE-SYNTAX
                distinguishedNameSyntax
        ::= {attributeType 34}

userPassword UserPassword
        ::= {attributeType 35}

userCertificate UserCertificate
        ::= {attributeType 36}

cACertificate CACertificate
        ::= {attributeType 37}

authorityRevocationList AuthorityRevocationList
        ::= {attributeType 38}

certificateRevocationList CertificateRevocationList
        ::= {attributeType 39}

crossCertificatePair CrossCertificatePair
        ::= {attributeType 40}


-- attribute syntaxes --

undefined ATTRIBUTE-SYNTAX
        ANY
        ::= {attributeSyntax 0}

distinguishedNameSyntax DistinguishedNameSyntax
        ::= {attributeSyntax 1}

objectIdentifierSyntax ObjectIdentifierSyntax
        ::= {attributeSyntax 2}

caseExactStringSyntax ATTRIBUTE-SYNTAX
        CHOICE {T61String, PrintableString}
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeSyntax 3}

caseIgnoreSyntax ATTRIBUTE-SYNTAX
        CHOICE {T61String, PrintableString}
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeSyntax 4}

printableStringSyntax ATTRIBUTE-SYNTAX
        PrintableString
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeSyntax 5}

numericStringSyntax ATTRIBUTE-SYNTAX
        NumericString
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeSyntax 6}

caseIgnoreListSyntax ATTRIBUTE-SYNTAX
        SEQUENCE OF
                CHOICE {T61String, PrintableString}
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeSyntax 7}

booleanSyntax ATTRIBUTE-SYNTAX
        BOOLEAN
        MATCHES FOR EQUALITY
        ::= {attributeSyntax 8}
```

```
integerSyntax ATTRIBUTE-SYNTAX
        INTEGER
        MATCHES FOR EQUALITY ORDERING
        ::= {attributeSyntax 9}

octetStringSyntax ATTRIBUTE-SYNTAX
        OCTET STRING
        MATCHES FOR EQUALITY SUBSTRINGS ORDERING
        ::= {attributeSyntax 10}

uTCTimeSyntax ATTRIBUTE-SYNTAX
        UTCTime
        MATCHES FOR EQUALITY ORDERING
        ::= {attributeSyntax 11}

telephoneNumberSyntax ATTRIBUTE-SYNTAX
        PrintableString
                (SIZE(1..ub-telephone-number))
        MATCHES FOR EQUALITY SUBSTRINGS
        ::= {attributeSyntax 12}
```

ANNEXE B

(à la Recommandation X.520)

Index des types et des syntaxes d'attributs

TYPES D'ATTRIBUTS

SYNTAXES D'ATTRIBUTS

---

* Connu de l'annuaire et utilisé par lui.

TYPES D'ATTRIBUTS

	Postal Code	§ 5.6.2
	Preferred Delivery Method	§ 5.8.1
	Presentation Address	§ 5.9.1
R	Registered Address	§ 5.7.7
	Role Occupant	§ 5.10.3
S	Search Guide	§ 5.5.2
	See Also	§ 5.10.4
	Serial Number	§ 5.2.3
	State or Province Name	§ 5.3.2
	Street Address	§ 5.3.4
	Supported Application Context	§ 5.9.2
	Surname	§ 5.2.2
T	Telephone Number	§ 5.7.1
	Teletex Terminal Identifier	§ 5.7.3
	Telex Number	§ 5.7.2
	Title	§ 5.4.3
U	User Certificate	§ 5.11.2
	User Password	§ 5.11.1
X	X.121 Address	§ 5.7.5

ANNEXE C

(à la Recommandation X.520)

Limites supérieures

Cette annexe fait partie de la Recommandation.

