UNIÓN INTERNACIONAL DE TELECOMUNICACIONES

UIT-T X.501

(02/2001)

SECTOR DE NORMALIZACIÓN
DE LAS TELECOMUNICACIONES
DE LA UIT

SERIE X: REDES DE DATOS Y COMUNICACIÓN ENTRE SISTEMAS ABIERTOS

Directorio

Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Modelos

Recomendación UIT-T X.501

RECOMENDACIONES UIT-T DE LA SERIE X

REDES DE DATOS Y COMUNICACIÓN ENTRE SISTEMAS ABIERTOS

REDES PÚBLICAS DE DATOS	
Servicios y facilidades	X.1–X.19
Interfaces	X.20–X.49
Transmisión, señalización y conmutación	X.50–X.89
Aspectos de redes	X.90–X.149
Mantenimiento	X.150–X.179
Disposiciones administrativas	X.180–X.199
INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Modelo y notación	X.200–X.209
Definiciones de los servicios	X.210–X.219
Especificaciones de los protocolos en modo conexión	X.220–X.229
Especificaciones de los protocolos en modo sin conexión	X.230–X.239
Formularios para declaraciones de conformidad de implementación de protocolo	X.240–X.259
Identificación de protocolos	X.260–X.269
Protocolos de seguridad	X.270–X.279
Objetos gestionados de capa	X.280–X.289
Pruebas de conformidad	X.290–X.299
INTERFUNCIONAMIENTO ENTRE REDES	
Generalidades	X.300–X.349
Sistemas de transmisión de datos por satélite	X.350–X.369
Redes basadas en el protocolo Internet	X.370–X.399
SISTEMAS DE TRATAMIENTO DE MENSAJES	X.400–X.499
DIRECTORIO	**X.500–X.599**
GESTIÓN DE REDES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS Y ASPECTOS DE SISTEMAS	
Gestión de redes	X.600–X.629
Eficacia	X.630–X.639
Calidad de servicio	X.640–X.649
Denominación, direccionamiento y registro	X.650–X.679
Notación de sintaxis abstracta uno	X.680–X.699
GESTIÓN DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Marco y arquitectura de la gestión de sistemas	X.700–X.709
Servicio y protocolo de comunicación de gestión	X.710–X.719
Estructura de la información de gestión	X.720–X.729
Funciones de gestión y funciones de arquitectura de gestión distribuida abierta	X.730–X.799
SEGURIDAD	X.800–X.849
APLICACIONES DE INTERCONEXIÓN DE SISTEMAS ABIERTOS	
Compromiso, concurrencia y recuperación	X.850–X.859
Procesamiento de transacciones	X.860–X.879
Operaciones a distancia	X.880–X.899
PROCESAMIENTO DISTRIBUIDO ABIERTO	X.900–X.999

Para más información, véase la Lista de Recomendaciones del UIT-T.

NORMA INTERNACIONAL ISO/CEI 9594-2
RECOMENDACIÓN UIT-T X.501

Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Modelos

Resumen

La presente Recomendación | Norma Internacional proporciona varios modelos diferentes para el directorio como un marco para las otras Recomendaciones UIT-T de la serie X.500. Los modelos son: el modelo (funcional) global, el modelo de autoridad administrativa, los modelos genéricos de información de directorio que proporcionan la visión de los usuarios del directorio y de los usuarios administrativos sobre la información de directorio, los modelos genéricos de agente de sistema de directorio (DSA) y de información de DSA, el marco operacional y un modelo de seguridad.

Orígenes

La Recomendación UIT-T X.501, revisada por la Comisión de Estudio 7 (2001-2004) del UIT-T, fue aprobada el 2 de febrero de 2001. Se publica también un texto idéntico como Norma Internacional ISO/CEI 9594-2.

Nota

Los implementadores y los usuarios deben saber que existe un proceso de resolución de defectos y que pueden introducirse correcciones en esta Recomendación | Norma Internacional en forma de corrigenda técnicos. Las mismas correcciones también prodrán introducirse en esta Recomendación en forma de Guía del implementador. La lista de corrigenda técnicos de esta Norma Internacional aprobados puede obtenerse en el sition web de la ISO, y los corrigenda técnicos publicados en las organizaciones nacionales de normalización. Los corrigenda técnicos y las Guías del implementador de esta Recomendación pueden obtenerse en el sition web del UIT-T.

PREFACIO

La UIT (Unión Internacional de Telecomunicaciones) es el organismo especializado de las Naciones Unidas en el campo de las telecomunicaciones. El UIT-T (Sector de Normalización de las Telecomunicaciones de la UIT) es un órgano permanente de la UIT. Este órgano estudia los aspectos técnicos, de explotación y tarifarios y publica Recomendaciones sobre los mismos, con miras a la normalización de las telecomunicaciones en el plano mundial.

La Asamblea Mundial de Normalización de las Telecomunicaciones (AMNT), que se celebra cada cuatro años, establece los temas que han de estudiar las Comisiones de Estudio del UIT-T, que a su vez producen Recomendaciones sobre dichos temas.

La aprobación de Recomendaciones por los Miembros del UIT-T es el objeto del procedimiento establecido en la Resolución 1 de la AMNT.

En ciertos sectores de la tecnología de la información que corresponden a la esfera de competencia del UIT-T, se preparan las normas necesarias en colaboración con la ISO y la CEI.

NOTA

En esta Recomendación, la expresión "Administración" se utiliza para designar, en forma abreviada, tanto una administración de telecomunicaciones como una empresa de explotación reconocida de telecomunicaciones.

PROPIEDAD INTELECTUAL

La UIT señala a la atención la posibilidad de que la utilización o aplicación de la presente Recomendación suponga el empleo de un derecho de propiedad intelectual reivindicado. La UIT no adopta ninguna posición en cuanto a la demostración, validez o aplicabilidad de los derechos de propiedad intelectual reivindicados, ya sea por los miembros de la UIT o por terceros ajenos al proceso de elaboración de Recomendaciones.

En la fecha de aprobación de la presente Recomendación, la UIT no ha recibido notificación de propiedad intelectual, protegida por patente, que puede ser necesaria para aplicar esta Recomendación. Sin embargo, debe señalarse a los usuarios que puede que esta información no se encuentre totalmente actualizada al respecto, por lo que se les insta encarecidamente a consultar la base de datos sobre patentes de la TSB.

ÍNDICE

Página

Introducción

Esta Recomendación | Norma Internacional, junto con otras Recomendaciones | Normas Internacionales, ha sido elaborada para facilitar la interconexión de los sistemas de procesamiento de información con el fin de proporcionar servicios de directorio. El conjunto de todos estos sistemas, junto con la información de directorio que contienen, puede considerarse como un todo integrado, denominado el *directorio*. La información contenida en el directorio, denominada colectivamente base de información de directorio (DIB, *directory information base*) se utiliza típicamente para facilitar la comunicación entre, con o sobre objetos tales como entidades de aplicación, personas, terminales y listas de distribución.

El directorio desempeña un papel importante en la interconexión de sistemas abiertos (OSI), cuyo objetivo es permitir, con un mínimo de acuerdos técnicos fuera de las propias normas de interconexión, la interconexión de sistemas de procesamiento de información:

- de diferentes fabricantes;
- sometidos a gestiones diferentes;
- de diferentes grados de complejidad; y
- de diferentes fechas de construcción.

Esta Recomendación | Norma Internacional proporciona varios modelos diferentes para el directorio como un marco para las otras Recomendaciones UIT-T de la serie X.500 | partes de ISO/CEI 9594. Los modelos son el modelo (funcional) global; el modelo de autoridad administrativa; modelos de información de directorio genéricos que proporcionan perspectivas de los usuarios de directorio y de los usuarios administrativos sobre información de directorio, modelos de agente de sistema de directorio (DSA) y de información de DSA genéricos, un marco operacional y un modelo de seguridad.

Los modelos de información de directorio genéricos describen, por ejemplo, la manera de agrupar informaciones sobre objetos para formar inserciones de esos objetos y cómo esas informaciones proporcionan nombres para objetos.

Los modelos de DSA y de información de DSA genéricos y el marco operacional sirven de soporte a la distribución de directorio.

Esta Recomendación | Norma Internacional proporciona una especialización de los modelos de información de directorio genéricos para soportar la administración de esquemas de directorio.

Esta cuarta edición revisa y mejora técnicamente la tercera edición de la presente Recomendación | Norma Internacional, pero no la sustituye. Las implementaciones pueden seguir alegando conformidad con la tercera edición. Sin embargo, en algún punto, no se soportará la tercera edición (es decir, los defectos informados ya no serán resueltos). Se recomienda que las implementaciones se conformen con esta cuarta edición lo antes posible.

Esta cuarta edición especifica las versiones 1 y 2 de los protocolos de directorio.

Las ediciones primera y segunda especifican solamente la versión 1. La mayor parte de los servicios y protocolos especificados en esta edición están diseñados para funcionar según la versión 1. No obstante, algunos servicios y protocolos mejorados, por ejemplo, los errores signados, no funcionarán a menos que todas las entidades de directorio que participan en la operación hayan negociado la versión 2. Cualquiera que sea la versión negociada, las diferencias entre los servicios y entre los protocolos definidos en las cuatro ediciones, salvo las asignadas específicamente a la versión 2, se acomodan utilizando las reglas de extensibilidad definidas en la edición de la Rec. UIT-T X.519 | ISO/CEI 9594-5.

El anexo A, que es parte integrante de esta Recomendación | Norma Internacional, resume la utilización de identificadores de objeto ASN.1 en las Recomendaciones UIT-T de la serie X.500 | partes de ISO/CEI 9594.

El anexo B, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el módulo ASN.1 que contiene todas las definiciones asociadas con el marco de información.

El anexo C, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el esquema de administración del subesquema en ASN.1.

El anexo D, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el módulo ASN.1 para la administración de servicio.

El anexo E, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el módulo ASN.1 para el control de acceso básico.

El anexo F, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el módulo ASN.1 que contiene todas las definiciones asociadas con los tipos de atributos operacionales de DSA.

El anexo G, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el módulo ASN.1 que contiene todas las definiciones asociadas con operaciones de gestión de vinculaciones operacionales.

El anexo H, que es parte integrante de esta Recomendación | Norma Internacional, proporciona el módulo ASN.1 que contiene todas las definiciones asociadas con la seguridad mejorada.

El anexo I, que no es parte integrante de esta Recomendación | Norma Internacional, resume la terminología matemática asociada con estructuras de árbol.

El anexo J, que no es parte integrante de esta Recomendación | Norma Internacional, describe algunos criterios que pueden considerarse para el diseño de nombres.

El anexo K, que no es parte integrante de esta Recomendación | Norma Internacional, presenta algunos ejemplos de distintos aspectos de esquemas.

El anexo L, que no es parte integrante de esta Recomendación | Norma Internacional, proporciona una descripción de la semántica asociada con permisos de control de acceso básico.

El anexo M, que no es parte integrante de esta Recomendación | Norma Internacional, proporciona un ejemplo ampliado de la utilización del control de acceso básico.

El anexo N, que no es parte integrante de esta Recomendación | Norma Internacional, describe algunas combinaciones de inserciones específicas de DSA.

El anexo O, que no es parte integrante de esta Recomendación | Norma Internacional, proporciona un marco para el modelado del conocimiento.

El anexo P, que no es parte integrante de esta Recomendación | Norma Internacional, describe los criterios según los cuales un nombre puede ser un nombre distinguido alternativo o el nombre distinguido primario, si puede contener valores alternativos o si puede incluir información de contexto.

El anexo Q, que no es parte integrante de esta Recomendación | Norma Internacional, describe el concepto de subfiltros.

El anexo R, que no es parte integrante de esta Recomendación | Norma Internacional, contiene recomendaciones y ejemplos de cómo pueden ser denominados los miembros de familia.

El anexo S, que no es parte integrante de esta Recomendación | Norma Internacional, enumera alfabéticamente los términos definidos en esta Recomendación | Norma Internacional.

El anexo T, que no es parte integrante de esta Recomendación | Norma Internacional, enumera las enmiendas e Informes de Defectos incorporados en esta edición de esta Recomendación | Norma Internacional.

NORMA INTERNACIONAL

RECOMENDACION UIT-T

Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Modelos

SECCIÓN 1 – GENERALIDADES

1 Alcance

Los modelos definidos en esta Recomendación | Norma Internacional proporcionan un marco conceptual y terminológico para las otras Recomendaciones UIT-T de la serie X.500 | partes de ISO/CEI 9594 que definen diversos aspectos de directorio.

Los modelos funcionales y de autoridad administrativa definen las maneras en que el directorio puede ser distribuido, tanto funcional como administrativamente. Se proporcionan también modelos de DSA y de información de DSA genéricos y un marco operacional para soportar la distribución de directorio.

Los modelos de información de directorio genéricos describen la estructura lógica de la DIB desde la perspectiva de los usuarios de directorio y de los usuarios administrativos. En estos modelos no es visible el hecho de que el directorio está distribuido y no centralizado.

Esta Recomendación | Norma Internacional proporciona una especialización de los modelos de información de directorio genéricos para soportar la administración de esquemas de directorio.

Las otras Recomendaciones de la serie X.500 | partes de ISO/CEI 9594 utilizan los conceptos definidos en esta Recomendación | Norma Internacional con el fin de definir especializaciones de modelos de DSA y de información de DSA genéricos para proporcionar modelos de DSA y de información de DSA y operacionales específicos que soportan capacidades particulares de directorio (por ejemplo, replicación):

a) el servicio proporcionado por el directorio se describe (en la Rec. UIT-T X.511 | ISO/CEI 9594-3) en términos de los conceptos del marco de información; esto permite que el servicio proporcionado en cierta medida independiente de la distribución física de la DIB;

b) la operación distribuida de directorio se especifica (en la Rec. UIT-T X.518 | ISO/CEI 9594-4) con el fin de proporcionar ese servicio, y por tanto mantener esa estructura de información lógica, puesto que la DIB es de hecho sumamente distribuida;

c) se especifican (en la Rec. UIT-T X.525 | ISO/CEI 9594-9) capacidades de replicación ofrecidas por las partes componentes de directorio para mejorar el funcionamiento global de éste).

El modelo de seguridad establece un marco para la especificación de mecanismos de control de acceso. Proporciona un mecanismo para identificar el método de control de acceso en vigor en una porción particular del DIT y define tres métodos de control de acceso flexibles y específicos, que son adecuados para una amplia variedad de aplicaciones y estilos de uso. El modelo de seguridad proporciona también un marco para la protección de la confidencialidad y la integridad de las operaciones de directorio que emplea mecanismos tales como la criptación y las firmas digitales. Se utiliza para ello el marco de autenticación definido en la Rec. UIT-T X.509 | ISO/CEI 9594-8 así como las herramientas de seguridad genérica de capas superiores definidas en la Rec. UIT-T X.830 | ISO/CEI 11586-1.

Los modelos de DSA establecen un marco para la especificación de la operación de los componentes de directorio. Específicamente:

a) el modelo funcional de directorio describe la manera en que el directorio se manifiesta como un conjunto de uno o más componentes, cada uno de los cuales es un DSA;

b) el modelo de distribución de directorio describe los criterios principales de acuerdo con los cuales las inserciones de la DIB y las copias de inserciones pueden ser distribuidas entre los DSA;

c) el modelo de información de DSA describe la estructura de la información de usuario de directorio y de la información operacional contenidas en un DSA;

d) el marco operacional de DSA describe los medios por los cuales se estructura la definición de formas específicas de cooperación entre los DSA para alcanzar determinados objetivos (por ejemplo, sombreado).

2 Referencias normativas

Las siguientes Recomendaciones y Normas Internacionales contienen disposiciones que, mediante su referencia en este texto, constituyen disposiciones de la presente Recomendación | Norma Internacional. Al efectuar esta publicación, estaban en vigor las ediciones indicadas. Todas las Recomendaciones y Normas son objeto de revisiones, por lo que se aconseja que los participantes en acuerdos basados en la presente Recomendación | Norma Internacional investiguen la posibilidad de aplicar las ediciones más recientes de las Recomendaciones y las Normas citadas a continuación. Los Miembros de la CEI y de la ISO mantienen registros de las Normas Internacionales actualmente vigentes. La Oficina de Normalización de las Telecomunicaciones de la UIT mantiene una lista de las Recomendaciones UIT-T actualmente vigentes.

2.1 Recomendaciones | Normas Internacionales idénticas

- Recomendación UIT-T X.200 (1994) | ISO/CEI 7498-1:1994, *Tecnología de la información – Interconexión de sistemas abiertos – Modelo de referencia básico: El modelo básico*.
- Recomendación UIT-T X.500 (2001) | ISO/CEI 9594-1:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Visión de conjunto de conceptos, modelos y servicios.*
- Recomendación UIT-T X.509 (2000) | ISO/CEI 9594-8:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Marcos para certificados de claves públicas y de atributos.*
- Recomendación UIT-T X.511 (2001) | ISO/CEI 9594-3:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Definición de servicio abstracto*.
- Recomendación UIT-T X.518 (2001) | ISO/CEI 9594-4:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Procedimientos para operación distribuida*.
- Recomendación UIT-T X.519 (2001) | ISO/CEI 9594-5:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Especificaciones de protocolo.*
- Recomendación UIT-T X.520 (2001) | ISO/CEI 9594-6:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Tipos de atributos seleccionados.*
- Recomendación UIT-T X.521 (2001) | ISO/CEI 9594-7:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Clases de objeto seleccionadas.*
- Recomendación UIT-T X.525 (2001) | ISO/CEI 9594-9:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Replicación.*
- Recomendación UIT-T X.530 (2001) | ISO/CEI 9594-10:2001, *Tecnología de la información – Interconexión de sistemas abiertos – El directorio: Utilización de la gestión de sistemas para la administración del directorio*.
- Recomendación CCITT X.660 (1992) | ISO/CEI 9834-1:1993, *Tecnología de la información – Interconexión de sistemas abiertos – Procedimientos para la operación de autoridades de registro para interconexión de sistemas abiertos: Procedimientos generales*.
- Recomendación UIT-T X.680 (1997) | ISO/CEI 8824-1:1998, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación de la notación básica.*
- Recomendación UIT-T X.681 (1997) | ISO/CEI 8824-2:1998, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación de objetos de información*.
- Recomendación UIT-T X.682 (1997) | ISO/CEI 8824-3:1998, *Tecnología de la información – Notación de sintaxis abstracta uno: Especificación de constricciones*.
- Recomendación UIT-T X.683 (1997) | ISO/CEI 8824-4:1998, *Tecnología de la información – Notación de sintaxis abstracta uno: Parametrización de especificaciones de notación de sintaxis abstracta uno.*
- Recomendación UIT-T X.803 (1994) | ISO/CEI 10745:1995, *Tecnología de la información – Interconexión de sistemas abiertos – Modelo de seguridad de capas superiores*.
- Recomendación UIT-T X.811 (1995) | ISO/CEI 10181-2:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de autenticación*.
- Recomendación UIT-T X.812 (1995) | ISO/CEI 10181-3:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad para sistemas abiertos: Marco de control de acceso*.

- Recomendación UIT-T X.813 (1996) | ISO/CEI 10181-4:1997, *Tecnología de la información – Interconexión de sistemas abiertos – Marcos de seguridad en sistemas abiertos: Marco de no rechazo*.
- Recomendación UIT-T X.830 (1995) | ISO/CEI 11586-1:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Seguridad genérica de las capas superiores: Sinopsis, modelo y notación.*
- Recomendación UIT-T X.833 (1995) | ISO/CEI 11586-4:1996, *Tecnología de la información – Interconexión de sistemas abiertos – Seguridad genérica de las capas superiores: Especificación de la sintaxis de transferencia de protección.*

2.2 Pares de Recomendaciones | Normas Internacionales de contenido técnico equivalente

- Recomendación CCITT X.800 (1991), *Arquitectura de seguridad para la interconexión de sistemas abiertos para aplicaciones del CCITT*.

  ISO 7498-2:1989, *Information processing systems – Open Systems Interconnection – Basic Reference Model – Part 2: Security Architecture*.

3 Definiciones

A los efectos de la presente Recomendación | Norma Internacional, se aplican las siguientes definiciones.

3.1 Definiciones del modelo de referencia de OSI

Los siguientes términos se definen en la Rec. UIT-T X.200 | ISO/CEI 7498-1:

a) *contexto de aplicación*;

b) *entidad de aplicación*;

c) *proceso de aplicación*.

3.2 Definiciones de directorio básico

Los siguientes términos se definen en la Rec. UIT-T X.500 | ISO/CEI 9594-1:

a) *directorio*;

b) *protocolo de acceso a directorio*;

c) *base de información de directorio*;

d) *protocolo de gestión de vinculación operacional de directorio*;

e) *protocolo de sistema de directorio*;

f) *usuario (de directorio).*

3.3 Definiciones de operaciones distribuidas

Los siguientes términos se definen en la Rec. UIT-T X.518 | ISO/CEI 9594-4:

a) *punto de acceso*;

b) *vinculación operacional jerárquica*;

c) *resolución de nombre*;

d) *vinculación operacional jerárquica no específica*;

e) *vinculación operacional jerárquica pertinente*.

3.4 Definiciones de replicación

Los siguientes términos se definen en la Rec. UIT-T X.525 | ISO/CEI 9594-9:

a) *copia velada;*

b) *referencia de consumidor;*

c) *copia de inserción;*

d) *DSA maestro;*

e) *sombreado primario;*

f) *zona replicada;*

g) *replicación;*

h) *sombreado secundario;*

i) *consumidor de sombra;*

j) *suministrador de sombra;*

k) *entrada específica de DSA sombreado;*

l) *sombreado;*

m) *referencia de suministrador*.

Las definiciones de los términos definidos en esta Recomendación | Norma Internacional figuran al principio de cada cláusula, según procede. Para facilitar la referencia, el anexo S contiene un índice alfabético de estos términos.

4 Abreviaturas

A los efectos de esta Recomendación | Norma Internacional se aplican las siguientes siglas.

ACDF	Función de decisión de control de acceso (*access control decision function*)
ACI	Información de control de acceso (*access control information*)
ACIA	Zona interior de control de acceso (*access control inner area*)
ACSA	Zona específica de control de accesos (*access control specific area*)
ADDMD	Dominio de gestión de directorio de Administración (*Administration directory management domain*)
ASN.1	Notación de sintaxis abstracta uno (*abstract syntax notation one*)
AVA	Aserción de valor de atributos (*attribute value assertion*)
BER	Reglas básicas de codificación (ASN.1) (*basic encoding rules*)
DACD	Dominio de control de acceso a directorio (*directory access control domain*)
DAP	Protocolo de acceso a directorio (*directory access protocol*)
DIB	Base de información de directorio (*directory information base*)
DISP	Protocolo de sombreado de información de directorio (*directory information shadowing protocol*)
DIT	Árbol de información de directorio (*directory information tree*)
DMD	Dominio de gestión de directorio (*directory management domain*)
DMO	Organización de gestión de dominio (*domain management organization*)
DOP	Protocolo de gestión de vinculaciones operacionales de directorio (*directory operational binding management protocol*)
DSA	Agente de sistema de directorio (*directory system agent*)
DSE	Inserción específica de DSA (*DSA-specific entry*)
DSP	Protocolo de sistema de directorio (*directory system protocol*)
DUA	Agente de usuario de directorio (*directory user agent*)
HOB	Vinculación operacional jerárquica (*hierarchical operational binding*)
NHOB	Vinculación operacional jerárquica no específica (*non-specific hierarchical operational binding*)
NSSR	Referencia de subordinado no específica (*non-specific subordinate reference*)
PRDMD	Dominio de gestión de directorio privado (*private directory management domain*)
RHOB	Vinculación operacional jerárquica pertinente (es decir HOB, o NHOB, según proceda) (*relevant hierarchical operational binding*)
RDN	Nombre distinguido relativo (*relative distinguished name*)
SDSE	DSE sombreada (*shadowed DSE*)

5 Convenios

Con pequeñas excepciones, esta Especificación de directorio se ha preparado con arreglo a las directrices "Presentación de textos comunes UIT-T | ISO/CEI" que figuran en la Guía para la cooperación entre el UIT-T y el JTC 1 de ISO/CEI, octubre de 1996.

El término "Especificación de directorio" (como en "esta Especificación de directorio") designa a la Rec. UIT-T X.501 | ISO/CEI 9594-2. El término "Especificaciones de directorio" designa a todas las Recomendaciones de la serie X.500 | partes de ISO/CEI 9594.

Esta Especificación de directorio utiliza el término "sistemas de la edición 1988" para hacer referencia a los sistemas conformes a la primera edición de las Especificaciones de directorio, es decir, la edición de 1988 de las Recomendaciones CCITT de la serie X.500 y la edición de ISO/CEI 9594:1990. Esta Especificación de directorio utiliza el término "sistemas de la edición 1993" para hacer referencia a los sistemas conformes a la segunda edición de las Especificaciones de directorio, es decir la edición de 1993 de las Recomendaciones UIT-T de la serie X.500 y la edición de ISO/CEI 9594:1995. Esta Especificación de directorio utiliza el término "sistemas de la edición 1997" para hacer referencia a los sistemas conformes a la tercera edición de las Especificaciones de directorio, es decir la edición de 1997 de las Recomendaciones UIT-T de la serie X.500 y la edición de ISO/CEI 9594:1998. Esta Especificación de directorio utiliza el término "sistemas de la cuarta edición" para hacer referencia a los sistemas conformes a esta cuarta edición (2001) de las Especificaciones de directorio, es decir las ediciones de 2001 de las Recomendaciones UIT-T X.500, X.501, X.511, X.518, X.519, X.520, X.521, X.525 y X.530 y las ediciones de 2000 de la Rec. UIT-T 509, y las partes 1-10 de ISO/CEI 9594:2001.

Esta Especificación de directorio presenta la notación ASN.1 con el tipo negrita Helvética. Cuando en el texto normal se hace referencia a tipos y valores ASN.1 éstos se distinguen del texto normal presentándolos en tipo negrita Helvética. Los nombres de los procedimientos referenciados típicamente cuando se especifica la semántica del procesamiento, se diferencian del texto normal representándolos en tipo negrita Times. Los permisos de control de acceso se presentan en tipo Times cursiva.

SECCIÓN 2 – VISIÓN DE CONJUNTO DE LOS MODELOS DE DIRECTORIO

6 Modelos de directorio

6.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

6.1.1 autoridad administrativa: Un agente de la organización de gestión de dominio que se ocupa de varios aspectos de administración de directorio. El término *autoridad administrativa* (en minúsculas) se refiere a las facultades conferidas a una Autoridad Administrativa por la organización de gestión de dominio para ejecutar la política.

6.1.2 dominio de gestión de directorio de Administración (ADDMD, *administration directory management domain*): Un DMD que es gestionado por una Administración.

NOTA – El término Administración denota una administración pública de telecomunicaciones u otra organización que ofrece servicios públicos de telecomunicaciones.

6.1.3 información administrativa y operacional de directorio: Información utilizada por el directorio para fines administrativos y operacionales.

6.1.4 dominio del árbol de información de directorio (DIT, *directory information tree*): La parte del DIT global contenida por los DSA que forman un DMD.

6.1.5 dominio de gestión de directorio (DMD, *directory management domain*): Un conjunto de uno o más DSA y ninguno o más DUA gestionados por una sola organización.

6.1.6 organización de gestión de dominios: Una organización que gestiona un DMD (y el dominio de DIT asociado).

6.1.7 información de usuario de directorio: Información de interés para usuarios y sus aplicaciones.

6.1.8 agente de sistema de directorio (DSA, *directory system agent*): Un proceso de aplicación OSI que forma parte del directorio.

6.1.9 usuario (de directorio): El usuario de extremo de directorio, es decir, la entidad o persona que accede al directorio.

6.1.10 agente de usuario de directorio (DUA, *directory user agent*): Un proceso de aplicación OSI que representa a un usuario al acceder al directorio.

NOTA – Los DUA pueden también proporcionar una gama de facilidades locales para ayudar a los usuarios a hacer preguntas e interpretar las respuestas.

6.1.11 dominio de gestión de directorio privado (PRDMD, *private directory management domain*): Un DMD gestionado por una organización que no es una Administración.

6.2 El directorio y sus usuarios

El *directorio* es un depósito de información. Este depósito se conoce como la base de información de directorio (DIB, *directory information base*). Los servicios de directorio proporcionados a los usuarios se relacionan con distintas clases de acceso a esta información.

Los servicios proporcionados por el directorio se definen en la Rec. UIT-T X.511 | ISO/CEI 9594-3.

Un usuario de directorio (por ejemplo, una persona o un proceso-aplicación) obtiene servicios de directorio accediendo al directorio. Dicho con mayor precisión, un *agente de usuario de directorio (DUA*, *directory user agent)*, accede en efecto al directorio e interactúa con él para obtener el servicio a nombre de un determinado usuario. El directorio proporciona uno o más *puntos de acceso* en los cuales puede efectuarse ese acceso. Estos conceptos se ilustran en la figura 1.

Un DUA se manifiesta como un proceso de aplicación. En cualquier ejemplar de comunicación, cada DUA representa precisamente a un usuario de directorio.

El directorio se presenta como un conjunto de uno o más procesos de aplicación denominados *agentes del sistema de directorio (DSA*, *directory system agents)*, cada uno de los cuales proporciona uno o más puntos de acceso. Para una descripción más detallada de los DSA, véase 21.2.

NOTA 1 – Algunos sistemas pueden proporcionar una función DUA centralizada que extrae información para los usuarios reales (procesos de aplicación, personas, etc.). Es transparente al directorio.

NOTA 2 – Las funciones de DUA y un DSA pueden estar dentro del mismo sistema abierto, y es una decisión de implementación el que uno o más DUA sean visibles dentro del entorno OSI como entidades de aplicación.

NOTA 3 – Un DUA puede presentar un comportamiento y una estructura locales que estén fuera del alcance de las Especificaciones de directorio; por ejemplo, un DUA que representa a un usuario humano de directorio puede proporcionar una gama de facilidades locales para ayudar a su usuario a hacer indagaciones e interpretar las respuestas.

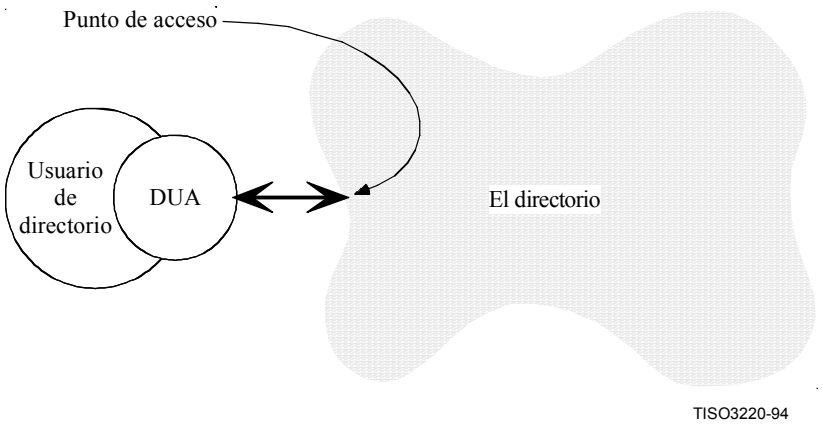


Figura 1 – Acceso a directorio

6.3 Modelos de directorio y de información de DSA

6.3.1 Modelos genéricos

La información de directorio puede clasificarse como:

- información de usuario, colocada en el directorio por los usuarios o en nombre de éstos y administrada a continuación por los usuarios, o en nombre de éstos. En la sección 3 figura un modelo de esta información; o
- información administrativa y operacional, mantenida por el directorio para satisfacer diversas necesidades administrativas y operacionales. En la sección 5 se proporciona un modelo de esta información, así como una especificación de la relación entre el usuario y los modelos de información administrativos y operacionales.

Estos modelos, que presentan visiones de la DIB desde perspectivas diferentes, se denominan modelos de información de directorio genéricos.

Los modelos de información de directorio describen cómo el directorio, en tanto que conjunto, presenta la información. La composición de directorio como un conjunto de DSA potencialmente cooperantes se abstrae a partir del modelo. Por otra parte, el modelo de información de DSA se relaciona especialmente con los DSA y la información que habrán de mantener estos DSA para que todos los DSA que comprenden el directorio puedan realizar juntos el modelo de información de directorio. El modelo de información de DSA figura en las cláusulas 22 a 23.

El modelo de información de DSA es un modelo genérico que describe la información mantenida por los DSA y la relación entre esta información y la DIB y el DIT.

Una parte de la información, pero no toda la información representada por el modelo de información de DSA, es accesible mediante el servicio abstracto de directorio. Por tanto, no es posible administrar toda la información descrita en estas Especificaciones de directorio mediante el servicio abstracto de directorio. Se prevé que la administración de la información de DSA será inicialmente un asunto local, pero que a la larga se empleará algún tipo de servicio de gestión de sistema genérico para proporcionar el acceso a toda la información descrita en el modelo de información del DSA.

6.3.2 Modelos de información específicos

Después del desarrollo de estos modelos genéricos para el directorio en su totalidad y para sus componentes, se requieren modelos de información específicos para la normalización de aspectos particulares del funcionamiento de directorio y sus componentes.

Los modelos de información de directorio establecen un marco para los siguientes modelos de información específicos:

- un modelo de información de control de acceso;
- un modelo de información de subesquema;
- un modelo de información de atributo colectivo.

El modelo de información de DSA genérico a su vez establece un marco para los siguientes modelos de información específicos:

- un modelo para un conocimiento de distribución del DSA;
- un modelo para un conocimiento de replicación del DSA.

6.4 Modelo de autoridad administrativa de directorio

Un conjunto de uno o más DSA y ninguno o más DUA gestionados por una sola organización forman un dominio de gestión de directorio (DMD, *directory management domain*).

La parte del DIT global mantenida por (los DSA que forman) un DMD se denomina un *dominio de DIT*. Existe una correspondencia de uno a uno entre los DMD y dominios de DIT. El término DMD se utiliza cuando se hace referencia a la gestión de los componentes funcionales de directorio. El término dominio de DIT se utiliza cuando se hace referencia a la gestión de información de directorio. Dos puntos importantes en relación con esta terminología son:

- Un dominio de DIT consta de uno o más subárboles disjuntos del DIT (véase 11.5). Un dominio de DIT no contendrá la raíz del DIT global.
- El término DMD puede utilizarse también como un término general cuando ambos aspectos de la gestión se consideran conjuntamente.

Una organización que gestiona un DMD (y el dominio DIT asociado) se conoce por una *organización de gestión de dominio (DMO, domain management organization)*.

NOTA 1 – Una DMO puede ser una Administración (es decir una administración pública de telecomunicaciones u otra organización que ofrece servicios públicos de telecomunicaciones) en cuyo caso se dice que el DMD gestionado es un DMD de Administración (ADDMD, *Administration directory management domain*); en los demás casos es un DMD privado (PRDMD, *private directory management domain*). Debe reconocerse que el aprovisionamiento de soporte para sistemas de directorio privados por miembros del UIT-T cae dentro de la reglamentación nacional. Así, las posibilidades técnicas descritas pueden ser ofrecidas o no por una Administración que proporciona servicios de directorio. La operación y configuración internas de los DMD privados no está dentro del alcance de las Especificaciones de directorio contempladas.

La figura 2 ilustra la relación entre una DMO, un DMD y un dominio de DIT.

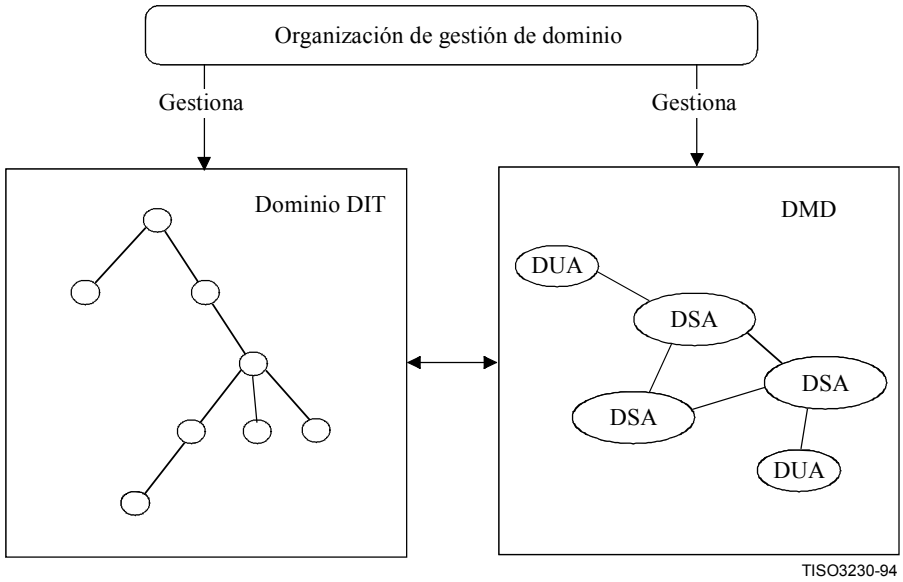


Figura 2 – Gestión de directorio

La gestión de un DUA por una DMO implica una responsabilidad continua, por parte de la DMO, de dar servicio a ese DUA, por ejemplo, el mantenimiento, o en algunos casos asumir la propiedad. La DMO puede optar o no por hacer uso de la Especificación de directorio para regir cualesquiera interacciones entre los DUA y DSA que estén completamente dentro del DMD.

Un agente de una DMO que se ocupa de diversos aspectos de administración de directorio se conoce por una *Autoridad Administrativa.* El término *autoridad administrativa* (en minúsculas) se refiere a las facultades conferidas a una Autoridad Administrativa por una DMO para aplicar políticas.

NOTA 2 – En la sección 4 se especifica un modelo de autoridad administrativa de directorio.

Se puede asignar un identificador de objeto a un DMD (un DMD-id) para facilitar la referencia, por ejemplo, en las reglas de búsqueda.

SECCIÓN 3 – MODELO DE INFORMACIÓN DE USUARIO DE DIRECTORIO

7 Base de información de directorio

7.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

7.1.1 asiento de alias: Una inserción de la clase "alias" que contiene información utilizada para proporcionar un nombre alternativo para un objeto.

7.1.2 antepasado: La inserción en la raíz de la jerarquía de miembros de una familia que comprende una inserción compuesta.

7.1.3 inserción compuesta: La representación de un objeto en términos de miembros de una familia que están organizados jerárquicamente en una o más familias de inserciones.

7.1.4 inserción derivada: Información de inserción en un resultado de búsqueda que contiene valores de atributo obtenidos uniendo datos procedentes de más de una inserción de directorio.

7.1.5 superclase directa: Con relación a una subclase – una clase de objeto de la cual se deriva directamente la subclase.

7.1.6 base de información de directorio (DIB, *directory information base*): El conjunto completo de información al que el directorio proporciona acceso, y que incluye todas las piezas de información que pueden ser leídas o manipuladas utilizando las operaciones de directorio.

7.1.7 árbol de información de directorio (DIT, *directory information tree*): La DIB considerada como un árbol, cuyos vértices (que no son la raíz) son las inserciones de directorio.

NOTA – El término DIT se utiliza en lugar de DIB solamente en contextos donde la estructura de árbol de la información es pertinente.

7.1.8 inserción (de directorio): Una colección de información denominada dentro de la DIB. La DIB se compone de inserciones.

7.1.9 familia: Un subconjunto jerárquico de inserciones de miembros de una familia que representa una clase particular de información dentro de una inserción compuesta. La raíz de cada familia dentro de una inserción compuesta es el antepasado, pero aparte del antepasado compartido, las familias no tienen miembros comunes. Una familia se distingue de las demás familias de una inserción compuesta por tener una clase común (clase de objeto estructural) para cada miembro de la familia que es subordinado inmediato del antepasado.

7.1.10 miembro de familia: Un miembro de una colección jerárquica de inserciones que comprende una inserción compuesta.

7.1.11 superior inmediato (sustantivo): Con relación a una inserción u objeto particular (deberá estar claro en el contexto a que se refiere), la inserción u objeto inmediatamente superior.

7.1.12 inserción inmediatamente superior: Con relación a una inserción particular, una inserción que está en el vértice inicial de un arco en el DIT cuyo vértice final es el de la inserción en cuestión.

7.1.13 objeto inmediatamente superior: Con relación a un objeto particular – un objeto cuya inserción de *objeto* es el superior inmediato de *cualquier* (o cualesquiera) de las inserciones (de objeto o de alias) para el segundo objeto.

7.1.14 objeto (de interés): Cualquier cosa en algún 'mundo', generalmente el mundo de las telecomunicaciones y del procesamiento de información o alguna parte de él, que es identificable (puede ser denominada) y que ofrece interés para que la información del mismo este contenida en la DIB.

7.1.15 clase de objeto: Una familia identificada de objetos (u objetos concebibles) que comparten ciertas características.

7.1.16 inserción de objeto: Una inserción que es la colección primaria de la información en la DIB sobre un objeto y de la cual puede por tanto decirse, que representa a ese objeto en la DIB.

7.1.17 inserciones conexas: Un conjunto de inserciones (de directorio) cada una de las cuales se puede identificar como que contiene información en la DIB sobre un determinado objeto de interés del mundo real. Inserciones distintas en el conjunto pueden contener diferentes tipos de información acerca del objeto del mundo real y pueden incluso contener información contradictoria.

NOTA 1 – El valor de la información en el conjunto de inserciones conexas depende de la fiabilidad de la identificación de cada inserción con el mundo real.

NOTA 2 – Es posible, aunque no necesario, que las inserciones conexas existan en visiones disjuntas del directorio y tengan nombres distinguidos idénticos. De manera similar, es posible que inserciones no conexas tengan nombres distinguidos idénticos; no obstante, se recomienda que sólo se usen nombres distinguidos idénticos para inserciones conexas.

7.1.18 **subclase**: Con relación a una o más superclases, una clase de objeto derivada de una o más superclases. Los miembros de la subclase comparten todas las características de las superclases y características adicionales no poseídas por los miembros de esas superclases.

7.1.19 **subordinado**: El inverso de superior.

7.1.20 **superclase**: Con respecto a una subclase – una superclase directa, o superclase de una clase de objeto que es una superclase directa (recursivamente).

7.1.21 **superior**: (Aplicado a objeto o inserción) inmediatamente superior, o superior, a uno que es inmediatamente superior (recursivamente).

7.2 Objetos

La finalidad de directorio es contener información sobre *objetos de interés (objetos)* que existen en algún 'mundo', y proporcionar acceso a éstos. Un objeto puede ser cualquier cosa en ese mundo que sea identificable (que pueda ser denominada).

NOTA 1 – El 'mundo' es generalmente el mundo de las telecomunicaciones y del procesamiento de información, o alguna parte de él.

NOTA 2 – Los objetos conocidos para el directorio pueden no corresponder exactamente con el conjunto de cosas 'reales' en el mundo. Por ejemplo, una persona del mundo real puede ser considerada como dos objetos diferentes, una persona que tiene un negocio y una persona residente, en lo que concierne al directorio. La correspondencia no se define en esta Especificación de directorio, sino que es un asunto que corresponde a los usuarios o proveedores de directorio en el contexto de sus aplicaciones.

Una *clase de objeto* es una familia identificada de objetos u objetos concebibles, que comparten ciertas características. Cada objeto pertenece por lo menos a una clase. Una clase de objeto puede ser una *subclase* de otras clases de objeto, en cuyo caso los miembros de la primera clase, la subclase, se consideran también miembros de las segundas, la superclase. Puede haber subclases de subclases, etc., hasta una profundidad arbitraria.

7.3 Inserciones de directorio

La DIB se compone de *inserciones (de directorio*). Una inserción es un conjunto de información denominado.

Hay cuatro clases de inserciones:

- *Inserciones de objeto*: Que representan el conjunto primario de información en la DIB sobre un objeto particular. Para cualquier clase de objeto particular hay precisamente una inserción de objeto o inserción compuesta (véase 8.10). Se dice que la inserción de objeto representa al objeto. Una inserción de objeto es una inserción simple o una inserción compuesta que comprende una agrupación de inserciones que juntas representan conjuntamente el objeto.
- *Inserciones de alias*: Utilizadas para proporcionar nombres alternativos para inserciones de objetos (posiblemente el antepasado de una inserción compuesta, pero no miembros de familia vástagos).
- *Subinserciones*: Que representan un conjunto de información en la DIB utilizado para satisfacer necesidades administrativas y operacionales de directorio. Las subinserciones se examinan en la sección 5.
- *Miembros de familia*: Inserciones especiales que son componentes de una inserción compuesta. El antepasado de una inserción compuesta es también un miembro de la familia.

La visión del usuario de la estructura de las inserciones de directorio se muestra en la figura 3 y se describe en 8.2.

Cada inserción contiene una indicación de las clases de objetos y sus superclases a las cuales pertenece la inserción.

Algunas inserciones de objeto se designan especialmente para los fines de la administración de directorio. Estas inserciones se denominan inserciones administrativas. El usuario de directorio normalmente no está al corriente de esto y considera

7.4 Árbol de información de directorio (DIT)

Para satisfacer las exigencias de la distribución y gestión de una DIB muy grande, y asegurar que las inserciones puedan ser inequívocamente denominadas y rápidamente encontradas, no es probable que sea factible una estructura plana. En consecuencia, la relación jerárquica comúnmente encontrada entre objetos (por ejemplo, una persona trabaja para un departamento que pertenece a una organización, que tiene su sede en un país) puede aprovecharse organizando las inserciones en forma de árbol, lo que se conoce por el *árbol de información de directorio (DIT)*.

NOTA – En el anexo I figura una introducción a los conceptos y la terminología de las estructuras de árbol.

Las partes componentes del DIT tienen las siguientes interpretaciones:

a) los vértices son las inserciones. Las inserciones de objetos pueden ser vértices constitutivos o no constitutivos de hoja (vértice hoja, vértice no-hoja) en tanto que las inserciones de alias son siempre vértices hoja. La raíz, en sí, no es una inserción, pero, cuando sea conveniente [por ejemplo en las definiciones b) y c) más abajo], puede ser considerada como una inserción de objeto nula [véase d) más abajo];

b) los arcos definen la relación entre vértices (y en consecuencia entre inserciones). Un arco del vértice A al vértice B entraña que la inserción en A sea la *inserción inmediatamente superior (superior inmediata)* de la inserción en B y a la inversa, que la inserción en B sea una *inserción inmediatamente subordinada (subordinada inmediata)* de la inserción en A. Las *inserciones superiores* (los *superiores*) de una inserción particular son su superior inmediato junto con sus superiores (recursivamente). Las *inserciones subordinadas* (las *subordinadas*) de una inserción particular son sus subordinadas inmediatas junto con sus subordinadas (recursivamente);

c) el objeto representado por una inserción es la autoridad de denominación o está estrechamente asociada con ella, (véase la cláusula 8) para sus subordinados;

d) la raíz representa el nivel más alto de autoridad de denominación para la DIB.

Una relación superior/subordinado entre objetos puede derivarse a partir de la relación entre inserciones. Un objeto es un *objeto inmediatamente superior* (un *superior inmediato*) de otro objeto solamente si la inserción de objeto para el primer objeto es el superior inmediato de cualesquiera de las inserciones para el segundo objeto. Los términos *objeto inmediatamente subordinado*, *subordinado inmediato*, *superior* y *subordinado* (aplicados a objetos) tienen significados análogos.

Las relaciones superior/subordinado permitidas entre objetos se rigen por las definiciones de estructura del DIT (véase 13.7).

Además de la información concerniente a las inserciones de directorio, éste mantiene información adicional relativa a colecciones de inserciones de directorio. Esas colecciones pueden ser *subárboles* (del DIT) o *refinamientos de subárbol* (cuando no constituyen una verdadera estructura de árbol). Véase la cláusula 12.

8 Inserciones de directorio

8.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

8.1.1 atributo: Información de un tipo particular. Las inserciones se componen de atributos.

8.1.2 atributo de usuario: Un atributo que representa información de usuario.

8.1.3 jerarquía de atributos: El aspecto de un atributo que permite derivar un tipo de atributo de usuario a partir de un tipo de atributo de usuario más genérico. La relación de las dos definiciones de tipos de atributos (que impone cierto comportamiento de los atributos correspondiente a estos tipos de atributo es, por tanto, jerárquica.

8.1.4 subtipo de atributo (subtipo): Un tipo de atributo A se relaciona con otro tipo de atributo B por el hecho de que, o bien A ha sido derivado de B, en cuyo caso A es un subtipo *directo* de B, o A ha sido derivado de un tipo de atributo que es un subtipo de B, en cuyo caso A es un subtipo *indirecto* de B.

8.1.5 supertipo de atributo (supertipo): Un tipo de atributo B está relacionado con otro tipo de atributo A por el hecho de que, o bien A ha sido derivado de B, en cuyo caso B es el supertipo *directo* de A, o A ha sido derivado de un tipo de atributo que es un subtipo de B, en cuyo caso B es un supertipo *indirecto* de A.

8.1.6 tipo de atributo: El componente de un atributo que indica la clase de información dada por ese atributo.

8.1.7 valor de atributo: Un caso particular de la clase de información indicada por un tipo de atributo.

8.1.8 aserción de valor de atributo: Una proposición, que puede ser verdadera, falsa, o indefinida, de acuerdo con las reglas de concordancia especificadas para el tipo, concerniente a la presencia, en una inserción, de un valor de atributo de un tipo de atributo particular.

8.1.9 clase de objeto auxiliar: Una clase de objeto que es descriptiva de inserciones o clases de inserciones y que no se utiliza para la especificación estructural del DIT.

8.1.10 atributo colectivo: Un atributo de usuario cuyos valores son los mismos para cada miembro de una colección de inserciones.