```
UpperBounds    {joint-ISO-CCITT ds(5) modules(1)
               upperBounds(10)}
DEFINITIONS ::=
BEGIN

-- Exports everything --

ub-answerback                   INTEGER ::= 8

ub-common-name                  INTEGER ::= 64

ub-surname                      INTEGER ::= 64

ub-serial-number                INTEGER ::= 64

ub-locality-name                INTEGER ::= 128

ub-state-name                   INTEGER ::= 128
```

```
    ub-street-address                    INTEGER ::= 128
    ub-organization-name                 INTEGER ::= 64
    ub-organizational-unit-name          INTEGER ::= 64
    ub-title                             INTEGER ::= 64
    ub-description                       INTEGER ::= 1024
    ub-business-category                 INTEGER ::= 128
    ub-postal-line                       INTEGER ::= 6
    ub-postal-string                     INTEGER ::= 30
    ub-postal-code                       INTEGER ::= 40
    ub-post-office-box                   INTEGER ::= 40
    ub-physical-office-name              INTEGER ::= 128
    ub-telex-number                      INTEGER ::= 14
    ub-country-code                      INTEGER ::= 4
    ub-teletex-terminal-id               INTEGER ::= 24
    ub-telephone-number                  INTEGER ::= 32
    ub-x121-address                      INTEGER ::= 15
    ub-international-isdn-number         INTEGER ::= 16
    ub-destination-indicator             INTEGER ::= 128
    ub-user-password                     INTEGER ::= 128
END
```

Recommandation X.521

L'ANNUAIRE - CATEGORIES D'OBJETS SELECTIONNEES [1)]

(Melbourne, 1988)

TABLE DES MATIERES

---

1) La Recommandation X.521 et la norme ISO 9594-7 - Systèmes de traitement de l'information - Interconnexion des systèmes ouverts - L'annuaire - Catégories d'objets sélectionnées - ont été élaborées en étroite collaboration et sont alignées du point de vue technique.

0 Introduction

0.1 Le présent document, ainsi que les autres de cette série, a été élaboré en vue de faciliter l'interconnexion de systèmes informatiques visant à assurer des services d'annuaire. L'ensemble de tous ces systèmes, avec les informations d'annuaire qu'ils détiennent, peut être considéré comme un tout intégré, appelé *annuaire*. Les informations de l'annuaire, désignées collectivement comme la Base de Données d'annuaire (DIB) sont normalement utilisées pour faciliter la communication entre, avec ou à propos d'objets tels que des entités d'application OSI, des personnes, des terminaux et des listes de diffusion.

0.2 L'annuaire joue un rôle significatif dans l'Interconnexion des Systèmes Ouverts (OSI) dont l'objectif est de permettre, au prix d'un minimum d'accords techniques en dehors des normes d'interconnexion proprement dites, d'interconnecter des équipements informatiques:

- de constructeurs différents;
- gérés de façons différentes;
- de niveaux de complexité différents;
- d'âges différents.

0.3 La présente Recommandation définit (dans la section 1) plusieurs ensembles d'attributs et catégories d'objets qui peuvent être jugés utiles dans une gamme d'applications de l'annuaire.

0.4 L'annexe A, qui fait partie de la Recommandation, donne un module ASN.1 contenant toutes les définitions de types et de valeurs qui apparaissent dans le présent document.

0.5 L'annexe B, qui ne fait *pas* partie de la Recommandation, énonce quelques règles courantes d'appellation et de structuration, qui peuvent être utilisées ou non par les autorités administratives.

1 Objet et domaine d'application

1.1 La présente Recommandation définit plusieurs ensembles d'attributs et catégories d'objets sélectionnés qui peuvent être utiles dans une gamme d'applications de l'annuaire. La définition d'un ensemble d'attributs comporte l'identification des attributs contenus dans l'ensemble et facilite la définition des catégories d'objets. La définition d'une catégorie d'objets comporte facultativement l'attribution d'un Identificateur d'Objet à cette catégorie et l'énumération de plusieurs types d'attributs se rapportant aux objets de cette catégorie. Ces définitions sont utilisées par l'autorité administrative responsable de la gestion de l'information de l'annuaire.

1.2 Toute Autorité Administrative peut définir ses propres catégories ou sous-catégories d'objets pour n'importe quelle fin.

Remarque 1 - Ces définitions peuvent ou non utiliser la notation spécifiée dans la Recommandation X.501.

Remarque 2 - Il est recommandé d'utiliser une catégorie d'objets définie dans le présent document ou une sous-catégorie tirée d'une catégorie d'objets plutôt que de créer une nouvelle catégorie, chaque fois que la sémantique convient à l'application.

1.3 Les Autorités Administratives peuvent utiliser certaines des catégories d'objets sélectionnées ou toutes ces catégories et peuvent aussi ajouter des catégories d'objets.

Toutes les Autorités Administratives utilisent les catégories d'objets employées dans l'annuaire à ses propres fins (par exemple, les catégories d'objets top, pseudonyme et DSA).

2 Références

Recommandation X.200 - Modèle de référence pour l'interconnexion des systèmes ouverts pour les applications du CCITT (voir aussi ISO 7498)

Recommandation X.500 - L'annuaire - Aperçu des concepts, modèles et services (voir aussi ISO 9594-1)

Recommandation X.501 - L'annuaire - Modèles (voir aussi ISO 9594-2)

3 Définitions et abréviations

3.1 *Définitions relatives au modèle de référence OSI*

La présente Recommandation utilise les définitions suivantes tirées de la Recommandation X.200:

a) *entité d'application*;

b) *processus d'application.*

3.2 *Définitions relatives au modèle d'annuaire*

La présente Recommandation utilise les définitions suivantes tirées de la Recommandation X.501:

a) *attribut*;

b) *type d'attribut*;

c) *Arbre d'Information de l'annuaire (DIT)*;

d) *Agent du Système de l'annuaire (DSA)*;

e) *ensemble d'attributs*;

f) *entrées*;

g) *nom*;

h) *catégorie d'objets*;

i) *sous-catégorie.*

4 Notation

Dans le présent document, les catégories d'objets sont définies à l'aide d'une notation spéciale, laquelle est définie comme une macro ASN.1, **OBJECT-CLASS**, dans la Recommandation X.501. Un identificateur d'objet "générique" (**objectClass**) sert à définir les identificateurs d'objet attribués aux catégories d'objets. On peut en trouver la définition dans l'annexe B de la Recommandation susmentionnée.

Les ensembles d'attributs sont définis dans le présent document à l'aide d'une notation spéciale, définie comme une macro ASN.1, **ATTRIBUTE-SET**, dans la Recommandation X.501. Un identificateur d'objet "générique" (**attributeSet**) sert à spécifier les identificateurs d'objet affectés aux définitions d'ensembles d'attributs. On peut en trouver la définition dans l'annexe B de la Recommandation susmentionnée.

SECTION 1 - *Catégories d'objets sélectionnées*

5 Définition d'ensembles d'attributs utiles

5.1 *Ensemble d'attributs télécommunications*

Cet ensemble d'attributs sert à définir les attributs couramment utilisés dans les communications d'affaires.

```
telecommunicationAttributeSet ATTRIBUTE-SET
    CONTAINS {
        facsimileTelephoneNumber,
        iSDNAddress,
        telephoneNumber,
        teletexTerminalIdentifier,
        telexNumber, X121Address,
        preferredDeliveryMethod,
        destinationIndicator,
        registeredAddress}
        ::= {attributeSet 0}
```

5.2 *Ensemble d'attributs postes*

Cet ensemble d'attributs sert à définir les attributs directement associés à la remise postale.

```
postalAttributeSet ATTRIBUTE-SET
    CONTAINS {
        physicalDeliveryOfficeName,
        postalAddress,
        postalCode,
        postOfficeBox,
        streetAddress}
        ::= {attributeSet 1}
```

5.3 *Ensemble d'attributs localisation*

Cet ensemble d'attributs sert à définir les attributs couramment utilisés aux fins de recherche pour indiquer la localisation d'un objet.

```
localeAttributeSet ATTRIBUTE-SET
    CONTAINS {
        localityName,
        stateOrProvinceName,
        streetAddress}
        ::= {attributeSet 2}
```

5.4 *Ensemble d'attributs organisation*

Cet ensemble d'attributs sert à définir les attributs qu'une organisation ou une unité d'organisation peut couramment posséder.

```
organizationalAttributeSet ATTRIBUTE-SET
    CONTAINS {
        description,
        localeAttributeSet,
        postalAttributeSet,
        telecommunicationAttributeSet,
        businessCategory,
        seeAlso,
        searchGuide,
        userPassword}
        ::= {attributeSet 3}
```

6 Définition de catégories d'objets sélectionnées

6.1 *Top*

La catégorie d'objets *top*, dont chaque autre catégorie d'objets est une sous-catégorie, est définie, sauf en ce qui concerne l'affectation d'un identificateur d'objet, dans la Recommandation X.501.