8.1.11 **contexto**: Propiedad que puede asociarse a un valor atributo de usuario para especificar la información que puede utilizarse para determinar la aplicabilidad del valor.

8.1.12 **aserción de contexto**: Proposición, que puede ser verdadera o falsa, relativa a un tipo de contexto y a valores de contexto particulares de ese tipo y que determina la aplicabilidad de un valor de atributo.

8.1.13 **tipo de contexto**: Componente de un contexto que indica su tipo o su finalidad.

8.1.14 **lista de contextos**: Conjunto de contextos asociados a un valor de atributo.

8.1.15 **valor de contexto**: Caso particular de la propiedad indicada por un tipo de contexto.

8.1.16 **atributo derivado**: Un atributo cuyo valor o valores se calculan en todo o en parte en vez de almacenarlos directamente.

8.1.17 **valor de clase de objeto derivada**: El valor de una clase objeto cuya presencia no es administrada por un usuario sino que es calculada. Los valores de clases de objeto derivadas se clasifican como abstractos.

8.1.18 **referencia de atributo directa**: Referencia (en el servicio abstracto de directorio y del DSA) a uno o más valores de atributo mediante el empleo del identificador de su tipo de atributo.

8.1.19 **valor distinguido**: Valor de atributo en una inserción que puede aparecer en el nombre distinguido relativo de la inserción.

8.1.20 **colección de inserciones**: Una colección de inserciones que pertenecen a un subárbol o refinamiento de subárbol especificadas explícitamente del DIT.

8.1.21 **referencia de atributo indirecta**: Referencia (en el servicio abstracto de directorio y del DSA) a uno o más valores de atributo mediante el empleo del identificador de un supertipo de su tipo de atributo.

8.1.22 **regla de concordancia**: Una regla que forma parte del esquema de directorio y que permite seleccionar inserciones formulando un enunciado particular (una aserción de regla de concordancia) concerniente a sus valores de atributo.

8.1.23 **aserción de regla de concordancia**: Una proposición, que puede ser verdadera, falsa o indefinida, concerniente a la presencia, en una inserción, de valores de atributo que satisfacen los criterios definidos por la regla de concordancia.

8.1.24 **atributo operacional**: Un atributo que representa información operacional y/o administrativa.

8.1.25 **clase de objeto estructural**: Una clase de objeto utilizada para la especificación estructural del DIT.

8.1.26 **clase objeto estructural de una inserción**: Con respecto a una inserción particular, la *única* clase de objeto estructural utilizada para determinar la regla de contenido del DIT y la regla de estructura del DIT aplicables a la inserción. Esta clase de objeto es indicada por el atributo operacional **structuralObjectClass** (**clase de objeto estructural**). Ésta es la clase de objeto más subordinada de la cadena de superclase de clase de objeto estructural de la inserción.

8.2 Estructura global

Como se muestra en la figura 3, una inserción consiste en un conjunto de *atributos*.

Cada atributo proporciona una pieza de información sobre el objeto a que corresponde la inserción o describe una característica particular de éste.

NOTA 1 – Ejemplos de atributos que pudieran estar presentes en una inserción son información de denominación, como el nombre personal del objeto, e información de direccionamiento, como su número de teléfono.

Un atributo consta de un *tipo de atributo*, que identifica la clase de información proporcionada por un atributo y los *valores de atributo* correspondientes, que son casos particulares de esa clase que aparecen en la inserción. Un valor de atributo de usuario puede tener cero, uno o más contextos asociados a él en su lista de contexto. Los valores atributos operacionales carecerán de contextos.

NOTA 2 – Los tipos de atributo, valores de atributo y contextos se describen, respectivamente, en 8.4, 8.5 y 8.7. En la cláusula 12 se describen los atributos operacionales.

```
Attribute ::= SEQUENCE {
    type                ATTRIBUTE.&id ({ SupportedAttributes }),
    values              SET SIZE (0..MAX) OF ATTRIBUTE.&TYPE ({SupportedAttributes}{@type}),
    valuesWithContext   SET SIZE (1..MAX) OF SEQUENCE {
        value               ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
        contextList         SET SIZE (1..MAX) OF Context } OPTIONAL }
```

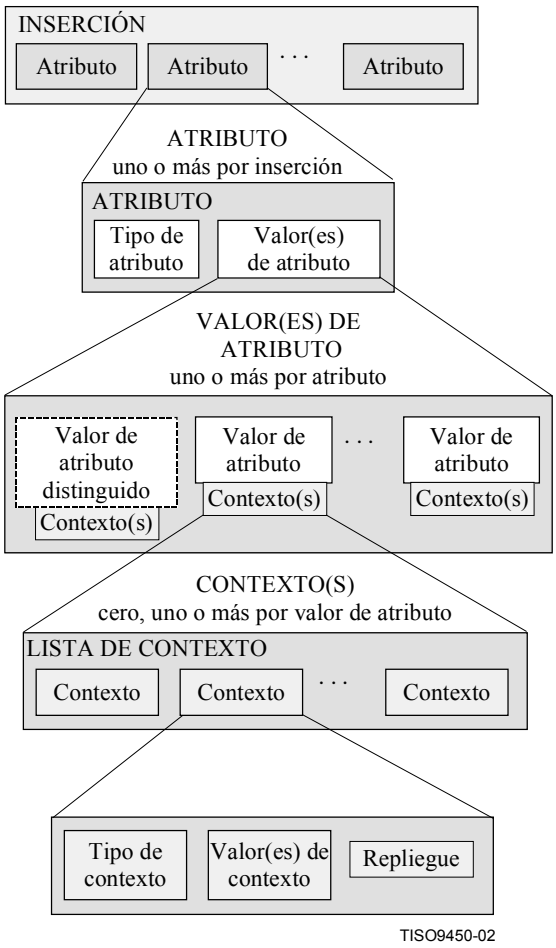


Figura 3 – Estructura de una inserción

Todo atributo puede diseñarse como univaluado o multivaluado. El directorio deberá asegurar que los atributos univaluados solo tienen un valor. Este valor puede tener una lista de contexto para asociar propiedades al valor de atributo. Los atributos almacenados tendrán al menos un valor pero a veces puede parecer que carecen de valores cuando se transfieren hacia o desde la memoria (por ejemplo, porque el control de acceso oculta los valores).

8.3 Clases de objeto

Las clases de objeto se utilizan en el directorio con varios fines:

- describir y clasificar objetos y las inserciones que corresponden a estos objetos;
- cuando proceda, controlar el funcionamiento de directorio;
- regular, en conjunción con especificaciones de reglas de estructuras del DIT de directorio, la posición de inserciones en el DIT;
- regular, junto con especificaciones de reglas de contenido del DIT, los atributos que están contenidos en inserciones;
- identificar clases de inserciones que van a ser asociadas con una política particular por una autoridad administrativa apropiada.

Algunas clases de objeto serán normalizadas internacionalmente. Otras serán definidas por autoridades administrativas nacionales y/u organizaciones privadas. Esto implica que cierto número de autoridades distintas serán responsables de definir clases de objetos e identificarlos inequívocamente. Esto se logra identificando cada clase de objeto mediante un identificador de objeto cuando se define la clase de objeto. En 13.3.3 se proporciona una notación para este fin.

NOTA – Una autoridad administrativa puede utilizar clases de objeto diferentes de las clases de objeto útiles, definidas y registradas en la Especificación de directorio sobre el directorio. Una autoridad administrativa puede por sí misma especificar y registrar clases de objeto, por ejemplo para complementar las definidas en las Especificaciones de directorio.

Una clase de objeto (una *subclase*) puede ser derivada de una clase de objeto (su *superclase* directa) que a su vez ha sido derivada de una clase de objeto más genérica. Para clases de objeto estructurales y alias, este proceso se detiene en la clase de objeto más genérica, **top (tope)**. Un conjunto ordenado de superclases hasta la clase de objeto más alta de una clase de objeto es su *cadena de superclase*.

Una clase de objeto puede ser derivada de dos o más superclases directas (superclases que no forman parte de la misma cadena de superclase). Esta característica de la formación de subclases se denomina *herencia múltiple*.

La especificación de una clase de objeto de inserción o de miembro de familia identifica si un atributo es obligatorio o facultativo; esta especificación se aplica también a sus subclases. Se puede decir que la subclase *hereda* la especificación de atributos obligatorios y facultativos de su superclase. La especificación de una subclase puede indicar que un atributo facultativo de la superclase es obligatorio en la subclase.

Hay tres modalidades de clases de objeto:

- clases de objeto abstractas;
- clases de objeto estructurales; y
- clases de objeto auxiliares.

Cada clase de objeto es exactamente de una de estas modalidades, y sigue siendo de esta modalidad en cualquier situación en que se encuentre en el directorio. La definición de cada clase de objeto especificará la modalidad de objeto de que se trata.

Todas las inserciones serán un miembro de la clase de objeto **top** y por lo menos de otra clase de objeto.

8.3.1 Clases de objeto abstractas

Una clase de objeto abstracta se utiliza principalmente para derivar otras clases de objeto que proporcionan las características comunes de esta clase de objeto. Una inserción no pertenecerá solamente a clases de objeto abstractas.

top es una clase de objeto abstracta utilizada como una superclase de todas las clases de objetos estructurales.

Además de utilizarlo para derivar otras clases de objeto, un valor de clase de objeto abstracta puede ser un valor derivado; es decir, su presencia es calculada o deducida por el directorio. Por ejemplo, el valor de la clase de objeto **parent (progenitor)** para una inserción particular se calcula o deduce a partir de la presencia de un miembro de la familia, de la clase de objeto auxiliar **child (vástago)**, subordinado inmediato de la inserción.

8.3.2 Clases de objeto estructurales

Una clase de objeto definida para uso en la especificación estructural del DIT se llama una *clase de objeto estructural*. Se utilizan clases de objeto estructurales en la definición de la estructura de los nombres de los objetos para inserciones conformes.

Una inserción de objeto o de alias se caracteriza exactamente por una cadena de superclases de clase de objeto estructural que tiene una sola clase de objeto estructural como la clase de objeto más subordinada. Esta clase de objeto estructural se denomina clase de *objeto estructural de la inserción*.

Las clases de objeto estructurales están relacionadas con inserciones asociadas:

- una inserción conforme a una clase de objeto estructural tiene que representar un objeto del mundo real descrito por la clase de objeto;
- las reglas de estructura del DIT sólo hacen referencia a clases de objeto estructurales; la clase de objeto estructural de una inserción se utiliza para especificar la posición de la inserción en el DIT;
- la clase de objeto estructural de una inserción se utiliza, junto con una regla de contenido del DIT asociada, para controlar el contenido de una inserción.

No se cambiará la clase de objeto estructural de una inserción.

8.3.3 Clases de objeto auxiliares

Las aplicaciones específicas que emplean el directorio con frecuencia encuentran útil especificar una *clase de objeto auxiliar* que puede utilizarse en la construcción de inserciones de varios tipos. Por ejemplo, los sistemas de tratamientos de mensajes utilizan la clase auxiliar usuario de MHS (véase la Rec. UIT-T X.402 | ISO/CEI 10021-2) para especificar un lote de atributos de tratamiento de mensaje obligatorios y facultativos para tipos de inserciones cuya clase de objeto estructural es variable, por ejemplo, persona de una organización o persona residencial.

En ciertos entornos es necesario poder añadir o suprimir en la lista de atributos permitidos de una inserción, una determinada clase o clases, quizá normalizadas.

Este requisito puede satisfacerse mediante la definición y el uso de una clase de objeto auxiliar que tenga una semántica conocida y mantenida dentro de una comunidad local, la cual cambia con el transcurso del tiempo según se vaya necesitando.

Este requisito puede satisfacerse también utilizando las facilidades de las definiciones de la regla de contenido del DIT para permitir dinámicamente (es decir, sin registro) la adición o exclusión de atributos de inserciones en puntos determinados del DIT (véase 13.3.3.)

Las clases de objeto auxiliares describen inserciones o clases de inserciones.

Por tanto, además de ser un miembro de la clase de objeto estructural, una inserción puede ser facultativamente un miembro de una o más clases de objetos auxiliares.

Las clases de objeto auxiliares de una inserción pueden cambiar en el tiempo.

NOTA – La facilidad de clase de objeto no registrada, disponible en la edición de 1988 de estas Especificaciones de directorio para soportar los requisitos examinados en esta cláusula, se desaconseja ahora en favor de la utilización de las reglas de contenido del DIT.

8.3.4 Definición de clase de objeto y la edición 1988 de esta Especificación de directorio

Las clases de objeto definidas utilizando la terminología de la edición 1988 de esta Especificación de directorio no serán clasificadas en clases de objeto estructurales, auxiliares o abstractas.

Las clases de objeto alias especificadas utilizando la terminología de la edición 1988 de esta Especificación de directorio pueden considerarse especificadas como clases de objeto abstractas, auxiliares o estructurales y desplegadas consiguientemente en un subesquema.

8.4 Tipos de atributo

Algunos tipos de atributo se normalizarán internacionalmente. Otros tipos de atributo serán definidos por autoridades administrativas nacionales y organizaciones privadas. Esto entraña que varias autoridades distintas serán responsables de definir tipos e identificarlos inequívocamente. Esto se realiza identificando cada tipo de atributo con un identificador de objeto cuando se define el tipo. Utilizando la notación de la clase de objeto de información **ATTRIBUTE (ATRIBUTO)** definida en 13.4.7, se define un tipo de atributo como:

```
AttributeType ::= ATTRIBUTE.&id
```

Todos los atributos en una inserción serán tipos de atributo distintos.

Determinados atributos no pueden ser almacenados ni ser accesibles en inserciones, sino que su finalidad es ser transportados en operaciones para llevar información, por ejemplo, información de diagnóstico, que pueda ser expresada convenientemente como atributos. Otros atributos, llamados *atributos de control*, pueden especificar en su definición un procedimiento especial que se ha de ejecutar en base a la información del atributo. Un atributo de control puede ser especificado en una operación, colocado en inserciones, etc. Véase un ejemplo en 6.5.3 de la Rec. UIT-T X.520 | ISO/CEI 9594-6.

Hay varios tipos de atributo que el directorio conoce y utiliza para sus propios fines, entre los que cabe citar:

a) **objectClass (clase de objeto)** – Un atributo de este tipo aparece en cada inserción, e indica las clases y superclases de objeto a las cuales pertenece el objeto.

b) **aliasedEntryName (nombre de inserción con alias)** – Un atributo de este tipo aparece en cada inserción de alias, y mantiene el nombre de la inserción (véase 8.5) al que hace referencia la inserción de alias.

Estos atributos se definen en 13.4.7.

Los tipos de atributos de usuario que aparecerán o pueden aparecer dentro de una inserción de objeto o alias están regidos por reglas aplicables a las clases de objeto indicadas así como por la regla de contenido del DIT para esa inserción (véase 13.8). Los tipos de atributo que pueden aparecer en una subinserción están regidos por las reglas del esquema de sistema.

Algunas inserciones de directorio pueden contener atributos especiales que normalmente no son visibles al usuario de directorio. Estos atributos se llaman atributos operacionales y se utilizan para satisfacer las necesidades administrativas y operacionales de directorio. En la sección 5 se examinan más detalladamente los atributos operacionales.

8.5 Valores de atributo

La definición de un atributo conlleva también la especificación de la sintaxis y, por tanto, el tipo de datos, a la cual se conformará cada valor de estos atributos. Mediante la notación de la clase de objeto de información **ATTRIBUTE** definido en 13.4.7, un valor de atributo se define como:

```
AttributeValue ::= ATTRIBUTE.&Type
```

Un valor de atributo puede ser designado como un *valor distinguido*, en cuyo caso el valor de atributo puede formar parte del nombre distinguido relativo de la inserción (véase 9.3). Es posible que haya múltiples valores distinguidos diferenciados por su contexto, como se describe en 9.3.

8.6 Jerarquías de tipos de atributo

Cuando se define un tipo de atributo, las características de un tipo de atributo más genérico pueden facultativamente utilizarse como base de la definición. El nuevo tipo de atributo es un *subtipo directo* del tipo de atributo más genérico, el *supertipo*, del cual se deriva.

Las jerarquías de atributos permiten el acceso a la DIB con diversos grados de granularidad. Esto se consigue permitiendo que los componentes de valores de atributos sean accedidos utilizando su identificador de tipo de atributo específico (una referencia directa al atributo), o el identificador de un identificador de tipo de atributo más genérico (una referencia indirecta).

Los atributos relacionados semánticamente pueden ser colocados en una relación jerárquica, situando el más especializado como subordinado del más generalizado. La búsqueda o extracción de atributos y sus valores se facilita citando el tipo de atributo más generalizado, un elemento de filtro así especificado se evalúa para los tipos más especializados así como para el tipo citado; se aplica también al tipo más especializado una aserción de contexto especificada para el tipo de atributo más generalizado.

Cuando se seleccionan tipos especializados subordinados para devolverlos como parte de un resultado de búsqueda, estos tipos se devolverán si están disponibles. Cuando se seleccionan tipos más generales para retornarlos como parte de un resultado de búsqueda, se devolverán tanto los tipos generales como los especializados, si están disponibles. Un valor de atributo se devolverá siempre como un valor de su propio tipo de atributo.

Para que una inserción contenga un valor de un tipo de atributo perteneciente a una jerarquía de atributos, dicho tipo se incluirá explícitamente en la definición de una clase de objeto a la que pertenece la inserción, o porque la regla de contenido DIT aplicable a esa inserción lo permite.

Todos los tipos de atributo en una jerarquía de atributos se tratan como tipos distintos y no relacionados, a los efectos de la administración de la inserción, y para la modificación, por el usuario, del contenido de la inserción.

Un valor de atributo almacenado en una inserción de objeto o alias de directorio es exactamente de un tipo de atributo. El tipo es indicado cuando el valor se añade por primera vez a la inserción.

8.7 Contextos

Puede perfeccionarse el modelo de información asociando a los valores de atributo unas propiedades denominadas contextos. Con cualquier valor de atributo de usuario puede asociarse una lista de contextos que proporciona una información adicional que puede utilizarse para determinar la aplicabilidad del valor del atributo.

NOTA 1 – Por ejemplo, es posible utilizar los contextos para asociar un idioma, tiempo o ubicación determinados con un valor de atributo.

Cada contexto consta de un campo de tipo, un valor de campo cuya sintaxis es función del tipo y una bandera de **fallback**. Utilizando la notación de la clase de objeto de información **CONTEXT** definida en 13.9, se define un contexto como sigue:

```
Context ::= SEQUENCE {
    contextType     CONTEXT.&id ({SupportedContexts}),
    contextValues   SET SIZE (1..MAX) OF CONTEXT.&Type ({SupportedContexts}{@contextType}),
    fallback        BOOLEAN DEFAULT FALSE }
```

El **contextType** es un **OBJECT IDENTIFIER** y se especifica utilizando la clase de objeto de información **CONTEXT** definida en 13.9. Especifica la propiedad concreta representada por el contexto.

El **contextValues** es el conjunto de uno o más valores de la propiedad especificados por **contextType** que están asociados con el valor de atributo particular.

Se utiliza **fallback** para designar uno o más valores de atributo para un comportamiento específico en relación con un tipo de contexto. Además de tener asociados **contextValues** específicos de ese tipo de contexto, un valor de atributo para el cual el repliegue es **TRUE** para un **contextType** determinado:

– Se considera asociado con cualquier valor del **contextType** dado para el cual no hay asociados de otra forma otros valores del mismo atributo. Por lo tanto, una aserción de contexto de este tipo de contexto que no concuerda con ninguno de los valores del atributo basados en las reglas de concordancia de **contextValues**, concordará con cualquier valor de atributo para el que **fallback** es **TRUE** para este tipo de contexto.

  NOTA 2 – Por ejemplo, la tentativa de seleccionar un valor de atributo asociado con un idioma concreto producirá valores con **fallback** fijado a **TRUE**, si ninguno de los valores de atributo está asociado de otra manera con el lenguaje escogido.

– Se considera como un valor a preservar en una operación que reinicia valores de atributo para un tipo de atributo determinado. Una modificación (valor de reiniciación) suprime todos los valores de un tipo de atributo elegido que tienen un contexto asociado para el cual el **fallback** está fijado a **FALSE**.

  NOTA 3 – La modificación (valor de reiniciación) se describe ulteriormente en 11.3.2 de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

Un valor de atributo sin contextos o cuya lista de contexto no contenga un contexto de un tipo especificado, se considerará aplicable para todos los valores de contexto de ese tipo especificado.

NOTA 4 – Por ejemplo, una selección basada en el valor de contexto francés de un contexto de lenguaje seleccionará un valor de atributo que no tiene ningún contexto de lenguaje asociado específicamente con él (así como aquellos valores de atributo que tengan el contexto de lenguaje francés asociado específicamente a ellos).

Todos los contextos de una lista de contexto de valores de atributo serán de tipos de contexto distintos.

La información de contexto asociada con valores de atributo puede recuperarse junto con los valores de atributo (por ejemplo para poder distinguir entre esos valores de atributo). Un usuario de directorio puede también utilizar los contextos para perfeccionar la selección y recuperación de información en operaciones de directorio.

8.8 Reglas de concordancia

8.8.1 Visión de conjunto

Es importantísimo para el directorio poder seleccionar un conjunto de inserciones de la DIB sobre la base de aserciones concernientes a valores de atributo contenidos por estas inserciones.

Una regla de concordancia permite seleccionar inserciones mediante una aserción particular concerniente a sus valores de atributo.

El tipo más primitivo de aserción es la aserción de valor de atributo. Pueden soportarse aserciones más complejas utilizando aserciones de reglas de concordancia. Una aserción de regla de concordancia es una proposición, que puede ser verdadera, falsa o indefinida, concerniente a la presencia, en una inserción, de valores de atributo que satisfacen los criterios definidos por la regla de concordancia.

Una aserción de atributo o de regla de concordancia se evalúa basándose en la regla de concordancia asociada con la aserción.

Una regla de concordancia se define mediante la especificación de:

– la gama de las sintaxis de atributo soportadas por la regla;

– los tipos específicos de concordancia soportados por la regla;

– la sintaxis requerida para expresar una aserción de cada tipo específico de concordancia;

– reglas para derivar un valor de la sintaxis de aserción a partir de un valor de la sintaxis de atributo, si es necesario.

  NOTA – No se imponen restricciones a las reglas de concordancia que pueden ser definidas para soportar una aplicación determinada. Sin embargo, las reglas definidas para soportar una aplicación determinada no pueden ser soportados de una manera general por los DUA y los DSA. Donde sea posible, debe preferirse la utilización de las reglas de concordancia definidas en la Rec. UIT-T X.520 | ISO/CEI 9594-6 a la especificación de reglas nuevas.

Algunas veces habrá una correspondencia de uno a uno entre una regla de concordancia y los tipos de concordancias soportados. Por ejemplo, el servicio abstracto de directorio soporta una regla de concordancia de presencia para detectar la presencia de un atributo en una inserción.

Otras veces habrá una correspondencia de muchos a muchos entre una regla y los tipos de concordancia soportados. Por ejemplo, el servicio abstracto de directorio soporta una regla de ordenación genérica que permite los tipos, de concordancia mayor o igual que y menor o igual que.

8.8.2 Aserciones de valor de atributo

Una aserción de valor de atributos (AVA, *attribute value assertion*) es una proposición que puede ser verdadera, falsa o indefinida, de acuerdo con las reglas de concordancia especificadas para el tipo, concernientes a la presencia en una inserción de un valor de atributo de un tipo determinado. Entraña un tipo de atributo, una aserción de valor de atributo y opcionalmente una aserción acerca de los contextos asociados con el valor de atributo:

```
AttributeValueAssertion ::= SEQUENCE {
      type                  ATTRIBUTE.&id ({SupportedAttributes}),
      assertion             ATTRIBUTE.&equality-match.&AssertionType ({SupportedAttributes}{@type}),
      assertedContexts      CHOICE {
            allContexts             [0]   NULL,
            selectedContexts        [1]   SET SIZE (1..MAX) OF ContextAssertion } OPTIONAL }

ContextAssertion ::= SEQUENCE {
      contextType           CONTEXT.&id({SupportedContexts}),
      contextValues         SET SIZE (1..MAX) OF
                  CONTEXT.&Assertion ({SupportedContexts}{@contextType})}
```

La sintaxis del componente **assertion (aserción)** de una AVA es determinada por la regla de concordancia de igualdad definida para el tipo de atributo, y puede ser diferente de la sintaxis del propio atributo.

8.8.2.1 Evaluación de una AVA

Una AVA es:

a) indefinida, si se cumple cualquiera de las condiciones siguientes:

   1) el tipo de atributo es desconocido;

   2) el tipo de atributo no tiene regla de concordancia de igualdad;

   3) el valor no se conforma al tipo de datos indicado por la sintaxis de la aserción de la regla de concordancia de igualdad del atributo;

   NOTA – 2) y 3) normalmente indican una AVA defectuosa; 1) sin embargo, puede ocurrir como una situación local (por ejemplo, un DSA determinado no ha sido configurado con soporte para ese tipo de atributo particular).

b) verdadera, si la inserción contiene un atributo de ese tipo y el atributo contiene un valor de ese valor, y el valor contiene un contexto que concuerda con los **assertedContexts** descritos en 8.8.2.2;

c) falsa, en los demás casos.

8.8.2.2 Utilización de assertedContexts o valores por defecto de aserción de contexto

La inclusión de **assertedContexts** dentro de un **AttributeValueAssertion** es facultativa. Si se especifica **assertedContexts**, deberá evaluarse la **assertion** únicamente para aquellos valores del atributo para los cuales **assertedContexts** es verdadero, como se define en 8.8.2.3.

Si dentro de **AttributeValueAssertion** no se proporciona **assertedContexts**, deberá aplicarse una aserción de contexto por defecto de la misma forma; es decir, deberá evaluarse **assertion** únicamente para aquellos valores del atributo para los cuales, como se define en 8.8.2.3, la aserción de contexto por defecto es verdadera. Hay tres fuentes potenciales para la aserción de contexto por defecto: la especificada para la operación como un todo, la disponible dentro de las subinserciones del DIT y la disponible localmente en el DSA. Se aplican como sigue:

1) Si no se proporciona **assertedContexts** dentro de **AttributeValueAssertion**, se aplicará cualquier aserción de contexto para el tipo de atributo dado que se haya suministrado para la operación como un todo, como parte de **operationContexts**, según se describe en 7.3 de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

2) Si el usuario no ha proporcionado **assertedContexts** para la AVA y no hay aserción de contexto para el tipo de atributo dado proporcionado para la operación como un todo, se aplicará la aserción de contexto por defecto para el tipo de atributo dado en las subinserciones de aserción de contexto (si existen) que controlan la inserción, como se describe en 14.7.

3) Si, según los pasos 1) y 2) anteriores, no hay aserción de contexto, el DSA puede aplicar una aserción de contexto por defecto definida localmente para el tipo de atributo dado. Tal valor por defecto reflejará típicamente parámetros locales tales como el lenguaje o la ubicación del lugar de despliegue del DSA o la hora actual del día, aunque el DSA puede diseñarlos de forma diferente para cada DUA a la que responda.

4) Si no hay ninguna aserción de contexto disponible desde alguna de estas fuentes, se evaluará la **assertion** para todos los valores del atributo.

8.8.2.3 Evaluación de assertedContexts

assertedContexts es verdadero si:

a) se especifica **allContexts** (esto permite que una aserción de contexto sustituya cualquier aserción de contexto por defecto que se hubiera aplicado si se hubiese omitido **assertedContexts** de **AttributeValueAssertion**); o

b) cada **ContextAssertion** de **selectedContexts** es verdadera como se describe en 8.8.2.4.

En cualquier otro caso **assertedContexts** es falso.

8.8.2.4 Evaluación de ContextAssertion

Una **ContextAssertion** es verdadera, para un valor de atributo particular, si:

a) el valor de atributo tiene un contexto del mismo **contextType** de la **ContextAssertion** y cualquiera de los **contextValues** almacenados de ese contexto concuerda con cualquiera de los **contextValues** de aserción, según la definición de concordancia determinada para ese **contextType**; o

b) el valor de atributo no contiene contextos del **contextType** de aserción; o

c) ninguno de los demás valores de atributo de ese atributo satisface la **ContextAssertion** según los pasos 1) ó 2) en 8.8.2.2, pero el valor de atributo contiene un contexto del **contextType** de aserción con el **fallback** fijado a **TRUE**.

En cualquier otro caso una **ContextAssertion** es falsa.

8.8.3 Aserciones de tipos de atributo

Una aserción de tipo de atributo es una proposición que puede ser verdadera, falsa o indefinida, de acuerdo con los contextos asociados.

```
AttributeTypeAssertion  ::=  SEQUENCE {
      type                       ATTRIBUTE.&id ({SupportedAttributes}),
      assertedContexts           SEQUENCE SIZE (1..MAX) OF ContextAssertion  OPTIONAL }
```

8.8.3.1 Evaluación de una aserción de tipo de atributo

Una aserción de tipo de atributo es:

a) indefinida, si el tipo de atributo es desconocido o si el atributo no está presente en la inserción;

b) VERDADERA, si la inserción contiene un atributo de ese tipo, y el atributo contiene uno o más valores que contienen un contexto que concuerda con los **assertedContexts** como se describe en 8.8.3.2;

c) FALSA, en los demás casos.

8.8.3.2 Utilización de assertedContexts o aserciones de contexto por defecto

La inclusión de **assertedContexts** dentro de una **AttributeTypeAssertion** es opcional. Si se especifica **assertedContexts**, **assertedContexts** será verdadero al menos para un valor de atributo de acuerdo con las reglas definidas en 8.8.2.4.

Si no se proporciona **assertedContext** dentro de una **AttributeTypeAssertion**, se puede aplicar una aserción de contexto por defecto de la misma manera; es decir, la aserción de contexto por defecto será verdadera al menos para un valor de atributo de acuerdo con las reglas definidas en 8.8.2.4. Los orígenes potenciales de una aserción de contexto por defecto se especifican en 8.8.2.2.

8.8.4 Aserciones de reglas de concordancia incorporadas

Varias categorías de reglas de concordancia conexas, cuyas semánticas son generalmente comprendidas y aplicables a valores de muchos tipos diferentes de atributos, son comprendidas por el directorio:

– presente;

– igualdad;

- subcadenas;
- ordenación;
- concordancia aproximada.

La sintaxis para sostener ciertos tipos de concordancias asociadas con estas categorías de reglas de concordancia se ha incorporado en el servicio abstracto de directorio:

- una sintaxis **present (presente)** para la regla presente;
- una sintaxis **equality (igualdad)** para reglas de igualdad;
- sintaxis **greaterOrEqual (mayor o igual)** y **lessOrEqual (menor o igual)** para las reglas de ordenación;
- sintaxis **initial (inicial)**, **any (cualquiera)** y **final (final)** para las reglas de subcadenas;
- una sintaxis **approximateMatch (concordancia aproximada)** para las reglas de concordancia aproximada.

La sintaxis **present** puede ser utilizada para cualquier atributo de cualquier tipo. La concordancia presente prueba la presencia de cualquier valor de un tipo determinado.

Las reglas de concordancia de igualdad, subcadenas y ordenación específicas pueden ser asociadas con un tipo de atributo cuando éste se define. Estas reglas específicas se utilizan cuando se evalúan aserciones de las reglas de igualdad, ordenación y subcadenas utilizando la sintaxis incorporada en el servicio abstracto de directorio. Si no se proporcionan reglas específicas, las aserciones concernientes a estos atributos son indefinidas.

La sintaxis **approximateMatch** soporta una regla de concordancia aproximada cuya definición es un asunto local del DSA.

8.8.5 Requisitos de las reglas de concordancia

Para que el sistema de directorio se comporte de una manera coherente y bien definida, es necesario que se impongan ciertas restricciones a las reglas de concordancia que se utilizarán junto con la sintaxis que ha sido incorporada en el servicio abstracto de directorio.

Para una regla de concordancia de igualdad en la cual la sintaxis de la aserción es diferente de la sintaxis de atributo a la cual se aplica la regla de concordancia, se suministrarán reglas para derivar un valor de la sintaxis de la aserción a partir de un valor de la sintaxis de atributo.

Las reglas de concordancia de igualdad para los atributos utilizados para denominación serán transitivas, conmutativas y tendrán una sintaxis de aserción idénticas a la sintaxis de atributo.

Una regla de concordancia transitiva se caracteriza por el hecho de que si un valor **a** concuerda con en valor **b**; y si ese valor **b** concuerda con un tercer valor **c**, entonces el valor **a** concuerda con el valor **c** cuando se utiliza esta regla.

Una regla de concordancia conmutativa se caracteriza por el hecho de que si un valor **a** concuerda con un valor **b**, el valor **b** concuerda con el valor **a**. El atributo **presentationAddress (dirección de presentación)** es un ejemplo de un atributo que soporta una sintaxis de atributo cuya regla de concordancia no es conmutativa.

Con respecto a un tipo de atributo específico, las reglas de igualdad y ordenación (si ambas están presentes) siempre estarán relacionadas por lo menos con respecto a lo siguiente: dos valores son iguales utilizando la relación de igualdad solamente si son iguales utilizando la relación de ordenación. Además, la relación de ordenación estará bien establecida; es decir, para todas las x, y, y z para las cuales x precede a y e y precede a z de acuerdo con la relación, x precede a z.

NOTA – Estos requisitos implican que, cuando se define una relación de ordenación, se define también una relación de igualdad.

Con respecto a un tipo de atributo específico, las reglas de igualdad y de subcadenas (si ambas están presentes) siempre estarán relacionadas por lo menos con respecto a lo siguiente: para todas las x e y que concuerdan según la relación de igualdad se dará que, para todos los valores z de la relación subcadena, el resultado de evaluar la aserción con respecto al valor x será igual al resultado de evaluar la aserción con respecto al valor y. Es decir, dos valores que son indistinguibles cuando se utiliza la relación de igualdad también serán indistinguibles cuando se utiliza la relación de subcadenas.

8.8.6 Reglas de concordancia de igualdad de identificador de objeto y nombre distinguido

Hay varias reglas de concordancia de igualdad utilizadas para evaluar aserciones de valores de atributos que el directorio conoce y utiliza para sus propios fines, y que comprenden:

- **objectIdentifierMatch**: Esta regla se utiliza para concordar atributos con la sintaxis de **ObjectIdentifier**.
- **distinguishedNameMatch**: Esta regla se utiliza para concordar atributos con la sintaxis de **DistinguishedName**.

8.9 Colecciones de inserciones

8.9.1 Visión general

Una colección de aserciones de objeto y de alias pueden tener ciertas características comunes (por ejemplo, ciertos atributos que tienen el mismo valor para cada inserción de la colección) debido a alguna característica común o relación compartida de los objetos correspondientes. Esta agrupación de inserciones se denomina una colección de inserciones.

Las colecciones de inserciones pueden contener inserciones de objeto y de alias que están relacionadas por su posición en el DIT. Estas colecciones se especifican como subárboles o refinamientos de subárbol como se describe en la sección 5.

Una inserción puede pertenecer a varias colecciones de inserciones, a reserva de las limitaciones administrativas impuestas en la sección 5.

8.9.2 Atributos colectivos

Cuando unos atributos de usuario son compartidos por las inserciones de una colección de inserciones se conocen por *atributos colectivos*.

Se permite también que el mismo atributo colectivo esté asociado independientemente con dos o más de estas colecciones. En tales casos, el atributo colectivo de la inserción tiene múltiples valores. En consecuencia los atributos colectivos deberán ser siempre multievaluados.

Los atributos colectivos, aunque son percibidos por los usuarios de operaciones de interrogación de directorio como atributos de inserción, se tratan diferentemente de los atributos de inserción en el modelo de información de directorio. Esta diferencia se manifiesta a los usuarios de operaciones de modificación de directorio por el hecho de que los atributos colectivos no pueden ser administrados (esto es, modificados) por las inserciones en que aparecen, sino que serán administrados por sus inserciones asociadas.

NOTA – Las fuentes independientes de estos valores no son percibidas por los usuarios de las operaciones de interrogación de directorio.

Para que un atributo colectivo aparezca en una inserción, es necesario que la presencia de ese tipo de atributo se permita de acuerdo con la regla de contenido del DIT que gobierna la inserción.

Las inserciones pueden excluir específicamente un determinado atributo colectivo. Esto se obtiene mediante la utilización del atributo **collectiveExclusions (exclusiones colectivas)**, descrito en 12.7 y definido en 14.6.

8.10 Inserciones compuestas y familias de inserciones

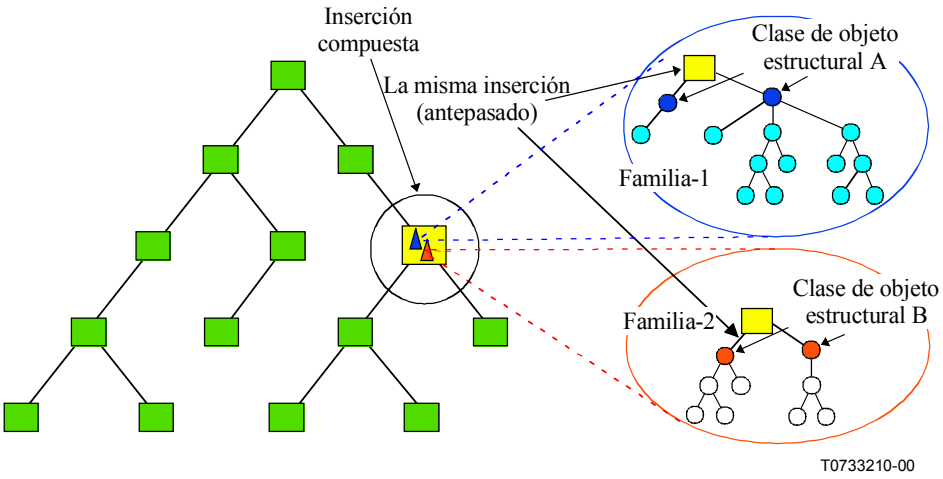


Figura 4 – Familias de inserciones

Una inserción compuesta es una inserción especial que contiene inserciones de miembros de familia subordinados con información organizada jerárquicamente sobre el objeto representado por la inserción compuesta. La inserción compuesta se representa en el DIT mediante un miembro de familia antepasado, que se halla en el tope de un árbol que contiene otros miembros de la familia.

Los propios miembros de la familia se pueden organizar en una o más familias a efectos de filtrado y recuperación de la información. Cada familia es un subárbol; familias distintas no tienen miembros de familia comunes aparte de la raíz compartida que es el antepasado. Una familia comprende así un antepasado más un conjunto de miembros de la familia subordinados.

La familia está compuesta, además de por el antepasado, por todos los miembros de la familia subordinados inmediatamente que son de la misma clase de objeto estructural. Sus miembros subordinados, si hay alguno, forman parte también de la misma familia con independencia de sus clases de objeto estructurales.

Estos conceptos se ilustran en la figura 4.

Un miembro de familia que es un vástago dentro de un árbol de familia se marca con la clase de objeto auxiliar **child**. La presencia del valor de la clase de objeto **child** para una inserción hace que se marque de manera automática la inserción inmediatamente superior con el valor de la clase de objeto abstracta **parent**. Una inserción que sea tanto **parent** como **child** dentro de un árbol de familia se marca con ambos valores de clase de objeto. El antepasado es el único miembro de la familia que no es de la clase de objeto **child**. La construcción de inserciones compuestas se lleva a cabo marcando inserciones con valores de la clase de objeto **child**.

Cada subordinado de un miembro de familia no antepasado será, él mismo, un miembro de la familia, y deberá marcarse con un valor de la clase de objeto **child**.

En 13.3.3 figura la definición ASN.1 de estas clases de objeto.

Todos los miembros de la familia de una inserción compuesta deberán situarse en el mismo contexto de denominación que el antepasado. No se permite que los miembros de la familia sean inserciones de alias. Un alias no deberá apuntar a un miembro de familia vástago.

9 Nombres

9.1 Definiciones

A los efectos de la presente Especificación de directorio, se aplican las siguientes definiciones.

9.1.1 alias, nombre de alias: Nombre alternativo de un objeto proporcionado por el uso de inserciones de alias.

9.1.2 desreferenciación (de alias): Proceso de convertir un nombre de alias de un objeto en su nombre distinguido.

9.1.3 nombre distinguido (de una inserción): Cada inserción de objeto, inserción de alias y subinserción tiene al menos un nombre distinguido. Si cualquier RDN de la inserción o de cualquier inserción superior incluye un atributo para el que existen múltiples nombres distinguidos diferenciados por su contexto (como se describe en 9.3), la inserción tendrá múltiples nombres distinguidos diferenciados por su contexto. El *nombre distinguido primario* es el nombre distinguido en el que cada RDN tiene el valor distinguido primario de cada atributo contribuyente como valor principal de la construcción RDN.

9.1.4 nombre (de directorio): Un constructivo que distingue un objeto determinado de los demás objetos. Un nombre debe ser inequívoco (es decir, debe designar un solo objeto), aunque no tiene que ser único (es decir, el único nombre que designa inequívocamente al objeto).

9.1.5 nombre (de una inserción): Un constructivo que distingue una inserción determinada de todas las demás inserciones.

9.1.6 nombre de miembro local: El nombre de un miembro de la familia formado por la secuencia de nombres distinguidos relativos (RDN) desde el antepasado hasta el miembro en cuestión sin incluir el RDN del antepasado.

9.1.7 autoridad denominadora: Una autoridad responsable de la atribución de nombres en alguna región del DIT.

9.1.8 nombre contemplado: Un constructivo que es sintácticamente un nombre, pero que no se ha demostrado (todavía) que es un nombre válido.

9.1.9 nombre distinguido relativo (RDN, *relative distinguished name*): Un conjunto de uno o más pares de tipo y valor de atributo, cada uno de los cuales concuerda con un valor de atributo distinguido distinto de la inserción.

9.2 Nombres en general

Un *nombre* (*de directorio*) es un constructivo que identifica un objeto determinado entre el conjunto de todos los objetos. Un nombre deberá ser inequívoco, es decir, designar un solo objeto. Sin embargo, un nombre no tiene que ser único, es decir, ser el único nombre que designa inequívocamente al objeto. Un nombre (de directorio) identifica también una inserción. Esta inserción es una inserción de objeto que representa al objeto o una inserción de alias que contiene información que ayuda al directorio a localizar la inserción que representa el objeto.

NOTA 1 – El conjunto de nombres de un objeto comprende por tanto el conjunto de nombres de alias para el objeto, junto con los nombres distinguidos del objeto.

Se puede asignar un nombre distinguido a un objeto sin que esté representado por una inserción en el directorio, pero este nombre es el nombre que su inserción de objeto tendría si estuviese representada en el directorio.

Desde el punto de vista sintáctico, cada nombre de un objeto o inserción constituye una secuencia ordenada de nombres distinguidos relativos (véase 9.3).

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

RDNSequence  ::=  SEQUENCE OF  RelativeDistinguishedName

DistinguishedName  ::=  RDNSequence
```

NOTA 2 – En una posible futura extensión, podrán tenerse en cuenta nombres formados de una manera diferente de la aquí descrita.

Cada subsecuencia inicial del nombre de un objeto es también el nombre de un objeto. La secuencia de objetos así identificados, que comienza por la raíz y termina por el objeto que está siendo denominado, es tal que cada uno es el inmediato superior del que le sigue en la secuencia.

Un *nombre contemplado* es un constructivo que, sintácticamente, es un nombre, pero que (todavía) no se ha demostrado que sea un nombre válido.

9.3 Nombres distinguidos relativos

Cada objeto e inserción tiene al menos un *nombre distinguido relativo* (*RDN, relative distinguished name*). El RDN de una inserción de objeto o de alias consta de un conjunto de pares de tipo y valor de atributo (con una lista de contexto facultativa), cada uno de los cuales concuerda, utilizando la regla de concordancia y la regla de concordancia de contexto aplicables, con un valor de atributo distinguido distinto de la inserción.

Todo atributo que contribuya a un RDN puede tener más de un valor distinguido, diferenciado por el contexto, como se describe seguidamente. Esto proporciona RDN alternativos para el mismo objeto. Dentro de un conjunto de atributos de valores distinguidos (diferenciados por el contexto) se designa, precisamente, uno de ellos como valor distinguido primario. El *nombre distinguido relativo primario* de un objeto comprende el conjunto de valores distinguidos primarios del conjunto de atributos que constituyen el RDN. Cada atributo de un RDN, cuando se transporta en un protocolo, indica el valor distinguido primario (si está presente) y, opcionalmente, puede incluir un contexto para el valor y valores de atributo alternativos adicionales con contexto. En este caso cada valor de atributo con su contexto concuerda con un valor de atributo distinguido diferente de la inserción para el tipo de atributo, de acuerdo con la regla de concordancia de igualdad aplicable y las reglas de concordancia de contexto.

NOTA 1 – Se puede utilizar la regla de concordancia de igualdad porque los atributos de denominación, la sintaxis del atributo y la sintaxis de aserción de la regla de concordancia igualdad son iguales. Análogamente, para contextos que pueden utilizarse para diferenciar valores distinguidos en un atributo de denominación, la sintaxis del contexto y la sintaxis de aserción del contexto son las mismas.

Los RDN de todas las inserciones que tienen un superior inmediato determinado son diferentes, con independencia de cualquier lista de contexto asociada. Corresponde a la autoridad denominadora pertinente de una inserción asegurar que esto es así, asignando de manera apropiada valores de atributos distinguidos. La atribución de los RDN se considera una tarea administrativa que puede o no requerir alguna negociación entre las organizaciones o administraciones afectadas. Esta Especificación de directorio no proporciona ese mecanismo de negociación ni efectúa ninguna hipótesis sobre su realización.