```
top Top ::= {objectClass 0}
```

6.2 *Pseudonyme*

La catégorie d'objets *alias*, dont peuvent être tirées des catégories de pseudonymes, est définie dans la Recommandation X.501, sauf en ce qui concerne l'affectation d'un identificateur d'objet.

```
alias Alias ::= {objectClass 1}
```

6.3 *Pays*

Une catégorie d'objets *Country* sert à définir les entrées de pays dans le DIT.

```
country OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        countryName}
    MAY CONTAIN {
        description,
        searchGuide}
    ::= {objectClass 2}
```

6.4 *Localisation*

La catégorie d'objets *Locality* sert à définir la localisation dans le DIT.

```
locality OBJECT-CLASS
    SUBCLASS OF top
    MAY CONTAIN {
        description,
        localityName,
        stateOrProvinceName,
        searchGuide,
        seeAlso,
        streetAddress}
    ::= {objectClass 3}
```

Un nom de localisation ou un nom d'Etat ou de province au moins doit être présent.

6.5 *Organisation*

La catégorie d'objets *Organization* sert à définir des entrées d'organisations dans le DIT.

```
organization OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        organizationName}
    MAY CONTAIN {
        organizationalAttributeSet}
    ::= {objectClass 4}
```

6.6 *Unité d'organisation*

La catégorie d'objets *Organizational Unit* sert à définir des entrées représentant des subdivisions d'organisations.

```
organizationalUnit OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        organizationalUnitName}
    MAY CONTAIN {
        organizationalAttributeSet}
    ::= {objectClass 5}
```

6.7 *Personne*

La catégorie d'objets *Person* sert à définir des entrées représentant génériquement une personne.

```
person OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        surname}
    MAY CONTAIN {
        description,
        seeAlso,
        telephoneNumber,
        userPassword}
    ::= {objectClass 6}
```

6.8 *Personne associée à une organisation*

La catégorie d'objets *Organizational Person* sert à définir des entrées représentant des personnes qui sont employées par une organisation ou lui sont associées d'une autre manière pertinente.

```
organizationalPerson OBJECT-CLASS
    SUBCLASS OF person
    MAY CONTAIN {
        localeAttributeSet,
        organizationalUnitName,
        postalAttributeSet,
        telecommunicationAttributeSet,
        title}
    ::= {objectClass 7}
```

6.9 *Rôle dans l'organisation*

La catégorie d'objets *Organizational Role* sert à définir des entrées représentant un rôle dans l'organisation, c'est-à-dire une position ou un rôle dans l'organisation. On considère généralement qu'un rôle est confié à une personne dans l'organisation mais, tant qu'il existe, ce rôle peut être confié successivement à différentes personnes au sein de l'organisation. En principe, un rôle peut être confié à une personne ou à une entité non humaine.

```
organizationalRole OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName}
    MAY CONTAIN {
        description,
        localeAttributeSet,
        organizationalUnitName,
        postalAttributeSet,
        preferredDeliveryMethod,
        roleOccupant,
        seeAlso,
        telecommunicationAttributeSet}
    ::= {objectClass 8}
```

6.10 *Groupe de noms*

La catégorie d'objets *Group of Names* sert à définir des entrées représentant un ensemble non ordonné de noms qui représentent des objets individuels ou d'autres groupes de noms. La composition d'un groupe est statique, c'est-à-dire qu'elle est explicitement modifiée par une mesure administrative, sans être déterminée dynamiquement chaque fois qu'il est fait référence au groupe considéré.

La composition d'un groupe peut être réduite à un ensemble de noms d'objets individuels par substitution à ce groupe des membres du groupe. Un tel processus pourrait être effectué de manière récurrente jusqu'à ce que tous les noms du groupe aient été éliminés, laissant seulement les noms des objets individuels.

```
groupOfNames OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        member}
    MAY CONTAIN {
        description,
        organizationName,
        organizationalUnitName,
        owner,
        seeAlso,
        businessCategory}
    ::= {objectClass 9}
```

6.11 *Personne du secteur résidentiel*

La catégorie d'objets *Residential Person* sert à définir des entrées représentant une personne qui habite dans un secteur résidentiel.

```
residentialPerson OBJECT-CLASS
    SUBCLASS OF person
    MUST CONTAIN {
        localityName}
    MAY CONTAIN {
        localeAttributeSet,
        postalAttributeSet,
        preferredDeliveryMethod,
        telecommunicationAttributeSet,
        businessCategory}
    ::= {objectClass 10}
```

6.12 *Processus d'application*

La catégorie d'objets *Application Process* sert à définir des entrées représentant des processus d'application. Un processus d'application est un élément d'un système ouvert réel qui exécute le traitement de l'information pour une application particulière (voir la Recommandation X.200).

```
applicationProcess OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName}
    MAY CONTAIN {
        description,
        localityName,
        organizationalUnitName,
        seeAlso}
    ::= {objectClass 11}
```

6.13 *Entité d'application*

La catégorie d'objets *Application Entity* sert à définir des entrées représentant des entités d'application. Une entité d'application se compose des aspects d'un processus d'application pertinent à l'OSI.

```
applicationEntity OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName,
        presentationAddress}
    MAY CONTAIN {
        description,
        localityName,
```

organizationName,
organizationalUnitName,
seeAlso,
supportedApplicationContext}
::= {objectClass 12}

Remarque - Si l'Entité d'Application est représentée comme un objet d'annuaire distinct d'un Processus d'Application, l'attribut **commonName** sert à véhiculer la valeur du Qualificateur d'Entité d'Application.

6.14 *DSA*

La catégorie d'objets *DSA* sert à définir des entrées représentant des DSA. Un DSA est défini dans la Recommandation X.501.

dSA OBJECT-CLASS
SUBCLASS OF applicationEntity
MAY CONTAIN {
knowledgeInformation}
::= {objectClass 13}

6.15 *Dispositif*

La catégorie d'objets *Device* sert à définir des entrées représentant des dispositifs. Un dispositif est un élément physique capable de communiquer, tel qu'un modem, une unité de disques, etc.

device OBJECT-CLASS
SUBCLASS OF top
MUST CONTAIN {
commonName}
MAY CONTAIN {
description,
localityName,
organizationName,
organizationalUnitName,
owner,
seeAlso,
serialNumber}
::= {objectClass 14}

Remarque - Un Nom de localité, Numéro de série, de propriétaire au moins doit être présent. Le choix dépend du type de dispositif.

6.16 *Utilisateur d'authentification poussée*

La catégorie d'objets *Strong Authentication User* sert à définir des objets qui participent à l'authentification poussée, tels qu'ils sont définis dans la Recommandation X.509.

strongAuthenticationUser OBJECT-CLASS
SUBCLASS OF top
MUST CONTAIN {userCertificate}
::= {objectClass 15}

6.17 *Autorité de certification*

La catégorie d'objets *Certification Authority* sert à définir des entrées d'objets utilisés comme autorités de certification, tels qu'ils sont définis dans la Recommandation X.509.

certificationAuthority OBJECT-CLASS
SUBCLASS OF top
MUST CONTAIN {
cACertificate,
certificateRevocationList,
authorityRevocationList}
MAY CONTAIN {crossCertificatePair}
::= {objectClass 16}

ANNEXE A

(à la Recommandation X.521)

Catégories d'objets sélectionnées en ASN.1

Cette annexe comprend toutes les définitions concernant le type et la valeur ASN.1 contenues dans la présente Recommandation sous la forme du module ASN.1 **SelectedObjectClasses.**

```
SelectedObjectClasses    {joint-ISO-CCITT ds(5) modules(1)
                          selectedObjectClasses(6)}
DEFINITIONS ::=
BEGIN