```
RelativeDistinguishedName  ::=  SET SIZE (1..MAX) OF AttributeTypeAndDistinguishedValue

AttributeTypeAndDistinguishedValue  ::=  SEQUENCE  {
      type                    ATTRIBUTE.&id ({SupportedAttributes}),
      value                   ATTRIBUTE.&Type({SupportedAttributes}{@type}),
      primaryDistinguished    BOOLEAN DEFAULT TRUE,
      valuesWithContext       SET SIZE (1..MAX) OF SEQUENCE {
            distingAttrValue        [0]    ATTRIBUTE.&Type ({SupportedAttributes}{@type}) OPTIONAL,
            contextList                    SET SIZE (1..MAX) OF Context } OPTIONAL }
```

El conjunto que forma un RDN contiene exactamente un **AttributeTypeAndDistinguishedValue** para cada atributo que contiene valores distinguidos en la inserción; esto es, un atributo determinado no puede aparecer dos veces en el mismo RDN.

Un valor de atributo designado para que aparezca en un RDN se denomina valor distinguido. Puede haber otros valores del mismo atributo que no sean distinguidos, por lo que no pueden utilizarse en un RDN. Un atributo puede tener múltiples valores distinguidos solamente si éstos se diferencian por el contexto asociado. Esto permite que un objeto tenga nombres alternativos diferenciados por los contextos. Éste es el único caso en el que un atributo puede tener más de un valor distinguido. En ese caso, los valores distinguidos poseerán listas de contexto que contengan los mismos tipos de contexto, cuyos valores de contexto asegurarán que únicamente uno de los valores distinguidos es aplicable para un contexto específico dado.

Se forma el RDN de una inserción dada utilizando un valor distinguido de cada atributo que posee valores distinguidos. El caso más simple es una inserción que tiene un valor distinguido; posee un solo RDN que se forma utilizando ese valor distinguido. Al RDN puede contribuir más de un atributo de una inserción. Si cada atributo contribuyente tiene solamente un valor distinguido, la inserción tiene un único RDN que se construye utilizando el valor distinguido de cada atributo. Si alguno de los atributos contribuyentes tiene múltiples valores distinguidos diferenciados por el contexto, la inserción poseerá múltiples RDN, cada uno de los cuales se construye utilizando una de las posibles combinaciones en la que se elige un valor distinguido para cada tipo de atributo que forma el RDN.

Cada RDN de una inserción contendrá un par **type** y **value** para cada atributo dado que forma parte del RDN. Se emplea **primaryDistinguished** para indicar que el valor es un **value** distinguido primario de ese tipo de atributo. Se utiliza **valuesWithContext** para transportar la lista de contexto para el valor de atributo distinguido en **value** cuando es necesario hacerlo así. Se utiliza también para transportar en un RDN único algunos o todos los demás valores distinguidos del mismo tipo de atributo. Cada **distingAttrValue** va acompañado de su **contextList**. Únicamente se omite **distingAttrValue** para el valor distinguido que aparece en **value**; así es como la lista de contexto de este valor se hace presente en el RDN.

Uno y solo uno de los valores distinguidos para un tipo de atributo determinado de una inserción será el *valor distinguido primario* para ese tipo de atributo. Este valor se utilizará como **value** en el **AttributeTypeAndDistinguishedValue** cuando se forme el nombre distinguido relativo primario del objeto (véanse 9.8 y 9.6). El *nombre distinguido relativo primario* es un RDN en el cual los valores distinguidos primarios del atributo RDN aparecen en las componentes **value** de cada **AttributeTypeAndDistinguishedValue** del RDN. En el componente **valuesWithContext** de cada **AttributeTypeAndDistinguishedValue** pueden aparecer valores distinguidos alternativos y contexto.

El RDN puede modificarse, de ser necesario, sustituyendo completamente todos los valores distinguidos de todos los atributos contribuyentes.

Los miembros de la familia, al igual que otras inserciones, tienen nombres distinguidos relativos (RDN). Un RDN puede constar de múltiples pares de tipo y valor de atributo. Sólo se pueden utilizar los RDN primarios. El *nombre de miembro local* de un miembro de la familia es la secuencia de los RDN desde el antepasado hasta ese miembro. El nombre de miembro local del antepasado es una secuencia vacía.

NOTA 2 – Se ha previsto que los RDN sean de larga duración por lo que los usuarios de directorio pueden almacenar los nombres distinguidos de los objetos (por ejemplo en el propio directorio) sin ninguna preocupación sobre su obsolescencia. Por lo tanto los RDN deben modificarse con precauciones.

NOTA 3 – La modificación del RDN de una inserción que no es hoja cambia automáticamente el nombre de las inserciones subordinadas.

NOTA 4 – El contexto en el que un tipo de atributo particular y un valor que forma parte de un RDN son aplicables, es independiente de los contextos asociados con cualquier otra parte de ese RDN o de otros RDN de un nombre distinguido.

NOTA 5 – Por ejemplo puede formarse un nombre distinguido válido de una inserción combinando un RDN designado como la variante idioma = francés de ese RDN de la inserción con el DN idioma = inglés de su inserción superior.

9.4 Concordancia de nombres

A veces, en la operación de directorio, es necesario determinar si dos nombres concuerdan. Esto exige la concordancia de los RDN correspondientes. Se describe aquí el enfoque general de la concordancia de nombres. Los métodos específicos para utilizaciones particulares de la concordancia de nombres se describen cuando sean procedente.

Se dice que un RDN propuesto concuerda con un RDN objetivo si cada **AttributeTypeAndDistinguishedValue** del RDN propusto concuerda con el **AttributeTypeAndDistinguishedValue** para el mismo tipo de atributo en el RDN objetivo. Existe una concordancia si el **value** o cualquier **distingAttrValue** propuestos del **AttributeTypeAndDistinguishedValue** propuesto concuerda con el **value** objetivo o cualquier **distingAttrValue** en el **AttributeTypeAndDistinguishedValue** objetivo. Para la concordancia se ignora el **primaryDistinguished**, si está presente en los **AttributeTypeAndDistinguishedValue** propuesto u objetivo.

NOTA 1 – Puede utilizarse la regla de concordancia de igualdad porque para los atributos de denominación, la sintaxis de atributo y la sintaxis de aserción de la regla de concordancia de igualdad son las mismas.

NOTA 2 – No hay ninguna garantía de que cada valor distinguido de un atributo de denominación determinado esté presente en el **AttributeTypeAndDistinguishedValue** de ese atributo en un RDN determinado. Pueden formarse dos RDN para el mismo objeto utilizando valores distinguidos distintos (diferenciados por el contexto) para el mismo tipo de atributo. Si no existe superposición entre los conjuntos de valores distinguidos para un atributo determinado que utiliza cada uno de ellos, éstos pueden dejar de concordar aún cuando el RDN propuesto y el RDN objeto sean RDN alternativos del mismo objeto. La forma en que esto puede suceder y su influencia dependen del motivo de la concordancia de nombres (por ejemplo resolución de nombre, control de acceso, filtrado).

Si al aplicar lo anterior se observa que los valores atributos no concuerdan, tampoco lo harán los RDN. Si los valores de atributo concuerdan, deberá haber también una concordancia entre los contextos asociados a esos valores, si existen, para considerar concordantes los pares tipo y valor de atributo. Se considera que cada contexto de la lista de contexto de valores de atributo propuestos es una aserción de contexto para el objetivo de concordancia de la lista de contexto de valores de atributo y se evaluará como verdadero como se describe en 8.8.2.4 para considerar los contextos concordantes. Cuando se forman las aserciones de contexto se ignora el **fallback** de los contextos propuestos.

NOTA 3 – Pueden utilizarse los contextos propuestos como aserciones de contexto de esta forma ya que la sintaxis de la aserción de contexto es la misma que la sintaxis de contexto para tipos de contexto que pueden utilizarse con valores distinguidos.

Si en un RDN propuesto no está presente **valuesWithContext**, las aserciones de contexto suministradas como parte de la operación o los valores por defecto establecidos para aplicarse a una operación, se aplicarán también como se describe en 8.8.2.2, salvo en el caso de concordancia de nombre durante la resolución de nombre en una operación de directorio. En ese caso no se aplican aserciones de contexto si no se dispone de ninguna en **valuesWithContext**.

9.5 Nombres devueltos durante las operaciones

Muchas operaciones de directorio devuelven el nombre de una inserción. Cuando una operación devuelva un nombre de una inserción o nombres de múltiples inserciones, devolverá también el nombre distinguido primario de cada inserción y puede, además devolver información de nombres distinguidos alternativos e información de contextos (véase 7.7 de la Rec. UIT-T X.511 | ISO/CEI 9594-3).

9.6 Nombres mantenidos como valores de atributo o utilizados como parámetros

Cuando se mantiene un nombre como valor de atributo dentro de algún otro atributo o se transfiere como valor de atributo en algún intercambio (por ejemplo un puntero de alias) subsiste siempre la cuestión de si el nombre mantenido puede ser un nombre distinguido alternativo o será el nombre distinguido primario que puede contener valores distinguidos alternativos o puede incluir información de contexto. En estas Especificaciones de directorio se mencionan las limitaciones específicas cuando sea necesario.

NOTA – El anexo O contiene sugerencias para mejorar la interoperabilidad con sistemas de ediciones anteriores a 1997 y asegurar un comportamiento predecible con respecto a la utilización de contextos con nombres.

9.7 Nombres distinguidos

El *nombre distinguido* de un objeto dado se define como la secuencia de los RDN de la inserción que representa al objeto y los de todas sus inserciones superiores (en orden descendente). Debido a la correspondencia de uno a uno entre los objetos e inserciones de objeto, puede considerarse que el nombre distinguido de un objeto es el nombre distinguido de la inserción del objeto.

NOTA 1 – Es preferible que los nombres distinguidos de los objetos que deban ser presentados a las personas sean fáciles para éstas.

NOTA 2 –La Rec. UIT-T X.650 | ISO/CEI 7498-3 define el concepto de un nombre primitivo. Un nombre distinguido puede utilizarse como un nombre primitivo del objeto que él identifica.

NOTA 3 – Debido a que sólo intervienen la inserción del objeto y sus superiores, los nombres distinguidos de objetos nunca pueden incluir inserciones de alias.

Las inserciones de alias tienen también nombres distinguidos; no obstante, este nombre puede ser el nombre distinguido de un objeto. Cuando hay que hacer esta distinción, se utiliza el término completo "nombre distinguido de una inserción de alias". El nombre distinguido de una inserción de alias se define, al igual que el nombre distinguido de una inserción de objeto, como la secuencia de los RDN de la inserción de alias y los de todas sus inserciones superiores (en orden descendente).

Es también conveniente definir el 'nombre distinguido' de la raíz, aunque éste nunca podrá ser el nombre distinguido de un objeto. El nombre distinguido de la raíz se define como la secuencia vacía.

Si cualquier atributo que contribuye a un RDN dentro del nombre distinguido de un objeto tiene múltiples valores distinguidos diferenciados por contextos, ese objeto tiene también múltiples nombres distinguidos. Cada uno de los cuales identifica inequívocamente al objeto. El nombre distinguido primario es el nombre distinguido para el que cada RDN es un RDN primario. Cuando se transporta en un protocolo, se forma el nombre distinguido primario utilizando el valor distinguido primario como **value** en el **AttributeTypeAndDistinguishedValue** de cada atributo en cada RDN que forma el nombre. Se forman los nombres distinguidos alternativos utilizando valores distinguidos alternativos para los atributos de uno o más RDN. En algunos casos puede utilizarse el nombre distinguido primario para un nombre. En otros pueden emplearse nombres distinguidos alternativos. Como el **AttributeTypeAndDistinguishedValue** de los RDN puede incluir valores distinguidos alternativos en el componente **valuesWithContext**, todo nombre distinguido puede incluir valores alternativos en sus RDN.

NOTA 4 – Se dice que el nombre distinguido incluye nombres alternativos cuando un RDN comprende múltiples valores distinguidos para cualquier atributo contribuyente.

En el componente **valuesWithContext** de cualquier RDN puede incluirse información de contexto con un valor distinguido. Cuando en estas Especificaciones de directorio se utilicen nombres, se especifica si el nombre es un nombre distinguido primario, si el nombre puede incluir valores alternativos y si puede incluirse la información de contexto. Cuando no se indique explícitamente, pueden utilizarse nombres distinguidos alternativos y el nombre puede comprender valores alternativos y/o información de contexto.

NOTA 5 – Cualquier requisito relativo a la utilización en un protocolo de un nombre distinguido primario en vez de un nombre distinguido alternativo no tiene porque reflejarse en el usuario final.

La figura 5 presenta un ejemplo que ilustra los conceptos de RDN y de nombre distinguido.

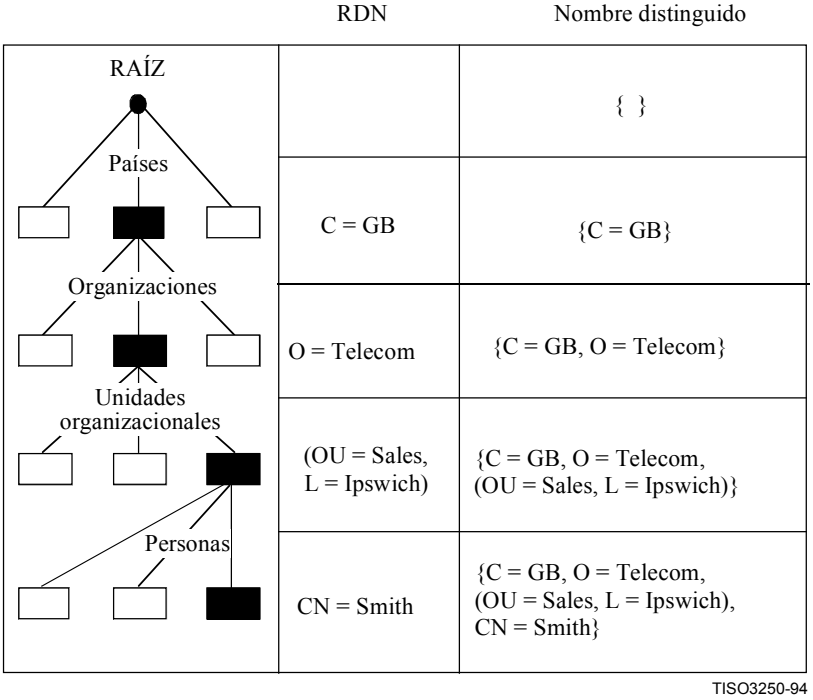


Figura 5 – Determinación de nombres distinguidos

9.8 Nombres de alias

Un *alias*, o un *nombre de alias*, para un objeto es un nombre alternativo para un objeto o inserción de objeto que es proporcionado por la utilización de inserciones de alias.

Cada inserción de alias puede contener, dentro del atributo **aliasedEntryName**, un nombre de algún objeto. El nombre distinguido de la inserción de alias es por tanto también un nombre para este objeto.

NOTA 1 – Se dice que el nombre dentro de **aliasedEntryName** es señalado por el alias. No tiene que ser el nombre distinguido de una inserción.

NOTA 2 – El valor de atributo **aliasedEntryName** pude ser el nombre distinguido primario o cualquier nombre distinguido alternativo, si existe. Pueden resultar afectados la coherencia y el interfuncionamiento con las DSA anteriores a 1997 si no se utiliza el nombre distinguido primario.

La conversión de un nombre de alias a un nombre de objeto se denomina desreferenciación (de alias) y comprende la sustitución sistemática de los nombres de alias, cuando se encuentran dentro de un nombre contemplado, por el valor del atributo correspondiente **aliasedEntryName**. El proceso puede requerir el examen de más de una inserción de alias.

Una inserción particular en el DIT puede tener ninguno o más nombres de alias. Como consecuencia de esto, varias inserciones de alias pueden apuntar a la misma inserción. Una inserción de alias puede apuntar a una inserción que no es una inserción hoja y puede apuntar a cualquier otra inserción de alias.

Una inserción de alias no tendrá subordinados, de modo que una inserción de alias es siempre una inserción hoja.

Cada inserción de alias pertenecerá a la clase de objeto **alias** que se define en 13.3.3.

No se permite que los miembros de la familia sean inserciones de alias.

10 Grupos jerárquicos

10.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

10.1.1 vástago jerárquico: Para una inserción, un vástago jerárquico es una inserción de la que es un progenitor jerárquico.

10.1.2 grupo jerárquico: Un grupo jerárquico es una colección de inserciones, incluidas las inserciones compuestas, que forma un árbol lógico no necesariamente relacionado con el árbol de información del directorio (DIT).

10.1.3 hoja jerárquica: Ésta es una inserción dentro de un grupo jerárquico que no tiene vástagos jerárquicos.

10.1.4 nivel jerárquico: Un entero que da la distancia desde una inserción dentro de un grupo jerárquico hasta el tope jerárquico en forma de número de enlaces jerárquicos entre la inserción y el tope jerárquico.

10.1.5 enlace jerárquico: Éste es un término general para la relación lógica entre dos inserciones que tienen una relación jerárquica inmediata de progenitor/vástago.

10.1.6 progenitor jerárquico: Para una inserción, los progenitores jerárquicos son el progenitor jerárquico inmediato, el progenitor jerárquico inmediato de éste, y así sucesivamente, de manera recursiva en sentido ascendente hasta, e incluido, el tope jerárquico.

10.1.7 hermano jerárquico: Para una inserción, los hermanos jerárquicos son las inserciones que tienen el mismo progenitor jerárquico inmediato que ella.

10.1.8 vástago-hermano jerárquico: Para una inserción, sus vástagos-hermanos jerárquicos son el conjunto completo de vástagos jerárquicos, en todos los niveles inferiores, de sus hermanos jerárquicos.

10.1.9 tope jerárquico: Ésta es la inserción, dentro de un grupo jerárquico, que constituye la raíz de la jerarquía. Un tope jerárquico no tiene progenitor jerárquico inmediato.

10.1.10 vástago jerárquico inmediato: Para una inserción, un vástago jerárquico inmediato es una inserción de la que es el progenitor jerárquico inmediato. No es necesario que este vástago jerárquico inmediato sea una inserción subordinada inmediata dentro del DIT.

10.1.11 progenitor jerárquico inmediato: Para una inserción, su progenitor jerárquico inmediato es la inserción que, dentro del grupo jerárquico, es su inserción inmediatamente superior. No es necesario que el progenitor jerárquico inmediato sea la inserción inmediatamente superior dentro del DIT.

10.2 Relaciones jerárquicas

Las inserciones del directorio tienen una relación jerárquica por la manera en que están situadas en el DIT. Pero también pueden tener relaciones jerárquicas no reflejadas en la estructura del DIT. Es posible, por ejemplo, que una organización dinámica no desee que se refleje su organización actual directamente en el DIT, porque quizás ello requiera frecuentes cambios de la estructura del DIT. El directorio ha de reflejar, por tanto, relaciones jerárquicas con independencia de la estructura del DIT. Esas relaciones se forman estableciendo *grupos jerárquicos*. Un grupo jerárquico constituye un árbol lógico cuya raíz se llama *tope jerárquico*.

Haciendo referencia a las relaciones jerárquicas es posible recuperar información, en una operación de búsqueda, no sólo a partir de una determinada inserción sino también a partir de otras inserciones dentro del mismo grupo jerárquico.

Una inserción compuesta se considera una inserción simple en el contexto de los grupos jerárquicos. Un miembro de familia vástago no puede formar parte de un grupo jerárquico por derecho propio.

NOTA – La finalidad de los grupos jerárquicos es permitir el modelado de colecciones de objetos distintos que tienen relaciones lógicas informales, y sobre todo relaciones que son, o podrían ser, temporales. Las inserciones compuestas, por el contrario, modelan objetos que comprenden subobjetos a los que se considera convenientemente como una jerarquía.

Para describir la navegación dentro de un grupo jerárquico conviene definir los términos de las relaciones que una inserción determinada tiene con otras inserciones dentro del grupo. Esto es lo que se hace en 10.1. Algunos de los términos definidos para relaciones directas son paralelos a los definidos para relaciones de inserciones dentro del DIT (*vástago jerárquico inmediato, vástago jerárquico, progenitor jerárquico inmediato* y *progenitor jerárquico*). Sin embargo, también conviene definir términos para relaciones más distantes. En algunas situaciones, es posible que un usuario desee recuperar información de *hermanos jerárquicos*, e incluso de sus vástagos jerárquicos (*vástagos-hermanos jerárquicos*).

Una inserción sólo puede ser miembro de un solo grupo jerárquico a la vez.

Una inserción que forma parte de un grupo jerárquico contiene un atributo operacional definido en 14.9, que refleja la relación con otras inserciones del grupo, incluyendo el *nivel jerárquico* de la inserción dentro del grupo. Cuando una inserción compuesta forma parte de un grupo jerárquico, este atributo operacional es mantenido por el antepasado.

Un grupo jerárquico ha de estar fuera por completo de cualquier zona administrativa específica de servicio (véase 16.3) o contenido por completo dentro de una zona administrativa específica de servicio. Un grupo jerárquico deberá estar confinado a un solo agente de sistema del directorio (DSA). El servicio del directorio deberá detectar y evitar los intentos de romper estas reglas.

SECCIÓN 4 – MODELO ADMINISTRATIVO DEL DIRECTORIO

11 Modelo de autoridad administrativa de directorio

11.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

11.1.1 zona administrativa: Un subárbol del DIT considerado desde la perspectiva de administración.

11.1.2 inserción administrativa: Una inserción situada en un punto administrativo.

11.1.3 punto administrativo: El vértice raíz de una zona administrativa.

11.1.4 usuario administrativo: Un representante de una autoridad administrativa. La definición completa del concepto usuario administrativo está fuera del alcance de esta Especificación de directorio.

11.1.5 zona administrativa autónoma: Un subárbol del DIT cuyas inserciones son administradas por la misma autoridad administrativa. Las zonas administrativas autónomas no se superponen.

11.1.6 autoridad administrativa de dominio de árbol de información de directorio: Una autoridad administrativa en su cometido de entidad que tiene la responsabilidad de la administración del DIT.

11.1.7 política de dominio de árbol de información de directorio: Una expresión de las metas generales y los procedimientos aceptables para un dominio DIT.

11.1.8 autoridad administrativa de dominio de gestión de directorio: Una autoridad administrativa en su cometido de entidad responsable de la administración de un DMD.

11.1.9 política de dominio de gestión de directorio: Una política que gobierna el funcionamiento de los DSA en un DMD.

11.1.10 política de organización de gestión de dominio: Una política definida por una DMO, expresada como políticas de dominios de DMD y de DIT.

11.1.11 zona administrativa interior: Una zona administrativa específica cuyo alcance está totalmente contenido dentro del alcance de otra zona administrativa específica del mismo tipo.

11.1.12 política: Una expresión, por una autoridad administrativa, de metas generales y procedimientos aceptables.

11.1.13 atributo de política: Un término genérico para cualquier atributo operacional de directorio, que expresa política.

11.1.14 objeto de política: Una entidad a la cual concierne una política.

11.1.15 procedimiento de política: Una regla que define cómo debe ser considerado un conjunto de objetos de política y qué acciones deben efectuarse como resultado de esta consideración.

11.1.16 parámetro de política: Un procedimiento de política se caracteriza por ciertos *parámetros de política* que están sujetos a configuración (es decir, a una elección) por una autoridad administrativa.

11.1.17 zona administrativa específica: Un subconjunto (en forma de subárbol) de una zona administrativa autónoma definido para un aspecto particular de administración: administración de control de acceso, de subesquema, o de colección de inserciones. Cuando están definidas, las zonas administrativas específicas de *una modalidad particular* dividen una zona administrativa autónoma.

11.1.18 punto administrativo específico: El vértice raíz de una zona administrativa específica.

11.2 Visión de conjunto

Un objetivo fundamental del modelo de información de directorio es considerar colecciones bien definidas de inserciones de manera que puedan ser administradas coherentemente como una unidad. Esta cláusula aclara la naturaleza y alcance de las autoridades responsables de esa administración y los medios por los cuales ejercen su autoridad.

El concepto de política, definido en 11.3, proporciona el mecanismo por el cual las autoridades administrativas ejercen el control de directorio.

Algunos aspectos del modelo administrativo de directorio son soportados por el modelo de información administrativa y operacional de directorio (véase la cláusula 12), para permitir el modelado de la información requerida para la reglamentación de la información de usuario de directorio y para otros fines administrativos.

Otros aspectos del modelo administrativo de directorio requieren soporte para la distribución de información administrativa y operacional entre las partes componentes de directorio, es decir, los DSA. En las cláusulas 22 a 24 se describe un modelo de información de DSA para soportar estas exigencias.

11.3 Política

Una *política* es una expresión, por una autoridad administrativa, que actúa como un agente de la DMO, de objetivos generales y procedimientos aceptables. Una política se define en términos de reglas que han de cumplirse (por el directorio, si procede) y desde el punto de vista de aspectos dentro de los cuales un usuario administrativo tiene cierto grado de libertad de acción y responsabilidades específicas.

Una autoridad administrativa expresa la política de la DMO desde el punto de vista de:

- la política de dominio del DIT;
- la política del DMD.

Estas políticas pueden expresarse como atributos de política. En 11.6 se define un modelo de políticas del DIT.

NOTA – La cláusula 14 define un esquema necesario para soportar la administración de atributos colectivos. La cláusula 15 define un marco para soportar políticas de administración de subesquema. La cláusula 17 define un marco que soporta políticas de control de acceso.

Las políticas de DMD se relacionan específicamente con los DSA como componentes de directorio distribuido. Estas políticas de DMD se describen en 11.7, que define un modelo para la administración de DSA.

Por último, hay políticas que se relacionan con asuntos externos (como acuerdos bilaterales entre las DMO), y por esa razón no se tratan aquí más detalladamente.

Un *objeto de política* es una entidad que es influida por una política (por ejemplo, una zona administrativa de subesquema es un objeto de política).

Un *procedimiento de política* es una regla que define cómo debe ser considerado un conjunto de objetos de política y qué acciones deben efectuarse como resultado de esa consideración (por ejemplo, la cláusula 15 define procedimientos de política de administración de subesquema).

Un procedimiento de política se caracteriza por ciertos *parámetros de política* que están sujetos a configuración (es decir, a elección) por una autoridad administrativa.

Se utilizan atributos operacionales para representar parámetros de política. Los valores de tal atributo forman una expresión de una parte o de la totalidad del parámetro de política que él representa.

11.4 Autoridades administrativas específicas

La administración de un dominio de DIT comprende la ejecución de cinco funciones relacionadas con diferentes aspectos de administración.

- administración de denominación;
- administración de subesquema;
- administración de seguridad;
- administración de atributo colectivo;
- administración de servicio.

Una *autoridad administrativa específica* es una autoridad administrativa en su cometido de entidad responsable de uno de estos aspectos específicos de política de dominio de DIT.

El término *autoridad de denominación* (véase la cláusula 9) identifica el cometido de la autoridad administrativa en lo pertinente a la atribución de nombres y a la administración de la estructura de estos nombres. Un cometido de la autoridad de subesquema es implementar estas estructuras de denominación en el subesquema.

El término *autoridad de subesquemas* identifica el cometido de una autoridad administrativa en lo pertinente al establecimiento, administración y ejecución de la política de subesquema que controla la denominación y el contenido de inserciones en un dominio de DIT. La cláusula 15 describe el soporte de directorio de la administración de subesquemas.

El término *autoridad de seguridad* (véase la Rec. UIT-T X.509 | ISO/CEI 9594-8) identifica el cometido de la autoridad administrativa en lo pertinente al establecimiento, administración y ejecución de una política de seguridad que rige el comportamiento de directorio con respecto a inserciones en un dominio de DIT.

El término *autoridad de atributo colectivo* identifica el cometido de la autoridad administrativa en lo pertinente al establecimiento y administración de atributos colectivos (véase 12.7) en un dominio de DIT.

El término *autoridad de servicio* identifica el cometido de la autoridad administrativa en lo que respecta al establecimiento y administración de constricciones y ajustes del servicio.

11.5 Zonas administrativas y puntos administrativos

11.5.1 Zonas administrativas autónomas

Cada inserción en el DIT es administrada exactamente por una autoridad administrativa (que puede ejecutar cometidos diferentes). Una *zona administrativa autónoma* es un subárbol del DIT cuyas inserciones son administradas por la misma autoridad administrativa.

El dominio de DIT puede ser dividido en una o más zonas administrativas autónomas que no se superponen.

El conjunto de una o más zonas administrativas autónomas para el cual una organización de gestión de dominio (DMO) tiene autoridad administrativa es su dominio de DIT. Esto se ilustra en la figura 6.

11.5.2 Zonas administrativas específicas

De la misma manera que una autoridad administrativa puede ejecutar un cometido específico, las inserciones de una zona administrativa pueden ser consideradas como una función administrativa específica. Cuando se considera en este contexto, una zona administrativa se llama una *zona administrativa específica*. Hay cinco clases de zonas administrativas específicas:

- zonas administrativas de subesquema;
- zonas administrativas de control de acceso;
- zonas administrativas de atributo colectivo;
- zonas administrativas por defecto de contexto;
- zonas administrativa de servicio.

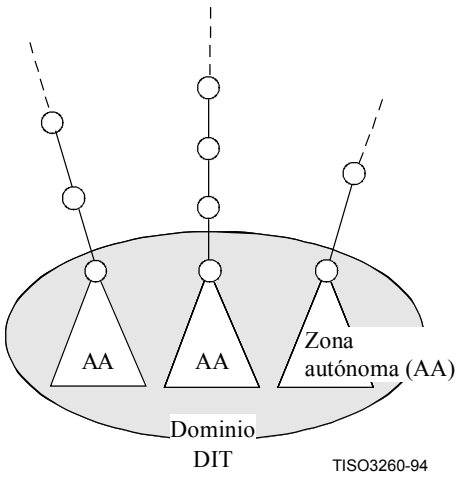


Figura 6 – Un dominio de DIT

Se puede considerar que una zona administrativa autónoma define explícitamente una sola zona administrativa específica para cada aspecto específico de administración. En este caso hay una correspondencia precisa entre cada zona administrativa específica y la zona administrativa autónoma.

Como otra posibilidad, para cada aspecto específico de administración, la zona administrativa autónoma puede estar dividida en zonas administrativas específicas que no se superponen.

Si está dividida de esta manera para un determinado aspecto de administración, cada inserción de la zona administrativa autónoma está contenida solamente en una zona administrativa específica de ese aspecto.

Una autoridad específica es responsable de cada zona administrativa específica. Si, para un determinado aspecto administrativo, una zona administrativa autónoma no está dividida, una autoridad administrativa específica es responsable de ese aspecto administrativo para toda la zona administrativa autónoma.

11.5.3 Áreas administrativas interiores

Para fines de administración de seguridad o de atributo colectivo, pueden definirse *zonas (administrativas) interiores* dentro de estas clases de zonas administrativas específicas:

a) para representar una forma limitada de delegación; o

b) por razones de conveniencia administrativa y operacional (por ejemplo, donde el punto administrativo de un subárbol está en un DSA que no es el que contiene las inserciones dentro del árbol, ese subárbol puede ser designado como una zona interior para permitir la administración por el DSA local).

Una zona administrativa interior puede estar contenida dentro de otra zona administrativa interior.

Las zonas interiores representan zonas de autonomía limitada. Las inserciones en zonas interiores son administradas por las autoridades administrativas específicas de las zonas administrativas específicas en las que están contenidas, y también por las autoridades administrativas de las zonas interiores en las que están contenidas. Las autoridades primeramente mencionadas tienen el control general de las políticas que rigen estas inserciones, mientras que las autoridades mencionadas en segundo término tienen un control (limitado) sobre los aspectos de política que les han sido delegados por las autoridades mencionadas en primer término.

Las reglas para zonas interiores jerarquizadas, si se permiten, se definirán como parte de la definición de los aspectos administrativos específicos dentro de los que están contenidas.

11.5.4 Puntos administrativos

La especificación de la extensión de una zona administrativa autónoma es implícita y consiste en la identificación de un punto en el DIT (la raíz del subárbol de la zona administrativa autónoma), un *punto administrativo autónomo*, desde el cual la zona administrativa parte en sentido descendente hasta que se encuentra otro punto administrativo autónomo, en el que comienza otra zona administrativa autónoma.

NOTA 1 – Los subordinados inmediatos de la raíz del DIT son puntos administrativos.

Cuando una zona administrativa autónoma *no está* dividida para un aspecto específico de administración, la zona administrativa para ese aspecto coincide con la zona administrativa autónoma. En este caso, el punto administrativo autónomo es también un *punto administrativo específico* para todos los aspectos de administración.

Cuando una zona administrativa autónoma *está* dividida para un aspecto específico de administración, la especificación de la extensión de cada zona administrativa específica consiste en la identificación de la raíz del subárbol de la zona administrativa específica, un *punto administrativo específico*, desde el cual la zona administrativa específica parte en sentido descendente hasta que se encuentra otro punto administrativo específico (del mismo aspecto administrativo), en el que comienza otra zona administrativa específica.

Las zonas administrativas específicas están siempre confinadas por la zona administrativa autónoma de cuya partición proceden.

Un punto administrativo particular puede ser la raíz de una zona administrativa autónoma y puede ser la raíz de una o más zonas administrativas específicas.

La especificación de la extensión de una zona administrativa interior (dentro de una zona administrativa específica) consiste en la identificación de la raíz del subárbol de la zona administrativa interior, un *punto administrativo interior*. Una zona administrativa interior está confinada por la zona administrativa específica dentro de la cual está definida.

Un punto administrativo que corresponde a la raíz de una zona administrativa autónoma representa una frontera de dominio de DIT (y de DSA). Esto es, su superior inmediato en el DIT tiene que estar bajo la autoridad administrativa de otro DMD.

NOTA 2 – Esto implica que una DMO no puede dividir arbitrariamente un dominio de DIT en zonas administrativas autónomas.

Un punto administrativo se representa en el modelo de información de directorio por una inserción que contiene un atributo **administrativeRole**. El valor o los valores de este atributo identifican el tipo de punto administrativo. Este atributo se define en 14.3.

En las cláusulas 22 a 24 se describe la correspondencia de zonas administrativas con los DSA y el modelo de información de DSA.

La figura 7 representa una zona administrativa autónoma que ha sido dividida en dos zonas administrativas específicas para un aspecto específico de administración (por ejemplo, control de acceso). En una zona administrativa específica se ha creado una zona administrativa interior jerarquizada (por ejemplo, porque el subárbol debe estar contenido en un DSA diferente del resto de la zona administrativa específica).

En la figura 7 se utilizan las abreviaturas AAP (*autonomous administrative point* – punto administrativo autónomo), SAP (*specific administrative point* – punto administrativo específico) e IAP (*inner administrative point* – punto administrativo interior).

11.5.5 Inserciones administrativas

Una inserción situada en un punto administrativo, se denomina una *inserción administrativa*. Las inserciones administrativas pueden tener inserciones especiales, llamadas *subinserciones*, como subordinadas inmediatas. La inserción administrativa y sus subinserciones asociadas se utilizan para controlar las inserciones abarcadas por la zona administrativa asociada.

Cuando se utilizan las zonas administrativas interiores, los alcances de estas zonas pudieran superponerse.

En consecuencia, para cada aspecto específico de autoridad administrativa, se requiere una definición del método de combinación de información administrativa cuando sea posible incluir inserciones en más de un subárbol o refinamiento de subárbol asociado con una zona interior definida para ese aspecto.

NOTA – No es necesario que cada punto administrativo represente cada aspecto específico de autoridad administrativa. Por ejemplo, podría haber un punto administrativo, subordinado a la raíz de la zona administrativa autónoma, que se utiliza exclusivamente para fines de control de acceso.

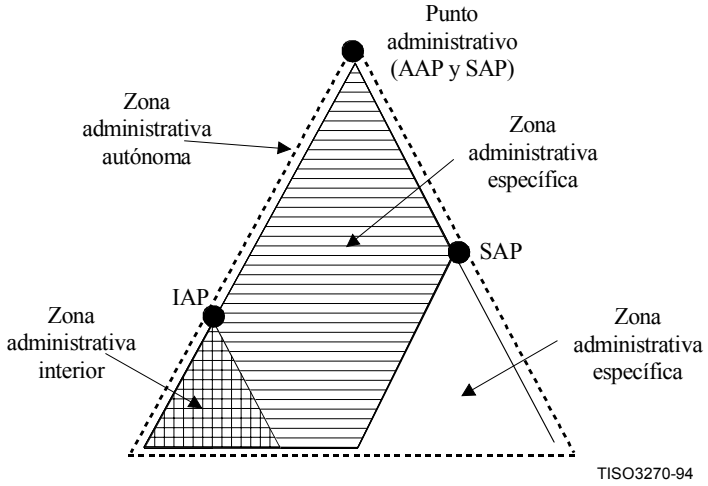


Figura 7 – Puntos y zonas administrativos

11.6 Políticas de dominio de DIT

Una política de dominio de DIT tiene los siguientes componentes: objetos de política DIT, procedimientos de política DIT y parámetros de política DIT.

Un atributo operacional que representa un parámetro de política DIT se llama un *atributo de política DIT* (por ejemplo, los atributos operacionales de administración de subesquema definidos en la cláusula 14 son atributos de política de dominio de DIT).

Para un DSA particular, los posibles valores de un parámetro pueden no corresponder a acciones distintas, realizables para ese componente. Esta situación puede darse, por ejemplo, cuando el DSA carece de la capacidad técnica para efectuar todos los aspectos del procedimiento de política (por ejemplo, implementar un esquema de control de acceso particular). Para que un procedimiento de política esté bien definido se tendrán en cuenta esas circunstancias en su definición.

Los objetos y atributos específicos de política de dominio de DIT se definen en la cláusula 15 para soportar la administración de subesquema.

11.7 Políticas de DMD

Una *política de DMD* es una política referente a uno o más de los DSA en el DMD. Una política DMD podrá aplicarse a todos los DSA en el DMD de una manera uniforme, a un subconjunto de los DSA en el DMD, o podrá aplicarse a un DSA específico.

Un tipo de política de DMD consiste en restringir o controlar en otra forma el servicio abstracto de directorio y de DSA proporcionado por uno o más DSA.

Son ejemplos de esas restricciones:

a) Limitación del servicio básico proporcionado a usuarios de directorio (es decir, no administrativos) a operaciones de interrogación solamente, de acuerdo con la Rec. CCITT F.500.

b) Limitación del servicio proporcionado a usuarios que acceden al DSA indirectamente, por concatenación, incluidas distinciones basadas en si la petición del usuario ha atravesado o no un trayecto de confianza.

c) Limitaciones a peticiones aceptadas de usuarios que acceden al DSA directamente cuando se requieren concatenaciones a DSA en el DMD que se sabe están sujetas a limitaciones de la forma indicada en el punto anterior.

d) Constricciones de los tipos de búsqueda que determinados usuarios pueden efectuar y de las características de tales búsquedas (por ejemplo, políticas de relajación).

SECCIÓN 5 – MODELO DE INFORMACIÓN ADMINISTRATIVA Y OPERACIONAL DE DIRECTORIO

12 Modelo de información administrativa y operacional de directorio

12.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

12.1.1 base: El vértice de la raíz del subárbol o refinamiento del subárbol producido por la evaluación de una especificación de subárbol.

12.1.2 parte: Un conjunto de aserciones relativas a los nombres de los subordinados de una base.

12.1.3 atributo operacional de directorio: Un atributo operacional definido y visible en el modelo de información administrativo y operacional de directorio.

12.1.4 esquema de sistema de directorio: El conjunto de reglas y constricciones relativas a los atributos operacionales y subinserciones.

12.1.5 inserción: Una inserción de directorio o inserción de directorio ampliada, según el contexto en el cual se utiliza el término (contexto de usuarios y sus aplicaciones o contexto de administración y operación de directorio).

12.1.6 subinserción: Una clase especial de inserción, conocida por el directorio, utilizada para mantener información asociada con un subárbol o refinamiento de subárbol.

12.1.7 subárbol: Una colección de inserciones de objetos y de alias situadas en los vértices de un árbol. El prefijo "sub" destaca que el vértice de la base (o raíz) de este árbol suele estar subordinado a la raíz del DIT.

12.1.8 refinamiento de subárbol: Un subconjunto especificado explícitamente de las inserciones en un subárbol, cuando las inserciones no están situadas en los vértices de un solo subárbol.

12.1.9 especificación de subárbol: La especificación *explícita* de un subárbol o refinamiento de subárbol. Una especificación de subárbol consiste en ninguno o más elementos de especificación: base, parte y filtro de especificación. La definición se califica de explícita (en contraste con la de una zona administrativa) porque la porción del DIT subordinada a la base que se incluye en el subárbol o refinamiento de subárbol se especifica explícitamente.

12.2 Visión de conjunto

Desde una perspectiva administrativa, la información de usuario mantenida en la DIB es complementada por la información administrativa y operacional representada por:

- los *atributos operacionales*, que representan información utilizada para controlar la operación de directorio (por ejemplo, información de control de accesos) o que es utilizada por el directorio para representar algún aspecto de su funcionamiento (por ejemplo, información de indicación de tiempo); y
- *subinserciones*, que asocian los valores de un conjunto de atributos (por ejemplo, atributos colectivos) con inserciones dentro del alcance de la subinserción. El alcance de una subinserción es un subárbol o refinamiento de subárbol.

Esta información, ilustrada en la figura 8, puede ser colocada en el directorio por las autoridades administrativas o por los DSA, y es utilizada por el directorio en el curso de su funcionamiento.

Dos mecanismos en el servicio abstracto de directorio que se relacionan con esta visión de la información de directorio son:

- se ha ampliado **EntryInformationSelection (selección de información de inserción)** para permitir la selección de atributos operacionales en una inserción; y
- se ha añadido el control del servicio **subentries (subinserción)** para permitir las operaciones de enumeración y búsqueda que han de aplicarse a inserciones de objetos y de alias o a subinserciones.

El acceso a la información operacional, como la información de usuario, se puede limitar mediante un mecanismo de control de acceso.

Las inserciones se hacen visibles a los usuarios de directorio mediante el servicio abstracto de directorio, pero sus relaciones con los DSA que las mantienen a la larga no son visibles. El modelo de información de DSA, descrito en las cláusulas 22 a 24, expone la correspondencia de estas inserciones con los depósitos de información de los DSA.

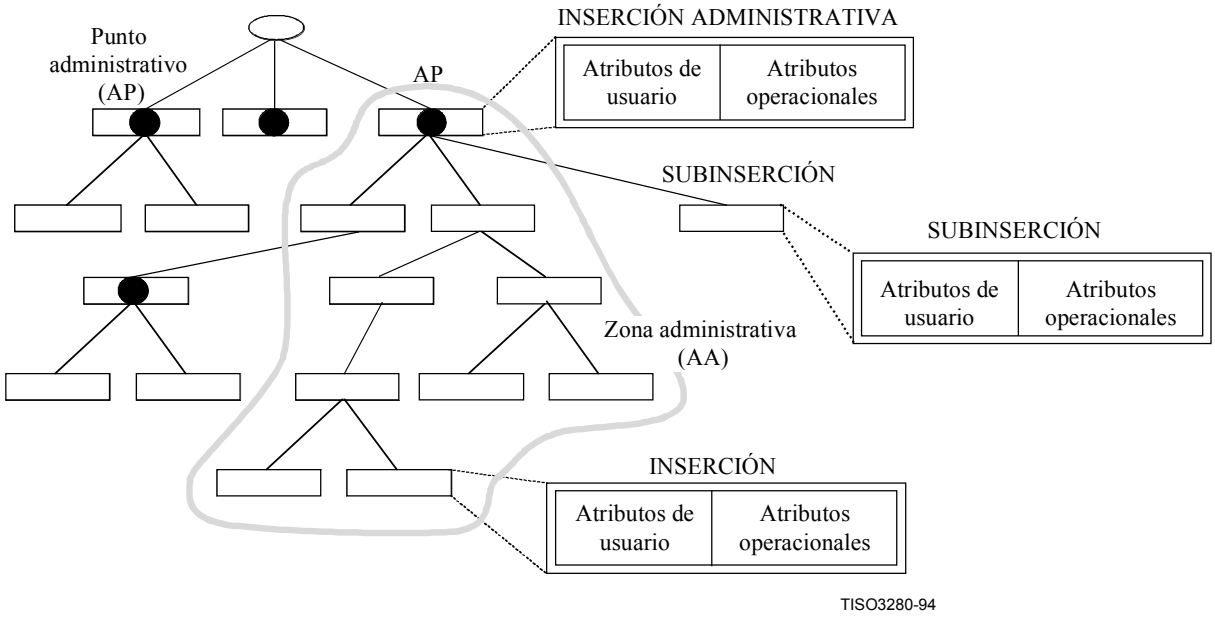


Figura 8 – Modelo de información administrativa y operacional de directorio

12.3 Subárboles

12.3.1 Visión de conjunto

Un subárbol es una colección de inserciones de objetos y de alias situadas en los vértices de un árbol. Los subárboles no contienen subinserciones. El prefijo "sub", en subárbol, destaca que el vértice de la base (o raíz) de este árbol suele estar subordinado a la raíz del DIT.

Un subárbol comienza en algún vértice y se extiende hasta alguna frontera más baja identificable, que posiblemente se extiende a hojas. Un subárbol se define siempre dentro de un contexto que limita implícitamente el subárbol. Por ejemplo, el vértice y fronteras más bajas de un subárbol que define una zona replicada están limitados por un contexto de denominación. De manera similar, el alcance de un subárbol que define una zona administrativa específica está limitado al contexto de una zona administrativa autónoma contenedora.

12.3.2 Especificación de subárbol

La especificación de subárbol es la definición de un subconjunto de las inserciones por debajo de un vértice especificado que forma la base del subárbol o refinamiento de subárbol.

El vértice y/o la frontera más baja del subárbol se puede especificar implícitamente, en cuyo caso están determinados por el contexto dentro del cual se utiliza el subárbol.

El vértice y/o la frontera más baja se puede especificar explícitamente utilizando el mecanismos especificado en esta cláusula. Este mecanismo se puede usar también para especificar refinamientos de subárbol que no son estructuras de árbol verdaderas.

NOTA – El concepto topológico de un (sub)árbol es útil para considerar estas especificaciones, aunque una especificación particular puede determinar una colección de inserciones que *no* están situadas en los vértices de un solo (sub)árbol. Se prefiere el término *refinamiento de subárbol* cuando las inserciones de la colección no están así situadas.

La especificación de un subárbol consiste en tres elementos de especificación facultativos que identifican la base del subárbol y reducen así la colección de inserciones subordinadas. Estos elementos de especificación son:

a) *Base* – el vértice raíz del subárbol o refinamiento de subárbol producido por la evaluación de una especificación de subárbol;

b) *Parte* – un conjunto de aserciones concernientes a los nombres de las inserciones subordinadas; y

c) *Filtro de especificación* – un subconjunto apropiado de la capacidad de aserción de un filtro aplicado a los subordinados.

La especificación de un subárbol o refinamiento de subárbol puede representarse por el siguiente tipo ASN.1:

```
SubtreeSpecification ::= SEQUENCE {
    base                [0]  LocalName DEFAULT { },
                             COMPONENTS OF  ChopSpecification,
    specificationFilter [4]  Refinement OPTIONAL }
    -- empty sequence specifies whole administrative area
```

Los tres componentes de este conjunto corresponden a los tres elementos de especificación identificados más arriba.

Cuando un valor de **SubtreeSpecification (especificación de subárbol)** identifica una colección de inserciones que están situadas en los vértices de un solo árbol, la colección se denomina un subárbol, en los demás casos, la colección se denomina un refinamiento de subárbol.

El tipo **SubtreeSpecification** proporciona un mecanismo de propósito general para la especificación de subárboles y refinamientos de subárbol. Cualquier uso particular de este mecanismo define la semántica específica exactamente de lo que es especificado y puede imponer limitaciones o constricciones a los componentes de **SubtreeSpecification.**

Cuando cada uno de los componentes de **SubtreeSpecification** está ausente (es decir, un valor del tipo **SubtreeSpecification** que es una secuencia vacía, **{}**), el subárbol así especificado está determinado implícitamente por el contexto dentro del cual se utiliza **SubtreeSpecification**.

Estos términos se ilustran en la figura 9 para el caso cuando los subárboles se despliegan dentro del contexto de zonas administrativas.

12.3.3 Base

El componente **base** de **SubtreeSpecification** representa el vértice de la raíz del subárbol o refinamiento de subárbol. Puede ser una inserción subordinada al vértice de la raíz del alcance identificado o puede ser el vértice de la raíz del propio alcance identificado (por defecto).

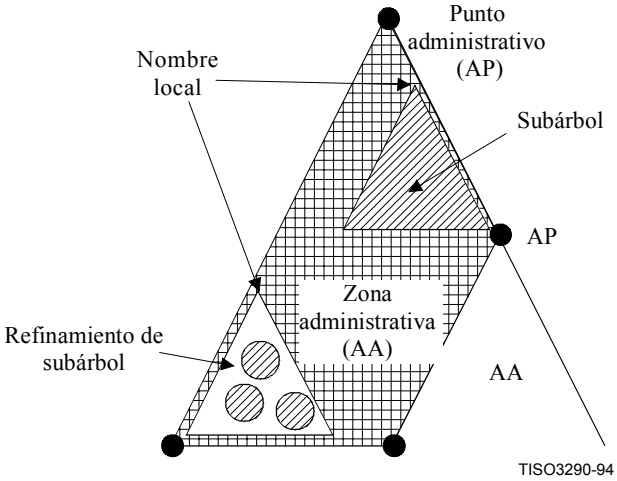


Figura 9 – Especificación de subárboles y refinamientos de subárbol dentro del contexto de zonas administrativas

El nombre relativo de vértice de la raíz del subárbol con respecto al vértice de la raíz del alcance identificado es un valor de tipo **LocalName**:

```
LocalName ::= RDNSequence
```

Obsérvese que cuando **LocalName** se omite en **SubtreeSpecification**, coinciden el vértice de la raíz del alcance identificado y el vértice de la raíz del subárbol.

Los RDN utilizados para denominar el vértice de la raíz del subárbol serán RDN primarios.

12.3.4 Especificación de parte

El componente **ChopSpecification (especificación de parte)** consiste en un conjunto de aserciones relativas a los nombres de los subordinados de una base. Consiste en un valor de tipo **ChopSpecification**:

```
ChopSpecification ::= SEQUENCE {
    specificExclusions    [1]   SET SIZE (1..MAX) OF CHOICE {
        chopBefore              [0]   LocalName,
        chopAfter               [1]   LocalName } OPTIONAL,
    minimum               [2]   BaseDistance DEFAULT 0,
    maximum               [3]   BaseDistance OPTIONAL }
```

Este tipo tiene por finalidad permitir la especificación de una estructura de árbol (o un subconjunto de ella) comenzando en la base por dos métodos: exclusiones específicas y distancia de base.

Cuando cualquier atributo de un RDN en **chopBefore** o **chopAfter** tenga múltiples valores distinguidos diferenciados por el contexto, se utilizará como **value** en el **LocalName** del RDN, el valor distinguido primario.

12.3.4.1 Exclusiones específicas

El componente **specificExclusions** tiene dos formas **chopBefore** y **chopAfter**, que pueden utilizarse individual o combinadamente.

El componente **chopBefore** define una lista de exclusiones, cada una de las cuales tiene la forma de algún punto límite que deberá ser excluido, junto con sus subordinados, del subárbol o refinamiento de subárbol. Los puntos límite son las inserciones identificadas por un **LocalName**, relativo a la base.

El componente **chopAfter** define una lista de exclusiones, cada una de las cuales tiene la forma de algún punto límite cuyos subordinados deberán ser excluidos del subárbol o refinamiento de subárbol. Los puntos límite son las inserciones identificadas por un **LocalName** relativo a la base.

12.3.4.2 Mínimos y máximos

Estos componentes permiten excluir todas las inserciones que son superiores a inserciones que son arcos RDN **minimum (mínimos)** por debajo de la base así como las inserciones que son subordinadas a inserciones que son arcos RDN **maximum (máximos)** por debajo de la base. Estas distancias se expresan por valores del tipo **BaseDistance**:

```
BaseDistance ::= INTEGER (0..MAX)
```

A los efectos de las especificaciones de parte, una inserción compuesta se cuenta como una inserción simple. En una inserción compuesta, todos los miembros de la familia se cuentan como si tuvieran la misma distancia básica que el antepasado, ya que todos ellos forman parte de la misma inserción lógica.

Un valor de **minimum** igual a cero (el valor por defecto) corresponde a la base. Un componente **maximum** ausente indica que no debe imponerse un límite inferior al subárbol o refinamiento de subárbol.

12.3.5 Filtro de especificación

El componente **specificationFilter** consiste en un subconjunto propio de la capacidad de aserción de un filtro (véase la Rec. UIT-T X.511 | ISO/CEI 9594-3) aplicado a los subordinados de una base. Sólo las inserciones para las cuales el filtro evalúa a verdadero son incluidas en el refinamiento de subárbol resultante. Consiste en un valor de tipo **Refinement**:

```
Refinement ::= CHOICE {
    item      [0]   OBJECT-CLASS.&id,
    and       [1]   SET OF Refinement,
    or        [2]   SET OF Refinement,
    not       [3]   Refinement }
```

Un **Refinement** evalúa a VERDADERO (TRUE) como si fuese un filtro que estuviese efectuando una aserción **equality** con respecto a valores de tipo de atributo **objectClass** solamente.

Si un miembro de una familia es excluido de un subárbol por esta especificación, también lo son todos sus miembros de la familia subordinados.

12.4 Atributos operacionales

Hay tres variedades de atributos operacionales: atributos operacionales de directorio, atributos operacionales compartidos por DSA y atributos operacionales específicos de DSA.

Los *atributos operacionales de directorio* aparecen en el modelo de información de directorio y se utilizan para representar información de control (por ejemplo, información de control de acceso) o información de otro género proporcionada por el directorio (por ejemplo, una indicación de si una inserción es o no una inserción hoja).

Los *atributos operacionales compartidos por DSA* aparecen solamente en el modelo de información de DSA y no son visibles en todos los modelos de información de directorio.

Los *atributos operacionales específicos de DSA* aparecen solamente en el modelo de información de DSA y no son visibles en todos los modelos de información de directorio.

NOTA – Se describen en las cláusulas 23 y 24.

La definición y utilización de cada atributo operacional deben ser especificadas en la Especificación de directorio apropiada.

12.5 Inserciones

12.5.1 Visión general

Desde la perspectiva administrativa, la información de usuario mantenida en una inserción puede ser complementada por información administrativa y operacional representada por atributos operacionales.

El directorio utiliza el atributo de clase de objeto y las reglas de contenido del DIT aplicables a una inserción para controlar los atributos de usuario requeridos y permitidos en la inserción. Los atributos operacionales de una inserción están regidos por el esquema del sistema de directorio (véase la cláusula 14) aplicable a la inserción.

12.5.2 Acceso a atributos operacionales

Aunque normalmente no son visibles, los atributos operacionales de directorio dentro de inserciones pueden hacerse visibles a usuarios autorizados (por ejemplo, usuarios administrativos) del servicio abstracto de directorio. Ciertos atributos operacionales pudieran estar disponibles también para usuarios ordinarios (por ejemplo, **entryACI** o **modifyTimestamp**).

12.6 Subinserciones

12.6.1 Visión de conjunto

Una *subinserción* es una clase especial de inserción inmediatamente subordinada a un punto administrativo. Contiene atributos que pertenecen a un subárbol (o refinamiento de subárbol) asociado con su punto administrativo. Las subinserciones y su punto administrativo forman parte del mismo contexto de denominación (véase la cláusula 21).

Una sola subinserción puede servir a todos o a varios aspectos de la autoridad administrativa. Como otra posibilidad, un aspecto específico de la autoridad administrativa puede ser tratado a través de una o más de sus propias subinserciones. Como máximo, se permite una subinserción para una autoridad administrativa de subesquema. Las autoridades de control de acceso y de atributos colectivos pueden tener varias subinserciones.

No se considera una subinserción en las operaciones **list (enumeración)** y **search (búsqueda)** a menos que se incluya en la petición el control de servicio de **subentries (subinserciones)**.

Una subinserción no tendrá subordinados.

La estructura de una subinserción correspondiente a un punto administrativo se muestra en la figura 10.

Una subinserción consiste en:

- Un atributo **commonName (nombre común)** especificado en la Rec. UIT-T X.521 | ISO/CEI 9594-6 que contiene el RDN de la subinserción.
- Un atributo **subtreeSpecification (especificación de subárbol)**, especificado en la cláusula 14.
- Un atributo **objectClass (clase de objeto)**, especificado en la cláusula 13, que indica la finalidad o finalidades de la subinserción en el funcionamiento de directorio.
- Otros atributos, que dependen de los valores del atributo **objectClass**.

Las subinserciones pueden contener también atributos operacionales con semántica apropiada (véase 12.6.4)

12.6.2 Atributo RDN de subinserción

El atributo **commonName** utilizado como el identificador de subárbol sirve para distinguir las diversas subinserciones que pueden ser definidas como subordinadas inmediatas de una inserción administrativa específica.

NOTA – El valor de este atributo podría elegirse de modo que sirviera de mnemónico a representantes de la autoridad administrativa.

El atributo **commonName** de una subinserción puede no contener múltiples valores distinguidos diferenciados por el contexto; únicamente se permite un solo valor distinguido.

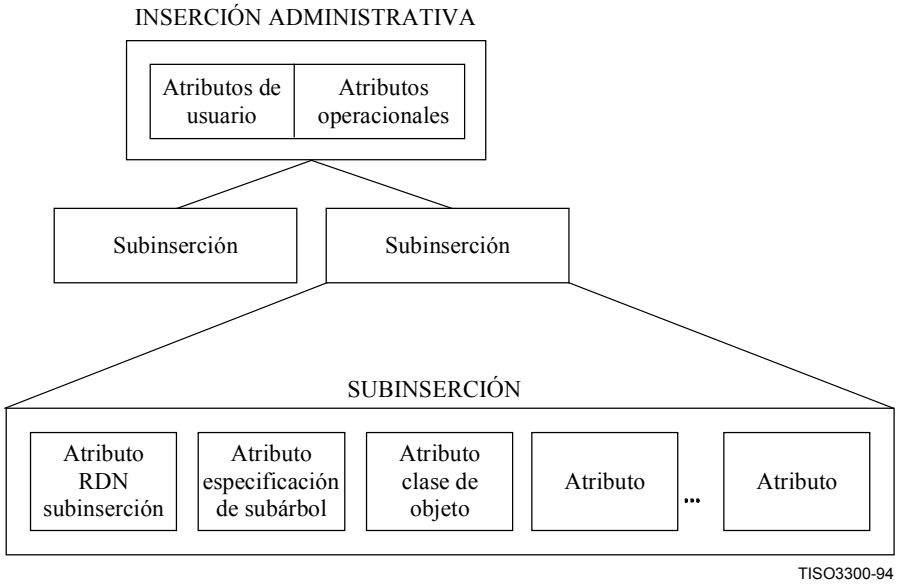


Figura 10 – Estructura de una subinserción

12.6.3 Atributo especificación de subárbol

El atributo **subtreeSpecification** define la colección de inserciones dentro de la zona administrativa con la que se relaciona el subárbol.

12.6.4 Utilización del atributo clase de objeto

El contenido de una subinserción está regido por los valores del atributo **objectClass** de la subinserción.

El atributo **objectClass** de todas las subinserciones contendrá el valor **subentry (subinserción)**. La clase de objeto **subentry** es una clase de objeto estructural, definida en la cláusula 14, y que se utiliza para incluir los atributos **commonName**, **subtreeSpecification** y **objectClass** en todas las subinserciones.

Para reglamentar los atributos restantes, se utilizarán otros valores del atributo **objectClass**, que representan las clases de objetos auxiliares permitidos para la subinserción.

La definición de la semántica de uno de estos valores incluye una identificación y especificación de ninguno o más tipos de atributo que aparecerán o pueden aparecer en la subinserción cuando el atributo **objectClass** toma el valor. La definición de la semántica de un valor para el atributo **objectClass** tiene que incluir también:

- una indicación de si una inserción puede ser incluida en más de un subárbol o refinamiento de subárbol asociado con el propósito particular (por ejemplo, puede que no se permita en el caso de **subschema** pero puede permitirse para control de acceso); y, si se permite
- los efectos de la combinación de atributos de subinserciones asociadas, si hubiere.

Una subinserción de una clase de objeto particular sólo puede estar subordinada a una inserción administrativa si el atributo de objeto **administrativeRole** permite esa clase de subinserción como subordinada.

Al igual que para las inserciones de objetos y de alias, la información mantenida en una subinserción puede ser complementada por información administrativa y operacional representada por atributos operacionales. Por ejemplo, una subinserción puede contener ACI de inserción, a condición solamente de que esta ACI esté permitida por el valor del atributo **accessControlScheme (esquema de control de acceso)** del punto específico de control de acceso correspondiente, y concuerde con dicho valor. De manera similar, una subinserción puede contener un **modifyTimestamp (indicación de tiempo de modificación)**.

12.6.5 Otros atributos de subinserción

Los atributos restantes en una subinserción dependen de los valores del atributo **objectClass**. Por ejemplo, un subesquema sólo puede colocarse en una subinserción si su atributo **objectClass** tiene **subschema (subesquema)** como uno de sus valores.

12.7 Modelo de información para atributos colectivos

Una zona administrativa autónoma puede ser designada como una zona administrativa específica de atributos colectivos para desplegar y administrar atributos colectivos. Esto se indicará mediante la presencia del valor **id-ar-collectiveAttributeSpecificArea** en el atributo **administrativeRole** de la inserción administrativa asociada [además de la presencia del valor **autonomousArea (zona autónoma)**, y posiblemente otros valores].

Esta zona administrativa autónoma puede ser dividida para desplegar y administrar atributos colectivos en las partes específicas. En este caso, las inserciones administrativas para cada una de las zonas administrativas específicas de atributos colectivos se indican mediante la presencia del valor **id-at-collectiveAttributeSpecificArea** en los atributos **administrativeRole** de estas inserciones.

Si una zona administrativa autónoma no está dividida, hay una sola zona administrativa específica para atributos colectivos que abarcan toda la zona administrativa autónoma.

Además, una zona administrativa específica definida a los efectos de la administración de atributos colectivos puede dividirse en zonas interiores jerarquizadas para el mismo fin. El atributo **administrativeRole** de las inserciones administrativas para cada una de estas zonas administrativas interiores indicará mediante esto la presencia del valor **id-ar-collectiveAttributeInnerArea**.

Una colección de inserciones y sus atributos colectivos asociados se representan en el modelo de información de directorio mediante una subinserción, denominada una *subinserción de atributos colectivos,* cuyo atributo **objectClass** tiene el valor **id-sc-collectiveAttributeSubentry** definido en la cláusula 14. Una subinserción de esta clase puede ser la subordinada inmediata de una inserción administrativa cuyo atributo **administrativeRole** contiene el valor **id-ar-collectiveAttributeSpecificArea** o **id-ar-collectiveAttributeInnerArea**.

Cuando hay diferentes colecciones de inserciones dentro de una zona de atributos colectivos dada, cada una tendrá su propia subinserción.

La colección de inserciones propiamente dicha está definida por el valor del atributo operacional **subtreeSpecification** de la subinserción. Este valor define el *alcance* de la subinserción de atributo colectivo. Los atributos de usuario de la subinserción son los atributos colectivos de la colección de inserción.

NOTA 1 – Puesto que el refinamiento de subárbol se basa en clase de objeto, la asociación de atributos colectivos con inserciones de objeto puede efectuarse de una manera que extienda naturalmente el esquema para estas inserciones. Por ejemplo, las inserciones **organizationalPerson** de una organización podrían ser ampliadas con un conjunto de atributos colectivos apropiados para todas las personas pertenecientes a la organización por la creación de una subinserción cuyo subárbol asociado se ha refinado para que incluya solamente inserciones **organizationalPerson** y que contenga el conjunto de atributos colectivos de la organización. Además, habría que definir una regla de contenido de DIT para esas inserciones con el fin de permitir que atributos colectivos sean visibles en las inserciones.

Los tipos de atributo colectivo y los tipos de atributo no colectivo difieren en lo tocante a la semántica. Un tipo de atributo capaz de expresar semántica colectiva se designará como tipo de atributo colectivo en el momento de su definición.

NOTA 2 – Los procedimientos de fusión empleados por el directorio en el caso de fuentes independientes de valores de un tipo de atributo colectivo se describen en la Rec. UIT-T X.511 | ISO/CEI 9594-3.

Mediante el atributo **collectiveExclusions**, definido en la cláusula 14 se puede impedir que aparezcan atributos colectivos en una inserción determinada.

12.8 Modelo de información para valores por defecto del contexto

Puede concebirse una zona administrativa autónoma como área administrativa específica por defecto del contexto para desplegar y administrar valores por defecto del contexto. Esto se indicará mediante la presencia del valor **id-ar-contextDefaultSpecificArea** en el atributo **administrativeRole** de la inserción administrativa asociada (junto con la presencia del valor **id-ar-autonomousArea** y, posiblemente, otros valores).

Puede dividirse esa zona administrativa autónoma para desplegar y administrar valores por defecto del contexto en las partes específicas. En este caso, las inserciones administrativas de cada una de las zonas específicas por defecto del contexto se indican mediante la presencia del valor **id-ar-contextDefaultSpecificArea** en los atributos **administrativeRole** de esas inserciones.

Cuando una zona administrativa autónoma no está dividida, hay una sola zona administrativa específica para los valores por defecto del contexto que abarcan la zona administrativa autónoma total.

En el modelo de información de directorio se representan los valores por defecto del contexto mediante una subinserción denominada *subinserción por defecto del contexto*, cuyo atributo **objectClass** tiene el valor **id-sc-contextAssertionSubentry**, como se define en 14.7. Una subinserción de este tipo debe ser la subordinada inmediata de una entidad administrativa cuyo atributo **administrativeRole** contiene el valor **id-ar-contextDefaultSpecificArea**.

La subinserción por defecto del contexto define un conjunto de aserciones de contexto cualquiera de las cuales se aplica cuando no haya aserciones de contextos aplicables a un tipo de atributo determinado especificado por el usuario cuando accede a la parte del DIT definida por el atributo operacional **subtreeSpecification** de la subinserción. La aplicación de las aserciones de contexto por defecto se describe en 8.8.2.2 y en 7.6.1 de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

SECCIÓN 6 – ESQUEMA DE DIRECTORIO

13 Esquema de directorio

13.1 Definiciones

A los fines de esta Especificación de directorio, se aplican las siguientes definiciones:

13.1.1 sintaxis de atributo: El tipo de datos ASN.1 utilizado para representar valores de un atributo.

13.1.2 esquema de directorio: El conjunto de reglas y limitaciones relativas a la estructura del DIT, al contenido del DIT, al uso del contexto del DIT, las clases de objeto y tipos de atributo, las sintaxis y las reglas de concordancia caracterizan la DIB. El esquema de directorio se manifiesta como un conjunto de subesquemas que no se superponen, cada uno de los cuales gobierna las inserciones de una zona administrativa autónoma (o una división de la misma específica de un subesquema). El esquema de directorio solo se relaciona con información del usuario de directorio.

13.1.3 subesquema (de directorio): El conjunto de reglas y constricciones concernientes a la estructura del DIT, el contenido del DIT, las clases de objeto y los tipos de atributo, las sintaxis, y reglas de concordancia que caracterizan las inserciones de la DIB dentro de una zona administrativa autónoma (o una división de la misma, específica de un esquema).

13.1.4 regla de contenido de árbol de información de directorio: Una regla que gobierna el contenido de inserciones de una determinada clase de objeto estructural, o alias. Especifica las clases de objeto auxiliares y el (los) tipo(s) de atributo adicional(es) cuya aparición está permitida, o prohibida, en inserciones de la clase de objeto estructural, o alias, indicada.

13.1.5 uso del contexto de árbol de información de directorio: Regla que gobierna los tipos de contexto que pueden estar asociados con valores de atributo de tipos de atributo determinados. Especifica los tipos de contextos permitidos y obligatorios para el tipo de atributo.

13.1.6 regla de estructura de árbol de información de directorio: Una regla que gobierna la estructura del DIT especificando una relación permitida de inserción superior a subordinado. Una regla de estructura relaciona una vinculación de nombre, y por tanto una clase de objeto estructural o alias, con reglas de estructura superior. Esto permite que inserciones de la clase de objeto estructural o alias identificadas por la vinculación de nombre existan en el DIT como subordinados de inserciones gobernados por las reglas de estructura superior indicadas.

13.1.7 regla de estructura gobernante (de una inserción): Con respecto a una determinada inserción, la regla de estructura DIT *singular* que se aplica a la inserción. Esta regla se indica por el atributo operacional **governingStructureRule**.

13.1.8 forma de nombre: Una forma de nombre especifica un RDN admisible para inserciones de una determinada clase de objeto estructural o alias. Identifica una clase de objeto denominado o uno o más tipos de atributo que serán utilizados para denominación (por ejemplo para el RDN). Las formas de nombre son piezas primitivas de especificación utilizadas en la definición de reglas de estructura del DIT.

NOTA – Las formas de nombre están registradas y tienen alcance global. Las reglas de estructura DIT no están registradas y tienen el alcance de la zona administrativa con la que están asociadas.

13.1.9 regla de estructura superior: Con respecto a una inserción dada, la regla de estructura DIT que gobierna al superior de la inserción.

13.2 Visión de conjunto

El esquema de directorio es un conjunto de definiciones y constricciones concernientes a la estructura del DIT, las maneras posibles de denominar las inserciones, la información que puede estar contenida en una inserción, los atributos utilizados para representar esa información y su organización en jerarquías para facilitar la búsqueda y extracción de la información, y las formas en las cuales valores de atributos pueden ser concordados en aserciones de valor de atributo y de reglas de concordancia.

NOTA 1 – El esquema permite al sistema de directorio, por ejemplo:

- evitar la creación de inserciones subordinadas de una clase de objeto incorrecta (por ejemplo, un país como subordinado de una persona);
- evitar la adición de tipos de atributo a una inserción inapropiada para la clase de objeto (por ejemplo, un número de serie a una inserción de una persona);
- evitar la adición de un valor de atributo a una sintaxis que no concuerda con la definida para el tipo de atributo (por ejemplo, una cadena imprimible a una cadena de bits).

Desde el punto de vista formal, el esquema de directorio comprende un conjunto de:

a) definiciones de *forma de nombre* que definen relaciones de denominación primitivas para clases de objeto estructurales;

b) definiciones de *regla de estructura del DIT* que definen los nombres que las inserciones pueden tener y las maneras de relacionar las inserciones, entre sí, en el DIT;

c) definiciones de *regla de contenido del DIT* que extienden la especificación de atributos admisibles para inserciones más allá de los indicados por la(s) clase(s) de objeto estructurales de las inserciones;

d) definiciones de *clase de objeto* que definen el conjunto básico de atributos obligatorios y facultativos que están o pueden estar presentes, respectivamente, en una inserción de una clase dada y que indican el tipo de clase de objeto que se está definiendo (véase 7.3);

e) definiciones de *tipo de atributo* que determinan el identificador de objeto por el cual se conoce un atributo, su sintaxis, reglas de concordancia asociadas, si se trata de un atributo operacional, y de ser así su tipo, si se trata de un atributo colectivo, si se permite tener múltiples valores, y si ha sido derivado o no de otro tipo de atributo;

f) definiciones de *regla de concordancia* que definen reglas de concordancia;

g) definiciones de *uso del contexto DIT* que gobiernan los tipos de contexto que pueden asociarse con valores de atributo de cualquier tipo de atributo determinado.

La figura 11 ilustra las relaciones entre definiciones de esquema y de subesquema por un lado, y el DIT, e inserciones, atributos, y valores de atributo de directorio, por otro lado.

Interpretación de la figura 11:

– los ítems listados verticalmente a la izquierda representan elementos de esquema;

– los ítems listados verticalmente a la derecha representan ejemplares de esquemas correspondientes;

– la relación entre ítems de esquema se ilustra por la relación "usa";

– la relación entre diferentes ejemplares de esquema se ilustra por la relación "pertenece a".

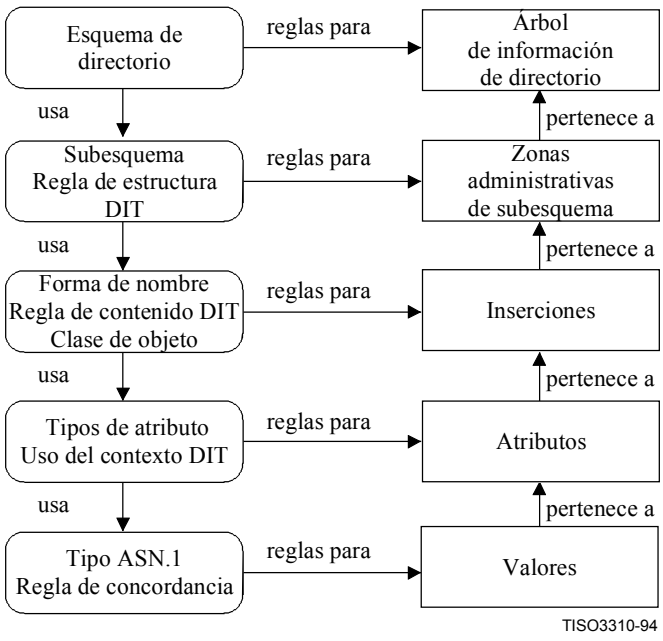


Figura 11 – Visión de conjunto de directorio

El esquema de directorio es distribuido, como lo es la propia DIB. Se manifiesta como un conjunto de subesquemas que no se superponen cada uno de los cuales gobierna inserciones de una zona administrativa autónoma (o una parte de ella específica del subesquema). Una autoridad administrativa de subesquema establece las reglas y constricciones que constituyen el subesquema.

La autoridad administrativa de subesquema puede decidir utilizar elementos individuales del esquema de directorio que tienen alcance global y que están definidos en estas Especificaciones de directorio: formas de nombre, clases de objeto y atributos (tipos, sintaxis, reglas de concordancia). Puede también optar por definir alternativas a estos elementos que sean más apropiadas para su propio entorno, u optar por una solución intermedia basada en la utilización de elementos de esquema tanto normalizados como de propiedad privada.

La autoridad administrativa del subesquema define los elementos del esquema cuyo alcance está limitado al subesquema: reglas de estructura del DIT, reglas del contenido del DIT y uso del contexto del DIT. Además, la autoridad administrativa del subesquema puede también especificar las reglas de concordancia aplicables a cada tipo de atributo.

El esquema de directorio se ocupa solamente de información de usuario de directorio. Aunque en la notación definida en esta cláusula se proporciona algún soporte para la especificación de información operacional, la reglamentación de la información administrativa y operacional de directorio es tarea del *esquema de sistema de directorio*.

NOTA 2 – El esquema de sistema de directorio se describe en la cláusula 14.

13.3 Definición de clase de objeto

La definición de una clase de objeto comprende:

a) indicación de las clases de las cuales esta clase de objeto será una subclase;

b) indicación de la modalidad de clase de objeto que se está definiendo;

c) listado de los tipos de atributo *obligatorios* que deberá contener una inserción de la clase de objeto, además de los tipos de atributo obligatorios de todas sus superclases;

d) listado de los tipos de atributo *facultativos* que una inserción de la clase de objeto en cuestión puede contener además de los atributos facultativos de todas sus superclases.

e) asignación de un identificador de objeto para la clase de objeto.

NOTA – Los atributos colectivos no aparecerán en los tipos de atributo de una definición de clase de objeto.

13.3.1 Subclases

Las subclases están sujetas a diversas restricciones, a saber:

– sólo clases de objeto abstractas serán superclases de otras clases de objeto abstractas.

Hay una sola clase de objeto especial, de la cual toda clase de objeto estructural es una subclase. Esta clase de objeto se denomina **top**. **top** es una clase de objeto abstracta.

13.3.2 El atributo clase de objeto

Toda inserción contendrá un atributo de tipo **objectClass** para identificar la(s) clase(s) de objeto y las superclases a que pertenece. La definición de este atributo se da en 13.4.7. Este atributo es multivaluado.

Habrá un valor del atributo **objectClass** para la clase de objeto estructural de la inserción y un valor para cada una de sus superclases. **top** puede ser omitido.

La(s) clase(s) de objeto estructural de una inserción no deberá(n) cambiarse. Los valores iniciales del atributo **objectclass** son proporcionados por el usuario cuando la inserción es creada.

Cuando se utilicen clases de objeto auxiliares, una inserción puede contener un valor (o valores) del atributo **objectClass** para la(s) clase(s) de objeto auxiliar(es) y sus superclases autorizadas por una regla de contenido de DIT. Si está presente un valor para una clase de objeto auxiliar autorizada, estarán presentes también valores para las superclases de la clase de objeto auxiliar.

Cuando el atributo **objectClass** contiene un valor de identificador de objeto para una clase de objeto auxiliar, la inserción tendrá siempre que contener los atributos obligatorios indicados por esa clase de objeto.

NOTA 1 – La exigencia de que el atributo **objectClass** esté presente en todas las inserciones se refleja en la definición de **top**.

NOTA 2 – Puesto que se considera que una clase de objeto pertenece a todas sus superclases, cada miembro de la cadena de superclases hasta **top** se representa por un valor en el atributo **objectClass** (y la concordancia de cualquier valor en la cadena puede ser establecida por un filtro).

NOTA 3 – Se pueden imponer restricciones de control de acceso a la modificación del atributo **objectClass**.

Junto con las reglas de contenido de DIT aplicables, el directorio hace cumplir, para todos las inserciones en la DIB, la clase de objeto definida. Todo intento de modificar una inserción que viole la definición de la clase de objeto de dicha inserción fracasará a menos que ello esté explícitamente autorizado por la regla de contenido DIT de la inserción.

NOTA 4 – En particular, el directorio generalmente impedirá que:

a) tipos de atributo que estén ausentes de una definición de clase de objeto estructural de la inserción y que no estén permitidos por la regla de contenido DIT de la inserción sean añadidos a una inserción de esa clase de objeto;

b) se cree una inserción de la cual estén ausentes uno o más tipos de atributo que son obligatorios para una clase de objeto de la inserción;

c) se suprima un tipo de atributo que es obligatorio para la clase de objeto de la inserción.

13.3.3 Especificación de clase de objeto

Las clases de objeto pueden definirse como valores de la clase de objeto de información **OBJECT-CLASS**:

```
OBJECT-CLASS  ::=    CLASS {
      &Superclasses          OBJECT-CLASS OPTIONAL,
      &kind                  ObjectClassKind DEFAULT structural,
      &MandatoryAttributes   ATTRIBUTE OPTIONAL,
      &OptionalAttributes    ATTRIBUTE OPTIONAL,
      &id                    OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
      [ SUBCLASS OF          &Superclasses ]
      [ KIND                 &kind ]
      [ MUST CONTAIN         &MandatoryAttributes ]
      [ MAY CONTAIN          &OptionalAttributes ]
      ID                     &id }

ObjectClassKind  ::=  ENUMERATED {
      abstract     (0),
      structural   (1),
      auxiliary    (2) }
```

Para una clase de objeto que se define utilizando esta clase de objeto de información:

a) **&Superclasses** es el conjunto de clases de objeto que son sus superclases directas;

b) **&kind** es su genero;

c) **&MandatoryAttributes** es el conjunto de atributos que contendrán las inserciones de esa clase;

d) **&OptionalAttributes** es el conjunto de atributos que deben ser contenidos por inserciones de esa clase, con la excepción de que si aparece un atributo en ambas categorías de conjuntos, es decir, obligatorio y opcional, se considerará obligatorio;

e) **&id** es el identificador de objeto que le ha sido asignado.

A continuación se definen las clases de objeto mencionados anteriormente (**top** y **alias**):

```
top  OBJECT-CLASS  ::=  {
      KIND               abstract
      MUST CONTAIN       { objectClass }
      ID                 id-oc-top }

alias  OBJECT-CLASS  ::=  {
      SUBCLASS OF        { top }
      MUST CONTAIN       { aliasedEntryName }
      ID                 id-oc-alias }
```

NOTA 1 – La clase de objeto **alias** no especifica tipos de atributos apropiados para el RDN de una inserción de alias. Las autoridades administrativas pueden especificar subclases de la clase **alias** que especifican tipos de atributos útiles para los RDN de inserciones de alias.

```
parent  OBJECT-CLASS  ::=  {
      KIND               abstract
      ID                 id-oc-parent }

child  OBJECT-CLASS  ::=  {
      KIND               auxiliary
      ID                 id-oc-child }
```

Las clases de objeto **parent (progenitor)** y **child (vástago)** no se combinarán con la clase de objeto **alias** para formar una inserción de alias.

La clase de objeto **parent** se deriva por la presencia de un miembro de familia subordinado inmediato, marcado por la presencia de un valor de la clase de objeto **child**. Puede no ser administrada directamente. El valor de la clase de objeto **child** sólo puede ser añadido o eliminado cuando el resultado sea coherente con la arquitectura de las inserciones compuestas (por ejemplo, los subordinados de los miembros de la familia deberán tener siempre una clase de objeto **child**).

NOTA 2 – Las clases de objeto **parent** y **child** no especifican ningún tipo de atributo apropiado para los RDN de los miembros de la familia. Esto se hará de forma normal por medio de las clases de objeto estructurales y formas de nombre apropiadas para esas inserciones.

13.4 Definición de tipo de atributo

Un atributo puede ser un atributo de usuario o un atributo operacional:

a) La indicación facultativa de que el tipo de atributo es un subtipo de un tipo de atributo definido previamente, su supertipo directo;

b) la especificación de la sintaxis de atributo para el tipo de atributo;

c) la indicación facultativa de la regla o reglas de concordancia de igualdad, ordenación y/o de subcadenas para el tipo de atributo;

d) la indicación de si un atributo de este tipo sólo tendrá un valor o puede tener más de uno;

e) la indicación de si el tipo de atributo es operacional o de usuario;

f) la indicación facultativa de si un tipo de atributo de usuario es colectivo;

g) la indicación facultativa de que un atributo operacional no es modificable por el usuario;

h) la indicación de la aplicación para los atributos operacionales;

i) la asignación de un identificador de objeto al tipo de atributo.

13.4.1 Atributos operacionales

Algunos atributos operacionales están bajo un control de usuario directo. En otros casos, el valor o valores del atributo operacional son controlados por el directorio. En este último caso, la definición del atributo operacional indicará que no se permiten modificaciones, por el usuario, de los valores de atributo.

La especificación de un tipo de atributo operacional indicará su aplicación, que será una de las siguientes:

– atributo operacional de directorio (por ejemplo, atributo de control de acceso);

– atributo operacional compartido por DSA (por ejemplo, atributo de punto de acceso maestro);

– atributo operacional específico de DSA (por ejemplo, un atributo de estado de copia).

13.4.2 Jerarquías de atributo

Una jerarquía de atributo contendrá atributos de usuario, o atributos operacionales, pero no ambos. De aquí que un atributo de usuario no será derivado de un atributo operacional y que un atributo operacional no será derivado de un atributo de usuario.

Un atributo operacional que es un subtipo de otro atributo operacional tendrá la misma aplicación que su supertipo.

Si un tipo de atributo no es un subtipo de otro tipo de atributo, la sintaxis de atributo y las reglas de concordancia (si son aplicables) deberán especificarse en la definición de tipo de atributo. La especificación de una sintaxis de atributo se hará especificando directamente el tipo de datos ASN.1.

Si un tipo de atributo es un subtipo de un tipo indicado, la definición no tiene que especificar una sintaxis de atributo, en cuyo caso su sintaxis de atributo es la de su supertipo directo. Si la sintaxis de atributo está indicada y el atributo tiene un supertipo directo, la sintaxis indicada será compatible con la sintaxis del supertipo, esto es, todos los valores posibles que satisfagan la sintaxis del atributo satisfarán también la sintaxis del supertipo.

Si un tipo de atributo es un subtipo de otro tipo de atributo, las reglas de concordancia aplicables al supertipo son aplicables al subtipo, a menos que hayan sido extendidas o modificadas en la definición del subtipo. Una regla de concordancia definida para un supertipo no puede ser suprimida cuando se define un subtipo.

13.4.3 Atributos colectivos

Un atributo operacional no será definido como colectivo.

Un atributo de usuario puede ser definido como colectivo. Con esto se indica que los mismos valores de atributo aparecerán en las inserciones de una colección de inserciones con arreglo a la utilización del atributo **collectiveExclusions**.

Los atributos colectivos deberán ser multivaluados.

13.4.4 Atributos derivados

Un atributo derivado es un atributo que contiene información en la que se utiliza la sintaxis de información de atributos, pero en donde los valores se computan como devueltos y no como prealmacenados.

El atributo derivado **family-information** se introduce para su utilización en el servicio de directorio a efectos de contenencia de la información de la familia. Sus características se definen en 7.7.1 de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

Los DSA pueden utilizar también tecnología del atributo derivado para proporcionar otros atributos. Por ejemplo, todos los atributos operacionales que incluyen el valor **AccessPoint** de un DSA específico pueden (y probablemente deban) derivar el valor de una sola fuente de información, que pueda ser administrada convenientemente.

13.4.5 Sintaxis de atributo

Si se especifica una regla de concordancia de igualdad para el tipo de atributo, el directorio asegurará que se utilice la sintaxis correcta para cada valor de este tipo de atributo.

13.4.6 Reglas de concordancia

En la definición de tipo de atributo pueden indicarse reglas de concordancia de igualdad, ordenamiento, y subcadenas. La misma regla de concordancia puede utilizarse para uno o más de estos tipos de concordancias si la semántica de la regla permite más de uno de estos tipos de concordancias.

NOTA 1 – Este hecho deberá ser reflejado en la regla de concordancia indicada.

Si no se indica una regla de concordancia de igualdad, el directorio:

a) trata los valores de este atributo como si fuesen del tipo **ANY**, es decir, el directorio no puede verificar que esos valores son conformes al tipo de datos o a cualquier otra regla de concordancia indicada para el atributo;

b) no permite que el atributo sea utilizado para denominación;

c) no autoriza que se añadan o supriman valores individuales de atributos multivaluados;

d) no efectúa comparaciones de valores del atributo;

e) no intentará evaluar las **AVA** utilizando valores de tal tipo de atributo.

Si se indica una regla de concordancia de igualdad, el directorio:

a) trata los valores de este atributo como si tuviesen el tipo definido en el campo **&Type** en la definición del atributo (o el del atributo del cual se deriva el atributo);

b) utilizará la regla de concordancia de igualdad indicada para evaluar las aserciones de valor de atributo concernientes al atributo;

c) sólo concordará un valor presentado de un tipo de datos adecuado especificado en la definición de tipo de atributo.

NOTA 2 – Esta subcláusula se aplica asimismo a un atributo cuya regla de concordancia de igualdad utiliza una sintaxis de aserción diferente de la sintaxis del tipo de atributo.

Si no se indica una regla de concordancia de ordenamiento, el directorio tratará cualquier aserción de una concordancia de ordenamiento utilizando la sintaxis proporcionada como indefinida por el servicio abstracto de directorio.

Si no se indica una regla de concordancia de subcadenas, el directorio tratará cualquier aserción de una concordancia de subcadena utilizando la sintaxis proporcionada como indefinida por el servicio abstracto de directorio.

Un tipo de atributo sólo especificará reglas de concordancia cuya definición se aplique a la sintaxis de atributo del atributo.

13.4.7 Especificación de atributo

Los atributos pueden definirse como valores de la clase de objeto de información **ATTRIBUTE (ATRIBUTO)**:

```
ATTRIBUTE ::= CLASS {
        &derivation                     ATTRIBUTE OPTIONAL,
        &Type                           OPTIONAL,       -- either &Type or &derivation required --
        &equality-match                 MATCHING-RULE OPTIONAL,
        &ordering-match                 MATCHING-RULE OPTIONAL,
        &substrings-match               MATCHING-RULE OPTIONAL,
        &single-valued                  BOOLEAN DEFAULT FALSE,
        &collective                     BOOLEAN DEFAULT FALSE,
        -- operational extensions --
        &no-user-modification           BOOLEAN DEFAULT FALSE,
        &usage                          AttributeUsage DEFAULT userApplications,
        &id                             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        [ SUBTYPE OF                    &derivation ]
        [ WITH SYNTAX                   &Type ]
        [ EQUALITY MATCHING RULE        &equality-match ]
        [ ORDERING MATCHING RULE        &ordering-match ]
        [ SUBSTRINGS MATCHING RULE      &substrings-match ]
        [ SINGLE VALUE                  &single-valued ]
        [ COLLECTIVE                    &collective ]
        [ NO USER MODIFICATION          &no-user-modification ]
        [ USAGE                         &usage ]
        ID                              &id }

AttributeUsage ::= ENUMERATED {
        userApplications        (0),
        directoryOperation      (1),
        distributedOperation    (2),
        dSAOperation            (3) }
```

Para un atributo que se define utilizando esta clase de objeto de información:

a) **&derivation** es el atributo, si lo hubiere, del cual él es un subtipo;

b) **&Type** es su sintaxis de atributo. Será un tipo ASN.1, pero no un tipo que contenga un **EmbeddedPDV**;

c) **&equality-match** es su regla de concordancia de igualdad (si la hubiere);

d) **&ordering-match** es su regla de concordancia de ordenación (si la hubiere);

e) **&substrings-match** es su regla de concordancia de subcadenas (si la hubiere);

f) **&single-valued** es VERDADERO si es univaluado, y falso si no lo es;

g) **&collective** es VERDADERO si es un atributo colectivo, y falso si no lo es;

h) **&no-user-modification** es VERDADERO si es un atributo operacional que no puede ser modificado por el usuario;

i) **&usage** indica el uso operacional del atributo. **userApplications** significa que se trata de un atributo de usuario, **directoryOperation**, **distributedOperation**, y **dSAOperation** significan que se trata de un atributo de directorio distribuido u operacional de DSA, respectivamente;

j) **&id** es el identificador de objeto que le ha sido asignado.

Los tipos de atributo definidos en la edición de 1988 de esta Especificación de directorio que son conocidos y utilizados por el directorio para sus propios fines se definen como sigue:

```
objectClass ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        ID                              id-at-objectClass }

aliasedEntryName ATTRIBUTE ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        ID                              id-at-aliasedEntryName }
```

NOTA – Las reglas de concordancia mencionadas en estas definiciones se definen en 13.5.2.

Los atributos **objectClass** y **aliasedEntryName** se definen como atributos de usuario aunque sólo se utilizan para operaciones de directorio y deben definirse semánticamente como operacionales. Esto se debe a que estos atributos se definieron como atributos de usuario antes del concepto de atributo operacional y deben permanecer como atributos de usuario para facilitar el interfuncionamiento entre sistemas que aplican diferentes ediciones de esta Especificación de directorio.

13.5 Definición de reglas de concordancia

13.5.1 Visión de conjunto

La definición de una regla de concordancia comprende:

a) la definición facultativa de las reglas de concordancia de progenitor de las que puede derivarse la regla de concordancia presente;

b) la definición de la sintaxis de una aserción de la regla de concordancia;

c) la especificación de diferentes tipos de concordancias soportados por la regla;

d) la definición de reglas apropiadas para evaluar una aserción presentada con respecto a valores de atributos fijados como objetivo en la DIB;

e) la asignación de un identificador de objeto a la regla de concordancia.

Una regla de concordancia debe utilizarse para evaluar aserciones de valor de atributo que indican la regla como su regla de concordancia de igualdad. La sintaxis utilizada en la aserción de valor de atributo (es decir, el componente **assertion** de la aserción de valor de atributo) es la sintaxis de aserción de la regla de concordancia.

Una regla de concordancia puede ser aplicable a muchos tipos diferentes de atributos con sintaxis diferentes de atributo.

La definición de una regla de concordancia incluirá una especificación de la sintaxis de una aserción de la regla de concordancia y la manera de utilizar valores de esta sintaxis para efectuar una concordancia. Esto no requiere una especificación completa de la sintaxis de atributo a la que se puede aplicar la regla de concordancia. Una definición de una regla de concordancia para uso con atributos con sintaxis ASN.1 diferentes especificará cómo deben efectuarse las concordancias.

La aplicabilidad de reglas de concordancia definidas de los atributos contenidos en una especificación de subesquema (por encima de las reglas de concordancia utilizadas en la definición de estos tipos de atributo) se indica mediante el atributo operacional de especificación de subesquema **matchingRuleUse**, definido en 15.7.7.

13.5.2 Especificación de reglas de concordancia

Las reglas de concordancia pueden definirse como valores de la clase de objeto de información **MATCHING-RULE**:

```
MATCHING-RULE ::= CLASS {
    &ParentMatchingRules        MATCHING-RULE       OPTIONAL,
    &AssertionType                                  OPTIONAL,
    &uniqueMatchIndicator       ATTRIBUTE           OPTIONAL,
    &id                         OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
    [ PARENT                    &ParentMatchingRules ]
    [ SYNTAX                    &AssertionType ]
    [ UNIQUE-MATCH-INDICATOR    &uniqueMatchIndicator ]
    ID                          &id }
```

Para una regla de concordancia que se define utilizando esta clase de objeto de información:

a) Se utiliza el campo **&ParentMatchingRules** si la regla de concordancia definida combina las características de otras dos o más reglas de concordancia. Se da como un conjunto de uno o más identificadores de objeto para las reglas de concordancia que suministran las características básicas de la regla de concordancia que se define (por ejemplo, el algoritmo de concordancia); se omite para una regla de concordancia básica.

b) **&AssertionType** es la sintaxis para una aserción que utiliza esta regla de concordancia; si se ha omitido, las sintaxis de aserción es la misma sintaxis utilizada para el atributo al que se aplica la regla, a no ser que la regla de concordancia especifique otra cosa. Si está presente, puede especificar una restricción impuesta a las reglas de concordancia del progenitor, pero en este caso será compatible con la sintaxis para las reglas de concordancia del progenitor (es decir, un valor que cumpla con **&AssertionType** debe cumplir también con **&AssertionType** para las reglas de concordancia del progenitor);

c) **&UniqueMatchIndicator** es un tipo de atributo de notificación. Cuando está presente, se requiere concordancia única. Para una regla de concordancia basada en el establecimiento de la correspondencia (véase 13.6), esa contraposición con el cuadro de correspondencias produce un resultado inequívoco. Si hay múltiples concordancias con relación al cuadro de correspondencias, se rechaza la petición de búsqueda con un **serviceError** con problema **ambiguousKeyAtrributes**. Además, es decir, se provoca el fallo de la búsqueda si la regla de concordancia asocia múltiples valores se colocará un atributo de notificación del tipo especificado por este campo en **CommonResults** del error devuelto.

NOTA 1 – Esa situación puede producirse en una concordancia geográfica cuando, por ejemplo, una aserción especifique "Newton" como una localidad del Reino Unido; hay numerosas localidades con este nombre, que han de ser distinguidas mediante un calificador (por ejemplo, "Newton, Cambs").

d) **&id** es el identificador de objeto asignado a la regla de concordancia.

Si se utilizan dos o más reglas de concordancia para **ParentMatchingRules**, el resultado es una regla de concordancia combinada que devuelve un resultado, para valores compatibles con **AssertionType**, según lo prescrito por las siguientes reglas:

a) si el resultado de cualquier regla de concordancia de progenitor es VERDADERO, la regla de concordancia combinada devolverá VERDADERO;

b) de otro modo, si el resultado de cualquier regla de concordancia de progenitor es FALSO, la regla de concordancia combinada devolverá FALSO; o

c) de no ser así, la regla de concordancia combinada devolverá indefinido.

El cuadro que sigue muestra la manera de combinar dos reglas de concordancia A y B; el cuadro podría en principio ampliarse a múltiples dimensiones, con patrones de resultados similares, para abarcar el caso de tres o más reglas de concordancia de progenitor:

		Regla A		
		VERDADERO	FALSO	INDEFINIDO
Regla B	VERDADERO	VERDADERO	VERDADERO	VERDADERO
	FALSO	VERDADERO	FALSO	FALSO
	INDEFINIDO	VERDADERO	FALSO	INDEFINIDO

Combinando las reglas de concordancia, como se especifica más arriba, es posible obtener una concordancia válida en casos en los que de otro modo fallaría.

NOTA 2 – Un caso específico de utilización de regla de concordancia de progenitor es la combinación de una regla de concordancia cualquiera con la regla de concordancia especial **ignoreIfAbsentMatch**. Esta última hace que un elemento de filtro devuelva VERDADERO si el atributo está ausente; si está presente, se aplican las reglas normales. De esta manera, un filtro de búsqueda puede investigar si están ausentes inserciones de algunos atributos especificados en el filtro **search**. Véase 6.7.1 de la Rec. UIT-T X.520 | ISO/CEI 9594-6.

La regla de concordancia **objectIdentifierMatch** se define como sigue:

```
objectIdentifierMatch  MATCHING-RULE  ::= {
        SYNTAX      OBJECT IDENTIFIER
        ID          id-mr-objectIdentifierMatch }
```

Un valor presentado de tipo identificador de objeto concuerda con un valor fijado como objetivo de identificador de tipo de objeto solamente si ambos tienen el mismo número de componentes integrales y cada componente integral del primero es igual al correspondiente componente del segundo. Esta regla de concordancia es inherente a la definición del identificador de objeto del tipo ASN.1. **objectIdentifierMatch** es una regla de concordancia de igualdad.

distinguishedNameMatch se define como sigue**:**

```
distinguishedNameMatch  MATCHING-RULE  ::= {
        SYNTAX     DistinguishedName
        ID         id-mr-distinguishedNameMatch }
```

Un valor de nombre distinguido presentado concuerda con un valor de nombre distinguido pretendido solamente si todo lo que sigue es verdadero:

a) ambos tienen el mismo número de los RDN;

b) los RDN correspondientes tienen el mismo número de **AttributeTypeAndValue**;

c) los **AttributeTypeAndValue** correspondientes (es decir, los que estén en los RDN correspondientes y tengan tipos de atributo idénticos) tienen valores de atributo concordantes, como se describe en 9.4.

distinguishedNameMatch es una regla de concordancia de igualdad.

13.6 Relajaciones y restricciones

Relajación y *restricción* son funciones que modifican de manera sistemática la concordancia de uno o más elementos de filtro. Si se efectúa una relajación, la modificación de la concordancia se produce de manera que aumenta la probabilidad de obtener más inserciones concordantes. La relajación se lleva a cabo cuando el número de inserciones concordantes está por debajo de un cierto mínimo. La restricción se efectúa de manera similar cuando el número de inserciones concordantes está por encima de un máximo determinado. Hay dos modos de relajación/restricción:

a) la regla de concordancia aplicada para un tipo de atributo particular puede ser reemplazada por una sustitución de regla de concordancia paso a paso hasta que se logre el efecto requerido o se hayan agotado las posibilidades, como se expone con detalle en 13.6.1; y

b) la relajación/restricción puede ser aplicada como parte de una *concordancia basada en el establecimiento de la correspondencia*, como se expone con detalle en 13.6.2.

13.6.1 Sustitución de regla de concordancia

La sustitución de la regla de concordancia se puede controlar mediante una regla de búsqueda directora dentro de una zona administrativa específica de servicio (véase 16.10.7). También la puede controlar el usuario en la petición **search** (véase 10.2.1 de la Rec. UIT-T X.511 | ISO/CEI 9594-3). En ambos casos, la sustitución se controla utilizando los constructivos **RelaxationPolicy** definidos en 16.10.

La relajación/restricción mediante la sustitución de la regla de concordancia modifica la acción de un filtro reemplazando sistemáticamente las reglas de concordancia aplicables con anterioridad para atributos seleccionados con reglas de concordancia que proporcionan una concordancia menos rigurosa (o más rigurosa). Una vez relajado, o restringido, por la sustitución de la regla de concordancia, el proceso de búsqueda se reevalúa en su totalidad en el mismo conjunto de inserciones dentro del alcance de la búsqueda. La reevaluación puede continuar hasta que no existan más relajaciones, o hasta que se efectúe una devolución satisfactoria (inferior o igual a **maximum**, o superior a **minimum**, por referencia a los elementos **RelaxationPolicy** controladores).

El resultado es que el filtro sigue siendo el mismo para cada reevaluación, pero cada una de las reglas de concordancia utilizadas para evaluar el filtro experimentan la sustitución que haga falta (véase la figura 12). Se puede evaluar DSA por DSA, aplicando una relajación no coordinada entre los DSA, o bien se puede utilizar, de manera alternativa, el componente **chainedRelaxation** de **ChainingArguments** para definir la relajación que se ha de aplicar.

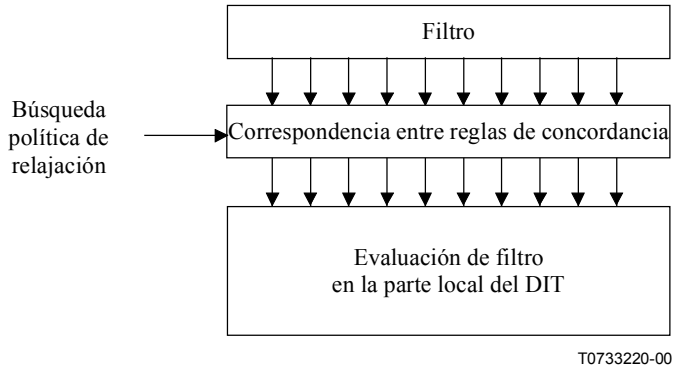


Figura 12 – Sustitución de regla de concordancia

Cuando hay que aplicar una política de relajación, el DSA, antes de efectuar la búsqueda local, realiza una *sustitución básica* para cada tipo de atributo para el que se define una sustitución básica, según lo especificado por la política de relajación.

NOTA 1 – Una aplicación particularmente útil de la sustitución básica, por ejemplo para el tipo de atributo **localityName,** consiste en sustituir la regla de concordancia **caseIgnoreSubstringMatch** por la regla de concordancia **generalWordMatch** en aquellas situaciones en las que esta regla de concordancia es más apropiada y lo previsto es que el usuario formule en consecuencia un elemento de filtro **substrings**.

Si la búsqueda, aplicada a este DSA particular, genera demasiado pocas inserciones, se aplica la primera política de relajación; si el resultado es todavía demasiado pocas inserciones, se aplica la siguiente política de relajación; y así sucesivamente.

De manera similar, si la búsqueda genera demasiadas inserciones, se aplica la primera política de restricción, si es necesario, la segunda, etc. No es posible pasar de una restricción a una relajación, o viceversa.

La relajación aplicada por un conjunto de **MRSubstitution** para un atributo particular se mantiene hasta que sea revocada por otra **MRMapping**. La revocación puede ser explícita especificando la regla de concordancia, o implícita omitiendo el identificador **oldMatchingRule**.

Si se lleva a cabo una evaluación relajada porque la evaluación previa ha generado demasiado pocos resultados, y la evaluación relajada devuelve demasiados resultados, se devolverán algunos o la totalidad de los resultados de la evaluación relajada. Si se lleva a cabo una evaluación endurecida porque la evaluación previa produjo demasiados resultados, y la evaluación endurecida devuelve demasiados resultados, algunos o la totalidad de los resultados de la evaluación previa deberán ser devueltos. En cualquier caso, el proceso de relajación o restricción se detiene.

La política de relajación conveniente se aplica a **filter** o **extendedFilter**, según proceda.

NOTA 2 – Puesto que la relajación permite efectuar evaluaciones más o menos rigurosas de los elementos de filtro en el caso de un filtro *ordinario,* se reduce la necesidad de conseguir un filtrado más complejo con filtros ampliados.

Un DSA puede suministrar el atributo **proposedRelaxation** (véase 5.12.15 de la Rec. UIT-T X.520 | ISO/CEI 9594-6) en el resultado de una **search** dentro del subcomponente **notification** del **PartialOutcomeQualifier**. La información ahí contenida se puede utilizar a continuación en una petición **search** a modo de política de relajación proporcionada por el usuario.

Como caso último de relajación, una política puede hacer que un elemento de filtro determinado sea evaluado a VERDADERO (o FALSO si se niega el elemento de filtro) de acuerdo con la regla de concordancia **nullMatch**.

Dentro de una zona administrativa específica de servicio, la validación con relación a las reglas de búsqueda se efectúa una vez que se hayan llevado a cabo las posibles sustituciones básicas, según lo especificado por la regla de búsqueda con relación a la cual se evalúa la petición **search**. Se selecciona una regla de búsqueda directora antes de cualquier sustitución de regla de concordancia subsiguiente, incluyendo posibles sustituciones básicas especificadas en la petición **search**.

13.6.2 Concordancia basada en el establecimiento de la correspondencia

La concordancia basada en el establecimiento de la correspondencia es pertinente en el caso de la operación de búsqueda, cuando la concepción del mundo real por parte de los usuarios difiere de varias maneras del modelo idealizado utilizado a menudo por el directorio. Por ejemplo, las nociones de los usuarios respecto a los nombres de las localidades y la forma en que las localidades se relacionan entre sí pueden ser muy distintas de cómo se representan las mismas en el directorio. Para reducir esa diferencia y aumentar la proporción de búsquedas exitosas, es fundamental establecer la correspondencia entre el concepto que tienen los usuarios de algunos objetos del mundo real, incluidas sus relaciones mutuas, y el modelo del directorio para los mismos objetos. Esa misma correspondencia deberá hacer posible además una especie de concordancia "difusa", es decir, deberá permitir que algunos valores de atributos reflejen más de lo que indica su definición estricta.

NOTA 1 – Por ejemplo, un usuario puede especificar el nombre de una localidad en el filtro, pero el objeto buscado puede estar próximo a la frontera en una localidad vecina.

La concordancia basada en el establecimiento de la correspondencia se puede aplicar a aspectos geográficos de las búsquedas en páginas blancas, aspectos de tipo empresarial de las búsquedas en páginas amarillas, etc.

La concordancia basada en el establecimiento de la correspondencia emplea algún tipo de cuadro intermedio, un *cuadro de correspondencias*, para controlar ese proceso. El comportamiento exacto de una concordancia basada en el establecimiento de la correspondencia y la estructura del cuadro de correspondencias son asuntos locales. De todos modos, el principio básico de la técnica es común, como se ilustra en la figura 13.

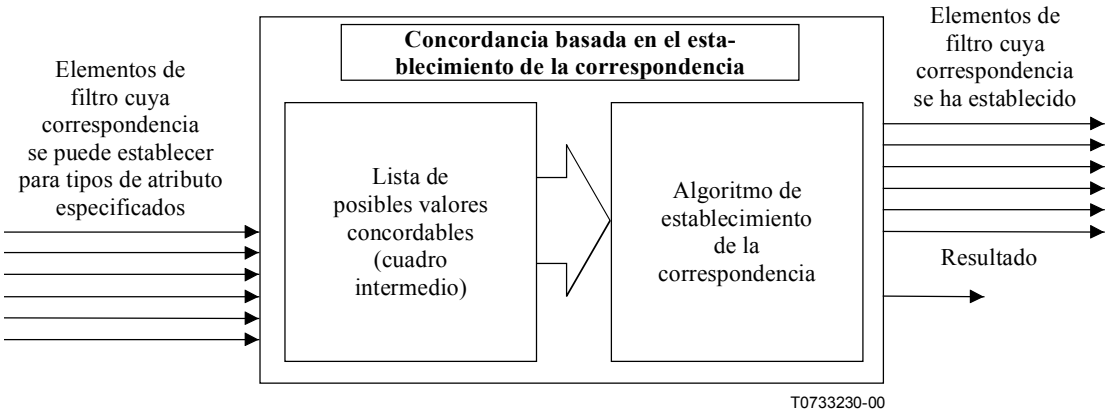


Figura 13 – Concordancia basada en el establecimiento de la correspondencia

Cuando se utiliza esta técnica, los elementos de filtro para los tipos de atributo designados (*elementos de filtro cuya correspondencia se puede establecer*) son sometidos a un proceso de establecimiento de la correspondencia basado en un cuadro de correspondencias y alguna forma de algoritmo de establecimiento de la correspondencia. El resultado de ese proceso son unos elementos de filtro llamados *elementos de filtro cuya correspondencia se ha establecido* que sustituyen a los elementos de filtro cuya correspondencia se pretendía establecer. En casos excepcionales, no se efectúa el establecimiento de la correspondencia y se devuelve información relativa a la naturaleza exacta de la excepción.

El número de elementos de filtro cuya correspondencia se establece no es necesariamente el mismo que el número de elementos de filtro cuya correspondencia se pretende, y por lo general será diferente.

Un elemento de filtro del tipo **extensibleMatch** con la especificación **type** ausente no puede ser un elemento de filtro del que se pueda establecer una correspondencia.

Una correspondencia basada en el establecimiento de la correspondencia puede ser local para un DSA. Si la evaluación de la búsqueda está distribuida, otros DSA participantes en la fase de evaluación de la búsqueda pueden aplicar su propia correspondencia basada en el establecimiento de la correspondencia. Sin embargo, el establecimiento de correspondencia aplicado se puede hacer llegar a otros DSA en el componente **chainedRelaxation** de los **chainedArguments**.

NOTA 2 – Para poder prestar un servicio coherente a los usuarios, los administradores de los DSA potencialmente participantes en una evaluación de búsqueda distribuida deberán considerar la armonización de sus cuadros y funciones de correspondencias.

La figura 14 ilustra el principio que se halla detrás del establecimiento de la función de correspondencia entre el mundo real y el modelo de directorio del mundo real. Los usuarios tienen una cierta percepción de ese mundo, en la que quizás no se tengan en cuenta todos los aspectos del mismo. Los aspectos del mundo real con cierta relevancia desde el punto de vista de cómo formulan los usuarios una petición de búsqueda constituyen un modelo del mundo real. Dicho modelo pasa a ser entonces la base del procedimiento de establecimiento de la correspondencia. El modelo preciso del mundo real se ha de basar en la experiencia y probablemente requiera actualizaciones periódicas basadas en el comportamiento de búsqueda observado por los usuarios.

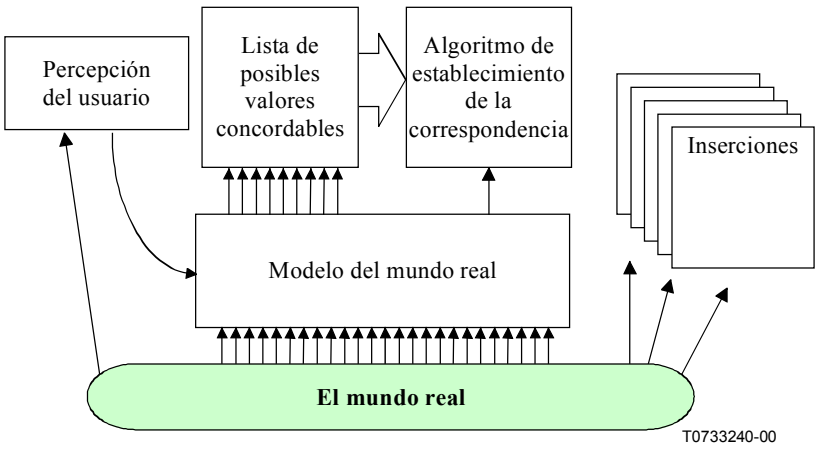


Figura 14 – Derivación de la información

Este modelo del mundo real quizás sólo requiera un subconjunto de los tipos de atributo utilizados por un usuario en una petición de búsqueda, y posiblemente sólo interese un único tipo de atributo. Por ejemplo, al considerar un modelo del mundo real con respecto a las localidades, sólo haría falta tener en cuenta los tipos de atributo relacionados con la localidad. No se establece la correspondencia de los elementos de filtro que no hagan referencia a esos tipos de atributo, sino que se retienen y utilizan junto con los elementos de filtro cuya correspondencia sí se haya establecido a efectos de concordancia de inserciones.

Se emplea un modelo del mundo real para establecer un cuadro de correspondencias de *valores concordables*, es decir, un conjunto de valores que posiblemente haya que hacer concordar con los elementos de filtro cuya correspondencia se puede establecer. La manera de establecer ese cuadro de correspondencias de valores concordables es un asunto local. La contraposición con el cuadro de correspondencias puede dar lugar entonces a cero o más concordancias. De cada concordancia se derivan uno o más elementos de filtro correspondidos. El algoritmo de establecimiento de la correspondencia determina cómo se aplican los elementos de filtro correspondidos con relación a las inserciones. La forma de hacerlo es un asunto local. Podría basarse en valores de atributos tradicionales de las inserciones o en valores fijados en las inserciones que carecen de significado fuera del diccionario, por ejemplo, identificadores numéricos.

La manera de emplear el establecimiento de la correspondencia y la manera de tratar los elementos de filtro correspondidos resultantes se especifican convenientemente haciendo referencia a los subfiltros definidos en 16.5 y expuestos con más detalle en el anexo Q. El concepto de subfiltros sólo se utiliza aquí como herramienta descriptiva. Una implementación puede utilizar cualquier otro algoritmo que dé el mismo resultado.

Cada subfiltro se evalúa contraponiéndolo al cuadro de correspondencias, y los elementos de filtro correspondidos resultantes se combinan con los elementos de filtro no correspondidos de la forma que determine el algoritmo pormenorizado de establecimiento de la correspondencia. Las inserciones concordantes que de ello resulten son la unión de las inserciones concordadas por cada uno de los subfiltros.

NOTA 3 – En muchas situaciones, los elementos de filtro cuya correspondencia se pretende establecer serán sustituidos por un operador OR lógico de los elementos de filtro cuya correspondencia se haya establecido.

En principio, hay dos modos diferentes de establecer la correspondencia. La correspondencia de cada elemento de filtro podría establecerse a razón de uno en cada momento, o podrían utilizarse múltiples elementos de filtro de posible correspondencia *combinables* para satisfacer una sola concordancia con relación al cuadro de correspondencias. Se pueden aplicar múltiples elementos de filtro a una sola concordancia basada en el establecimiento de la correspondencia si, y solamente si, son elementos de filtro *combinables*, es decir, están contenidos como elementos dentro de un subfiltro único.

NOTA 4 – Por ejemplo, se pueden utilizar dos nombres geográficos separados, unidos mediante un operador AND lógico, para referirse a una ubicación geográfica única de tamaño conveniente, cuando la utilización de un solo nombre geográfico pudiera especificar una ubicación geográfica poco precisa o de tamaño excesivo.

La contraposición de un elemento de filtro con un cuadro de correspondencias se efectúa utilizando la regla de concordancia inherente a, o especificada por, ese elemento de filtro, posiblemente después de una sustitución de regla de concordancia básica especificada en la regla de búsqueda directora (si la hubiere) y en la petición **search**.

NOTA 5 – Lo anterior podría requerir una regla de concordancia compleja, tal como la **generalWorldMatch** definida en la Rec. UIT-T X.520 | ISO/CEI 9594-6, que permitiera la rotación de palabras, la truncación de palabras, la concordancia de palabras aproximadas, etc.

NOTA 6 – Estas Especificaciones de directorio no especifican cómo combina una implementación las reglas de concordancia pertinentes en una concordancia combinada. Lo previsto es que la implementación pueda restringir las combinaciones de elementos de filtro y reglas de concordancia soportadas.

Si la contraposición intentada por un elemento de filtro o por elementos de filtro combinables frente a un cuadro de correspondencias no da de sí concordancia alguna para ningún subfiltro, es decir, la contraposición da como resultado FALSO o bien indefinido, no se establecerá la correspondencia de ningún elemento de filtro. Si en cada subfiltro, hay elementos de filtro, cuya correspondencia puede establecerse, la búsqueda no producirá ningún resultado. Se devolverá entonces un error al usuario.

En algunas situaciones, por ejemplo al establecer la concordancia de zonas geográficas, existe el requisito de que la contraposición con el cuadro de correspondencias dé un resultado único e inequívoco. Si un subfiltro concuerda con más de una inserción del cuadro de correspondencias o si diferentes subfiltros concuerdan con diferentes inserciones de dicho cuadro, la búsqueda podría dar como resultado demasiadas inserciones no deseadas. En vez de eso, se devuelve al usuario información que le permita iniciar una búsqueda nueva y mejor orientada.

NOTA 7 – En una situación más sencilla, los elementos de filtro que pueden corresponder sólo son comprobados contra el cuadro de correspondencias. Si esta comparación tiene éxito, los elementos de filtro son utilizados inalterados.

El establecimiento de la correspondencia puede ser dinámico en el sentido de que se puede ajustar (relajar) si la búsqueda da como resultado cero o muy pocas inserciones concordantes. Los detalles de cómo se lleva a cabo esa relajación quedan fuera del ámbito de aplicación de estas Especificaciones de directorio. Es algo que determinan los requisitos locales. La relajación se puede efectuar por pasos, haciendo así posible encontrar más inserciones. Se efectuará de tal manera que, cuando se dé un paso adicional en ese proceso, todas las inserciones devueltas en pasos anteriores, sean devueltas entonces de manera conjunta, quizás con algunas inserciones nuevas.

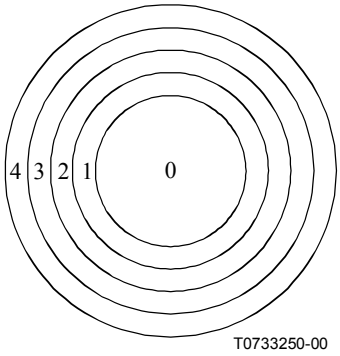


Figura 15 – Relajación de las condiciones de búsqueda

La relajación se efectúa paso a paso especificando diferentes niveles de relajación. El nivel cero corresponde a ausencia de relajación. El nivel uno corresponde a un primer nivel de relajación, etc. La figura 15 es una manera abstracta de ilustrar el mecanismo de relajación paso a paso. En estas Especificaciones de directorio no se define qué es lo que entrañan exactamente los diferentes niveles de relajación. El nivel de relajación lo puede controlar el constructivo **RelaxationPolicy**, que puede ser suministrado en una regla de búsqueda, en una petición **search**, o en ambas. Esto permite que la relajación de la correspondencia basada en el establecimiento de la correspondencia y la relajación mediante sustituciones de la regla de concordancia se sincronicen entre sí, ya que ambas pueden ser determinadas desde cada paso de la relajación según lo especificado por la **RelaxationPolicy**.

El control de búsqueda **extendedArea** es un entero que proporciona una manera alternativa de controlar el nivel de relajación para un algoritmo de concordancia basado en el establecimiento de la correspondencia. La posibilidad de controlarlo o no mediante este control de búsqueda forma parte de la personalización del algoritmo de correspondencia basado en el establecimiento de la correspondencia.

Si el control de búsqueda **extendedArea** está presente en una **request** de búsqueda y un algoritmo basado en el establecimiento de la correspondencia está autorizado a utilizarlo, se pasa por alto cualquier especificación de nivel de la **RelaxationPolicy**, tanto si está incluida en **search** como si está incluida en la regla de búsqueda directora.

La opción de control de búsqueda **includeAllAreas** especifica el modo de relajación cuando ésta es controlada por el control de búsqueda **extendedArea**. Si se ha fijado esta opción, la relajación se lleva a cabo como se ha descrito más arriba, es decir, se devuelven potencialmente más inserciones para niveles mayores de relajación (*relajación inclusiva*). Si no se ha fijado esta opción, al usuario sólo le interesa el resultado correspondiente a la relajación incremental (*relajación exclusiva*). Esto último podría ser de interés, si el usuario está procediendo a una relajación paso a paso y no le interesa obtener inserciones que fueron devueltas en resultados anteriores, sino únicamente inserciones adicionales resultantes del último paso de la relajación.

NOTA 8 – No se garantiza (sobre todo con un filtro complejo) que el usuario no obtenga ninguna inserción recibida anteriormente, ni que se devuelvan todas las inserciones que pudieran ser de interés. Por ejemplo, la búsqueda de restaurantes franceses en Winkfield podría fallar; la relajación de la búsqueda de todos los restaurantes en la zona Winkfield, pero excluyendo Winkfield, haría que el restaurante de cocina mixta White Hart Inn de Winkfield quedara fuera de los resultados de la búsqueda.

Es posible que algunos algoritmos de concordancia basados en el establecimiento de la correspondencia no permitan la relajación exclusiva o pueden estar personalizados para no permitirla. En este caso, la opción de control de búsqueda **includeAllAreas** será ignorada para esa función de establecimiento de la correspondencia y se llevará a cabo una posible relajación a modo de relajación inclusiva.

En algunos entornos, quizás también sea importante poder especificar un nivel negativo de relajación, lo que corresponde a un *endurecimiento de la concordancia*. En este caso, la opción de control de búsqueda **includeAllAreas** carece de significado y se prescinde de ella si está presente. Es posible que el endurecimiento no sea importante para todos los tipos de concordancia basada en el establecimiento de la correspondencia.

Un DSA puede soportar simultáneamente varias funciones de establecimiento de la correspondencia, es decir, mantener cuadros de correspondencias múltiples con los correspondientes algoritmos de establecimiento de la correspondencia. Los motivos por los que se justificarían funciones de establecimiento de la correspondencia múltiples podrían ser:

a) La dependencia de la función de establecimiento de la correspondencia que se ha de aplicar con respecto al tipo de aplicación. La concordancia de zonas geográficas (véase 6.8 de la Rec. UIT-T X.520 | ISO/CEI 9594-6) es una aplicación particular e importante de la concordancia basada en el establecimiento de la correspondencia. Otros ejemplos son la concordancia basada en el establecimiento de la correspondencia para búsquedas en páginas amarillas, las búsquedas bibliográficas, etc.

b) La dependencia, dentro de una determinada aplicación, de la especificación detallada de cómo se establece la correspondencia con respecto a las condiciones específicas. El establecimiento de la correspondencia para la concordancia de zonas geográficas, por ejemplo, puede depender de la zona geográfica (según lo reflejado por el **baseObject** de la búsqueda, por ejemplo) o del tipo de búsqueda que el usuario está intentando, esto es, basada en la información del filtro de búsqueda. Un caso similar es la dependencia del establecimiento de la correspondencia con respecto al idioma utilizado en la petición.

Si se pueden aplicar simultáneamente múltiples funciones de establecimiento de la correspondencia y la ejecución de una de ellas da lugar a una condición excepcional que debe ser notificada al usuario, no es preciso que la implementación verifique si existen o no excepciones múltiples (pero puede hacerlo).

Una especificación de correspondencia basada en el establecimiento de la correspondencia (véase más adelante) determina si el control de búsqueda **extendedArea** será aplicable para la función de establecimiento de la correspondencia en cuestión. Si están activas múltiples funciones de establecimiento de la correspondencia para la misma operación de búsqueda y algunas de ellas pueden ser controladas por el control de búsqueda **extendedArea**, todas las funciones aplicarán una relajación o un endurecimiento simultáneo de acuerdo con el control de búsqueda **extendedArea** y, si es aplicable, también la opción de control de búsqueda **includeAllAreas**.

NOTA 9 – El ejemplo dado anteriormente muestra que la utilización de **includeAllAreas** con más de una correspondencia basada en el establecimiento de la correspondencia puede crear dificultades.

Si el control de búsqueda **extendedArea** especifica un nivel Si el control de búsqueda **extendedArea** especifica un nivel de relajación o endurecimiento no soportado por el DSA para algunas de las funciones de establecimiento de la correspondencia afectadas por el control de búsqueda, el DSA llevará a cabo el establecimiento de la correspondencia en base al mejor esfuerzo. Si el control de búsqueda **extendedArea** especifica un nivel de relajación o endurecimiento no soportado por el DSA para ninguna de las funciones de establecimiento de la correspondencia afectadas por el control de búsqueda, se devolverá un atributo de notificación **searchServiceProblem** con el valor **id-pr-unavailableRelaxationLevel** en el parámetro **notification** de **CommonResults**.

NOTA 10 – Si la evaluación de la operación búsqueda está distribuida entre múltiples DSA, es posible que dichos DSA empleen funciones de establecimiento de la correspondencia diferente que den resultados incoherentes a menos que se establezca un cierto grado de coordinación entre los DSA.

Aunque los detalles de la concordancia basada en el establecimiento de la correspondencia son un asunto local, es posible fijar las características globales de esa concordancia definiendo un tipo especial de reglas de concordancia llamadas *reglas de concordancia basada en el establecimiento de la correspondencia*. Esas reglas de concordancia se definen como ejemplares de la clase de objeto de información **MATCHING-RULE**. Sin embargo, difieren de las reglas de concordancia habituales porque no especifican la concordancia en el sentido tradicional y, por tanto, no especifican la sintaxis de la misma. Ahora bien, como parte de su definición, especifican cuál es su objetivo e indican cómo se aplican y cómo se tratan las condiciones excepcionales. El comportamiento específico de una regla de concordancia basada en el establecimiento de la correspondencia se puede describir parcialmente mediante un ejemplar de la clase de objeto de información ASN.1 derivada de la clase de objeto de información por debajo de lo genérico (parametrizada) **MAPPING-BASED-MATCHING**. La finalidad única de esta clase de objeto de información es especificar aquellos aspectos que pudieran ser personalizables. La Especificación de directorio no indica cómo y dónde se almacena un ejemplar de esa clase de objeto de información, sino simplemente que se ponga de alguna manera a disposición del DSA.

```
MAPPING-BASED-MATCHING
      { SelectedBy, BOOLEAN:combinable, MappingResult, OBJECT IDENTIFIER:matchingRule } ::=
CLASS {
      &selectBy                SelectedBy              OPTIONAL,
      &ApplicableTo            ATTRIBUTE,
      &subtypesIncluded        BOOLEAN                 DEFAULT TRUE,
      &combinable              BOOLEAN                 (combinable),
      &mappingResults          MappingResult           OPTIONAL,
      &userControl             BOOLEAN                 DEFAULT FALSE,
      &exclusive               BOOLEAN                 DEFAULT TRUE,
      &matching-rule           MATCHING-RULE           (matchingRule),
      &id                      OBJECT IDENTIFIER       UNIQUE }
```

```
WITH SYNTAX  {
        [ SELECT BY                 &selectBy ]
        APPLICABLE TO               &ApplicableTo
        [ SUBTYPES INCLUDED         &subtypesIncluded ]
        COMBINABLE                  &combinable
        [ MAPPING RESULTS           &mappingResults ]
        [ USER CONTROL              &userControl ]
        [ EXCLUSIVE                 &exclusive ]
        MATCHING RULE               &matching-rule
        ID                          &id }
```

La clase de objeto información **MAPPING-BASED-MATCHING** tiene las siguientes especificaciones de campos:

a) El campo **&selectBy**, referencia ficticia para una especificación de cómo se selecciona un ejemplar de la clase de objeto de información derivada para una correspondencia basada en el establecimiento de la correspondencia. La clase de objeto de información derivada especificará, si procede, un tipo ASN.1 determinante junto con una descripción textual de la manera de efectuar la selección. Este componente se pasará por alto si el usuario suministra en la petición **search** un componente **mapping** no vacío del constructivo **RelaxationPolicy**.

NOTA 11 – En principio, varios ejemplares, posiblemente de diferentes clases de objeto de información derivadas, pueden ser seleccionados por la misma petición **search**.

b) El campo **&ApplicableTo**, que especifica los elementos de filtro que deberán considerarse candidatos al establecimiento de una correspondencia especificando los tipos de atributo de esos elementos de filtro. A cualquier elemento de filtro para un tipo de atributo indicado por este subcomponente se le aplica la concordancia basada en el establecimiento de la correspondencia. Este componente deberá estar siempre presente. No es necesario que todos los tipos de atributo por él indicados estén presentes en el filtro. El valor lo determina el ejemplar del objeto de información de la clase de objeto de información derivada.

c) El campo **&subtypesIncluded**, que es un valor de tipo booleano que especifica si el ejemplar de una clase de objeto de información derivada puede aceptar subtipos de atributo **&ApplicableTo**, además de los tipos de atributo especificados. Si está ausente, se permiten los subtipos siempre que no sean invalidados por otros mecanismos. El valor lo determina el ejemplar de objeto de información de la clase de objeto de información derivada.

d) El campo **&combinable**, que es un valor de tipo booleano que, si es **TRUE**, permite que la concordancia basada en el establecimiento de la correspondencia utilice múltiples elementos de filtro combinables para el cumplimiento de la concordancia. **Combinable** es una referencia ficticia para el valor de este componente que ha de ser determinado por la clase de objeto de información derivada.

e) El campo **&mappingResults**, referencia ficticia para una especificación de cómo se notifican las condiciones de excepción. La clase de objeto de información derivada deberá especificar un tipo ASN.1 para notificar condiciones de excepción importantes.

f) El campo **&userControl**, que es un valor de tipo booleano que especifica si un ejemplar de una clase de objeto de información derivada y su regla de concordancia basada en el establecimiento de la correspondencia asociada pueden ser controladas por el control de búsqueda **extendedArea**.

NOTA 12 – Si se aplican simultáneamente varias concordancias basadas en el establecimiento de la correspondencia, quizás convenga que sólo una de ellas utilice el control de búsqueda **extendedArea**.

g) El campo **&exclusive**, que es un valor de tipo booleano que especifica si un ejemplar de una clase de objeto de información derivada y su regla de concordancia basada en el establecimiento de la correspondencia asociada permiten que se aplique una relajación exclusiva. El valor, si está presente, lo determina el ejemplar de objeto de información de la clase de objeto de información derivada. Si el valor es **FALSE** o si el DSA no soporta concordancia exclusiva para esta concordancia basada en el establecimiento de la correspondencia, este establecimiento particular de la correspondencia actuará como si se hubiera fijado la opción de control de búsqueda **includeAllAreas**.

NOTA 13 – Si se aplican simultáneamente varias concordancias basadas en el establecimiento de la correspondencia, quizás convenga hacer que sólo una de ellas recurra a la relajación exclusiva.

h) El campo **&matching-rule**, que es un valor de tipo de identificador de objeto que identifica la regla de concordancia basada en la concordancia para la que este ejemplar proporciona una especificación adicional y que deberá ser aplicada para la concordancia basada en el establecimiento de la correspondencia. La referencia ficticia **matchingRule** para el valor de este componente ha de ser determinada por la clase de objeto de información derivada. La regla de concordancia especificada deberá ser utilizada para la concordancia basada en el establecimiento de la correspondencia particular.

i) El campo **&id**, que es un identificador de objeto atribuido a la correspondencia basada en el establecimiento de la correspondencia particular.

13.7 Definición de estructura del DIT

13.7.1 Visión de conjunto

Un aspecto fundamental del esquema de directorio es la especificación del lugar, en el DIT, donde puede colocarse una inserción de una determinada clase, y de cómo debe ser denominada. Deben considerarse varios aspectos de esta especificación, a saber:

- la relación jerárquica de inserciones en el DIT (reglas de estructura del DIT);
- el atributo o atributos utilizados para formar el RDN de la inserción (formas de nombre).

13.7.2 Definición de forma de nombre

La definición de una forma de nombre comprende:

a) especificación de la clase de objeto denominado;

b) indicación de los atributos obligatorios que van a ser utilizados para los RDN para inserciones de esta clase de objeto en que se aplique esta forma de nombre;

c) indicación de los atributos facultativos, si los hubiere, que pueden ser utilizados para los RDN para inserciones de esta clase de objeto en que se aplique esta forma de nombre;

d) asignación de un identificador de objeto para la forma de nombre.

Si se requiere un conjunto diferente de atributos de denominación para inserciones de una clase de objeto estructural o alias dada, se especificará una forma de nombre para cada conjunto distinto de atributos que se vaya a utilizar para denominación.

En las formas de nombre sólo se usan clases de objeto estructurales y alias.

Para que inserciones de una determinada clase de objeto estructural o alias existan en una porción de la DIB, por lo menos una forma de nombre para esa clase de objeto estructural o alias deberá estar contenida en la parte aplicable del esquema. El esquema contiene las formas de nombre adicionales que se requieran.

El atributo (o los atributos) del RDN no tienen que ser escogidos entre los contenidos en la lista de los atributos permitidos de la clase de objeto estructural o alias como especificados en su definición de clase de objeto estructural o alias.

NOTA – Los atributos de denominación vienen gobernados por las reglas de contenido del DIT y por el uso del contexto del DIT de la misma manera que otros atributos.

Una forma de nombre es sólo un elemento primitivo de la especificación completa requerida para constreñir la forma del DIT a la requerida por las autoridades administrativas y de denominación que determinan las políticas de denominación de una región dada del DIT. Los aspectos restantes de la especificación de la estructura del DIT se examinan en 13.7.5.

13.7.3 Especificación de formas de nombre

Las formas de nombre pueden definirse como valores de la clase de objeto de información **NAME-FORM**:

```
NAME-FORM ::= CLASS {
        &namedObjectClass       OBJECT-CLASS,
        &MandatoryAttributes    ATTRIBUTE,
        &OptionalAttributes     ATTRIBUTE OPTIONAL,
        &id                     OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        NAMES                   &namedObjectClass
        WITH ATTRIBUTES         &MandatoryAttributes
        [ AND OPTIONALLY        &OptionalAttributes ]
        ID                      &id }
```

Para una forma de nombre que se define utilizando esta clase de objeto de información:

a) **&namedObjectClass** es la clase de objeto estructural que denomina;

b) **&MandatoryAttributes** es el conjunto de atributos que estarán presentes en el RDN de la inserción que rige;

c) **&OptionalAttributes** es el conjunto de atributos que puede estar presente en el RDN de la inserción que rige;

d) **&id** es el identificador de objeto que le ha sido asignado.

Todos los tipos de atributos en las listas de atributos obligatorios y facultativos tienen que ser diferentes.

13.7.4 Clase de objeto estructural de una inserción

Algunas especificaciones de subesquema incluyen formas de nombre para no más de una clase de objeto estructural por cada cadena de superclase de clase de objeto estructural representada en el subesquema.

Algunas especificaciones de subesquema pueden incluir formas de nombre para más de una clase de objeto estructural por cada cadena de superclase de clase de objeto estructural representada en el subesquema.

En ambos casos, con respecto a una determinada inserción, sólo la clase de objeto estructural más subordinada en la cadena de superclase de clase estructural presente en el atributo **objectClass** de la inserción determina la regla de contenido del DIT y la regla de estructura del DIT que se aplica a la inserción. Esta clase se conoce por la clase de objeto estructural de la inserción y se indica por el atributo operacional **structuralObjectClass**.

13.7.5 Definición de regla de estructura de DIT

Una regla de estructura de DIT es una especificación proporcionada por la autoridad administrativa de subesquema que el directorio utiliza para controlar la colocación y denominación de inserciones dentro del alcance del subesquema. Cada inserción de objeto y de alias es gobernado por una sola regla de estructura de DIT. Un subesquema que gobierna un subárbol del DIT consta típicamente de varias reglas de estructura DIT que permiten varios tipos de inserciones en el subárbol.

Una definición de regla de estructura de DIT incluye:

a) un identificador constituido por un número entero que es único dentro del alcance del subesquema;

b) una indicación de la forma de nombre para inserciones gobernadas por la regla de estructura de DIT;

c) el conjunto de reglas de estructura superior autorizadas, si se requiere.

El conjunto de reglas de estructura de DIT para un subesquema especifica las formas de nombres distinguidos para inserciones gobernadas por el subesquema.

Una regla de estructura de DIT autoriza que las inserciones en un subesquema dado se adhieran a una determinada forma de nombre. La forma del último componente de RDN de un **DistinguishedName** de la inserción es determinada por la forma de nombre de la regla de estructura de DIT que gobierna la inserción.

El componente **namedObjectClass** de la forma de nombre (la clase de objeto de la forma de nombre) corresponde a la clase de objeto estructural de la inserción.

Una regla de estructura de DIT sólo permitirá inserciones pertenecientes a la clase de objeto estructural identificada por su forma de nombre asociada. No permite inserciones pertenecientes a cualquiera de las subclases de la clase de objeto estructural.

Con respecto a una inserción particular, la regla de estructura de DIT que gobierna la inserción se llama *regla de estructura gobernante* de la inserción. Esta regla puede identificarse examinando el atributo **governingStructureRule** de la inserción.

Con respecto a una determinada inserción, la regla de estructura de DIT que gobierna el superior de la inserción se llama *regla de estructura superior* de la inserción.

Una inserción sólo puede existir en el DIT como un subordinado de otra inserción (el superior), si en el esquema gobernante existe una regla de estructura de DIT que:

– indica una forma de nombre para la clase de objeto estructural de la inserción, e

– incluye la regla de estructura superior de la inserción como una posible regla de estructura superior *o* no especifica una regla de estructura superior, en cuyo caso la inserción será un punto administrativo de subesquema.

Si una inserción que es un punto administrativo de subesquema no está incluida para los fines de administración de subesquema en su subinserción de subesquema, se utiliza entonces el subesquema de la zona administrativa de subesquema inmediatamente superior para gobernar la inserción.

Las inserciones que son puntos administrativos pero no tienen subinserción de subesquema (por ejemplo, inserciones de punto administrativo recientemente creadas), no tienen regla de estructura gobernante. El directorio no permitirá la creación de subordinados por debajo de tales inserciones hasta que no se agregue una subinserción de subesquema.

Si se convierte una inserción en un nuevo punto administrativo de subesquema, la regla de estructura gobernante de todas las inserciones en la nueva zona administrativa de subesquema se cambia automáticamente a la indicada por el nuevo subesquema.

13.7.6 Especificación de la regla de estructura de DIT

La sintaxis abstracta de una regla de estructura DIT se expresa por el siguiente tipo ASN.1:

```
DITStructureRule  ::=  SEQUENCE {
      ruleIdentifier              RuleIdentifier ,
                                  -- shall be unique within the scope of the subschema
      nameForm                    NAME-FORM.&id,
      superiorStructureRules      SET SIZE (1..MAX) OF RuleIdentifier OPTIONAL }

RuleIdentifier  ::=  INTEGER
```

La correspondencia entre las partes de la definición, tal como están indicadas en 13.7.5, y los diversos componentes del tipo ASN.1 definido más arriba es la siguiente:

a) el componente **ruleIdentifier** identifica la regla de estructura de DIT inequívocamente dentro de un subesquema;

b) el componente **nameForm** de la regla de estructura de DIT especifica la forma de nombre para inserciones gobernadas por la regla de estructura de DIT;

c) el componente **superiorStructureRules** identifica reglas de estructura superior permitidas para inserciones gobernadas por la regla. Si se omite este componente, la regla de estructura de DIT se aplica un punto administrativo de subesquema.

La clase de objeto de información **STRUCTURE-RULE** se proporciona para facilitar la documentación de las reglas de estructura de DIT:

```
STRUCTURE-RULE  ::=  CLASS {
      &nameForm                   NAME-FORM,
      &SuperiorStructureRules     STRUCTURE-RULE OPTIONAL,
      &id                         RuleIdentifier }
WITH SYNTAX {
      NAME FORM                   &nameForm
      [ SUPERIOR RULES            &SuperiorStructureRules ]
      ID                          &id }
```

13.8 Definición de la regla de contenido de DIT

13.8.1 Visión de conjunto

La regla de contenido de DIT especifica el contenido admisible de inserciones de una clase de objeto estructural particular mediante la identificación de un conjunto facultativo de clases de objeto auxiliares, y de atributos obligatorios, facultativos y excluidos. Los atributos colectivos deberán ser incluidos en reglas de contenido de DIT si han de ser permitidos en una inserción.

Una definición de regla de contenido de DIT incluye:

a) una indicación de la clase de objeto estructural a que se aplica;

b) facultativamente, una indicación de las clases de objeto auxiliares permitidas para inserciones gobernadas por la regla;

c) facultativamente, una indicación de los atributos obligatorios, por encima de los invocados por las clases de objeto estructurales y auxiliares, requeridos por inserciones gobernadas por la regla de contenido de DIT;

d) facultativamente, una indicación de los atributos facultativos, por encima de los invocados por las clases de objeto estructurales y auxiliares, permitidos por inserciones gobernadas por la regla de contenido de DIT;

e) facultativamente, una indicación de los atributos *facultativos* de las clases de objeto estructurales y auxiliares cuya aparición en inserciones gobernadas por la regla está excluida.

Para cualquier subesquema de especificación válido, hay, como máximo, una regla de contenido de DIT para cada clase de objeto estructural.

Todas las inserciones en el DIT están gobernadas, como máximo, por una regla de contenido de DIT. Esta regla puede ser identificada examinando el valor del atributo **structuralObjectClass** de la inserción.

Si no hay una regla de contenido DIT presente para una clase de objeto estructural, las inserciones de esas clases contendrán solamente los atributos permitidos por la definición de clase de objeto estructural.

Las reglas de contenido de DIT de superclases de la clase de objeto estructural para una inserción no se aplican a esa inserción.

Dado que una regla de contenido DIT está asociada con una clase de objeto estructural, se desprende de ello que todas las inserciones de la misma clase de objeto estructural tendrán la misma regla de contenido de DIT cualquiera que sea la regla de estructura de DIT que gobierne su ubicación en el DIT.

Una inserción gobernada por una regla de contenido DIT puede estar asociada, además de con la clase de objeto estructural de la regla de estructura DIT, con un subconjunto de clases de objeto auxiliares identificado por la regla de contenido de DIT. Esta asociación se refleja en el atributo **objectClass** de la inserción.

El contenido de una inserción deberá ser coherente con las clases de objeto indicadas por su atributo **objectClass** de la siguiente forma:

- los atributos obligatorios de clases de objeto indicadas por el atributo **objectClass** estarán *siempre* presentes en la inserción;
- los atributos facultativos (no indicados como adicionales, facultativos u obligatorios, en la regla de contenido de DIT) de clases de objeto auxiliares indicadas por la regla de contenido de DIT sólo pueden estar presentes si el atributo **objectClass** indica estas clases de objeto auxiliares.

Los atributos obligatorios asociados con las clases de objeto estructurales o con las clases de objeto auxiliares indicadas no pueden ser excluidos de una regla de contenido de DIT.

13.8.2 Especificación de la regla de contenido de DIT

La sintaxis abstracta de una regla de contenido de DIT se expresa por el siguiente tipo ASN.1:

```
DITContentRule ::= SEQUENCE {
    structuralObjectClass       OBJECT-CLASS.&id,
    auxiliaries                 SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
    mandatory           [1]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
    optional            [2]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
    precluded           [3]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL }
```

La correspondencia entre las partes de la definición enumeradas en 13.8.1 y los diversos componentes del tipo ASN.1 definido más arriba, es la siguiente:

a) el componente **structuralObjectClass** identifica la clase de objeto estructural a que se aplica la regla de contenido de DIT;

b) el componente **auxiliaries** identifica las clases de objeto auxiliares permitidas para una inserción a la cual se aplica la regla de contenido del DIT;

c) el componente **mandatory** especifica los tipos de atributos de usuario que contendrá una inserción a la cual se aplica la regla de contenido del DIT además de aquellos que contendrá de acuerdo con sus clases de objeto estructurales y auxiliares;

d) los componentes **optional** especifican los tipos de atributos de usuario que puede contener una inserción a la cual se aplica la regla de contenido del DIT además de los que puede contener de acuerdo con sus clases de objetos estructurales y auxiliares;

e) el componente **precluded** especifica un subconjunto de tipos de atributos de usuario facultativos de las clases de objeto estructurales y auxiliares que están excluidos de una inserción a la cual se aplica la regla de contenido del DIT.

La clase de objeto de información **CONTENT-RULE** se proporciona para facilitar la documentación de reglas de contenido del DIT.

```
CONTENT-RULE ::= CLASS {
    &structuralClass        OBJECT-CLASS    UNIQUE,
    &Auxiliaries            OBJECT-CLASS    OPTIONAL,
    &Mandatory              ATTRIBUTE       OPTIONAL,
    &Optional               ATTRIBUTE       OPTIONAL,
    &Precluded              ATTRIBUTE       OPTIONAL }
```

```
WITH SYNTAX {
        STRUCTURAL OBJECT-CLASS     &structuralClass
        [ AUXILIARY OBJECT-CLASSES  &Auxiliaries ]
        [ MUST CONTAIN              &Mandatory ]
        [ MAY CONTAIN               &Optional ]
        [ MUST-NOT CONTAIN          &Precluded ] }
```

13.9 Definición del tipo de contexto

La definición de un tipo de contexto implica:

a) la especificación de la sintaxis del contexto;

b) la especificación de la sintaxis de una aserción de contexto;

c) la definición de la semántica del contexto;

d) la especificación de como se efectúan las concordancias; y

e) la asignación de un identificador de objeto al tipo de contexto.

13.9.1 Concordancia del valor de contexto

Una aserción de contexto presentada concuerda con un valor de contexto almacenado del mismo tipo de contexto según la descripción de concordancia que forma parte de la definición del contexto.

13.9.2 Definición del contexto

Los contextos se definen utilizando la clase de objeto de información **CONTEXT**:

```
CONTEXT ::= CLASS {
        &Type,
        &Assertion          OPTIONAL,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        WITH SYNTAX         &Type
        [ ASSERTED AS       &Assertion ]
        ID                  &id }
```

Si se omite la **&Assertion**, la sintaxis de aserción del contexto es la misma que **&Type**.

Cuando se define un contexto, la especificación incluirá la descripción de la semántica del contexto y la forma de evaluar la concordancia.

En la Rec. UIT-T X.520 | ISO/CEI 9594-6, se especifican definiciones de contexto seleccionadas.

13.10 Definición de uso del contexto del DIT

13.10.1 Visión de conjunto

Un uso del contexto del DIT es una especificación proporcionada por la autoridad administrativa del subesquema para especificar los tipos de contexto admisibles que pueden almacenarse con un atributo y los tipos de contexto obligatorios que pueden almacenarse con un atributo.

Una definición de uso de contexto del DIT comprende:

a) una indicación del tipo de atributo a la que se aplica;

b) facultativamente, una indicación de los tipos de contexto obligatorios que se asociarán con valores del tipo de atributo cuando se almacene el atributo;

c) facultativamente, una indicación de los tipos de contexto facultativos que pueden asociarse con valores del tipo de atributo cuando se almacene el atributo.

Si, para un tipo de atributo determinado, no está presente ninguna definición de uso del contexto del DIT, los valores de atributos de ese tipo carecerán de listas de contexto. Para una zona administrativa de subesquema determinada, únicamente puede existir un solo uso de contexto del DIT para un tipo de atributo determinado. Puede definirse un uso de contexto del DIT aplicable a todos los tipos de atributo, en cuyo caso será el único uso de contexto del DIT en el subesquema.

13.10.2 Especificación del uso del contexto del DIT

La sintaxis abstracta del uso del contexto del DIT se expresa mediante el siguiente tipo ASN.1:

```
DITContextUse ::= SEQUENCE {
      attributeType                ATTRIBUTE.&id,
      mandatoryContexts      [1]   SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL,
      optionalContexts       [2]   SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL }
```

La correspondencia entre las partes de la definición, enumeradas en 13.10.1 y las distintas componentes del tipo ASN.1 definido anteriormente, se realiza como sigue:

a) el componente **attributeType** identifica el tipo de atributo al que se aplica el uso del contexto DIT o cualquier tipo de atributo (**id-oa-allAttributeTypes**);

b) el componente **mandatoryContexts** especifica los tipos de contexto que se asociarán con un valor de atributo del tipo dado cuando se almacene el atributo. Si éste se omite, puede haber valores de atributo sin listas de contexto;

c) el componente **optionalContexts** especifica tipos de contexto que pueden estar asociados con un valor de atributo del tipo dado cuando se almacene el atributo. Si éste se omite pero hay presentes contextos obligatorios, aparecerán todos los valores de atributo con los tipos de contextos obligatorios y ninguno más. Si se omiten éste y mandatoryContexts, ello equivale a la carencia de uso de contexto DIT para el tipo de atributo; es decir los valores de atributo del tipo de atributo dado no tendrán asociadas listas de contexto.

Se proporciona la clase de objeto de información **DIT-CONTEXT-USE-RULE** para facilitar la documentación de las reglas de uso del contexto DIT:

```
DIT-CONTEXT-USE-RULE ::= CLASS {
      &attributeType               ATTRIBUTE.&id  UNIQUE,
      &Mandatory                   CONTEXT        OPTIONAL,
      &Optional                    CONTEXT        OPTIONAL }
WITH SYNTAX {
      ATTRIBUTE TYPE               &attributeType
      [ MANDATORY CONTEXTS         &Mandatory ]
      [ OPTIONAL CONTEXTS          &Optional ] }
```

14 Esquema de sistema de directorio

14.1 Visión de conjunto

El esquema de sistema de directorio es un conjunto de definiciones y constricciones concernientes a la información que el propio directorio necesita conocer para funcionar correctamente. Esta información se especifica como subinserciones y atributos operacionales.

NOTA – El esquema de sistema permite, por ejemplo que el sistema de directorio:

- evite la asociación de subinserciones de tipo erróneo con inserciones administrativas (por ejemplo, la creación de una subinserción de subesquema subordinada a una inserción administrativa definida solamente como una inserción administrativa de seguridad);
- evite la adición de atributos operacionales inapropiados a una inserción o subinserción (por ejemplo, un atributo operacional de subesquema a la inserción de una persona).

Formalmente, el esquema de sistema de directorio comprende un conjunto de:

a) definiciones de clases de objetos que definen los atributos que estarán o pueden estar presentes en una subinserción de una clase dada;

b) definiciones de tipos de atributos operacionales que especifican las características de atributos operacionales conocidos y utilizados por el directorio.

La definición completa de un atributo operacional incluye una especificación de la manera en la que el directorio utiliza y (si procede) proporciona o gestiona, el atributo en el curso de su funcionamiento.

El esquema de sistema de directorio es distribuido, como la propia DIB. Cada autoridad administrativa establece la parte del esquema de sistema que aplicará para las porciones de la DIB administrada por la autoridad.

El esquema de sistema de directorio definido en esta Especificación de directorio es una parte integrante del propio sistema de directorio. Cada DSA que participa en un sistema de directorio necesita un conocimiento completo del esquema de sistema establecido por su autoridad administrativa. El esquema de sistema para una zona administrativa puede ser definido por la autoridad administrativa utilizando la notación indicada en esta cláusula.

El esquema de sistema de directorio no está reglamentado por las reglas de estructura o de contenido del DIT. Cuando se define un elemento de esquema de sistema, se proporciona una especificación de cómo se utiliza y dónde aparece en el DIT.

En las subcláusulas siguientes se especifican ciertos aspectos del esquema de sistema de directorio.

El esquema de sistema de directorio requerido para sustentar la distribución de directorio se especifica en las cláusulas 25 a 28.

14.2 Esquema de sistema que soporta el modelo de información administrativo y operacional

Aunque **subentry** y **subentryNameForm** se especifican utilizando la notación de la cláusula 13, las subinserciones no están reglamentadas por las reglas de estructura o de contenido del DIT.

14.2.1 Clase de objeto subinserción

La clase de objeto **subentry** es una clase de objeto estructural y se define como sigue:

```
subentry  OBJECT-CLASS ::= {
        SUBCLASS OF     { top }
        KIND            structural
        MUST CONTAIN    { commonName | subtreeSpecification }
        ID              id-sc-subentry }
```

14.2.2 Forma de nombre de subinserción

La forma de nombre **subentryNameForm (forma de nombre de subinserción)** permite denominar inserciones de la clase **subentry** utilizando el atributo **commonName**:

```
subentryNameForm  NAME-FORM  ::=  {
        NAMES                   subentry
        WITH ATTRIBUTES         { commonName }
        ID                      id-nf-subentryNameForm }
```

No se utilizará ninguna otra forma de nombre para las subinserciones.

14.2.3 Atributo operacional especificación de subárbol

El atributo operacional **subtreeSpecification**, cuya semántica se especifica en la cláusula 12, se define como sigue:

```
subtreeSpecification  ATTRIBUTE  ::=  {
        WITH SYNTAX     SubtreeSpecification
        USAGE           directoryOperation
        ID              id-oa-subtreeSpecification }
```

Este atributo está presente en todas las subinserciones; cada valor define un conjunto de inserciones (en términos de porción de una zona administrativa posiblemente con mejora por selección en un filtro de clases de objeto) que pueden ser sometidas a las políticas definidas por la subinserción.

NOTA – Lo anterior permite dirigir una política compleja única (por ejemplo, una regla de búsqueda) en múltiples combinaciones de clases de objeto, en regiones separadas de una zona administrativa, mientras que se definen en una sola subinserción.

14.3 Esquema de sistema que soporta el modelo administrativo

El modelo administrativo definido en la cláusula 11 requiere que las inserciones administrativas contengan un atributo **administrativeRole (cometido administrativo)** para indicar que la zona administrativa asociada se relaciona con uno o más cometidos administrativos.

El atributo operacional **administrativeRole** se especifica como sigue:

```
administrativeRole  ATTRIBUTE  ::= {
        WITH SYNTAX                     OBJECT-CLASS.&id
        EQUALITY MATCHING RULE          objectIdentifierMatch
        USAGE                           directoryOperation
        ID                              id-oa-administrativeRole }
```

Los valores de atributo definido por esta Especificación de directorio son:

id-ar-autonomousArea
id-ar-accessControlSpecificArea
id-ar-accessControlInnerArea
id-ar-subschemaAdminSpecificArea
id-ar-collectiveAttributeSpecificArea
id-ar-collectiveAttributeInnerArea
id-ar-contextDefaultSpecificArea
id-ar-serviceSpecificArea

La semántica de estos valores se definen en la cláusula 12.

El atributo operacional **administrativeRole** se utiliza también para reglamentar las subinserciones que se permiten sean subordinadas de una inserción administrativa. Una subinserción de una clase no permitida por el atributo **administrativeRole** no puede ser subordinada de la inserción administrativa.

14.4 Esquema de sistema que soporta los requisitos generales administrativos y operacionales

En las cláusulas que siguen se describen atributos operacionales del subesquema que no son atributos en el sentido general (es decir no se mantienen con una inserción), sino que deben considerarse como atributos "virtuales" que representan información deducible (por ejemplo, de atributos operacionales existentes, sus valores u otra información). Tales atributos virtuales son válidos para todas las inserciones de una zona administrativa. Esto tiene como efecto que esos atributos operacionales de subesquema parezcan estar presentes en cada inserción.

14.4.1 Indicación de tiempo

El atributo **createTimestamp (indicación de tiempo de creación)** indica el momento en que se creó una inserción:

```
createTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- as per 41.3 b) or c) of ITU-T Rec. X.680 | ISO/IEC 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-createTimestamp }
```

modifyTimestamp indica el momento en que se modificó una inserción por última vez:

```
modifyTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- as per 41.3 b) or c) of ITU-T Rec. X.680 | ISO/IEC 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-modifyTimestamp }
```

El **subschemaTimestamp** indica el momento en el que se creó o se modificó por última vez la subinserción de subesquema para la inserción (véase 15.3). Está disponible en cada inserción:

```
subschemaTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- as per 41.3 b) or c) of ITU-T Rec. X. 680 | ISO/IEC 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-subschemaTimestamp }
```

Las reglas de concordancia **generalizedTimeMatch** y **generalizedTimeOrderingMatch** se definen en la Rec. UIT-T X.520 | ISO/CEI 9594-6.

14.4.2 Atributos operacionales de modificador de inserción

El atributo operacional **creatorsName (nombre de creador)** indica el nombre distinguido del usuario de directorio que creó una inserción:

```
creatorsName  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-creatorsName }
```

El atributo operacional **modifiersName (nombre de modificador)** indica el nombre distinguido del usuario de directorio que modificó por última vez la inserción:

```
modifiersName  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-modifiersName }
```

Estos atributos operacionales utilizarán el nombre distinguido primario.

14.4.3 Atributos operacionales de identificación de subinserción

El atributo operacional **subschemaSubentryList** identifica la subinserción de subesquema que gobierna la inserción. Está disponible en cada inserción:

```
subschemaSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-subschemaSubentryList }
```

El atributo operacional **accessControlSubentryList** identifica todas las subinserciones de control de acceso que afectan a la inserción. Está disponible en cada inserción:

```
accessControlSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-accessControlSubentryList }
```

El atributo operacional **collectiveAttributeSubentryList** identifica todas las subinserciones de atributos colectivos que afectan a la inserción. Está disponible en cada inserción:

```
collectiveAttributeSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-collectiveAttributeSubentryList }
```

El atributo operacional **contextDefaultSubentryList** identifica todas las subinserciones por defecto del contexto que afectan a la inserción. Está disponible en cada inserción:

```
contextDefaultSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-contextDefaultSubentryList }
```

El atributo operacional **serviceAdminSubentryList** identifica todas las subinserciones del servicio de administración, si existen, que afectan a la inserción. Está disponible en toda inserción afectada por cualquiera de esas subinserciones.