-- exports everything

IMPORTS
    objectClass, attributeSet, informationFramework, selectedAttributeTypes
        FROM  UsefulDefinitions {joint-iso-ccitt ds(5) modules(1)  usefulDefinitions(0)}
    OBJECT-CLASS,ATTRIBUTE-SET, Top, Alias
        FROM  InformationFramework informationFramework
    authorityRevocationList, businessCategory, CACertificate, certificateRevocationList,
    commonName, countryName, description, destinationIndicator, facsimileTelephoneNumber,
    internationalISDNNumber, knowledgeInformation, localityName, member, organizationName,
    organizationalUnitName, owner, physicalDeliveryOfficeName, postOfficeBox, postalAddress,
    postalCode, preferredDeliveryMethod, presentationAddress, registeredAddress,
    roleOccupant, searchGuide, seeAlso, serialNumber, stateOrProvinceName, streetAddress,
    supportedApplicationContext, surname, telephoneNumber, teletexTerminalIdentifier,
    telexNumber, title, userCertificate, userPassword, x121Address
        FROM  SelectedAttributeTypes selectedAttributeTypes;

telecommunicationAttributeSet ATTRIBUTE-SET
    CONTAINS {
        facsimileTelephoneNumber,
        iSDNAddress,
        telephoneNumber,
        teletexTerminalIdentifier,
        telexNumber,
        x121Address, preferredDeliveryMethod, destinationIndicator,
        registeredAddress}
        ::= {attributeSet 0}

postalAttributeSet ATTRIBUTE-SET
    CONTAINS {
        physicalDeliveryOfficeName,
        postalAddress,
        postalCode,
        postOfficeBox,
        streetAddress}
        ::= {attributeSet 1}

localeAttributeSet ATTRIBUTE-SET
    CONTAINS {
        localityName,
        stateOrProvinceName,
        streetAddress}
        ::= {attributeSet 2}

organizationalAttributeSet ATTRIBUTE-SET
    CONTAINS {
        description,
        localeAttributeSet,
        postalAttributeSet,
        telecommunicationAttributeSet,
        businessCategory,
```

```
            seeAlso,
            searchGuide,
            userPassword}
            ::= {attributeSet 3}

top         Top ::= {objectclass 0}

alias       Alias ::= {objectClass 1}

country OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
            countryName}
      MAY CONTAIN {
            description,
            searchGuide}
      ::= {objectClass 2}

locality OBJECT-CLASS
      SUBCLASS OF top
      MAY CONTAIN {
            description,
            localityName,
            stateOrProvinceName,
            searchGuide,
            seeAlso,
            streetAddress}
      ::= {objectClass 3}

organization OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
            organizationName}
      MAY CONTAIN {
            organizationalAttributeSet}
      ::= {objectClass 4}

organizationalUnit OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
            organizationalUnitName}
      MAY CONTAIN {
            organizationalAttributeSet}
      ::= {objectClass 5}

person OBJECT-CLASS
      SUBCLASS OF top
      MUST CONTAIN {
            commonName,
            surname}
      MAY CONTAIN {
            description,
            seeAlso,
            telephoneNumber,
            userPassword}
      ::= {objectClass 6}

organizationalPerson OBJECT-CLASS
      SUBCLASS OF person
      MAY CONTAIN {
            localeAttributeSet,
            organizationalUnitName,
            postalAttributeSet,
            telecommunicationAttributeSet,
            title}
      ::= {objectClass 7}
```

```
organizationalRole OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
                commonName}
        MAY CONTAIN {
                description,
                localeAttributeSet,
                organizationalUnitName,
                postalAttributeSet,
                preferredDeliveryMethod,
                roleOccupant,
                seeAlso,
                telecommunicationAttributeSet}
        ::= {objectClass 8}

groupOfNames OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
                commonName,
                member}
        MAY CONTAIN {
                description,
                organizationName,
                organizationalUnitName,
                owner,
                seeAlso,
                businessCategory}
        ::= {objectClass 9}

residentialPerson OBJECT-CLASS
        SUBCLASS OF person
        MUST CONTAIN {
                localityName}
        MAY CONTAIN {
                localeAttributeSet,
                postalAttributeSet,
                preferredDeliveryMethod,
                telecommunicationAttributeSet,
                businessCategory}
        ::= {objectClass 10}

applicationProcess OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
                commonName}
        MAY CONTAIN {
                description,
                localityName,
                organizationalUnitName,
                seeAlso}
        ::= {objectClass 11}

applicationEntity OBJECT-CLASS
        SUBCLASS OF top
        MUST CONTAIN {
                commonName,
                presentationAddress}
        MAY CONTAIN {
                description,
                localityName,
                organizationName,
                organizationalUnitName,
                seeAlso,
                supportedApplicationContext}
        ::= {objectClass 12}
```

```
dSA OBJECT-CLASS
    SUBCLASS OF applicationEntity
    MAY CONTAIN {
        knowledgeInformation}
    ::= {objectClass 13}

device OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        commonName}
    MAY CONTAIN {
        description,
        localityName,
        organizationName,
        organizationalUnitName,
        owner,
        seeAlso,
        serialNumber}
    ::= {objectClass 14}

strongAuthenticationUser OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        userCertificate}
    ::= {objectClass 15}

certificationAuthority OBJECT-CLASS
    SUBCLASS OF top
    MUST CONTAIN {
        cACertificate,
        certificateRevocationList,
        authorityRevocationList}
    MAY CONTAIN {
        crossCertificatePair}
    ::= {objectClass 16}

END
```

ANNEXE B

(à la Recommandation X.521)

Formes de noms suggérées et structures de DIT

Cette annexe ne fait pas partie de la Recommandation.

Cette annexe suggère certaines pratiques courantes de formulation de noms et structures de DIT qui peuvent être utilisées ou non par une Autorité Administrative. Les pratiques de formulation de noms et les définitions de structures de DIT pour une catégorie d'objets comprennent la spécification des attributs utilisés pour la formulation de noms et les catégories d'objets que l'entrée supérieure ou l'entrée subordonnée peut avoir dans le DIT. Toutes les entrées d'une catégorie d'objets doivent comprendre au moins les attributs utilisés pour la formulation de noms. Les utilisateurs de l'annuaire devraient être informés des formulations de noms suggérées afin de mieux prévoir les noms des objets avec lesquels ils communiquent. On trouvera dans les paragraphes qui suivent des règles applicables à la formulation de noms et à l'établissement de structures pour certaines catégories d'objets.

Les règles de structure sont illustrées à la figure B-1/X.521.

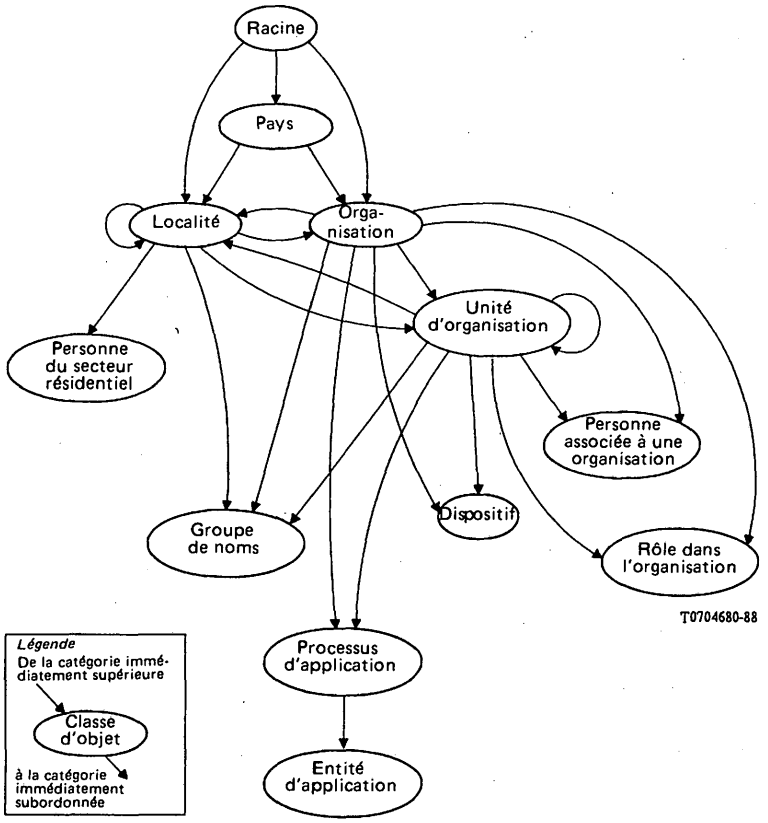


FIGURE B-1/X.521

Structure DIT suggérée

B.1 *Pays*

L'attribut **CountryName** est utilisé pour la formulation de noms.

La Racine se trouve immédiatement au-dessus des entrées de la catégorie d'objets **Country**.

B.2 *Organisation*

L'attribut **OrganizationName** est utilisé pour la formulation de noms.

La Racine, le **Pays** ou la **Localité** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **Organization**.

Remarque - Lorsque l'organisation est placée immédiatement au-dessous de la racine, cela désigne une organisation internationale. Les valeurs de l'attribut **OrganizationName** pour les organisations internationales doivent toutes être distinctes.

B.3 *Localité*

L'attribut **LocalityName** ou **State/ProvinceName** est utilisé pour la formulation de noms.

La Racine, le **Pays**, la **Localité**, l'**Organisation** ou l'**Unité d'Organisation** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **Locality**.

B.4 *Unité d'organisation*

L'attribut **OrganizationalUnitName** est utilisé pour la formulation de noms.