```
serviceAdminSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-serviceAdminSubentryList }
```

14.4.4 Atributo operacional que posee subordinados

El atributo operacional **hasSubordinates** indica si existen inserciones subordinadas por debajo de la inserción que posee este atributo. Un valor de **VERDADERO** indica que existen subordinados. Un valor de **FALSO** indica que no hay subordinados. Si este atributo está ausente, no se proporciona información sobre la existencia de inserciones subordinadas. Generalmente, el atributo desvelará la existencia de subordinados aún cuando los controles de acceso oculten los subordinados inmediatos. Para prevenir la revelación de la existencia de subordinados, deberá protegerse el propio atributo operacional mediante controles de acceso.

NOTA – Puede devolverse un valor de **VERDADERO** cuando no haya subordinados si únicamente están disponibles todos los subordinados posibles a través de una referencia subordinada no específica (véase la Rec. UIT-T X.518 | ISO/CEI 9594-4) o si los únicos subordinados son subinserciones o miembros de familia vástagos.

```
hasSubordinates ATTRIBUTE ::= {
        WITH SYNTAX                     BOOLEAN
        EQUALITY MATCHING RULE          booleanMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hasSubordinates }
```

14.5 Esquema de sistema que soporta el control de acceso

Si una subinserción contiene información de control de acceso prescriptiva, su atributo **objectClass** contendrá el valor **accessControlSubentry (subinserción de control de acceso)**:

```
accessControlSubentry  OBJECT-CLASS  ::= {
        KIND        auxiliary
        ID          id-sc-accessControlSubentry }
```

Una subinserción de esta clase de objeto contendrá precisamente un atributo de ACI prescriptiva de un tipo que concuerda con el valor del atributo **id-sc-accessControlScheme** del punto específico de control de acceso correspondiente.

14.6 Esquema de sistema que soporta el modelo de atributos colectivos

Las subinserciones que soportan zonas administrativas específicas de atributos colectivos o zonas administrativas interiores se definen como sigue:

```
collectiveAttributeSubentry  OBJECT-CLASS  ::= {
        KIND          auxiliary
        ID            id-sc-collectiveAttributeSubentry }
```

Una subinserción de esta clase de objetos contendrá por lo menos un atributo colectivo.

Los atributos colectivos contenidos dentro de una subinserción de esta clase de objeto están disponibles conceptualmente para interrogación y filtrado en cada inserción dentro del alcance del atributo **subtreeSpecification** de la subinserción, pero son administrados por la subinserción.

El atributo operacional **collectiveExclusions** permite excluir determinados atributos colectivos de una inserción:

```
collectiveExclusions  ATTRIBUTE  ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        USAGE                           directoryOperation
        ID                              id-oa-collectiveExclusions }
```

Este atributo es facultativo para cada inserción.

Se puede utilizar el valor de **OBJECT IDENTIFIER id-oa-excludeAllCollectiveAttributes**, mediante su presencia como un valor del atributo **collectiveExclusions**, para excluir todos los atributos colectivos de una inserción.

14.7 Esquema de sistema que soporta valores por defecto de la aserción de contexto

Las subinserciones que proporcionan valores por defecto para las aserciones de contexto, se definen como sigue:

```
contextAssertionSubentry  OBJECT-CLASS  ::= {
        KIND              auxiliary
        MUST CONTAIN      {contextAssertionDefaults}
        ID                id-sc-contextAssertionSubentry }
```

Una subinserción de esta clase de objeto contendrá un atributo **contextAssertionDefaults**:

```
contextAssertionDefaults  ATTRIBUTE  ::= {
        WITH SYNTAX                     TypeAndContextAssertion
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-oa-contextAssertionDefault }
```

Cuando se evalúe un contexto y el usuario no proporcione aserción de contexto, el directorio suministrará valores por defecto de la aserción de contexto iguales a los valores de este atributo en la subinserción de aserción de contexto que controla la inserción a la que se está accediendo, como se describe en 8.8.2.2.

NOTA – El **TypeAndContextAssertion** se define en 7.6 (y su evaluación se define en 7.6.3) de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

14.8 Esquema de sistema que soporta el modelo de administración de servicio

```
serviceAdminSubentry  OBJECT-CLASS  ::= {
        KIND                    auxiliary
        MUST CONTAIN            { searchRules }
        ID                      id-sc-serviceAdminSubentry }
```

Una subinserción de esta clase de objeto contendrá un atributo operacional **searchRules**:

```
searchRules ATTRIBUTE ::= {
        WITH SYNTAX                     SearchRuleDescription
        EQUALITY MATCHING RULE          integerFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-oa-searchRules }
```

```
SearchRuleDescription ::= SEQUENCE {
    COMPONENTS OF          SearchRule,
    name                [28] SET SIZE (1 .. MAX) OF DirectoryString { ub-search } OPTIONAL,
    description         [29] DirectoryString { ub-search } OPTIONAL }
```

Un valor del atributo operacional **searchRules** es una regla de búsqueda que contiene restricciones de búsqueda efectivas, o bien es una regla de búsqueda ficticia que no especifica ninguna restricción. Esta regla de búsqueda ficticia se identifica porque tiene un id de cero y porque no tiene ningún componente **serviceType** (ni cualquier otro componente de **SearchRule** distinto de **id** y **dmdId**) es un identificador para el DMD controlador (véase 6.4).

14.9 Esquema de sistema que soporta grupos jerárquicos

```
hierarchyLevel ATTRIBUTE ::= {
    WITH SYNTAX                 HierarchyLevel
    EQUALITY MATCHING RULE      integerMatch
    ORDERING MATCHING RULE      integerOrderingMatch
    SINGLE VALUE                TRUE
    NO USER MODIFICATION        TRUE
    USAGE                       directoryOperation
    ID                          id-oa-hierarchyLevel }

HierarchyLevel ::= INTEGER

hierarchyBelow ATTRIBUTE ::= {
    WITH SYNTAX                 HierarchyBelow
    EQUALITY MATCHING RULE      booleanMatch
    SINGLE VALUE                TRUE
    NO USER MODIFICATION        TRUE
    USAGE                       directoryOperation
    ID                          id-oa-hierarchyBelow }

HierarchyBelow ::= BOOLEAN

hierarchyParent ATTRIBUTE ::= {
    WITH SYNTAX                 DistinguishedName
    EQUALITY MATCHING RULE      distinguishedNameMatch
    SINGLE VALUE                TRUE
    USAGE                       directoryOperation
    ID                          id-oa-hierarchyParent }
```

El atributo operacional **hierarchyLevel** estará presente en cualquier inserción que sea miembro de un grupo jerárquico. El directorio creará y mantendrá este atributo, y lo suprimirá cuando la inserción ya no sea miembro de un grupo jerárquico. Este atributo tomará el valor cero para el tope jerárquico y no estará presente en un miembro de familia vástago.

El atributo operacional **hierarchyBelow** indica si la inserción tiene algún vástago jerárquico. Un valor de **TRUE** indica que el vástago jerárquico existe. Un valor de **FALSE** o la ausencia del tipo de atributo indica que no existe El directorio creará y mantendrá este atributo, y lo suprimirá cuando la inserción ya no sea miembro de un grupo jerárquico.

El atributo operacional **hierarchyParent** estará presente en una operación Añadir inserción o Modificar inserción cuando una nueva inserción o una inserción existente se convierte en un vástago jerárquico. El valor de atributo será el nombre distinguido del progenitor inmediatamente jerárquico. Si el progenitor inmediatamente jerárquico es una inserción compuesta, el valor será el nombre distinguido del antepasado. En los demás casos, el directorio devolverá un error de actualización con el problema **parentNotAncestor**. Este atributo no estará presente en un miembro de familia vástago, en una inserción que no esté dentro de un grupo jerárquico, ni en una inserción que sea el tope jerárquico.

Cuando una inserción dentro de un grupo jerárquico es suprimida por la operación Suprimir inserción, todos sus vástagos jerárquicos son suprimidos del grupo jerárquico.

14.10 Mantenimiento del esquema de sistema

Es responsabilidad de los DSA mantener la coherencia de las subinserciones y atributos operacionales con el esquema del sistema. No deberá haber incoherencia entre distintos aspectos del esquema del sistema y entre el esquema del sistema y subinserciones y atributos operacionales.

El directorio ejecuta los procedimientos de adición y modificación de inserciones siempre que se añade una nueva subinserción al DIT o se modifica una subinserción existente. El directorio debe determinar si la operación propuesta violaría el esquema del sistema. Si es así, la modificación fracasa.

En particular, el directorio asegura que las subinserciones añadidas al DIT concuerdan con los valores del atributo **administrativeRole**, y que los atributos dentro de la subinserción concuerdan con los valores del atributo **objectClass** de la subinserción.

Se puede modificar el valor del atributo **administrativeRole** para permitir que algunas clases de subinserciones sean subordinadas de la inserción administrativa que no está aún presente. El valor del atributo **administrativeRole** no será modificado de modo que las subinserciones existentes sean incoherentes.

El directorio asegura también, cuando los valores de atributos operacionales son proporcionados por el directorio que éstos son correctos.

14.11 Esquema de sistema para subordinadas de primer nivel

El directorio hará cumplir las siguientes reglas y limitaciones de inserciones creadas subordinadas inmediatamente a la raíz DIT:

- Todas estas inserciones serán creadas como inserciones de puntos administrativos.
- Los atributos clase de objeto y denominación de tales inserciones serán los especificados en la Rec. CCITT X.660 | ISO/CEI 9834-1.

15 Administración del esquema de directorio

15.1 Visión de conjunto

La administración generalizada del esquema de directorio del DIT global se realiza mediante la administración independiente de los subesquemas de las zonas administrativas autónomas de los dominios de DIT que constituyen el DIT global.

La coordinación de la administración del esquema de directorio en fronteras entre dominios de DIT está sujeto a acuerdo bilateral entre las DMO y se sitúa frente del ámbito de esta Especificación de directorio.

Las capacidades administrativas de subesquema definidas en esta cláusula a los efectos de gestionar un dominio de DIT comprenden:

a) creación, supresión y modificación de subinserciones de subesquemas;

b) soporte del mecanismo de publicación para permitir que los DSA incluyan información de esquemas en intercambios de protocolo de vinculación operacional y que los DUA extraigan información de subesquema mediante el DAP;

c) reglamentación de subesquema para asegurar que cualesquiera operaciones de modificación se realizarán de acuerdo con la especificación de subesquema aplicable.

15.2 Objetos de política

Un objeto de política de subesquema puede ser uno de los siguientes:

- una zona administrativa de subesquemas;
- una inserción de objeto o alias dentro de una zona administrativa de subesquema;
- un atributo de usuario de esa inserción de objeto o alias.

Una zona administrativa autónoma puede ser designada como una zona administrativa específica de subesquema para administrar el subesquema. Esto se indicará mediante la presencia del valor **id-oa-subschemaAdminSpecificArea** en el atributo **administrativeRole** de la inserción administrativa asociada (además de la presencia del valor **id-oa-autonomusArea**, y posiblemente otros valores).

Esta zona administrativa autónoma puede ser dividida para desplegar y administrar el subesquema de las partes específicas. En este caso, las inserciones administrativas para cada una de las zonas administrativas específicas de subesquemas son indicadas por la presencia del valor **id-oa-subschemaAdminSpecificArea** en los atributos **administrativeRole** de estas inserciones.

15.3 Parámetros de política

Se utilizan parámetros de política de subesquema para expresar las políticas de la autoridad administrativa de subesquema. Estos parámetros, y los atributos operacionales utilizados para representarlos, son:

- un parámetro *estructura de DIT*: utilizado para definir la estructura de la zona administrativa de subesquema y para almacenar información sobre reglas de estructura de DIT obsoletas que algunas inserciones pueden haber identificado como su regla de estructura de DIT gobernante. Este parámetro se representa por los atributos operacionales **dITStructureRules** y **nameForms**;
- un parámetro *contenido de DIT*: utilizado para definir el contenido de inserciones de objeto y alias contenidas en la zona administrativa de subesquema y para almacenar información sobre reglas de contenido de DIT obsoletas que el directorio puede haber utilizado para determinar el contenido de algunas inserciones. Este parámetro se representa por los atributos operacionales **dITContentRules**, **objectClasses**, **attributeTypes**, **contextTypes** y **diTContextUse**;
- un parámetro *capacidad de concordancia*: utilizado para definir las capacidades de concordancia soportadas por reglas de concordancia aplicadas a tipos de atributos definidos en una zona administrativa de subesquema. Este parámetro se representa por los atributos operacionales **matchingRules** y **matchingRuleUse**.

Una subinserción es utilizada por la autoridad de subesquema para administrar el subesquema para la zona administrativa de subesquema. Con este propósito la subinserción de subesquema contiene los atributos operacionales que representan los parámetros de política utilizados para expresar políticas de subesquema. El atributo **subtreeSpecification** de una subinserción de subesquema especificará la zona administrativa de subesquema completa, es decir, será una secuencia vacía.

La subinserción de subesquema se especifica como sigue:

```
subschema  OBJECT-CLASS  ::= {
      KIND              auxiliary
      MAY CONTAIN       {
                        dITStructureRules |
                        nameForms |
                        dITContentRules |
                        objectClasses |
                        attributeTypes |
                        contextTypes |
                        dITContextUse |
                        matchingRules |
                        matchingRuleUse }
      ID                id-soc-subschema }
```

Los atributos operacionales de la subinserción de subesquema se definen en 15.7.

15.4 Procedimientos de política

Hay dos procedimientos de política asociados con la administración de subesquema:

- un procedimiento de modificación de subesquema;
- un procedimiento de modificación de inserción.

15.5 Procedimientos de modificación de subesquema

Una autoridad de subesquema puede administrar un subesquema de manera dinámica, y hacer modificaciones restrictivas de subesquema. Esto se puede efectuar modificando los valores de los atributos operacionales de subesquema mediante operaciones de modificación de directorio, cambiando efectivamente el subesquema que está vigente en la zona administrativa de subesquema. Una autoridad de subesquema puede también crear nuevas zonas de subesquema, o eliminar las zonas de subesquema existentes mediante la creación o supresión de subinserciones de subesquema, respectivamente.

Antes de que la autoridad de subesquema extienda las reglas de estructura o de contenido de DIT añadiendo una nueva regla, o añadiendo una clase de objeto auxiliar, o un atributo obligatorio o facultativo a una regla existente, la información de esquema referenciada deberá describirse en el atributo apropiado en la subinserción de subesquema. Las formas de nombre, clases de objeto, tipos de atributo, y reglas de concordancia que son referenciadas (directa o indirectamente) por **dITStructureRule**, **dITContentRule** o por un atributo **matchingRuleUse** no se suprimirán de la subinserción de subesquema.

Las definiciones de objetos de información, tales como clases de objeto, tipos de atributo, reglas de concordancia y formas de nombre que han sido registrados (es decir, que se les ha asignado un nombre de tipo identificador de objeto) son estáticas y no pueden ser modificadas. Para efectuar cambios en la semántica de esos objetos de información se requiere la asignación de nuevos identificadores de objeto.

Las reglas de estructura del DIT y de contenido del DIT pueden ser activas u obsoletas. Sólo se utilizan reglas activas para reglamentar el DIT. La identificación y preservación de reglas obsoletas es una conveniencia administrativa que permite localizar (y posiblemente reparar) inserciones añadidas según reglas antiguas que han sido cambiadas.

Este mecanismo obsoleto se utilizará cuando se efectúen modificaciones restrictivas a las reglas de estructura del DIT o de contenido del DIT que crean incoherencia en la DIB; en los demás casos, la regla activa apropiada se puede modificar directamente. El directorio permite suprimir reglas obsoletas en cualquier momento.

NOTA – El mecanismo obsoleto proporcionado en atributos operacionales de subesquema asegura que todas las inserciones con esquema obsoleto puedan ser identificadas y reparadas antes de suprimir el atributo operacional de subesquema obsoleto.

Es responsabilidad de la autoridad administrativa de subesquema mantener la coherencia de inserciones con el subesquema activo por medio del servicio abstracto de directorio, o por otros medios locales. Esto puede hacerse según convenga a la autoridad administrativa de subesquema. Es decir, no está definido cuándo deberá hacerse ese ajuste de inserciones incoherentes. Sin embargo, la supresión de reglas obsoletas antes de la localización y reparación de inserciones incoherentes dificultará más esta tarea.

15.6 Procedimientos de adición y modificación de inserciones

El directorio ejecuta procedimientos de adición y modificación de inserciones cuando se añade una nueva inserción al DIT o se modifica una inserción existente. El directorio determinará si la operación propuesta violaría una política de subesquema.

En particular, el directorio asegurará que las inserciones añadidas al DIT concuerdan con las reglas activas de estructura del DIT y de contenido del DIT.

El directorio permitirá la interrogación de inserciones que son incoherentes con sus reglas activas.

El directorio cumple reglas activas cuando tiene que modificar el DIB. Si una inserción no concuerda con su regla activa, se permitirá una petición de modificar la inserción si se repara una incoherencia existente, o no introduce una nueva incoherencia. Una petición que introduce una nueva incoherencia, fracasará.

Para cualquier inserción válida en una zona administrativa de subesquema válida sólo puede haber una clase de objeto estructural más subordinada en la cadena de superclases de clase de objeto estructural. Cuando se añade una inserción al DIT, el directorio determina esta clase de objeto estructural más subordinada a partir de los valores del atributo **objectClass** proporcionados y asociados permanentemente con ésta con la inserción mediante el atributo **structuralObjectClass** de la inserción.

Cuando se crea una inserción, se proporcionarán valores del atributo **objectClass**, de modo que el contenido de la inserción concuerde con la regla de contenido del DIT que rige la inserción. En particular, cuando un valor del atributo **objectClass** identifica una clase de objeto particular que tiene superclases distintas de **top**, se proporcionarán también valores para todas estas superclases. En los demás casos, la operación de directorio que crea la inserción fracasará.

Los usuarios de directorio pueden subsiguientemente añadir o suprimir valores del atributo **objectClass** para las clases de objetos auxiliares de una inserción. El contenido de una inserción seguirá siendo coherente con la regla de contenido del DIT que rige la inserción después de un cambio de los valores del atributo **objectClass**. En particular, cuando se añade o se suprime un valor del atributo **objectClass** que identifica una clase de objeto particular que tiene superclases distintas a **top**, se añadirán o suprimirán también los valores para todas estas superclases, salvo cuando tales superclases están también presentes en las cadenas de superclase asociadas con otros valores que no se añaden o se suprimen respectivamente.

15.7 Atributos de política de subesquema

En las siguientes subcláusulas se especifican los atributos operacionales de política de subesquema. Estos atributos están:

- presentes en la subinserción de subesquema. Los valores de estos atributos son administrados por operaciones de modificación de directorio utilizando el nombre distinguido de la subinserción de subesquema;
- disponibles para interrogación en todas las inserciones regidas por el subesquema.

El tipo parametrizado ASN.1 **DirectoryString { ub-schema }** utilizado en las siguientes definiciones se define en la Rec. UIT-T X.520 | ISO/CEI 9594-6.

Las reglas de concordancia de igualdad **integerFirstComponentMatch** y **objectIdentifierFirstComponentMatch** se definen también en la Rec. UIT-T X.520 | ISO/CEI 9594-6.

Para fines de gestión, se permiten facultativamente varios componentes **name** y un componente **description** legibles por seres humanos como componentes de varios atributos operacionales de política de subesquema definidos en las subcláusulas siguientes.

Varios atributos operacionales de política de subesquema definidos en las siguientes subcláusulas contienen un componente **obsolete**. Este componente se utiliza para indicar si la definición está activa u obsoleta en la zona administrativa de subesquema.

15.7.1 Atributo operacional reglas de estructura de DIT

El atributo operacional **dITStructureRules** define las reglas de estructura de DIT que están en vigor en un subesquema:

```
dITStructureRules ATTRIBUTE  ::=  {
        WITH SYNTAX                     DITStructureRuleDescription
        EQUALITY MATCHING RULE          integerFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITStructureRule }

DITStructureRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF       DITStructureRule,
        name            [1] SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description         DirectoryString { ub-schema } OPTIONAL,
        obsolete            BOOLEAN DEFAULT FALSE }
```

El atributo operacional **dITStructureRules** es multivaluado; cada valor define una regla de estructura DIT.

Los componentes de **dITStructureRule** tienen la misma semántica de la correspondiente definición ASN.1 en 13.7.6.

15.7.2 Atributo operacional reglas de contenido de DIT

El atributo operacional **dITContentRules** define las reglas de contenido de DIT que están vigentes en un subesquema. Cada valor del atributo operacional está rotulado por el identificador de objeto de clase de objeto estructural que corresponde:

```
dITContentRules ATTRIBUTE  ::=  {
        WITH SYNTAX                     DITContentRuleDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContentRules }

DITContentRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF           DITContentRule,
        name                [4] SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE }
```

El atributo operacional **dITContentRules** es multivaluado; cada valor define una regla de contenido DIT.

Los componentes de **dITContentRules** tienen la misma semántica que la definición ASN.1 correspondiente en 13.8.2.

15.7.3 Atributo operacional reglas de concordancia

El atributo operacional **matchingRules** especifica las reglas de concordancia utilizadas dentro de un subesquema:

```
matchingRules  ATTRIBUTE  ::= {
        WITH SYNTAX                     MatchingRuleDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-matchingRules }

MatchingRuleDescription  ::=  SEQUENCE {
        identifier              MATCHING-RULE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information       [0]   DirectoryString { ub-schema }       OPTIONAL }
                        -- describes the ASN.1 syntax
```

El componente **identifier** de un valor del atributo **matchingRules** es el identificador de objeto que identifica la regla de concordancia.

El componente **description** contiene una descripción de lenguaje natural de los algoritmos asociados con la regla.

El componente **information** contiene la definición en ASN.1 de la sintaxis de aserción de la regla.

Esa definición en ASN.1 se suministrará en forma de una producción de importación ASN.1 facultativa, seguida de producciones de asignación ASN.1 facultativas, seguida de una producción de tipo ASN.1. Todos los nombres de tipo definidos en los módulos de directorio se importan implícitamente y no requieren una importación explícita. Todos los nombres de tipo, importados o definidos mediante una asignación son locales a la definición de esta sintaxis. Si el tipo ASN.1 incluye una limitación definida por el usuario y no es uno de los tipos ASN.1 definidos en los módulos de directorio, el último UserDefinedConstraintParameter de la limitación será un parámetro real cuyo tipo gobernante es **SyntaxConstraint** y cuyo valor es el identificador de objeto asignado a la limitación.

```
SyntaxConstraint ::= OBJECT IDENTIFIER
```

NOTA 1 – Las producciones importación, asignación y tipo de ASN.1 se definen en la Rec. UIT-T X.680 | ISO/CEI 8824-1. **UserDefinedConstraintParameter**, se define en la Rec. UIT-T X.682 | ISO/CEI 8824-3.

NOTA 2 – Una definición ASN.1 típica es, simplemente, un nombre de tipo.

El atributo operacional **matchingRules** es multivaluado; cada valor describe una regla de concordancia.

15.7.4 Atributo operacional tipos de atributo

El atributo operacional **attributeTypes** especifica los tipos de atributo utilizados en un subesquema:

```
attributeTypes ATTRIBUTE ::= {
        WITH SYNTAX                     AttributeTypeDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-attributeTypes }

AttributeTypeDescription  ::=  SEQUENCE  {
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information       [0]   AttributeTypeInformation }
```

El componente **identifier** de un valor del atributo **attributeTypes** es el identificador de objeto que identifica el tipo de atributo.

El atributo operacional **attributeTypes** es multivaluado; cada valor describe un tipo de atributo:

```
AttributeTypeInformation  ::=  SEQUENCE  {
        derivation          [0]   ATTRIBUTE.&id OPTIONAL,
        equalityMatch       [1]   MATCHING-RULE.&id             OPTIONAL,
        orderingMatch       [2]   MATCHING-RULE.&id             OPTIONAL,
        substringsMatch     [3]   MATCHING-RULE.&id             OPTIONAL,
```

```
attributeSyntax     [4]  DirectoryString { ub-schema }OPTIONAL,
multi-valued        [5]  BOOLEAN                      DEFAULT TRUE,
collective          [6]  BOOLEAN                      DEFAULT FALSE,
userModifiable      [7]  BOOLEAN                      DEFAULT TRUE,
application              AttributeUsage               DEFAULT  userApplications }
```

Los componentes **derivation**, **equalityMatch**, **attributeSyntax**, **multi-valued**, **collective** y **application** tienen las mismas semánticas que las correspondientes piezas de notación introducidas por la correspondiente clase de objeto de información.

El componente **attributeSyntax** contiene una cadena de texto que proporciona la definición en ASN.1 de la sintaxis del atributo. Se suministrará esa definición en ASN.1 como se especifica para el componente **information** del atributo operacional reglas de concordancia.

15.7.5 Atributo operacional clases de objeto

El atributo operacional **objectClasses** especifica las clases de objeto utilizadas en un subesquema:

```
objectClasses ATTRIBUTE  ::= {
    WITH SYNTAX                 ObjectClassDescription
    EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
    USAGE                       directoryOperation
    ID                          id-soa-objectClasses }

ObjectClassDescription  ::=  SEQUENCE  {
    identifier              OBJECT-CLASS.&id,
    name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
    description             DirectoryString { ub-schema }       OPTIONAL,
    obsolete                BOOLEAN                             DEFAULT FALSE,
    information       [0]   ObjectClassInformation }
```

El componente **identifier** de un valor del atributo **objectClasses** es el identificador de objeto que identifica la clase de objeto.

El atributo operacional **objectClasses** es multivaluado; cada valor describe una clase de objeto:

```
ObjectClassInformation  ::=  SEQUENCE  {
    subclassOf              SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
    kind                    ObjectClassKind                          DEFAULT structural,
    mandatories       [3]   SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
    optionals         [4]   SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL }
```

Los componentes **subclassOf**, **kind**, **mandatories**, y **optionals** tienen las mismas semánticas que las correspondientes piezas de notación introducidas por la correspondiente clase de objeto de información.

15.7.6 Atributo operacional formas de nombre

El atributo operacional **nameForms** especifica las formas de nombre utilizadas en un subesquema:

```
nameForms ATTRIBUTE  ::= {
    WITH SYNTAX                 NameFormDescription
    EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
    USAGE                       directoryOperation
    ID                          id-soa-nameForms }

NameFormDescription  ::=  SEQUENCE  {
    identifier              NAME-FORM.&id,
    name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
    description             DirectoryString { ub-schema } OPTIONAL,
    obsolete                BOOLEAN DEFAULT FALSE,
    information       [0]   NameFormInformation }
```

El componente **identifier** de un valor del atributo **nameForms** es el identificador de objeto que identifica la clase de objeto.

El atributo operacional **nameForms** es multivaluado; cada valor describe una forma de nombre:

```
NameFormInformation ::= SEQUENCE {
        subordinate             OBJECT-CLASS.&id,
        namingMandatories       SET OF ATTRIBUTE.&id,
        namingOptionals         SET SIZE (1..MAX) OF ATTRIBUTE.&id OPTIONAL }
```

Los componentes **subordinate**, **mandatoryNamingAttributes** y **optionalNamingAttributes** tienen la misma semántica que las piezas de notación correspondientes introducidas por la clase de objeto de información correspondiente.

15.7.7 Atributo operacional de utilización de reglas de concordancia

El atributo operacional **matchingRuleUse** se usa para indicar los tipos de atributo a los cuales se aplica una regla de concordancia en un subesquema:

```
matchingRuleUse ATTRIBUTE ::= {
        WITH SYNTAX                     MatchingRuleUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-matchingRuleUse }

MatchingRuleUseDescription ::= SEQUENCE {
        identifier              MATCHING-RULE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }               OPTIONAL,
        obsolete                BOOLEAN                                     DEFAULT FALSE,
        information     [0]     SET OF ATTRIBUTE.&id }
```

El componente **identifier** de un valor del atributo **matchingRulesUse** es el identificador de objeto que identifica la regla de concordancia.

El componente **information** de un valor identifica el conjunto de tipos de atributo a que se aplica la regla de concordancia.

15.7.8 Atributo operacional Clase de objeto estructural

Cada inserción en el DIT tiene un atributo operacional **structuralObjectClass** que indica la clase de objeto estructural de la inserción:

```
structuralObjectClass ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-soa-structuralObjectClass }
```

15.7.9 Atributo operacional regla de estructura gobernante

Cada inserción en el DIT, con excepción de las inserciones de puntos administrativos que no tengan subinserciones de subesquema, tiene un atributo operacional **governingStructureRule** que indica la regla de estructura gobernante de la inserción:

```
governingStructureRule ATTRIBUTE ::= {
        WITH SYNTAX                     INTEGER
        EQUALITY MATCHING RULE          integerMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-soa-governingStructureRule }
```

15.7.10 Atributo operacional tipos de contexto

El atributo operacional **contextTypes** especifica los tipos de contexto utilizados dentro de un subesquema:

```
contextTypes ATTRIBUTE  ::= {
        WITH SYNTAX                     ContextDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-contextTypes }

ContextDescription  ::=  SEQUENCE  {
        identifier              CONTEXT.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString {ub-schema}    OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                                 DEFAULT FALSE,
        information     [0]     ContextInformation }
```

El componente **identifier** de un valor del atributo operacional **contextTypes** es el identificador de objeto que identifica el tipo de contexto.

El atributo operacional **contextTypes** es multivaluado; cada valor describe un tipo de contexto:

```
ContextInformation  ::=  SEQUENCE  {
        syntax                  DirectoryString { ub-schema } ,
        assertionSyntax         DirectoryString { ub-schema } OPTIONAL }
```

Los componentes **syntax** y **assertionSyntax** tienen las mismas semánticas que los elementos correspondientes de la notación introducida en las clases de objeto de información correspondientes.

El componente **syntax** y el componente **assertionSyntax** contienen cada uno una cadena de texto que proporciona la definición en ASN.1 de la sintaxis de contexto y la sintaxis de aserción de contexto, respectivamente. Esa definición en ASN.1 se suministrará en forma de una producción de importación ASN.1 facultativa, seguida de producciones de asignación ASN.1 facultativas, seguida de una producción de tipo ASN.1. Todos los nombres de tipo definidos en los módulos de directorio se importan implícitamente y no requieren una importación explícita. Todos los nombres de tipo, importados o definidos mediante una asignación son locales a la definición de esta sintaxis. Si el tipo ASN.1 incluye una limitación definida por el usuario y no es uno de los tipos ASN.1 definidos en los módulos de directorio, el último UserDefinedConstraintParameter de la limitación será un parámetro real cuyo tipo gobernante es **SyntaxConstraint** y cuyo valor es el identificador de objeto asignado a la limitación.

NOTA 1 – Las producciones ASN.1 Imports, Assignment y Type se definen en la Rec. UIT-T X.680 | ISO/CEI 8824-1. **UserDefinedConstraintParameter** se define en la Rec. UIT-T X.682 | ISO/CEI 8824-3. **SyntaxConstraint** se define en 15.7.3.

NOTA 2 – Una definición ASN.1 típica es, simplemente, un nombre tipo.

15.7.11 Atributo operacional uso de contexto DIT

Se utiliza el atributo operacional **dITContextUse** para indicar los contextos que se utilizarán o pueden utilizarse con un atributo:

```
dITContextUse ATTRIBUTE  ::= {
        WITH SYNTAX                     DITContextUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContextUse }

DITContextUseDescription  ::=  SEQUENCE  {
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information     [0]     DITContextUseInformation }
```

El componente **identifier** de un valor del atributo operacional **dITContextUse** es el identificador de objeto del tipo de atributo al que se aplica. El valor **id-oa-allAttributeTypes** indica que se aplica a todos los tipos de atributo.

El componente **information** de un valor identifica los tipos de contexto obligatorios y facultativos asociados al tipo de atributo identificado por **identifier**:

```
DITContextUseInformation  ::=  SEQUENCE  {
        mandatoryContexts       [1]     SET SIZE (1..MAX) OF CONTEXT.&id  OPTIONAL,
        optionalContexts        [2]     SET SIZE (1..MAX) OF CONTEXT.&id  OPTIONAL }
```

SECCIÓN 7 – ADMINISTRACIÓN DE SERVICIO DE DIRECTORIO

16 Modelo de administración de servicio

Esta cláusula contiene un modelo de la forma según la cual la autoridad administrativa puede controlar, constreñir y ajustar el servicio con respecto a lo que un usuario puede especificar en una petición de búsqueda, lectura o modificación de inserción y la información que se ha de devolver.

16.1 Definiciones

A los fines de esta Especificación de directorio se aplican las siguientes definiciones.

16.1.1 tipo de atributo presente efectivamente: Un tipo de atributo que está presente en al menos un elemento de filtro no negado en cada subfiltro de un filtro de búsqueda y que cumple los requisitos especificados para ese tipo de atributo en la regla de búsqueda pertinente. Para las definiciones de elementos de filtro negados y no negados, véase 7.8.1 de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

16.1.2 regla de búsqueda directora: Una regla de búsqueda a la que se atiene una operación particular y que ha sido seleccionada para dirigir esa operación.

16.1.3 servicio denominado: Una colección de tipos de servicio que, juntos, proporcionan un servicio global, por ejemplo, un servicio de páginas blancas.

16.1.4 perfil de atributo de petición: La especificación de lo que se necesita para un elemento filtro de manera que el tipo de atributo correspondiente esté presente efectivamente.

16.1.5 tipo de atributo de petición: Un tipo de atributo que, de acuerdo con una especificación de regla de búsqueda, puede estar representado en el filtro de una operación de búsqueda.

16.1.6 regla de búsqueda: La especificación detallada de los aspectos relativos a las constricciones/mejoras del servicio de un tipo de servicio determinado destinado principalmente a una clase de usuario determinada y adaptada a un grupo particular de usuarios.

16.1.7 tipo de servicio: Identificación globalmente única de una capacidad de servicio para un fin determinado dentro de un ámbito bien definido, por ejemplo, la capacidad de búsqueda de un tipo particular de inserciones dentro de una zona del DIT. No todos los aspectos de un tipo de servicio pueden estar a disposición de todos los usuarios.

16.1.8 subfiltro: Un componente booleano de un filtro que comprende solamente operadores AND lógicos de elementos de filtro no negados y de elementos de filtro negados, es decir, que puede expresarse de manera informal como un operador NOT (elemento de filtro). Cualquier filtro puede expresarse en una forma canónica que comprenda un operador OR lógico de subfiltros, según se analiza en el anexo Q

16.1.9 clase de usuario: Un conjunto identificado de usuarios que, por sus funciones, su posición en una organización, etc., pueden invocar determinados aspectos de los tipos de servicio dentro de un servicio denominado. Diferentes grupos de usuarios identificados por sus nombres dentro de una clase de usuario pueden ver variaciones en el servicio proporcionado. Un grupo de usuarios puede abarcar varias clases de usuario.

16.2 Modelo de tipo de servicio/clase de usuario

El servicio abstracto del directorio especificado en la Rec. UIT-T-T X.511 | ISO/CEI 9594-3 es la representación de todas las capacidades de servicio ofrecidas por las Epecificaciones de directorio. Un *tipo de servicio* es un subconjunto de ese servicio para efectuar una función determinada, por ejemplo, la búsqueda de un cierto tipo de objetos dentro de un ámbito definido.

Un *servicio denominado* es una colección de tipos de servicio con un objetivo determinado, por ejemplo, proporcionar un servicio de páginas blancas, un tipo particular de servicio de páginas amarillas, etc.

Un tipo de servicio se realiza principalmente a través de la operación de búsqueda, pero también vía otras operaciones que pueden especificar selección de información de inserción, es decir, operaciones de lectura y modificación de inserción. A los efectos de la administración del servicio se considera que una petición **read** o **modifyEntry** es equivalente de alguna manera a una petición **search** con **subset** igual a **baseObject** y **filter** igual a **and: {}**. La administración del servicio no influye en la información que puede ser modificada por una operación de modificación de inserción. Esto es algo en lo que sólo influye el control de acceso.

Un tipo de servicio se identifica mediante un identificador de objeto que le da una identificación global única. Dependiendo de su cometido, posición en la organización, etc., diferentes *clases de usuario*, pueden tener percepciones algo distintas de un tipo de servicio. Una clase de usuario se identifica mediante un entero del que sólo se precisa que sea único en un DMD. Diferentes DMD podrían asignar un identificador diferente a lo que cabría considerar como la misma clase de usuario. Sin embargo, las autoridades administrativas, que cooperan proporcionando un servicio denominado común a través de varios DMD, coordinarán previsiblemente los identificadores de grupos de usuarios. Incluso para una clase de usuario particular, puede haber variaciones en el servicio a disposición de los usuarios de esa clase. Dichas variaciones se basan en los nombres distinguidos de los usuarios. Por ejemplo, los usuarios de una determinada clase de usuario de un país pueden no tener exactamente la misma visión de un tipo de servicio que los usuarios de la misma clase de usuario de otro país, debido, por ejemplo, a las leyes locales sobre privacidad. La definición de un tipo de servicio para un grupo de usuarios se expresa mediante una *regla de búsqueda* que especifica los detalles relativos a cómo se ha de efectuar la operación.

El tipo de servicio y la clase de usuario a la que está destinado principalmente se indican en una regla de búsqueda.

Un grupo de usuarios puede abarcar varias clases de usuario. Un usuario de una clase de usuario podría quizás utilizar también reglas de búsqueda destinadas sobre todo a otras clases de usuario, por ejemplo, a los usuarios de una clase de usuario con una capacidad mayor se les concederían también permisos destinados a clases de usuario a las que por lo general se ofrecen capacidades de servicio inferiores.

Un grupo de usuarios no es identificado directamente por una regla de búsqueda, sino que se identifica indirectamente por tener el permiso de invocación de esa regla de búsqueda. Un grupo de usuarios puede invocar cualquier regla de búsqueda cuya invocación le esté permitida. Si un determinado usuario tiene permiso para invocar varias reglas de búsqueda del mismo tipo de servicio pero para diferentes grupos de usuarios, los procedimientos definidos en estas Especificaciones de directorio seleccionarán, permaneciendo todo lo demás igual, la regla de búsqueda cuyo identificador de grupo de usuarios sea más alto. De esta manera, la autoridad administrativa puede, controlar la selección, mediante la asignación adecuada de identificadores de clase de usuario.

16.3 Zonas administrativas específicas de servicio

Una zona administrativa autónoma puede ser designada como *zona administrativa específica de servicio* para difundir y administrar las reglas de búsqueda. Esto se indicará por la presencia del valor **id-ar-serviceSpecificArea** en el atributo **administrativeRole** de la inserción administrativa asociada (además de por la presencia del valor **id-ar-autonomousArea**, y posiblemente otros valores).

Esa zona administrativa autónoma se puede dividir para difundir y administrar las reglas de búsqueda en divisiones específicas. En este caso, las inserciones administrativas de cada una de las zonas administrativas específicas de servicio son indicadas por la presencia del valor **id-ar-serviceSpecificArea** en los atributos **administrativeRole** de esas inserciones. Las políticas de servicio para zonas administrativas específicas de servicio superiores no son subordinadas importantes de dicha inserción administrativa.

Si esa zona administrativa autónoma no se divide, hay una zona administrativa específica de servicio única para las reglas de búsqueda que abarca la zona administrativa autónoma en su totalidad.

En el modelo de información de directorio se representan una o más reglas de búsqueda mediante una subinserción, denominada *subinserción de servicio*, cuyo atributo **objectClass** contiene el valor **id-sc-serviceAdminSubentry**, definido en 14.8. Una subinserción de esta clase será la subordinada inmediata de una inserción administrativa cuyo atributo **administrativeRole** contenga el valor **id-ar-serviceSpecificArea**.

La fase de evaluación dentro de una zona administrativa específica de servicio depende, entre otras cosas, del objeto básico utilizado para la operación, posiblemente después de eliminar la referencia de alias. Las reglas de búsqueda están vinculadas por tanto a las inserciones. Cuando el objeto básico de una operación ha sido determinado, las reglas de búsquedas vinculadas a esa inserción son candidatas a la dirección de la búsqueda. Los vínculos entre reglas de búsqueda dentro de una subinserción e inserciones dentro de la zona administrativa específica de servicio los establece el atributo operacional **subtreeSpecification** de la subinserción. Las inserciones identificadas por los valores del atributo operacional **subtreeSpecification** están vinculadas de esta manera a las reglas de búsqueda situadas en la misma subinserción.

Una inserción particular puede ser asociada con las reglas de búsqueda desde múltiples subinserciones; éstas pueden tener las mismas o diferentes especificaciones de subárbol. A la inversa, partes diferentes de la zona administrativa pueden ser el objetivo de una subinserción, utilizando valores múltiples de la especificación del subárbol.

Los argumentos de una operación se pueden validar con una regla de búsqueda utilizando un algoritmo llamado *función de validación de búsqueda*.

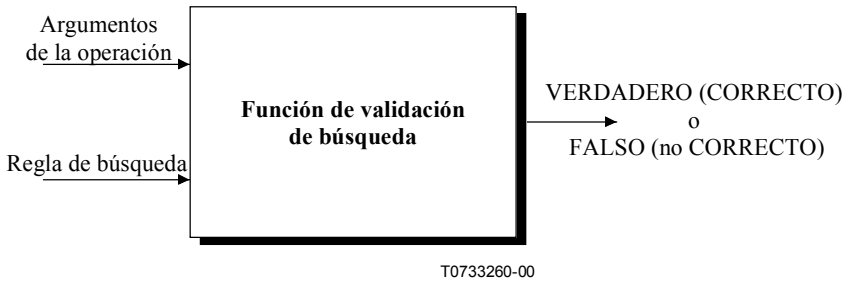


Figura 16 – Función de validación de búsqueda

Una operación es válida y está autorizada a seguir su curso si, y solamente si, la función de validación de búsqueda produce VERDADERO al menos para una de las reglas de búsqueda asociadas con el objeto básico disponibles para la operación. Para que una regla de búsqueda esté disponible para una operación, el solicitante debe tener permiso de *invocación* del valor del atributo que representa la regla de búsqueda. Si sólo hay una regla de búsqueda disponible con la que cumple la operación, dicha regla se denomina *regla de búsqueda directora* de esa operación, es decir, es la regla de búsqueda que se utiliza cuando la operación sigue avanzando. Si hay varias reglas de búsqueda como ésa, la opción local selecciona una de ellas como regla de búsqueda directora. El procedimiento de selección de una regla de búsqueda directora se indica en 19.3.2.2.1 de la Rec. UIT-T X.518 | ISO/CEI 9594-4. La regla de búsqueda directora está permanentemente asociada, por tanto, con la operación para su evaluación dentro de la zona administrativa específica de servicio. Éste es también el caso cuando parte de la operación la llevan a cabo otros DSA que retienen partes de esa zona administrativa específica de servicio.

Las autoridades administrativas pueden optar por:

- reunir varias reglas de búsqueda que requieren permisos de invocación diferentes en una sola subinserción (por lo que haría falta control de acceso hasta el nivel del valor del atributo si esos permisos de invocación variaran de un valor a otro); o
- reunir reglas de búsqueda con los mismos permisos de control de acceso en subinserciones distintas, con lo que los permisos de control de acceso podrían otorgarse en base a los permisos del atributo completo; diferentes subinserciones podrían contener entonces permisos de control de acceso diferentes.

Si no se dispone de ninguna regla de búsqueda para una operación en la que se especifica una inserción de objeto básico dentro de una zona administrativa específica de servicio, o si la función de validación de la búsqueda devuelve FALSO para todas las reglas de búsqueda disponibles, se rechaza la operación con un error.

Sin embargo, si una zona administrativa específica de servicio no tiene subinserciones, no hay ninguna constricción del servicio asociada con esa zona.

Puede haber usuarios que no estén limitados por las restricciones del servicio, por ejemplo, los administradores, y puede haber inserciones para las que no se requiera ninguna restricción cuando sirven como inserciones de objetos básicos, por ejemplo, inserciones situadas en la parte baja del DIT. La autoridad administrativa puede incluir por tanto reglas de búsqueda especiales, *reglas de búsqueda vacías*.

Un grupo jerárquico situado dentro de una zona administrativa específica del servicio ha de estar contenido por completo dentro esa zona.

El ámbito de aplicación de una operación de búsqueda no puede traspasar el límite de una zona administrativa específica de servicio. La Rec. UIT-T X.518 | ISO/CEI 9594-4 especifica procedimientos que no permiten a una operación de búsqueda que empieza dentro de una determinada zona administrativa específica de servicio extenderse fuera de esa zona incluso cuando se eliminan referencias de alias durante la evaluación de la búsqueda. De manera similar, una búsqueda que empieza fuera de una zona administrativa específica de servicio no puede extenderse dentro de esa zona.

16.4 Introducción a las reglas de búsqueda

Las reglas de búsqueda son expresiones de políticas que, por un lado, constriñen y ajustan las operaciones que se pueden llevar a cabo en una región del DIT y, por otro, ayudan a su ejecución guiando el proceso de la operación. Una regla de búsqueda tiene las siguientes características principales:

- establece los requisitos que deberá cumplir una operación si la operación se ha de llevar a cabo en base a esa regla de búsqueda;
- especifica el ajuste de la petición de operación;

- especifica los detalles de la evaluación de la operación, por ejemplo, las políticas de relajación si se encuentran demasiadas o demasiado pocas inserciones para la operación de búsqueda; y
- contiene especificaciones relativas a la selección de información de inserción.

Cuando comienza el procesamiento de una operación, la inserción básica de la misma corresponde a una o más subinserciones de servicio cuyos valores de especificación de subárbol incluyen esa inserción básica. Existe la posibilidad, por tanto, de que se identifique un cierto número de reglas de búsqueda candidatas. Los detalles de la operación se evalúan por comparación con estas reglas de búsqueda candidatas. Una búsqueda sólo se puede llevar a cabo si se puede encontrar una regla de búsqueda compatible.

16.5 Subfiltros

Si se elabora una regla de búsqueda para controlar la operación de búsqueda, la regla puede especificar un conjunto de tipos de atributo que quizás estén presentes en un filtro de una petición **search**. Estos atributos se llaman *tipos de atributo de petición* de la regla de búsqueda. En el filtro no estarán presentes en forma alguna otros tipos de atributo, ni negados ni no negados. Esta subcláusula especifica con más detalle lo que se requiere para que un tipo de atributo esté presente en un filtro de búsqueda. Una regla de búsqueda especifica además los requisitos que han de cumplir las combinaciones válidas de tipos de atributo de petición. Dichos requisitos podrían ser la obligatoriedad de la presencia de determinados tipos de atributo, que al menos uno de dos tipos de atributo esté presente, que no se permita un tipo de atributo sin que esté presente otro, etc. Para seguir profundizando en la manera de expresar combinaciones, conviene introducir el concepto de *subfiltros*.

De acuerdo con el cálculo proposicional, cualquier filtro se puede representar como una secuencia de subfiltros separados por operadores OR lógicos, como se indica a continuación:

$$f = f_1 + f_2 + ... + f_r$$

donde cada subfiltro, f_i, es una secuencia de elementos de filtro o elementos de filtro negados separados por operadores AND lógicos, que se puede expresar de la siguiente forma:

$$f_i = f_{i1} f_{i2} ... f_{is}$$

siendo f_{ij} un elemento de filtro o su negación.

En el anexo Q se expone con más detalle el concepto de subfiltro.

Para que un filtro cumpla una regla de búsqueda, cada subfiltro deberá cumplir con esa regla de búsqueda.

Para que un elemento de filtro represente efectivamente un tipo de atributo en un subfiltro, hace falta que cumpla los requisitos del *perfil de atributo de petición* del tipo de atributo. Los perfiles de atributo de petición forman parte de la especificación de reglas de búsqueda. Si al menos un elemento de filtro de un tipo de atributo de cada subfiltro cumple con el perfil de atributo de petición para ese tipo de atributo, se dice que el tipo de atributo es un *tipo de atributo presente efectivamente*.

16.6 Requisitos de filtro

Para que un tipo de atributo esté presente efectivamente en un filtro, el tipo de atributo o, si se fija la opción **includeSubtypes** del perfil de atributo de petición, uno de sus subtipos deberá estar presente al menos en un elemento de filtro no negado de cada subfiltro. Ese elemento de filtro no negado deberá cumplir los requisitos siguientes:

- Habrá de ser un elemento de filtro no negado que no sea de uno de los tipos siguientes:

  greaterOrEqual;

  lessOrEqual;

  present o **contextPresence** a menos que lo permita de manera explícita el perfil de atributo petición.
- Tendrá que cumplir la especificación del perfil de atributo de petición para ese tipo de atributo.
- Si es un elemento de filtro **extensibleMatch**, el tipo de atributo deberá estar especificado en el componente **type** de la **MatchingRuleAssertion**.

  NOTA – Si no se introduce la última restricción, este elemento de filtro podría incluir implícitamente un número no especificado de tipos de atributo en el filtro de búsqueda y, de esa manera, perjudicar el procedimiento de validación de búsqueda.

Si un tipo de atributo está representado en un filtro, estará presente efectivamente.

Se permite que haya elementos de filtro **extensibleMatch** sin el componente **type** en el filtro. Su presencia no afecta a la validación de la búsqueda con las reglas de búsqueda. Sin embargo, ese elemento de filtro sólo se aplicará a atributos cuyos tipos sean tipos de atributo de petición, es decir, que están representados en la regla de búsqueda directora por un perfil de atributo de petición (véase 16.10.2).

16.7 Selección de información de atributo en base a las reglas de búsqueda

Fuera de una zona administrativa específica de servicio, la información de atributo devuelta se basa en el componente **selection** de la petición de operación, modificada posiblemente por el **operationContext** de los **CommonArguments**, y cualesquiera valores por defecto del contexto establecidos dentro de una zona administrativa específica del valor por defecto del contexto o por valores locales por defecto del contexto. Para una operación de búsqueda, la selección de información puede ser modificada también por el componente **matchedValuesOnly** en el **SearchArgument**. Sin embargo, cuando una operación es controlada por una regla de búsqueda directora, esta regla de búsqueda puede especificar la información que ha de ser devuelta. Si tal es el caso, la información de atributo de usuario devuelta será la intersección de lo que especifica la regla de búsqueda directora y lo que habría sido devuelto si no hubiera habido regla de búsqueda directora. Si la selección de información de inserción en el componente **selection** especifica selección de atributos operacionales, se aplicará la misma regla para los atributos operacionales. Si la selección de información de inserción no especifica devolución de información de atributo operacional, la información de atributo operacional devuelta sólo será determinada por la regla de búsqueda directora.

Una regla de búsqueda directora puede especificar qué información de atributo se ha de devolver por completo con independencia de los tipos de atributo que puedan ser especificados en un filtro **search**.

Cuando se tenga que devolver información en base a grupos jerárquicos, la selección de información de atributo desde esas inserciones se basará en el principio anterior, con la salvedad de que las especificaciones **matchedValuesOnly** no tienen efecto.

NOTA – La selección de miembros de familia no se rige por el principio anterior (véase 16.10.6).

16.8 Aspectos de las reglas de búsqueda relativos al control de acceso

Las reglas de búsqueda proporcionan algunas capacidades adicionales de control de acceso, además de las capacidades descritas en la cláusula 18. En un planteamiento orientado al servicio, es necesario aplicar restricciones a la forma según la cual las operaciones pueden ser formuladas y sobre la información que puede ser devuelta. Esto es algo que debería basarse no sólo en la identidad del usuario sino también en el tipo de servicio y la clase de usuario, permitiendo de esa manera a las autoridades administrativas adaptar el servicio en base a la calidad de la información, consideraciones relativas a la tarificación, etc.

Las capacidades de control de acceso definidas en la cláusula 18 se utilizan para asegurar que sólo los grupos de usuario pertinentes pueden invocar las reglas de búsqueda. Estas capacidades pueden proteger también la información de manera que nunca puedan acceder a la misma grupos de usuarios particulares.

Es posible que un DSA que oculta información originada en una zona administrativa específica de servicio no tenga las reglas de búsqueda con las que controlar las restricciones impuestas a esa información. Al igual que ocurre en el control de acceso (véase 18.8.2), los administradores de seguridad deben saber que un DSA con capacidad de ocultación puede entrañar un importante riesgo para la seguridad de otros DSA.

16.9 Aspectos de las reglas de búsqueda relativos a los contextos

Puesto que las aserciones de contextos pueden ser parte de un elemento de filtro para la operación de búsqueda, las especificaciones de las reglas de búsqueda han de tenerlos en cuenta. La inclusión de contextos en la regla de búsqueda introduce nuevas capacidades en la característica del contexto que pueden simplificar los requisitos impuestos a las implementaciones de DUA y DSA.

La característica básica del contexto permite al usuario especificar contextos para el filtro de búsqueda y para la selección de información de inserción; y permite a las autoridades administrativas establecer valores por defecto de contextos dentro de una zona administrativa específica del valor por defecto del contexto. Dichos valores se aplican indiscriminadamente a todos los usuarios y a todo tipo de servicios. Sin embargo, la característica del contexto proporcionada por las reglas de búsqueda permite al usuario especificar una información de contexto mínima y, a las autoridades administrativas, formular especificaciones de contexto individuales para cada regla de búsqueda. Además, es posible, como se indica en 16.8, proporcionar control de acceso a modo de función formulando adecuadamente la especificación del contexto de la regla de búsqueda. La utilización de especificaciones de contexto en reglas de búsqueda podría provocar la redundancia del establecimiento de zonas administrativas específicas del valor por defecto del contexto.

16.10 Especificación de las reglas de búsqueda

El tipo de datos ASN.1 **SearchRule** da la sintaxis de una regla de búsqueda.