L'**Organisation**, l'**Unité d'Organisation** ou la **Localité** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **OrganizationalUnit**.

B.5 *Personne associée à une organisation*

Les attributs **CommonName** et, facultativement, **OrganizationalUnitName** sont utilisés pour la formulation de noms.

L'**Organisation** ou l'**Unité d'Organisation** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **OrganizationalPerson.**

Remarque - Il existe deux moyens permettant d'acquérir l'attribut **unité d'organisation** dans des noms: avoir un objet **unité d'organisation** supérieur ou avoir directement un tel attribut.

B.6 *Rôle dans l'organisation*

L'attribut **CommonName** est utilisé pour la formulation de noms.

L'**Organisation** ou l'**Unité d'Organisation** peuvent se trouver immédiatement au-dessus d'entrées de la catégorie d'objets **OrganizationalRole.**

Remarque - Il existe deux moyens permettant d'acquérir l'attribut **unité d'organisation** dans des noms: avoir un objet **unité d'organisation** supérieur ou avoir directement un tel attribut.

B.7 *Groupe de noms*

L'attribut **CommonName** est utilisé pour la formulation de noms.

La **Localité**, l'**Organisation** ou l'**Unité d'Organisation** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **GroupOfNames.**

Remarque - Il existe deux moyens permettant d'acquérir l'attribut **unité d'organisation** dans des noms: avoir un objet **unité d'organisation** supérieur ou avoir directement un tel attribut.

B.8 *Personne du secteur résidentiel*

Les attributs **CommonName** et, facultativement, **StreetAddress** sont utilisés pour la formulation de noms.

La **Localité** est immédiatement supérieure aux entrées de catégories d'objets **ResidentialPerson.**

B.9 *Entité d'application*

L'attribut **CommonName** est utilisé pour la formulation de noms. Le **CommonName** devrait contenir un qualificatif d'entité d'application (voir la Recommandation X.200).

Le **Processus d'Application** est immédiatement supérieur aux entrées de catégorie d'objets **ApplicationEntity.**

B.10 *Dispositif*

L'attribut **CommonName** est utilisé pour la formulation de noms.

L'**Organisation** ou l'**Unité d'Organisation** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **Device.**

Remarque - Il existe deux moyens permettant d'acquérir l'attribut **unité d'organisation** dans des noms: avoir un objet **unité d'organisation** supérieur ou avoir directement un tel attribut.

B.11 *Processus d'application*

L'attribut **CommonName** est utilisé pour la formulation de noms.

L'**Organisation** ou l'**Unité d'Organisation** peuvent se trouver immédiatement au-dessus des entrées de la catégorie d'objets **ApplicationProcess.**

Remarque 1 - On trouvera dans la Recommandation X.200 des précisions sur la manière de choisir un **Nom Commun** pour l'Entité d'Application.

Remarque 2 - Il existe deux moyens permettant d'acquérir l'attribut **unité d'organisation** dans des noms: avoir un objet **unité d'organisation** supérieur ou avoir directement un tel attribut.

ISBN 92-61-03732-1