```
SearchRule ::= SEQUENCE {
        COMPONENTS OF           SearchRuleId,
        serviceType         [1] OBJECT IDENTIFIER                           OPTIONAL,
        userClass           [2] INTEGER                                     OPTIONAL,
        inputAttributeTypes [3] SEQUENCE SIZE (0..MAX) OF RequestAttribute  OPTIONAL,
        attributeCombination [4] AttributeCombination                       DEFAULT and : { },
        outputAttributeTypes [5] SEQUENCE SIZE (1..MAX) OF ResultAttribute  OPTIONAL,
        defaultControls     [6] ControlOptions                              OPTIONAL,
        mandatoryControls   [7] ControlOptions                              OPTIONAL,
        searchRuleControls  [8] ControlOptions                              OPTIONAL,
        familyGrouping      [9] FamilyGrouping                              OPTIONAL,
        familyReturn        [10] FamilyReturn                               OPTIONAL,
        relaxation          [11] RelaxationPolicy                           OPTIONAL,
        additionalControl   [12] SEQUENCE SIZE (1..MAX) OF AttributeType    OPTIONAL,
        allowedSubset       [13] AllowedSubset                              DEFAULT '111'B,
        imposedSubset       [14] ImposedSubset                              OPTIONAL,
        entryLimit          [15] EntryLimit                                 OPTIONAL }

SearchRuleId ::= SEQUENCE {
        id                      INTEGER,
        dmdId               [0] OBJECT IDENTIFIER }

AllowedSubset ::= BIT STRING { baseObject (0), oneLevel (1), wholeSubtree (2) }

ImposedSubset ::= ENUMERATED { baseObject (0), oneLevel (1), wholeSubtree (2) }

RequestAttribute ::= SEQUENCE {
        attributeType           ATTRIBUTE.&id ({ SupportedAttributes }),
        includeSubtypes     [0] BOOLEAN                                     DEFAULT FALSE,
        selectedValues      [1] SEQUENCE SIZE (0..MAX) OF ATTRIBUTE.&Type
                                ({ SupportedAttributes }{ @attributeType })  OPTIONAL,
        defaultValues       [2] SEQUENCE SIZE (0..MAX) OF SEQUENCE {
                entryType           OBJECT-CLASS.&id  OPTIONAL,
                values              SEQUENCE OF ATTRIBUTE.&Type
                                ({ SupportedAttributes }{ @attributeType }) }   OPTIONAL,
        contexts            [3] SEQUENCE SIZE (0..MAX) OF ContextProfile    OPTIONAL,
        contextCombination  [4] ContextCombination                          DEFAULT and : { },
        matchingUse         [5] SEQUENCE SIZE (1..MAX) OF MatchingUse       OPTIONAL }

ContextProfile ::= SEQUENCE {
        contextType     CONTEXT.&id({SupportedContexts}),
        contextValue    SEQUENCE SIZE (1..MAX) OF CONTEXT.&Assertion
                        ({SupportedContexts}{@contextType})  OPTIONAL }

ContextCombination ::= CHOICE {
        context             [0] CONTEXT.&id({SupportedContexts}),
        and                 [1] SEQUENCE OF ContextCombination,
        or                  [2] SEQUENCE OF ContextCombination,
        not                 [3] ContextCombination }

MatchingUse ::= SEQUENCE {
        restrictionType     MATCHING-RESTRICTION.&id ({SupportedMatchingRestrictions}),
        restrictionValue    MATCHING-RESTRICTION.&Restriction
                                ({SupportedMatchingRestrictions}{@restrictionType}) }

-- Definition of the following information object set is deferred, perhaps to standardized
-- profiles or to protocol implementation conformance statements. The set is required to
-- specify a table constraint on the components of SupportedMatchingRestrictions

SupportedMatchingRestrictions  MATCHING-RESTRICTION ::= { ... }

AttributeCombination ::= CHOICE {
        attribute           [0] AttributeType,
        and                 [1] SEQUENCE OF AttributeCombination,
        or                  [2] SEQUENCE OF AttributeCombination,
        not                 [3] AttributeCombination }
```

```
ResultAttribute ::= SEQUENCE {
        attributeType           ATTRIBUTE.&id ({ SupportedAttributes }),
        outputValues            CHOICE {
                selectedValues          SEQUENCE OF ATTRIBUTE.&Type
                                                ({ SupportedAttributes }{ @attributeType }),
                matchedValuesOnly       NULL } OPTIONAL,
        contexts                [0] SEQUENCE SIZE (1..MAX) OF ContextProfile OPTIONAL }

ControlOptions ::= SEQUENCE {
        serviceControls         [0]     ServiceControlOptions DEFAULT { },
        searchOptions           [1]     SearchControlOptions DEFAULT { searchAliases },
        hierarchyOptions        [2]     HierarchySelections OPTIONAL }

EntryLimit ::= SEQUENCE {
        default             INTEGER,
        max                 INTEGER }

RelaxationPolicy ::= SEQUENCE {
        basic               [0]     MRMapping DEFAULT { },
        tightenings         [1]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        relaxations         [2]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        maximum             [3]     INTEGER OPTIONAL,    -- mandatory if tightenings is present
        minimum             [4]     INTEGER DEFAULT 1 }

MRMapping ::= SEQUENCE {
        mapping             [0]     SEQUENCE SIZE (1..MAX) OF Mapping           OPTIONAL,
        substitution        [1]     SEQUENCE SIZE (1..MAX) OF MRSubstitution    OPTIONAL }

Mapping ::= SEQUENCE {
        mappingFunction     OBJECT IDENTIFIER (CONSTRAINED BY { -- shall be an
                        -- object identifier of a mapping-based matching algorithm -- } ),
        level               INTEGER DEFAULT 0 }

MRSubstitution ::= SEQUENCE {
        attribute                   AttributeType,
        oldMatchingRule     [0]     MATCHING-RULE.&id OPTIONAL,
        newMatchingRule     [1]     MATCHING-RULE.&id OPTIONAL }
```

16.10.1 Componentes de identificación de regla de búsqueda

El componente **id** permite la identificación exclusiva de las reglas de búsqueda dentro de un DMD. El valor cero se reserva para la regla de búsqueda vacía. En 16.3 se describe la finalidad de una regla de búsqueda vacía.

El componente **dmdId** da la identificación única del DMD que ha establecido la regla de búsqueda. Este componente, junto con **id**, permite la definición de la identificación global y única de una regla de búsqueda.

NOTA – La manera de gestionar esta unicidad queda fuera del alcance de la presente especificación.

El componente **id** (con el valor de cero) y los componentes **dmdId** son los únicos componentes de interés para la regla de búsqueda vacía.

El componente **serviceType** es un identificador de objeto que identifica el tipo de servicio soportado por la regla de búsqueda. Este componente estará siempre presente salvo en el caso de una regla de búsqueda vacía.

El componente **userClass** indica la clase de usuario hacia la que se orienta principalmente la regla de búsqueda. Para un tipo de servicio dado puede haber varias reglas de búsqueda que especifiquen la misma clase de usuario. Este componente estará siempre presente excepto en el caso de una regla de búsqueda vacía.

16.10.2 Perfiles de atributo de petición

El componente **inputAttributeTypes** deberá especificar perfiles de atributo de petición para todos los tipos de atributo que estén o puedan estar representados en un filtro de **search**. Si un filtro de **search** incluye un elemento de filtro para un tipo de atributo no representado por un perfil de atributo de petición, fallará la validación de la búsqueda con esta regla de búsqueda. El tipo de datos **RequestAttribute** especifica el requisito que se impone a un elemento de filtro para el tipo de atributo especificado en el elemento de filtro que ha de estar presente efectivamente. Si este componente está

ausente, la regla de búsqueda no impone ninguna restricción a la presencia de tipos de atributo, es decir, cualquier operación cumple con este componente. Si el componente está presente, pero vacío, sólo una petición **read**, una petición **modifyEntry** o una petición **search** con filtro por defecto (**and: { }**) cumple con este componente.

Los subcomponentes siguientes son importantes en todos los tipos de operación controlados por reglas de búsqueda:

a) El subcomponente **attributeType**, que especifica el tipo de atributo para el que se aplica esta especificación. Es el único subcomponente obligatorio. Sólo puede haber una especificación **RequestAttribute** para un tipo de atributo determinado dentro de una regla de búsqueda. Si es éste el único subcomponente, excepto quizás el subcomponente **includeSubtypes**, no hay restricciones impuestas a los elementos de filtro de búsqueda para este tipo de atributo, salvo que si esos elementos de filtro están en el filtro, al menos uno de ellos deberá ser no negado.

b) El subcomponente **includeSubtypes**, que especifica que este perfil de atributo de petición puede ser satisfecho por un elemento de filtro para un subtipo de este tipo de atributo.

Los subcomponentes siguientes son importantes solamente en la operación de búsqueda:

c) El subcomponente **selectedValues**, que proporciona un conjunto de valores de atributo del tipo dado en **attributeType**. Si este tipo de atributo está representado en el filtro, habrá por lo menos un elemento de filtro no negado para este tipo de atributo que concuerde al menos con un valor de este subcomponente. De no ser así, este tipo de atributo no está presente efectivamente en el filtro.

   Si este subcomponente está ausente, la concordancia anterior evalúa a VERDADERO.

   Si se da un conjunto vacío de valores de atributo, este tipo de atributo sólo puede estar presente efectivamente en:

   – un elemento de filtro **present** si el subcomponente **Contexts** no está presente; o

   – un elemento de filtro **contextPresent** si el subcomponente **Contexts** está presente.

d) El subcomponente **defaultValues**, que no afecta a la evaluación de una petición **search** con la regla de búsqueda, pero controla la operación de búsqueda cuando se ha seleccionado una regla de búsqueda como regla de búsqueda directora. Este componente proporciona un conjunto de valores de atributo del tipo dado en **attributeType**. Si se define un elemento de filtro utilizando este tipo de atributo dentro del filtro, pero no está ningún atributo de este tipo presente en una inserción (o una agrupación de familias), el elemento de filtro evalúa a VERDADERO (o a FALSO si es negado) si el elemento de filtro concuerda con uno de los valores de este subcomponente. Si este subcomponente está ausente, no hay valores por defecto.

   Si este componente está presente, pero vacío, ello indica que toma todos los valores posibles, es decir, un elemento de filtro para este tipo de atributo siempre evalúa a VERDADERO (o a FALSO si es negado) si el tipo de atributo está ausente en una inserción.

   NOTA 1 – Lo anterior refleja la situación en la que un elemento de filtro será ignorado si está ausente un atributo del tipo referenciado.

   Si una inserción contiene un atributo de este tipo, se efectúa la concordancia normal con este atributo.

e) El subcomponente **contexts**, que especifica los tipos de contexto a los que se permite estar representados en un elemento de filtro para este tipo de atributo. Un tipo de contexto particular no deberá estar representado más de una vez en este subcomponente.

   – Si el subcomponente está ausente, cualquier información de contexto puede estar presente en un elemento de filtro para este tipo de atributo.

   – Si el subcomponente está presente, sólo los tipos de contexto especificados en este subcomponente pueden estar presentes en un elemento de filtro para este tipo de atributo. Si es una secuencia vacía, no puede estar presente ninguna información de contexto en un elemento de filtro para este tipo de atributo.

   – Si sólo se especifica un tipo de contexto, cualquier valor de contexto de ese tipo puede estar presente en la aserción de contexto.

   – Si en este subcomponente están presentes valores de contexto para un tipo de contexto determinado, sólo esos valores pueden estar presentes en la aserción de contexto correspondiente de un elemento de filtro.

   Si la especificación de contexto del elemento de filtro no cumple con lo anterior, el elemento de filtro no cumple con el perfil de atributo de petición para el tipo de atributo.

f) El subcomponente **contextCombination**, que especifica la combinación válida de los tipos de contexto indicados en el subcomponente **contexts** dentro de este perfil de atributo de petición. Si este subcomponente está ausente, no hay ninguna restricción impuesta a la combinación de estos tipos de contexto. Si está presente una combinación no válida de tipos de contexto, el elemento de filtro no cumple con el perfil de atributo de petición para el tipo de atributo. Este componente puede especificar que determinados tipos de contexto estén presentes incondicionalmente.

g) El subcomponente **matchingUse**, que se utiliza para especificar posibles constricciones impuestas a la utilización de la regla de concordancia aplicable, por ejemplo, longitudes mínimas para la concordancia de subcadenas. La regla de concordancia aplicable es la que de hecho vaya a utilizarse antes de cualquier relajación pero después de una posible sustitución básica. Los detalles respecto a las restricciones y cómo se evalúan se describen como parte de la especificación de la restricción. Si este subcomponente especifica una restricción de la concordancia definida para la regla de concordancia que se ha de utilizar, se comprueba si esa restricción de la concordancia es vulnerada o si se han de aplicar aspectos no sustentados de la regla de concordancia. Cuando tal sea el caso:

   – si no se ha fijado la opción de control de búsqueda **performExactly**, la implementación puede utilizar una regla local sobre la forma de aplicar la regla de concordancia de manera diferente;

     NOTA 2 – Esa regla local requiere que se aplique una capacidad de personalización para la regla de concordancia en cuestión.

   – si se ha fijado la opción de control de búsqueda **performExactly** o si no es posible aplicar una regla local, la petición **search** no cumple con esta regla de búsqueda.

16.10.3 Combinaciones de atributos

El componente **attributeCombination** especifica la combinación válida de los tipos de atributo de petición indicados en el componente **inputAttributeTypes**. Si este componente está ausente o tiene el valor por defecto (**and : { }**), no hay ninguna restricción impuesta a la combinación de tipos de atributo de petición y todos los tipos de operaciones pertinentes cumplen con este componente. Si está presente una combinación no válida de tipos de atributo de petición, fallará la validación de la búsqueda con esta regla de búsqueda. Este componente puede especificar que determinados tipos de atributo estén presentes efectivamente de manera incondicional en el filtro. Este componente estará ausente si **inputAttributeTypes** está ausente o vacío. Si está presente y tiene un valor no por defecto, sólo una operación de búsqueda con un filtro no por defecto tendría la posibilidad de cumplir con este componente.

16.10.4 Atributos en resultado

El componente **outputAttributeTypes** especifica qué tipos de atributo (o sus subtipos si no se ha fijado la opción de control de servicio **noSubtypeSelection**) podrían estar presentes en el resultado, a reserva del control de acceso (véase 16.7). Si una inserción concordada o una inserción compuesta no contiene ninguno de los atributos definidos en este componente, no se incluye en el resultado. Una regla similar se aplica en el caso de un miembro de familia individual marcado como el resultado de la concordancia o por operaciones especificadas por atributos de control en el componente **additionalControl**. Si miembros de esa familia no contienen ninguno de los tipos de atributo definidos por este componente, tal situación corresponde a la no marcación de manera explícita del miembro de la familia y de todos sus subordinados. El tipo de datos **ResultAttribute** especifica los detalles respecto a cómo se representará en el resultado ese tipo de atributo. Este componente no afecta a la validación de la búsqueda. Si está ausente, la regla de búsqueda no afecta a la selección de información de inserción, salvo que sea especificado posiblemente por los componentes **familyReturn** y **additionalControl**. Este componente tiene los siguientes subcomponentes:

a) El subcomponente **attributeType**, que especifica el tipo de atributo para el que se aplica esta especificación. Es el único subcomponente obligatorio. Sólo puede haber una especificación **ResultAttribute** para un tipo de atributo determinado dentro de una regla de búsqueda.

b) El subcomponente **outputValues**, que especifica qué valores de atributo de este tipo de atributo podrían ser devueltos. El conjunto de valores puede ser restringido más aún por el subcomponente de contexto, la selección de información de inserción proporcionada por el peticionario, el control de acceso, etc. Si este subcomponente está ausente, podrían ser devueltos todos los valores de atributo. La opción **selectedValues** proporciona un conjunto de valores de atributo del tipo dado en **attributeType**. Sólo los valores indicados pueden ser valores de atributo que han de ser devueltos. La opción **matchedValuesOnly** especifica que sólo los valores de atributo del atributo que contribuyó a que el filtro devolviera a VERDADERO vía elementos de filtro distintos de los presentes podrían ser devueltos (véase 10.2.2 de la Rec. UIT-T X.511 | ISO/CEI 9594-3 una definición del término "contribuyó").

c) El subcomponente **context**, que contiene un conjunto de perfiles de contexto que especifican la información de valor de atributo que se ha de devolver para este tipo de atributo.

- Si este subcomponente está ausente, la regla de búsqueda no impone ninguna restricción respecto a qué valores de atributo pueden ser devueltos en base a los contextos.
- Si en este subcomponente no está incluido un tipo de contexto, no se devuelve ninguna información de contexto de este tipo con ningún valor de atributo devuelto de este tipo de atributo.
- Si un perfil de contexto no incluye el tipo de datos **contextValue**, se devuelven todos los valores de contexto del tipo de contexto con cada valor de atributo.
- Si uno o más perfiles de contexto incluyen el tipo de datos **contextValue**, cada uno de esos perfiles de contexto se trata como una **ContextAssertion** que se ha de aplicar con relación a los valores de atributo especificados en 8.8.2.4. Sólo se devuelven aquellos valores de atributo para los que esta evaluación produzca VERDADERO para todos esos tipos de contexto. Si esta selección da como resultado la no devolución de ningún valor para este tipo de atributo, el atributo no se incluye en el resultado. De manera similar, si la selección da como resultado el que no quede ninguna información para una inserción, esa inserción no se devuelve.
- Si todos los valores de atributo de este tipo de atributo devueltos tienen el mismo par {tipo de contexto, valor de contexto}, para devolver, ese valor de contexto se elimina de todos los valores de atributo. Si un contexto se queda, por ello, sin ningún valor, se elimina por completo.

  NOTA – Esto permitirá adaptar un servicio de manera que un usuario con un equipo es de tipo sencillo reciba en la mayoría de los casos información sin contextos.

16.10.5 Controles de servicio y búsqueda

El componente **defaultControls**, si está presente, se utiliza para especificar la fijación de bits no fijados de manera explícita para una operación en las **ServiceControlOptions** dentro de los controles de servicio del argumento de la operación, y si la operación es una búsqueda, las **SearchControlOptions** y **HierarchySelections**. Si está ausente alguna opción específica, se utiliza el elemento **defaultControls**, caso de estar presente.

Si todo el subcomponente **hierarchyOptions** está ausente de **defaultControls**, o está ausente **defaultControls**, no se utilizará selección de jerarquía. Si el componente **hierarchySelection** está presente en un argumento **search** y no especifica más que **self**, fallará la validación de la búsqueda con la regla de búsqueda. Los elementos correspondientes en **mandatoryControls** y **searchRuleControls** serán omitidos.

Si el componente **defaultControls** está ausente por completo, se considerará que toma el valor por defecto normalizado **{ serviceControls { }, searchOptions {searchAliases} }.**

El componente **mandatoryControls** especifica, fijando bits específicos, las opciones de cadenas de bits que estarán presentes según lo especificado en **defaultControls**; si algún bit especificado por **mandatoryControls** difiere en las opciones suministradas por el usuario de **defaultControls**, fallará la validación de la búsqueda con la regla de búsqueda. Los bits no especificados por el componente **mandatoryControls** se supone que son cero. Si la operación es una lectura o una modificación de inserción, sólo se considera el subcomponente **serviceControls**.

El componente **searchRuleControls** especifica, fijando bits específicos, las opciones de cadenas de bits que se han de tomar del componente **defaultControls** en vez de las opciones suministradas por el usuario. Los bits no especificados por el componente **searchRuleControls** se supone que son cero. Si la operación es una lectura o una modificación de inserción, sólo se considera el subcomponente **serviceControls**.

NOTA – Si el usuario suministra $U_{0\ a\ p}$ en una operación de búsqueda, y los bits por defecto son $D_{0\ a\ N}$, el resultado de aplicar el componente **defaultControls** es una cadena de bits $C_{0\ a\ N}$ en donde los bits 0 a p se toman de U y los restantes de D. La validación de la búsqueda con la regla de búsqueda fallará si C&M no es igual a D&M, donde C significa $C_{0\ a\ N}$, "&" representa una operación AND binaria, y $M_{0\ a\ N}$ es la cadena de bits especificada por **mandatoryControls**. De otro modo, el valor de las opciones que se utiliza es (C&~S | D&S) siendo S la cadena de bits especificada por **searchRuleControls**, ~S es la negación binaria y '|' representa una operación OR binaria. Esta última manipulación tiene el efecto de eliminar los bits indicados por **searchRuleControls** y sustituirlos por los valores de bits por defecto. El componente **familyGrouping** especifica una agrupación de familia que, si está presente, tiene precedencia con respecto (es decir, sustituye) a la **familyGrouping** en **CommonArguments** del argumento de búsqueda.

16.10.6 Especificaciones de familia

El componente **familyGrouping** especifica una selección de agrupación de familia que, si está presente, tiene precedencia con respecto (es decir, sustituye) a la **familyGrouping** en **CommonArguments** del argumento **search**.

El componente **familyReturn** especifica la selección de devolución de miembros de la familia. Ajusta la especificación dada por **familyReturn** en la **EntryInformationSelection** (o su valor por defecto) del argumento **search**. La especificación de la regla de búsqueda tiene precedencia con respecto a la especificación en el componente **memberSelect**, mientras que la especificación del argumento **search** tiene precedencia con respecto al componente **familySelect**, es decir, si el componente **familySelect** está presente en el argumento **search**, cualquier posible componente **familySelect** de la regla de búsqueda deberá ser ignorado.

16.10.7 Control de relajación

El componente **relaxation** define la política de relajación utilizando el constructivo **RelaxationPolicy**. El mismo constructivo puede estar incluido en una petición **search** del componente **relaxation**. Aquí se describe el procedimiento asociado con este constructivo, abarcando tanto el caso en el que se incluye en una regla de búsqueda como el caso en que se incluye en una petición **search**. En 10.2.2 de la Rec. UIT-T X.511 | ISO/CEI 9594-3 se dan especificaciones adicionales para el caso en que **RelaxationPolicy** está incluido en la regla de búsqueda y en la petición **search**.

La **RelaxationPolicy** tiene los subcomponentes siguientes:

a) El subcomponente **basic** que, si está presente, define **MRMapping**, es decir, un conjunto de sustituciones de reglas de concordancia y/o funciones de concordancia basadas en el establecimiento de la correspondencia que se aplican a un filtro **search** para la primera evaluación (sin endurecimiento o relajación por tanto). Esto permite la selección de una concordancia más apropiada que la original. Si se omite el elemento o se suministra con un conjunto vacío, es preciso utilizar para la primera evaluación todas las reglas de concordancia normales sin aplicar ninguna concordancia basada en el establecimiento de la correspondencia.

b) El subcomponente **tightenings** que, si está presente, comprende una secuencia de sustituciones y establecimientos de correspondencias, definidos cada uno de ellos por **MRMapping**, que se han de aplicar en el orden dado, uno cada vez, cuando el número de inserciones concordadas sea muy elevado (superior a **maximum**).

c) El subcomponente **relaxations** que, si está presente, comprende una secuencia de sustituciones y establecimientos de correspondencias, definido cada uno de ellos por **MRMapping**, que se han de aplicar en el orden dado, uno cada vez, cuando el número de inserciones concordadas sea muy reducido (inferior a **minimum**).

d) El subcomponente **maximum**, que estará siempre presente si está presente **tightenings**, y especifica entonces el número de inserciones encontradas, por encima del cual se aplica un endurecimiento.

e) El subcomponente **minimum**, que especifica el número de inserciones encontradas para el que (o por debajo del que) se ha de aplicar una relajación; si está ausente, el valor por defecto es cero.

NOTA 1 – La relajación/restricción no son afectadas por la opción de control de búsqueda **performExactly**.

Las sustituciones de reglas de concordancia y los establecimientos de correspondencias se definen mediante elementos **MRMapping**, cada uno de los cuales consta de una secuencia de elementos **Mapping** y una secuencia de elementos **MRSubstitution**. El orden de secuencia de estos elementos carece de importancia.

Un elemento **Mapping** tiene los siguientes componentes:

a) El componente **mappingFunction**, que identifica una función de correspondencia basada en el establecimiento de la correspondencia con el cuadro de correspondencias asociado que se ha de aplicar.

b) El componente **level**, que identifica el nivel de relajación (o de endurecimiento si es negativo) que se ha de aplicar para la concordancia basada en el establecimiento de la correspondencia. Este componente será ignorado si se ha fijado el **&userControl** para la concordancia basada en el establecimiento de la correspondencia y el control de búsqueda **extendedArea** está incluido en la petición **search**, en cuyo caso se aplica el valor especificado en **extendedArea**.

NOTA 2 – Para la sustitución y el establecimiento de la correspondencia de **basic**, el **level** deberá fijarse en muchos casos en cero.

Un elemento **MRSubstitution** tiene los siguientes componentes:

a) **attribute**, que describe el atributo al que se ha de aplicar la sustitución.

b) **oldMatchingRule**, que es la regla de concordancia que se ha de sustituir. Si está ausente, se aplica la regla de concordancia aplicable previamente del tipo especificado para el atributo, si es que hay uno. Para sustitución básica o, si la sustitución básica no se lleva a cabo, para la primera sustitución de relajación/endurecimiento, la concordancia aplicable es la que se hubiera utilizado de otro modo. Para las sustituciones subsiguientes, la regla de concordancia aplicable es la aportada por la sustitución previa. Si este subcomponente especifica una regla de concordancia distinta de la regla de concordancia aplicable previamente, no se efectúa ninguna sustitución.

NOTA 3 – Por ejemplo, si el elemento de filtro es del tipo **equality** y por ello selecciona una regla de concordancia de igualdad, y este subcomponente especifica una regla de concordancia de subcadenas, no se efectúa ninguna sustitución.

c) **newMatchingRule**, que es el identificador de objeto para la regla de concordancia sustituta que se ha de utilizar en lugar de la antigua regla de concordancia. Si está ausente, cualquier elemento de filtro correspondiente se evalúa como VERDADERO para un elemento no negado, y FALSO para un elemento negado (es decir, de acuerdo con **id-mr-nullMatch**).

Si se especifica una regla de concordancia para la que existe una restricción de concordancia a propósito del tipo de atributo [véase 16.2.2, inciso g)], la petición **search** no cumplirá la regla de búsqueda directora; no se especifica una regla de concordancia no soportada, se abandona la sustitución y no se lleva a cabo ninguna otra para el tipo de atributo.

NOTA 4 - Se supone que un DSA no permitirá la presencia de sustituciones no válidas en una regla de búsqueda.

Un atributo puede tener múltiples elementos **MRSubstitution** en un **MRMapping** siempre que la combinación de atributo y **oldMatchingRule** (si está presente) sea única. Cuando **oldMatchingRule** está ausente de una **MRSubstitution**, pero está presente en otra **MRSubstitution**, esta última tiene precedencia al establecer la correspondencia de la regla de concordancia definida por **oldMatchingRule**.

16.10.8 Componente de control adicional

El componente **additionalControl** permite que el efecto de una regla de búsqueda directora se adapte a un entorno específico en el que se requiere control adicional de una operación de búsqueda. Especifica uno o más tipos de atributo. La semántica, la sintaxis y la ubicación de ese tipo de atributo de control referenciado por este componente se definirán como parte de la definición del atributo de control. La especificación correspondiente puede efectuarse fuera de estas Especificaciones de directorio. Un atributo de control especificado incluye una parte de los procedimientos de su definición que se han de ejecutar en base a la información proporcionada por el atributo de control.

Este componente no afecta a la función de validación de búsqueda.

Un atributo de control podría situarse de manera que afectara a varias inserciones, por ejemplo, en un punto administrativo específico de servicio o en una subinserción de administración de servicio. También puede situarse en inserciones individuales. Cuando un atributo de control se sitúa en inserciones individuales, sólo puede afectar a la selección de información de inserción de esas inserciones. Un atributo de control puede hacer que determinadas inserciones o miembros de familia sean *desmarcados explícitamente*, lo que evitará su presencia en el resultado de la búsqueda.

NOTA 1 – Situando un atributo de control en el punto administrativo específico de servicio, dicho atributo puede afectar a la manera en que se efectúa la concordancia. Por ejemplo, se puede establecer la correspondencia de un tipo de atributo especificado en un elemento de filtro con, o ser complementado por, un conjunto de tipos de atributo (tipos de atributo "amistosos") con relación a los cuales se puede efectuar la concordancia de alguna manera definida para obtener, por ejemplo, el mismo efecto que se deriva de la subtipificación de atributos. De manera similar, un atributo de control podría ajustar la información de inserción devuelta.

NOTA 2 – Situando un atributo de control en una inserción dada es posible tener en cuenta requisitos individuales, por ejemplo, para abarcar las necesidades de protección de los datos personales.

Si como resultado del procesamiento de uno o más atributos de control se marcan o desmarcan inserciones compuestas, esa marcación o desmarcación se efectuará aplicando la especificación de devolución de familia (especificada en **familyReturn** en la **EntryInformationSelection** o revocada por el componente regla de búsqueda **familyReturn**). Si la desmarcación explícita da como resultado el que no se devuelva ningún miembro de una inserción compuesta, la inserción compuesta se elimina por completo del resultado.

16.10.9 Componentes diversos

El componente **allowedSubset** especifica las opciones válidas de la especificación **subset** de la petición de búsqueda. Este componente de regla de búsqueda se ignora si el componente de regla de búsqueda **imposedSubset** está presente y no se ha fijado el control de búsqueda **useSubset** en una petición **search**. Como valor por defecto es posible cualquier opción **subset**. Si el parámetro **subset** de una petición **search** no especifica un valor compatible con este componente de regla de búsqueda, fallará la validación de la búsqueda con la regla de búsqueda. Para que una operación de lectura o modificación de inserción cumpla con este componente, debe incluir el valor **baseObject**.

El componente **imposedSubset** especifica un **subset** que sustituye a la especificación **subset** en una petición **search**. Si este componente no está presente o si se ha fijado el control de búsqueda **useSubset** en la petición **search**, no se efectúa ninguna sustitución y se aplica la restricción expresada por el **allowedSubset**. Este componente será ignorado cuando se evalúe una petición **read** o **modifyEntry** con relación a una regla de búsqueda.

El componente **entryLimit** tiene dos subcomponentes. El subcomponente **default** indica el límite de tamaño que ha de ser impuesto por el directorio si no se ha fijado el control de servicio **sizeLimit**. El subcomponente **max** indica cuál es el valor máximo admisible del control de servicio **sizeLimit**. Si se rebasa, se reduce el **sizeLimit** efectivo a este valor **max**. Este componente será ignorado cuando se evalúe una petición **read** o **modifyEntry** con una regla de búsqueda.

16.10.10 Clases de objeto de información ASN.1

Las clases de objeto de información **SEARCH-RULE**, **REQUEST-ATTRIBUTE** y **RESULT-ATTRIBUTE** se proporcionan para facilitar la documentación de las reglas de búsqueda:

```
SEARCH-RULE ::= CLASS {
        &dmdId                  OBJECT IDENTIFIER,
        &serviceType            OBJECT IDENTIFIER                       OPTIONAL,
        &userClass              INTEGER                                 OPTIONAL,
        &InputAttributeTypes    REQUEST-ATTRIBUTE                       OPTIONAL,
        &combination            AttributeCombination                    OPTIONAL,
        &OutputAttributeTypes   RESULT-ATTRIBUTE                        OPTIONAL,
        &defaultControls        ControlOptions                          OPTIONAL,
        &mandatoryControls      ControlOptions                          OPTIONAL,
        &searchRuleControls     ControlOptions                          OPTIONAL,
        &familyGrouping         FamilyGrouping                          OPTIONAL,
        &familyReturn           FamilyReturn                            OPTIONAL,
        &additionalControl      AttributeType                           OPTIONAL,
        &relaxation             RelaxationPolicy                        OPTIONAL,
        &allowedSubset          AllowedSubset                           DEFAULT '111'B,
        &imposedSubset          ImposedSubset                           OPTIONAL,
        &entryLimit             EntryLimit                              OPTIONAL,
        &id                     INTEGER UNIQUE }
WITH SYNTAX {
        DMD ID                  &dmdId
        [ SERVICE-TYPE          &serviceType ]
        [ USER-CLASS            &userClass ]
        [ INPUT ATTRIBUTES      &InputAttributeTypes ]
        [ COMBINATION           &combination ]
        [ OUTPUT ATTRIBUTES&OutputAttributeTypes ]
        [ DEFAULT CONTROL       &defaultControls ]
        [ MANDATORY CONTROL     &mandatoryControls ]
        [ SEARCH-RULE CONTROL   &searchRuleControls ]
        [ FAMILY-GROUPING       &familyGrouping ]
        [ FAMILY-RETURN         &familyReturn ]
        [ ADDITIONAL CONTROL    &additionalControl ]
        [ RELAXATION            &relaxation ]
        [ ALLOWED SUBSET        &allowedSubset ]
        [ IMPOSED SUBSET        &imposedSubset ]
        [ ENTRY LIMIT           &entryLimit ]
        ID                      &id }

REQUEST-ATTRIBUTE ::= CLASS {
        &attributeType          ATTRIBUTE.&id,
        &SelectedValues         ATTRIBUTE.&Type                                 OPTIONAL,
        &DefaultValues          SEQUENCE {
                                        entryType   OBJECT-CLASS.&id            OPTIONAL,
                                        values      SEQUENCE OF ATTRIBUTE.&Type } OPTIONAL,
        &contexts               SEQUENCE OF ContextProfile                      OPTIONAL,
        &contextCombination     ContextCombination                              OPTIONAL,
        &MatchingUse            MatchingUse                                     OPTIONAL,
        &includeSubtypes        BOOLEAN                                         DEFAULT FALSE
}
WITH SYNTAX {
        ATTRIBUTE TYPE          &attributeType
        [ SELECTED VALUES       &SelectedValues ]
        [ DEFAULT VALUES        &DefaultValues ]
        [ CONTEXTS              &contexts ]
        [ CONTEXT COMBINATION   &contextCombination ]
        [ MATCHING USE          &MatchingUse ]
        [ INCLUDE SUBTYPES      &includeSubtypes ] }
```

```
RESULT-ATTRIBUTE ::= CLASS {
        &attributeType              ATTRIBUTE.&id,
        &outputValues               CHOICE {
                selectedValues      SEQUENCE OF ATTRIBUTE.&Type,
                matchedValuesOnly   NULL }                          OPTIONAL,
        &contexts                   ContextProfile                  OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE              &attributeType
        [ OUTPUT VALUES             &outputValues ]
        [ CONTEXTS                  &contexts ] }
```

16.11 Definición de restricción de concordancia

A una autoridad administrativa quizás le convenga imponer restricciones sobre la manera de aplicar una regla de concordancia. Por ejemplo, una restricción impuesta a una regla de concordancia de subcadenas puede especificar longitudes mínimas de las subcadenas proporcionadas en un elemento de filtro de búsqueda. Esa restricción es de naturaleza permanente y no tiene características dinámicas, como ocurre en el caso de relajación de la búsqueda.

Dentro de una zona administrativa específica de servicio, las restricciones pueden ser aplicadas por la propia construcción de las reglas de búsqueda, y ese es el único lugar en donde es posible introducir restricciones de la concordancia.

Las restricciones de la concordancia se pueden definir como valores de la clase de objeto de información **MATCHING-RESTRICTION**:

```
MATCHING-RESTRICTION ::= CLASS {
        &Restriction,
        &Rules          MATCHING-RULE.&id,
        &id             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
        RESTRICTION     &Restriction
        RULES           &Rules
        ID              &id }
```

Para la restricción de una regla de concordancia definida utilizando esta clase de objeto de información:

a) **&Restriction** es la sintaxis de la restricción de la concordancia que se ha de aplicar.

b) **&Rules** es el conjunto de reglas de concordancia al que se puede aplicar esta restricción. Las restricciones sólo pueden ser especificadas por una regla de concordancia básica, es decir, una regla de concordancia que no tenga reglas de concordancia de progenitor en su definición.

c) **&id** es el identificador de objeto que se le ha asignado.

Para cualquier regla de concordancia se pueden definir varias restricciones de la concordancia, pero sólo se puede aplicar una en una situación determinada.

16.12 Función de validación de búsqueda

La función de validación de búsqueda es una función abstracta que se utiliza para determinar la compatibilidad de una petición de búsqueda con una regla de búsqueda particular. La función de validación de búsqueda da como resultado VERDADERO si la petición de búsqueda cumple la regla de búsqueda. De no ser así, da como resultado FALSO. Para que una petición de búsqueda cumpla una regla de búsqueda:

– no deberán estar presentes de ninguna forma tipos de atributo distintos de los representados por los **inputAttributeTypes** en el filtro de búsqueda, negado o no negado;

– si un tipo de atributo está presente en un filtro, debería estar presente efectivamente;

  NOTA – Esto significa que un tipo de atributo no deberá estar representado solamente por elementos de filtro negados.

– deberá cumplirse la condición para la presencia efectiva de atributos de petición especificados por el componente **attributeCombination** de la regla de búsqueda;

– si hay perfiles de atributo de petición que incluyen el subcomponente **selectedValues**, los atributos correspondientes deberán estar representados solamente por elementos de filtro no negados;

- la especificación **subset** del argumento de búsqueda deberá cumplir la especificación **subset** de la regla de búsqueda;
- el control obligatorio especificado por el componente **mandatoryControls** deberá ser exactamente como en **defaultControls** para la regla de búsqueda.

Para que un tipo de atributo representado por uno o más elementos de filtro en un subfiltro esté presente efectivamente en ese subfiltro, al menos uno de los elementos de filtro deberá cumplir con la especificación **RequestAttribute** para ese tipo de atributo, es decir:

- los elementos de filtro deberán ser del tipo especificado en 16.6;
- si el subcomponente **selectedValues** está presente y no vacío en el perfil de atributo de petición, el elemento de filtro deberá concordar con este subcomponente;
- la especificación del contexto en el elemento de filtro deberá cumplir las especificaciones de contexto en el perfil de atributo de petición;
- la especificación de la regla de concordancia en el elemento de filtro deberá cumplir las especificaciones de reglas de concordancia en el perfil de atributo de petición;
- deberá cumplirse cualquier restricción impuesta a la concordancia.

En la cláusula 13 de la Rec. UIT-T X.511 | ISO/CEI 9594-3 se especifica el procedimiento detallado de validación de la búsqueda.

SECCIÓN 8 – SEGURIDAD

17 Modelo de seguridad

17.1 Definiciones

Esta especificación de directorio utiliza los siguientes términos definidos en la Rec. CCITT X.800 | ISO 7498-2:

- control de acceso;
- autenticación;
- política de seguridad;
- confidencialidad;
- integridad.

En esta Especificación de directorio se definen los términos siguientes.

17.1.1 esquema de control de acceso: El medio que permite controlar el acceso a la información de directorio y posiblemente los propios derechos de acceso.

17.1.2 elemento protegido; ítem protegido: Un elemento de información de directorio al cual el acceso puede controlarse separadamente. Los ítems protegidos de directorio son inserciones, atributos, valores de atributo, nombres.

17.2 Políticas de seguridad

El directorio existe en un entorno donde diversas autoridades administrativas controlan el acceso a su porción de la DIB. Dicho acceso se ajusta generalmente a alguna política de seguridad controlada por administración (véase la Rec. UIT-T X.509 | ISO/CEI 9594-8).

Dos aspectos o componentes de la política de seguridad que influyen en el acceso al directorio son los procedimientos de autenticación y el esquema de control de acceso.

NOTA – La cláusula 18 define dos métodos de control de acceso conocidos como control de acceso básico y control de acceso simplificado y la cláusula 19 define el control de acceso reglado. Estos métodos se pueden utilizar junto con controles administrativos locales; sin embargo, como la política administrativa local no tiene representación normalizada, no puede ser comunicada en información sombreada.

17.2.1 Procedimientos y mecanismos de autenticación

Los procedimientos y mecanismos de autenticación en el contexto de directorio incluyen los métodos para verificar y propagar, donde sea necesario:

- la identidad de los DSA y usuarios de directorio;
- la identidad del origen de la información recibida en un punto de acceso.

NOTA 1 – La autoridad administrativa puede establecer disposiciones diferentes para la autenticación de usuarios administrativos, en comparación con las establecidas para la autenticación de usuarios no administrativos.

Los procedimientos de autenticación de uso general se definen en la Rec. UIT-T X.509 | ISO/CEI 9594-8 y pueden utilizarse junto con los esquemas de control de acceso definidos en esta Especificación de directorio para reforzar la política de seguridad.

NOTA 2 – En futuras ediciones de esta Especificación de directorio se pueden definir otros esquemas de control de acceso.

NOTA 3 – Una política administrativa local puede estipular que no se tenga en cuenta la autenticación que se efectúa en algunos otros DSA (por ejemplo, los DSA en otros DMD).

En general, habrá una función de correspondencia de la identidad autentificada (por ejemplo, identidad de usuario humano autentificada por un intercambio de autenticación) y la identidad de control de acceso (por ejemplo el nombre distinguido de una inserción, junto con un identificador único facultativo, que representa al usuario). Una determinada política de seguridad puede disponer que la identidad autenticada y la identidad de control de acceso sean la misma.

Los nombres utilizados en la identidad de control de acceso, serán nombres distinguidos primarios. De forma análoga, cuando el control de acceso utilice nombres para su especificación de concesiones y denegaciones, deberán emplearse nombres distinguidos primarios.

17.2.2 Esquema de control de acceso

La definición de un esquema de control de acceso en el contexto de directorio incluye métodos para:

- especificar información de control de acceso (ACI, *access control information*);
- garantizar el cumplimiento de derechos de acceso definidos por esa información de control de acceso;
- mantener información de control de acceso.

La garantía del cumplimiento de derechos de acceso consiste en controlar el acceso a:

- información de directorio relacionada con nombres;
- información de usuario de directorio;
- información operacional de directorio, incluida información de control de acceso.

Las autoridades administrativas pueden utilizar todo el esquema de control de acceso normalizado o partes de éste para aplicar sus políticas de seguridad, o pueden definir libremente sus propios esquemas, a discreción.

Sin embargo, las autoridades administrativas pueden estipular disposiciones distintas para la protección de una parte o la totalidad de la información operacional de directorio. Las autoridades administrativas no están obligadas a proporcionar a usuarios ordinarios los medios para detectar disposiciones para la protección de información operacional.

NOTA 1 – Una política administrativa puede conceder o denegar cualquier forma de acceso a determinados atributos (por ejemplo, atributos operacionales) independientemente de los controles de acceso que puedan aplicarse en otros casos.

El directorio proporciona un medio para identificar el esquema de control de acceso vigente en una porción particular de la DIB mediante el uso del atributo operacional **accessControlScheme**. El alcance de este esquema está definido por una zona específica de control de acceso (ACSA, *access control specific area*), que es una zona administrativa específica bajo la responsabilidad de la autoridad de seguridad correspondiente. Este atributo se coloca en la inserción administrativa para el punto administrativo correspondiente. Solamente las inserciones administrativas para puntos específicos de control de acceso pueden contener un atributo **accessControlScheme**.

NOTA 2 – Si este atributo operacional falta con respecto al acceso a una inserción dada, el directorio se comportará como en el caso de un DSA conforme a la edición de 1988 (es decir, la determinación de un mecanismo de control de acceso, así como de sus efectos sobre operaciones, resultados y errores, es un asunto local).

```
accessControlScheme  ATTRIBUTE  ::=  {
      WITH SYNTAX                    OBJECT IDENTIFIER
      EQUALITY MATCHING RULE         objectIdentifierMatch
      SINGLE VALUE                   TRUE
      USAGE                          directoryOperation
      ID                             id-aca-accessControlScheme }
```

Cualquier subinserción o inserción en una zona específica de control de acceso (ACSA), puede contener ACI de inserción solamente si esta ACI está permitida y concuerda con el valor de atributo **accessControlScheme** de la ACSA correspondiente.

17.3 Protección de las operaciones de directorio

Advertencia – Se sabe que 17.3.1 y 17.3.2 contienen especificaciones no válidas. Estas subcláusulas son por tanto desaprobadas. En una edición futura se eliminarán o actualizarán las especificaciones desaprobadas o se proporcionará texto actualizado.

Las especificaciones siguientes se facilitan para preservar la capacidad firmada opcional proporcionada por la segunda edición de estas Especificaciones de directorio y para permitir extender dichas capacidades a todas las operaciones y errores:

OPTIONALLY-PROTECTED es un tipo de datos parametrizado en que el parámetro es un tipo de datos cuyos valores puede, a opción del generador, ir acompañado de su firma digital. Esta capacidad se especifica mediante el siguiente tipo:

```
OPTIONALLY-PROTECTED { Type }  ::=  CHOICE {
      unsigned          Type,
      signed            SIGNED {Type} }
```

Se utiliza **OPTIONALLY-PROTECTED-SEQ** en lugar de **OPTIONALLY-PROTECTED** cuando el tipo de datos protegidos es un tipo de datos secuencial que no está rotulado.

```
OPTIONALLY-PROTECTED-SEQ { Type } ::= CHOICE {
    unsigned        Type,
    signed    [0]   SIGNED { Type } }
```

El tipo de datos parametrizado **SIGNED**, que describe la forma del formulario firmado de la información, se especifica en la Recomendación UIT-T X.509 | ISO/CEI 9594-8.

17.3.1 Protección y transformaciones de seguridad

Pueden protegerse las operaciones de directorio (argumentos, resultados y errores) utilizando las notaciones **PROTECTION**, **SECURITY-TRANSFORMATION** y **PROTECTION-MAPPING** basadas en las especificaciones de la seguridad genérica de capas superiores (definidas en la Rec. UIT-T X.830 | ISO/CEI 11586-1).

En función de los requisitos de protección, un sistema puede soportar uno o más de los siguientes mecanismos:

a) La **PROTECTION-MAPPING** firmada, definida en (Rec. UIT-T X.830 | ISO/CEI 11586-1), para proporcionar la integridad y la autenticación del origen de los datos utilizando técnicas de firma digital. Esto se pone en correspondencia con la **dirSignedTransformation** lo que proporciona una codificación idéntica al mismo tipo de datos utilizados en la construcción ASN.1 **SIGNED** (como se define en la Rec. UIT-T X.509 | ISO/CEI 9594-8).

b) La **PROTECTION-MAPPING** criptada para proporcionar confidencialidad (y con algunos algoritmos y métodos de gestión de la clave, integridad y autenticación del origen de los datos) empleando técnicas de criptación. Esto utiliza las siguientes definiciones:

```
genEncryptedTransform {KEY-INFORMATION: SupportedKIClasses }  SECURITY-TRANSFORMATION  ::=
    {
    IDENTIFIER                  { enhancedSecurity  gen-encrypted(2) }
    INITIAL-ENCODING-RULES      { joint-iso-itu-t  asn1(1)  ber(1) }
                                -- This default for initial encoding rules may be overridden
                                -- using a static protected parameter (initEncRules).
    XFORMED-DATA-TYPE           SEQUENCE {
            initEncRules        OBJECT IDENTIFIER DEFAULT { joint-iso-itu-t  asn1(1)  ber(1) },
            encAlgorithm        AlgorithmIdentifier OPTIONAL,  -- Identifies the encryption algorithm,
            keyInformation      SEQUENCE {
                kiClass         KEY-INFORMATION.&kiClass ({SupportedKIClasses}),
                keyInfo         KEY-INFORMATION.&KiType ({SupportedKIClasses} {@kiClass})
                                } OPTIONAL,
                -- Key information may assume various formats, governed by supported members
                -- of the KEY-INFORMATION information object class (defined in ITU-T
                -- Rec. X.830 | ISO/IEC 11586-1)
            encData     BIT STRING ( CONSTRAINED BY {
                -- the encData value shall be generated following
                -- the procedure specified below -- })
            }
    }
```

Otros detalles

Proceso de codificación:	Los datos de entrada se criptan.
Entradas locales al proceso de codificación:	Identificador del algoritmo de criptación, parámetros del algoritmo (facultativo), identificador de la clave.
Proceso de decodificación:	Se descriptan los datos.
Entradas locales al proceso de transportado como decodificación:	Algoritmo e identificador de la clave si no se han parámetros protegidos.
Salidas del proceso de decodificación:	Datos descriptados.
Parámetros:	Los parámetros protegidos estáticos facultativos son: reglas de codificación inicial, identificador del algoritmo de criptación, parámetros del algoritmo de criptación, información de la clave.

Calificadores de la transformación:	Identificador de la clave.
Errores:	Se produce una condición de error si fallan los procesos de criptación o descriptación.
Servicios de seguridad:	Confidencialidad.

```
encrypted PROTECTION-MAPPING ::= {
    SECURITY-TRANSFORMATION {genEncryptedTransform} }
```

c) La **signedAndEncrypt PROTECTION-MAPPING** para proporcionar una combinación de la protección descrita más arriba.

```
signedAndEncrypt PROTECTION-MAPPING ::= {
    SECURITY-TRANSFORMATION {signedAndEncryptedTransform} }

signedAndEncryptedTransform {KEY-INFORMATION: SupportedKIClasses}
    SECURITY-TRANSFORMATION ::= {
        IDENTIFIER                  { enhancedSecurity  dir-encrypt-sign (1) }
        INITIAL-ENCODING-RULES      { joint-iso-itu-t asn1 (1) ber-derived (2) distinguished-encoding (1) }
        XFORMED-DATA-TYPE
            PROTECTED
            {
                PROTECTED
                {
                unprotectedData ABSTRACT-SYNTAX.&Type,
                signed
                },
            encrypted
            }
    }
```

17.3.2 Selección de la protección que debe aplicarse

En general, la aplicación de las operaciones de protección de directorio es facultativa y utiliza la siguiente notación:

```
OPTIONALLY-PROTECTED {ToBeProtected, PROTECTION-MAPPING:generalProtection} ::=
    CHOICE {
        toBeProtected   ToBeProtected,
                            -- no DIRQOP specified for operation
        signed          PROTECTED {ToBeProtected, Signed},
                            -- DIRQOP is Signed
        protected       [APPLICATION 0]
                            PROTECTED { ToBeProtected, generalProtection } }
                            -- DIRQOP is other than Signed
```

Las correspondencias de protección específica que deben aplicarse a las operaciones en un sistema abierto determinado, se definen mediante la siguiente notación **DIRQOP**:

a) La protección necesaria para todos los argumentos, resultados y errores de operaciones (excluidos los resultados/errores procedentes de operaciones encadenadas – véase más adelante) en una vinculación, se indica en el establecimiento de la vinculación mediante una especificación **DIRQOP**. Si en el establecimiento de la vinculación no se identifica ningún **DIRQOP**, las operaciones de esa vinculación están desprotegidas.

b) En una especificación **DIRQOP**, si no se especifica ninguna protección para un argumento, resultado o errores de una operación determinada, ésta queda sin protección.

c) El solicitante puede indicar en una petición la necesidad de una protección alternativa en el resultado o los errores de una operación determinada.

d) Para operaciones en cadenas, a menos que el solicitante haya indicado requisitos de protección alternativos, la protección requerida en el resultado/errores se indica en la petición, de forma que refleje la DIRQOP seleccionada en la vinculación de DUA con DSA para la operación que se está encadenando.

e) Si un DSA no puede proporcionar la protección requerida en una respuesta o error, puede devolver una información no protegida aunque esta información puede ser rechazada por el destinatario.

En la Rec. UIT-T X.530 | ISO/CEI 9594-10 se define un objeto gestionado para la DIRQOP. Puede disponerse de una DIRQOP por defecto para un componente de directorio utilizando el siguiente atributo en la inserción correspondiente a ese componente de directorio.

```
defaultDirQop ATTRIBUTE  ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        USAGE                           directoryOperation
        ID                              id-at-defaultDirQop }

DIRQOP  ::=  CLASS
-- This information object class is used to define the quality of protection
-- required throughout directory operation.
-- The Quality Of Protection can be signed, encrypted, signedAndEncrypt
{
        &dirqop-Id                                  OBJECT IDENTIFIER UNIQUE,
        &dirBindError-QOP                           PROTECTION-MAPPING:protectionReqd,
        &dirErrors-QOP                              PROTECTION-MAPPING:protectionReqd,
        &dapReadArg-QOP                             PROTECTION-MAPPING:protectionReqd,
        &dapReadRes-QOP                             PROTECTION-MAPPING:protectionReqd,
        &dapCompareArg-QOP                          PROTECTION-MAPPING:protectionReqd,
        &dapCompareRes-QOP                          PROTECTION-MAPPING:protectionReqd,
        &dapListArg-QOP                             PROTECTION-MAPPING:protectionReqd,
        &dapListRes-QOP                             PROTECTION-MAPPING:protectionReqd,
        &dapSearchArg-QOP                           PROTECTION-MAPPING:protectionReqd,
        &dapSearchRes-QOP                           PROTECTION-MAPPING:protectionReqd,
        &dapAbandonArg-QOP                          PROTECTION-MAPPING:protectionReqd,
        &dapAbandonRes-QOP                          PROTECTION-MAPPING:protectionReqd,
        &dapAddEntryArg-QOP                         PROTECTION-MAPPING:protectionReqd,
        &dapAddEntryRes-QOP                         PROTECTION-MAPPING:protectionReqd,
        &dapRemoveEntryArg-QOP                      PROTECTION-MAPPING:protectionReqd,
        &dapRemoveEntryRes-QOP                      PROTECTION-MAPPING:protectionReqd,
        &dapModifyEntryArg-QOP                      PROTECTION-MAPPING:protectionReqd,
        &dapModifyEntryRes-QOP                      PROTECTION-MAPPING:protectionReqd,
        &dapModifyDNArg-QOP                         PROTECTION-MAPPING:protectionReqd,
        &dapModifyDNRes-QOP                         PROTECTION-MAPPING:protectionReqd,
        &dspChainedOp-QOP                           PROTECTION-MAPPING:protectionReqd,
        &dispShadowAgreeInfo-QOP                    PROTECTION-MAPPING:protectionReqd,
        &dispCoorShadowArg-QOP                      PROTECTION-MAPPING:protectionReqd,
        &dispCoorShadowRes-QOP                      PROTECTION-MAPPING:protectionReqd,
        &dispUpdateShadowArg-QOP                    PROTECTION-MAPPING:protectionReqd,
        &dispUpdateShadowRes-QOP                    PROTECTION-MAPPING:protectionReqd,
        &dispRequestShadowUpdateArg-QOP             PROTECTION-MAPPING:protectionReqd,
        &dispRequestShadowUpdateRes-QOP             PROTECTION-MAPPING:protectionReqd,
        &dopEstablishOpBindArg-QOP                  PROTECTION-MAPPING:protectionReqd,
        &dopEstablishOpBindRes-QOP                  PROTECTION-MAPPING:protectionReqd,
        &dopModifyOpBindArg-QOP                     PROTECTION-MAPPING:protectionReqd,
        &dopModifyOpBindRes-QOP                     PROTECTION-MAPPING:protectionReqd,
        &dopTermOpBindArg-QOP                       PROTECTION-MAPPING:protectionReqd,
        &dopTermOpBindRes-QOP                       PROTECTION-MAPPING:protectionReqd
}
WITH SYNTAX
{
        DIRQOP-ID                                   &dirqop-Id
        DIRECTORYBINDERROR-QOP                      &dirBindError-QOP
        DIRERRORS-QOP                               &dirErrors-QOP
        DAPREADARG-QOP                              &dapReadArg-QOP
        DAPREADRES-QOP                              &dapReadRes-QOP
        DAPCOMPAREARG-QOP                           &dapCompareArg-QOP
        DAPCOMPARERES-QOP                           &dapCompareRes-QOP
        DAPLISTARG-QOP                              &dapListArg-QOP
        DAPLISTRES-QOP                              &dapListRes-QOP
        DAPSEARCHARG-QOP                            &dapSearchArg-QOP
        DAPSEARCHRES-QOP                            &dapSearchRes-QOP
        DAPABANDONARG-QOP                           &dapAbandonArg-QOP
        DAPABANDONRES-QOP                           &dapAbandonRes-QOP
        DAPADDENTRYARG-QOP                          &dapAddEntryArg-QOP
        DAPADDENTRYRES-QOP                          &dapAddEntryRes-QOP
```

```
        DAPREMOVEENTRYARG-QOP                   &dapRemoveEntryArg-QOP
        DAPREMOVEENTRYRES-QOP                   &dapRemoveEntryRes-QOP
        DAPMODIFYENTRYARG-QOP                   &dapModifyEntryArg-QOP
        DAPMODIFYENTRYRES-QOP                   &dapModifyEntryRes-QOP
        DAPMODIFYDNARG-QOP                      &dapModifyDNArg-QOP
        DAPMODIFYDNRES-QOP                      &dapModifyDNRes-QOP
        DSPCHAINEDOP-QOP                        &dspChainedOp-QOP
        DISPSHADOWAGREEINFO-QOP                 &dispShadowAgreeInfo-QOP
        DISPCOORSHADOWARG-QOP                   &dispCoorShadowArg-QOP
        DISPCOORSHADOWRES-QOP                   &dispCoorShadowRes-QOP
        DISPUPDATESHADOWARG-QOP                 &dispUpdateShadowArg-QOP
        DISPUPDATESHADOWRES-QOP                 &dispUpdateShadowRes-QOP
        DISPREQUESTSHADOWUPDATEARG-QOP          &dispRequestShadowUpdateArg-QOP
        DISPREQUESTSHADOWUPDATERES-QOP          &dispRequestShadowUpdateRes-QOP
        DOPESTABLISHOPBINDARG-QOP               &dopEstablishOpBindArg-QOP
        DOPESTABLISHOPBINDRES-QOP               &dopEstablishOpBindRes-QOP
        DOPMODIFYOPBINDARG-QOP                  &dopModifyOpBindArg-QOP
        DOPMODIFYOPBINDRES-QOP                  &dopModifyOpBindRes-QOP
        DOPTERMINATEOPBINDARG-QOP               &dopTermOpBindArg-QOP
        DOPTERMINATEOPBINDRES-QOP               &dopTermOpBindRes-QOP
}
```

18 Control de acceso básico

18.1 Alcance y aplicación

Esta cláusula define un esquema de control de acceso específico (posiblemente entre muchos) para el directorio. El esquema de control de acceso aquí definido, es identificado con el atributo operacional **accessControlScheme** dándole el valor **basic-access-control**. La subcláusula 17.2.2 describe cuáles inserciones contienen el atributo operacional **accessControlScheme**.

NOTA – En 18.9 se define otro esquema de control de acceso conocido por "Control de acceso simplificado". Se define como un subconjunto del esquema control de acceso básico. Cuando se utiliza el control de acceso simplificado, el atributo operacional **accessControlScheme** tendrá el valor **simplified-access-control**. El esquema de control de acceso adicional conocido por Control de acceso reglado se especifica en la cláusula 19.

El esquema aquí definido sólo trata del aprovisionamiento de medios para controlar el acceso a la información de directorio que está contenida en la DIB (la que pudiera incluir información de estructura de árbol y de control de acceso). No trata del control de acceso a los efectos de la comunicación con una entidad de aplicación de DSA. Por control de acceso a información ha de entenderse la prevención de la detección, revelación, o modificación no autorizadas de esa información.

18.2 Modelo de control de acceso básico

El modelo de control de acceso básico para el directorio define, para toda operación abstracta, uno o más puntos en los cuales se toman decisiones de control de acceso. Cada decisión de control de acceso comprende:

- el elemento de información de directorio al que se accede, llamado el *ítem protegido*;
- el usuario que pide la operación, llamado el *solicitante*;
- un derecho particular necesario para efectuar una porción de la operación, llamado el *permiso*;
- uno o más atributos operacionales que contienen colectivamente la política de seguridad que gobierna el acceso a ese ítem, llamados *ítems de ACI*.

Así, el modelo de control de acceso básico define:

- los ítems protegidos;
- las clases de usuario;
- las categorías de permiso requeridas para efectuar cada operación de directorio;
- el alcance de la aplicación y la sintaxis de ítems de ACI;
- el algoritmo básico, llamado función de decisión de control de acceso (ACDF, *access control decision function*), utilizada para decidir si un determinado solicitante tiene un determinado permiso en virtud de ítems de ACI aplicables.

18.2.1 Ítems protegidos

Un ítem protegido es un elemento de información de directorio cuyo acceso puede ser controlado separadamente. Los ítems protegidos de directorio son inserciones, atributos, valores de atributo y nombres. Por razones de conveniencia al especificar políticas de control de acceso, el control de acceso básico proporciona el medio de identificar colecciones de ítems conexos, tales como atributos en una inserción o todos los valores de atributo de un atributo dado, y especificar una protección común para ellos.

18.2.2 Permisos de control de acceso y su alcance

El acceso se controla concediendo o denegando permisos. Las categorías de permiso se describen en 18.2.3 y 18.2.4.

El alcance de controles de acceso puede ser una sola inserción o una colección de inserciones que están lógicamente relacionadas por el hecho de estar dentro del alcance de una subinserción para un punto administrativo dado.

Las categorías de permiso son generalmente independientes. Puesto que todas las inserciones de directorio tienen una posición relativa dentro del DIT, el acceso a información de usuario y operacional siempre implica alguna forma de acceso a información relacionada con el DIT. Por tanto, hay dos formas principales de decisión de control de acceso asociada con una operación abstracta: acceso a inserciones como objetos denominados (lo que se conoce por *acceso a inserción*), y acceso a atributos que contienen información de usuario y operacional (lo que se conoce por *acceso a atributo*). Para muchas operaciones de directorio se requieren las dos formas de permiso. Además, cuando es aplicable, permisos distintos controlan el tipo de nombre o de error devuelto. A continuación se indican algunos aspectos importantes de las categorías de permiso, las formas de acceso, y la toma de decisiones sobre control de acceso:

a) Para efectuar operaciones de directorio en inserciones enteras (por ejemplo, leer una inserción o añadir una inserción), por lo general es necesario que se conceda permiso con respecto a los atributos y valores contenidos en esa inserción. Son excepciones los permisos que controlan la redenominación y la supresión de inserciones; en ninguno de estos dos casos se tienen en cuenta permisos sobre atributo o valor de atributo.

b) Para efectuar operaciones de directorio que requieren acceso a atributos o valores de atributo es necesario tener permiso de acceso a la inserción o inserciones que contienen esos atributos o valores.

   NOTA 1 – Para la supresión de una inserción o de un atributo no es necesario el acceso al contenido de la inserción o del atributo.

c) La decisión de si ha de permitirse o no el acceso a una inserción está determinada exclusivamente por la posición de la inserción en el DIT, en términos de su nombre distinguido, y es independiente de la manera en que el directorio localiza esa inserción.

d) Un principio de diseño del control de acceso básico es que sólo pueda autorizarse el acceso cuando una concesión explícitamente otorgada esté presente en la información de control de acceso utilizada por el directorio para tomar la decisión de control de acceso. La concesión de una forma de acceso (por ejemplo, acceso a inserción) nunca concede automática, o implícitamente, la otra forma (por ejemplo, acceso a atributo). Para administrar políticas de control de acceso al directorio que tengan sentido, es necesario, por tanto, establecer explícitamente la política de acceso para ambas formas de acceso.

   NOTA 2 – Ciertas combinaciones de concesiones o denegaciones de permiso son ilógicas; no obstante, incumbe en primer término a los usuarios, más bien que al directorio, asegurar que esas combinaciones no aparezcan.

   NOTA 3 – De acuerdo con el principio de diseño antes mencionado, la concesión o denegación de permisos para un valor de atributo no controla automáticamente el acceso al atributo conexo. Además, para acceder a uno o más valores de atributo en el curso de una operación de interrogación de directorio, tiene que haberse concedido al usuario el acceso tanto al tipo de atributo como a su(s) valor(es).

e) La única decisión de acceso por defecto proporcionada en el modelo es la de negar el acceso en ausencia de una información de acceso explícita que lo conceda.

f) Una denegación especificada en información de control de acceso prevalece siempre sobre una concesión, si todos los demás factores permanecen iguales.

g) Un DSA dado puede no tener la información de control de acceso que gobierna los datos de directorio que oculta. Los administradores de seguridad deben estar atentos a que un DSA que tiene la facultad de tener datos velados puede exponer a otros DSA a riesgos de seguridad considerables, ya que podría revelar información a usuarios no autorizados.

h) A los efectos de la interrogación, los atributos colectivos que están asociados con alguna inserción están protegidos de manera precisa como si dichos atributos formasen parte de la inserción.

   NOTA 4 – A los efectos de la modificación, los atributos colectivos se asocian con la subinserción que los retiene, no con inserciones dentro del alcance de la subinserción. Los controles de acceso relacionados con una modificación no afectan por tanto a los atributos colectivos, salvo cuando son aplicables a un atributo colectivo y sus valores dentro de la subinserción.

18.2.3 Categorías de permiso para acceso a inserciones

Las categorías de permiso utilizadas para controlar el acceso a inserciones son *Leer* (*Read*), *Hojear* (*Browse*), *Añadir* (*Add*), *Suprimir* (*Remove*), *Modificar* (*Modify*), *Redenominar* (*Rename*), *RevelarError* (*DiscloseOnError*), *Exportar* (*Export*), e *Importar* (*Import*) y *RetornarDN* (*ReturnDN*). Su utilización se describe con más detalle en la Rec. UIT-T X.511 | ISO/CEI 9594-3. El anexo L recapitula su significado en situaciones generales. Esta subcláusula presenta las categorías indicando brevemente el propósito que se persigue al conceder permiso para cada una de ellas. La influencia real de un determinado permiso concedido sobre decisiones de control de acceso estará determinada, sin embargo, por el contexto completo de la ACDF y de los puntos de decisión de control de acceso para cada operación abstracta.

a) *Leer*, si se concede, permite el acceso de lectura para operaciones de directorio que denominan específicamente una inserción (es decir, lo opuesto a las operaciones de enumeración y búsqueda) y proporciona visibilidad de la información contenida en la inserción a la cual se aplica.

b) *Hojear*, si se concede, permite acceder a inserciones utilizando operaciones que no proporcionan explícitamente el nombre de la inserción.

c) *Añadir*, si se concede, crear una inserción en el DIT respetando los controles sobre todos los atributos y valores de atributo que habrán de insertarse en la nueva inserción en el momento de su creación.

   NOTA 1 – Para añadir una inserción se deberá conceder también el permiso para añadir, por lo menos, los atributos obligatorios y sus valores.

   NOTA 2 – No hay un "permiso para añadir subordinado" específico. El permiso para añadir una inserción se controla utilizando atributos operacionales **prescriptiveACI,** descritos en 18.3.

d) *Suprimir*, si se concede, permite suprimir la inserción en el DIT independientemente de los controles sobre los atributos o valores de atributo en la inserción.

e) *Modificar*, si se concede, permite modificar la información contenida en una inserción.

   NOTA 3 – Para modificar información contenida en una inserción que no sea los valores de atributo de nombre distinguido, se deberá conceder también permisos apropiados sobre los atributos y valores.

f) La concesión de *Redenominar* es necesaria para redenominar una inserción con un nuevo RDN, teniendo en cuenta los cambios consiguientes de los nombres distinguidos de inserciones subordinadas, si las hubiere; si el nombre del superior no se modifica, la concesión es suficiente.

   NOTA 4 – Para redenominar una inserción no se requieren permisos previos relativos a atributos o valores contenidos, incluidos los atributos RDN; esto es también aplicable cuando, al efectuar la operación, resulten añadidos o suprimidos valores de atributo como consecuencia de los cambios de RDN.

g) *RevelarError*, si se concede, permite revelar el nombre de una inserción en caso de resultado de error (o vacío).

h) *Exportar*, si se concede, permite que una inserción y sus subordinados (si los hubiere) sean exportados, es decir, retirados de la ubicación actual y colocados en una nueva ubicación, a reserva de la concesión de permisos adecuados en el destino. Si se cambia el último RDN, es necesario también *Redenominar* en la ubicación actual.

   NOTA 5 – Para exportar una inserción o sus subordinadas no se requieren permisos previos relativos a atributos o valores contenidos, incluidos los atributos RDN; esto es también aplicable cuando, al efectuar la operación, resulten añadidos o suprimidos valores de atributo como consecuencia de los cambios de RDN.

i) *Importar*, si se concede, permite importar una inserción, es decir, retirarla de una ubicación y colocarla en la ubicación a que se aplica el permiso (a reserva de la concesión de permisos adecuados en la fuente).

   NOTA 6 – Para importar una inserción o sus subordinados no se requieren permisos previos relativos a atributos o valores contenidos, incluidos los atributos RDN; esto es también aplicable cuando, al efectuar la operación, resulten añadidos o suprimidos valores de atributo como consecuencia de los cambios de RDN.

j) *RetornarDN*, si se concede, permite revelar el nombre distinguido de la inserción en un resultado de operación.

18.2.4 Categorías de permiso para acceso a atributo y valor de atributo

Las categorías de permiso utilizadas para controlar el acceso a atributo y valor de atributo son *Comparar* (*Compare*), *Leer* (*Read*), *ConcordarPorFiltro* (*FilterMatch*), *Añadir* (*Add*), *Suprimir* (*Remove*), y *RevelarError* (*DiscloseOnError*). Su utilización se describe con más detalle en la Rec. UIT-T X.511 | ISO/CEI 9594-3. El anexo L recapitula su significado en situaciones generales. Esta subcláusula presenta las categorías indicando brevemente el propósito que se persigue al conceder permiso para cada una de ellas. La influencia real de un determinado permiso concedido sobre decisiones de control de acceso está determinada, sin embargo, por el contexto completo de la ACDF y de los puntos de decisión de control de acceso para cada operación abstracta.

a) *Comparar*, si se concede, permite utilizar atributos y valores en una operación de comparar.

b) *Leer*, si se concede, permite devolver atributos y valores como información de inserción en operaciones de acceso para leer y buscar.

c) *ConcordarPorFiltro*, si se concede, permite la evaluación de un filtro dentro de un criterio de búsqueda.

d) *Añadir*, si se concede para un atributo, permite añadir un atributo a reserva de que se puedan añadir todos los valores de atributo especificados. Si se concede para un valor de atributo, permite añadir un valor a un atributo existente.

e) *Suprimir*, si se concede para un atributo, permite suprimir un atributo completo con todos sus valores. Si se concede para un valor de atributo, permite suprimir el valor de atributo de un atributo existente.

f) *RevelarError*, si se concede para un atributo, permite que la presencia del atributo sea revelada por un error de atributo o de seguridad. Si se concede para un valor de atributo, permite que la presencia del valor de atributo sea revelada por un error de atributo o de seguridad.

g) *Invocar*, si se concede, el objeto (siempre un atributo operacional o un valor de un atributo operacional) al que se aplica el permiso puede ser invocado en nombre del usuario autenticado por el DSA. La función llevada a cabo mediante la invocación depende del atributo. No se requieren otros permisos al usuario para el atributo operacional o respecto a la inserción/subinserción que lo contiene.

18.3 Zonas administrativas de control de acceso

El DIT está dividido en subárboles que constituyen zonas administrativas autónomas, cada una de las cuales está bajo la autoridad administrativa de una sola organización de gestión de dominio. Puede ser ulteriormente dividida en subárboles que constituyen zonas administrativas específicas para los fines de aspectos específicos de administración; como otra posibilidad, una zona administrativa autónoma completa puede comprender una sola zona administrativa específica. Cada zona administrativa específica está bajo la responsabilidad de una autoridad administrativa específica correspondiente. Una zona administrativa dada puede ser compartida por varias autoridades administrativas específicas. Véase la cláusula 11.

18.3.1 Zonas de control de acceso y dominios de control de acceso de directorio

En el caso de control de acceso, la autoridad administrativa específica es una autoridad de seguridad, y la zona administrativa específica se conoce por zona específica de control de acceso (ACSA). La raíz de ACSA se llama punto específico de control de acceso. Cada punto específico de control de acceso está representado en el DIT por una inserción administrativa que incluye **access-control-specific-area** como un valor de su atributo operacional **administrativeRole**; tiene (potencialmente) uno o más subárboles que contienen información de control de acceso. De manera similar, cada punto interior de control de acceso se representa en el DIT mediante una inserción administrativa que contiene **access-control-inner-area** como un valor de su atributo operacional **administrativeRole**, y tiene también (potencialmente) una o más inserciones que contienen información de control de acceso. Cada una de estas inserciones administrativas que tiene una subinserción que contiene información de ACI prescriptiva tiene **basic-access-control**, **simplified-access-control**, u otro valor pertinente como un valor de su atributo operacional **accessControlScheme**. Cada subinserción que es un punto específico de control de acceso y que contiene información de control de acceso, tiene **accessControlSubentry** como un valor de su atributo de clase de objeto. Una inserción administrativa y sus subinserciones pueden mantener atributos operacionales (como información de control de acceso) que se relacionan, respectivamente, con el punto administrativo (y posiblemente sus subinserciones) y con colecciones de inserciones (dentro de la zona administrativa) definidas por la subinserción **subtreeSpecification**.

El atributo **accessControlScheme** estará presente solamente si la inserción administrativa que lo tiene es una inserción específica de control de acceso. Una inserción administrativa nunca puede ser a la vez una inserción específica de control de acceso y una inserción interior de control de acceso; por tanto, nunca pueden estar presentes simultáneamente valores correspondientes en el atributo **administrativeRole**.

El alcance de una subinserción que contiene información de control de acceso, definida por su **subtreeSpecification** (que puede incluir refinamiento de subárbol), se denomina un dominio de control de acceso de directorio (DACD, *directory access control domain*).

NOTA – Un DACD puede no contener ninguna inserción, y puede abarcar inserciones que no han sido añadidas aún al DIT.

La autoridad de seguridad puede permitir que una zona específica de control de acceso se divida en subárboles denominados zonas (administrativas) interiores. Cada una de estas zonas interiores se denomina una zona interior de control de acceso (ACIA, *access control inner area*) con **access-control-inner-area** como el valor del atributo operacional **administrativeRole**. Cada subinserción del punto administrativo correspondiente que contiene ACI prescriptiva tiene, como antes, un valor **accessControlSubentry** en su atributo de clase de objeto.

El alcance (**subtreeSpecification**) especificado en una subinserción dentro de una ACIA es también un DACD y contiene inserciones dentro de la zona interior de control de acceso asociada.

Las ACIA permiten un grado de delegación de autoridad de control de acceso dentro de la ACSA. La autoridad de la ACSA retendrá la autoridad dentro de la ACIA porque la ACI en las subinserciones del punto administrativo de la ACSA se aplica también a la ACI en las subinserciones de las ACIA pertinentes (en 18.6 se explica cómo la ACSA controla la autoridad).

En resumen, al evaluar los controles de acceso, el tipo de esquema de control de acceso (por ejemplo, control de acceso básico) es indicado por el valor del atributo **accessControlScheme** de la inserción específica de control de acceso pertinente; el cometido de cada inserción administrativa pertinente dentro de la ACSA es indicada por los valores de su atributo **administrativeRole**; la presencia de control de acceso prescriptivo en una subinserción determinada es indicada por un valor **accessControlSubentry** en su atributo de clase de objeto.

Las subinserciones, como otras inserciones, pueden mantener un atributo **entryACI** para proteger su propio contenido.

18.3.2 Asociación de controles con zonas administrativas

El acceso a una inserción dada es controlado (potencialmente) por la totalidad de los puntos administrativos de control de acceso superiores (tanto interiores como específicos) hasta el primer punto administrativo de control de acceso no interior o punto administrativo autónomo inclusive encontrado cuando se asciende en el DIT desde la inserción hacia la raíz. Los puntos específicos de control de acceso superiores a este punto administrativo de control de acceso no producen ningún efecto sobre el control de acceso a la inserción dada.

NOTA 1 – Se considera que un punto administrativo autónomo es, implícitamente, un punto específico de control de acceso a los fines de esta descripción, incluso si no está asociado a ningún control prescriptivo.

A continuación se señalan algunos puntos importantes relativos a la asociación entre controles de acceso y zonas administrativas:

a) Los controles de acceso para información de directorio podrían sólo ser aplicables a inserciones seleccionadas, o tener un alcance que se extendiera a través de porciones de la DIB que están lógicamente relacionadas por una política de seguridad común y una administración de control de acceso común.

b) Se puede imponer control de acceso a inserciones dentro de las ACSA o dentro de las ACIA colocando atributos **prescriptiveACI** (véase 18.5) en una o más subinserciones de la correspondiente inserción administrativa de control de acceso, con un alcance definido por una **subtreeSpecification** apropiada.

   NOTA 2 – Los atributos **prescriptiveACI** no son atributos colectivos. Hay varias diferencias apreciables entre **prescriptiveACI** y atributos colectivos:

   – no se considera que una **prescriptiveACI**, aunque pueda influir en decisiones de control de acceso para cada asiento dentro del alcance de la subinserción que ella contiene, suministra información accesible a una tal inserción, ni forma, en modo alguno, parte de una inserción;

   – los atributos **prescriptiveACI** están asociadas con los aspectos de control de acceso de la administración, y están asociados con puntos específicos e interiores de control de acceso, y no con puntos administrativos de colección de inserciones;

   – un atributo **prescriptiveACI** tiene por finalidad expresar una política que ejerce una influencia a través de un conjunto definido de inserciones, mientras que un atributo colectivo tiene por finalidad proporcionar información que asocia un conjunto de atributos, accesible por el usuario, dentro de un conjunto definido de inserciones;

   – los atributos **prescriptiveACI** representan una información de política que, en general, no será ampliamente accesible por usuarios ordinarios. Los usuarios administrativos que necesitan acceder a información **prescriptiveACI** pueden acceder a ella como atributos operacionales dentro de subinserciones.

c) Un atributo operacional **prescriptiveACI** contiene **ACIItems** (véase 18.4.1) comunes a todas las inserciones dentro del alcance de la subinserción, es decir, el DACD, en el que aparece la **prescriptiveACI**. Un DACD contiene inserciones dentro de la zona específica de control de acceso asociada (pero puede no contener ninguna inserción).

d) Aunque **ACIItems** particulares pueden especificar atributos o valores como ítems protegidos, los **ACIItems** están asociados lógicamente con inserciones. El conjunto particular de **ACIItems** asociados con una inserción y con el contenido de esa inserción es una combinación de:

   – **ACIItems** que se aplican a esa inserción particular, especificados como valores del atributo operacional **entryACI**, si está presente (véase 18.5.2);

   – **ACIItems** procedentes de atributos operacionales **prescriptiveACI** aplicables a la inserción por el hecho de estar colocados en subinserciones de inserciones administrativas cuyo alcance incluye la inserción en cuestión (véase 18.5.1).

e) Cada inserción (controlada por **entryACI** y/o **prescriptiveACI**) tiene necesariamente que caer dentro de sólo una ACSA. Cada una de estas inserciones pueden también caer dentro de sólo una ACIA anidadas dentro de la ACSA que contiene la inserción. La **prescriptiveACI** que afecta potencialmente al resultado de decisiones de control de acceso para una inserción dada está ubicada dentro de subinserciones (de la inserción administrativa) para la ACSA y para cada ACIA que contiene la inserción. Otras subinserciones no pueden afectar a decisiones de control de acceso relativas a esa inserción.

f) Si una inserción está dentro del alcance de más de un DACD, el conjunto completo de **ACIItems** que puede potencialmente afectar a decisiones de control de acceso relativas a esa inserción incluye todos los atributos **prescriptiveACI** de esos DACD, además de cualesquiera atributos **entryACI** en la propia inserción. En la figura 17 se presenta un ejemplo. El control de acceso efectivo en la inserción E1 es una combinación de la **prescriptiveACI** para DACD1, DACD2, DACD3, y **entryACI** (si está presente) en la inserción E1. El control de acceso efectivo en la inserción E2 es una combinación de la **prescriptiveACI** para DACD1 y DACD3, y **entryACI** (si está presente) en la inserción E2.

NOTA 3 – La protección de la información de control de acceso se describe en 18.6.

g) El atributo **subtreeSpecification** en cada subinserción define una colección de inserciones dentro de una zona administrativa. Puesto que **subtreeSpecification** puede definir un refinamiento de subárbol, los DACD pueden superponerse arbitrariamente dentro de la intersección de sus respectivas zonas administrativas. Para simplificar la presentación, la figura 17 no muestra puntos administrativos, subinserciones, ni zonas administrativas; sin embargo, puede considerarse como tres DACD en la misma ACSA, correspondiendo cada DACD a una sola subinserción del punto administrativo para esa ACSA (y no hay ACIA). Otra posibilidad es considerar la figura 17 en el contexto de una sola ACSA que contiene una sola ACIA donde DACD1 es congruente con la ACSA y DACD3 es congruente con la ACIA (DACD1 y DACD2 corresponderían a subinserciones del punto administrativo ACSA, y DACD3 correspondería a una subinserciones del punto administrativo ACIA). Un zona administrativa es congruente con un DACD cuando la colección de inserciones en el DACD es la misma que la colección de inserciones en el subárbol definido implícitamente que corresponde a la zona administrativa. El anexo M contiene un ejemplo con figuras que representan las relaciones entre inserciones administrativas, zonas administrativas, subinserciones y DACD.

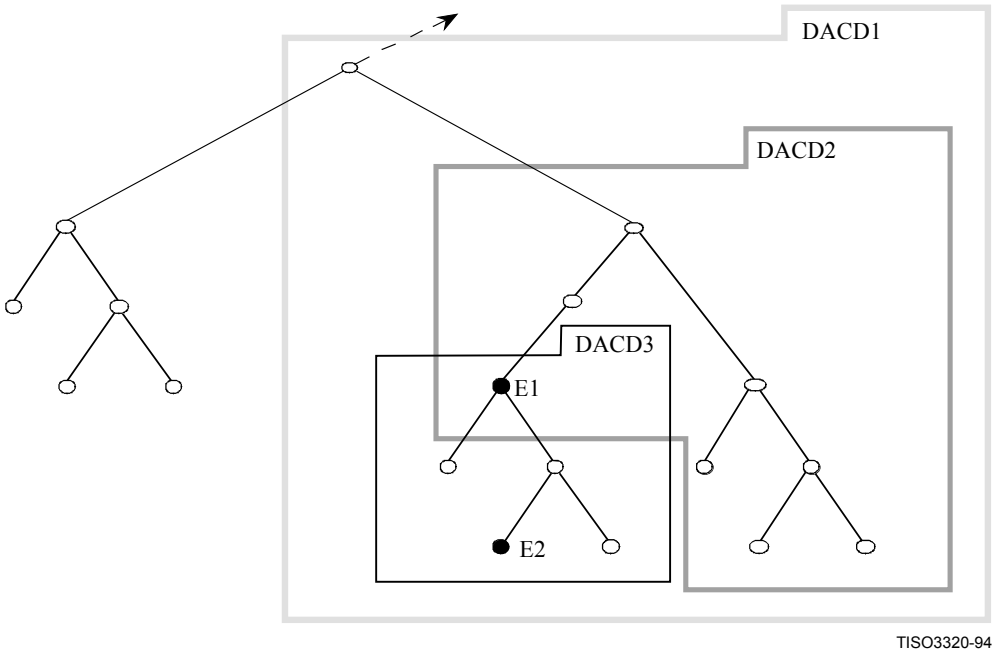


Figura 17 – Control de acceso efectivo mediante los DACD

18.4 Representación de información de control de acceso

18.4.1 ASN.1 para información de control de acceso

La información de control de acceso se representa como un conjunto de **ACIItems**, donde cada **ACIItem** concede o deniega permisos con respecto a ciertos usuarios e ítems protegidos.

En ASN.1, la información se expresa como:

```
ACIItem ::= SEQUENCE {
    identificationTag           DirectoryString { ub-tag },
    precedence                  Precedence,
    authenticationLevel         AuthenticationLevel,
    itemOrUserFirst             CHOICE {
        itemFirst       [0]     SEQUENCE {
            protectedItems          ProtectedItems,
            itemPermissions         SET OF ItemPermission },
        userFirst       [1]     SEQUENCE {
            userClasses             UserClasses,
            userPermissions         SET OF UserPermission } } }

Precedence ::= INTEGER (0..255)

ProtectedItems ::= SEQUENCE {
    entry                           [0]  NULL                                    OPTIONAL,
    allUserAttributeTypes           [1]  NULL                                    OPTIONAL,
    attributeType                   [2]  SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
    allAttributeValues              [3]  SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
    allUserAttributeTypesAndValues  [4]  NULL                                    OPTIONAL,
    attributeValue                  [5]  SET SIZE (1..MAX) OF AttributeTypeAndValue OPTIONAL,
    selfValue                       [6]  SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
    rangeOfValues                   [7]  Filter                                  OPTIONAL,
    maxValueCount                   [8]  SET SIZE (1..MAX) OF MaxValueCount      OPTIONAL,
    maxImmSub                       [9]  INTEGER                                 OPTIONAL,
    restrictedBy                    [10] SET SIZE (1..MAX) OF RestrictedValue    OPTIONAL,
    contexts                        [11] SET SIZE (1..MAX) OF ContextAssertion   OPTIONAL,
    classes                         [12] Refinement                              OPTIONAL }

MaxValueCount ::= SEQUENCE {
    type            AttributeType,
    maxCount        INTEGER }

RestrictedValue ::= SEQUENCE {
    type            AttributeType,
    valuesIn        AttributeType }

UserClasses ::= SEQUENCE {
    allUsers    [0] NULL                                    OPTIONAL,
    thisEntry   [1] NULL                                    OPTIONAL,
    name        [2] SET SIZE (1..MAX) OF NameAndOptionalUID OPTIONAL,
    userGroup   [3] SET SIZE (1..MAX) OF NameAndOptionalUID OPTIONAL,
                    -- dn component shall be the name of an
                    -- entry of GroupOfUniqueNames
    subtree     [4] SET SIZE (1..MAX) OF SubtreeSpecification OPTIONAL }

ItemPermission ::= SEQUENCE {
    precedence          Precedence OPTIONAL,
                    -- defaults to precedence in ACIItem
    userClasses         UserClasses,
    grantsAndDenials    GrantsAndDenials }

UserPermission ::= SEQUENCE {
    precedence          Precedence OPTIONAL,
                    -- defaults to precedence in ACIItem
    protectedItems      ProtectedItems,
    grantsAndDenials    GrantsAndDenials }

AuthenticationLevel ::= CHOICE {
    basicLevelsSEQUENCE {
        level           ENUMERATED { none (0), simple (1), strong (2) },
        localQualifier  INTEGER OPTIONAL,
        signed          BOOLEAN DEFAULT FALSE },
    other           EXTERNAL }
```

```
GrantsAndDenials ::= BIT STRING {
    -- permissions that may be used in conjunction
    -- with any component of ProtectedItems
    grantAdd              (0),
    denyAdd               (1),
    grantDiscloseOnError  (2),
    denyDiscloseOnError   (3),
    grantRead             (4),
    denyRead              (5),
    grantRemove           (6),
    denyRemove            (7),
    -- permissions that may be used only in conjunction
    -- with the entry component
    grantBrowse           (8),
    denyBrowse            (9),
    grantExport           (10),
    denyExport            (11),
    grantImport           (12),
    denyImport            (13),
    grantModify           (14),
    denyModify            (15),
    grantRename           (16),
    denyRename            (17),
    grantReturnDN         (18),
    denyReturnDN          (19),
    -- permissions that may be used in conjunction
    -- with any component, except entry, of ProtectedItems
    grantCompare          (20),
    denyCompare           (21),
    grantFilterMatch      (22),
    denyFilterMatch       (23),
    grantInvoke           (24),
    denyInvoke            (25) }

AttributeTypeAndValue ::= SEQUENCE {
    type      ATTRIBUTE.&id ({SupportedAttributes}),
    value     ATTRIBUTE.&Type({SupportedAttributes}{@type}) }
```

18.4.2 Descripción de parámetros ACIItem

18.4.2.1 Rótulo de identificación

identificationTag (Rótulo de identificación) se utiliza para identificar un **ACIItem** particular. Se emplea para discriminar entre **ACIItems** individuales con fines de protección, gestión, y administración.

18.4.2.2 Precedencia

Precedence (Precedencia) se utiliza para controlar el orden relativo en que los **ACIItems** son considerados durante el curso de una toma de decisión sobre control de acceso de conformidad con 18.8. Los **ACIItems** que tienen valores de precedencia más altos pueden prevalecer sobre otros con valores de precedencia más bajos, si todos los demás factores permanecen iguales. Los valores de precedencia son números enteros y se comparan como tales.

La precedencia puede ser utilizada por una autoridad superior dentro de la autoridad de seguridad para permitir la delegación parcial de la fijación de política de control de acceso dentro de una ACSA. Esto puede ser efectuado por la autoridad superior fijando una política general con una alta precedencia y autorizando a los usuarios que representan la autoridad subordinada (por ejemplo, asociados con una ACIA) a crear y modificar la ACI con una precedencia menor, con el fin de adaptar la política general a fines específicos. La delegación parcial requiere por tanto, los medios para que la autoridad superior limite la precedencia máxima que la autoridad subordinada puede asignar a la ACI bajo su control.

El control de acceso básico no especifica ni describe cómo limitar la precedencia máxima que puede ser utilizada por una autoridad subordinada. Esto ha de hacerse por medios locales.

18.4.2.3 Nivel de autenticación

AuthenticationLevel (Nivel de autenticación) define el nivel mínimo de seguridad del solicitante requerido para este **ACIItem**. Tiene dos formas:

- **basicLevels** (**Niveles básicos**), que indica el nivel de autenticación, facultativamente calificado por un entero positivo o negativo, **localQualifier** (**Calificador local**), y si la petición tiene que ser firmada;
- **other** (**otro**):una medida definida externamente.

Cuando se utiliza **basicLevels**, un **AuthenticationLevel** constituido por un **level (nivel)** y un **localQualifier** facultativo será asignado al solicitante por el DSA de acuerdo con la política local. Para que el nivel de autenticación de un solicitante satisfaga o rebase un nivel mínimo requerido, el **nivel** básico del solicitante deberá cumplir o rebasar el especificado en el **ACIItem** y, además, el **localQualifier** del solicitante deberá ser aritméticamente mayor o igual que el del **ACIItem**. Se considera que una autenticación fuerte del solicitante supera un requisito de autenticación simple o de ausencia de autenticación, y que una autenticación simple supera un requisito de ausencia de autenticación. A efectos de control de acceso, el nivel de autenticación "simple" requiere una contraseña. Se considera que el caso de identificación únicamente, sin aportar una contraseña, equivale a "ausencia" de autenticación. Si no se especifica un **localQualifier** en el **ACIItem**, el solicitante no necesita tener un valor correspondiente (si ese valor está presente, es pasado por alto). Además de satisfacer o rebasar los requisitos anteriores, la petición deberá estar firmada si el **ACIItem** especifica **signed** igual a **TRUE**.

Cuando se utiliza **other**, un **AuthenticationLevel** apropiado será asignado al solicitante por el DSA de acuerdo con la política local. La forma de este **AuthenticationLevel** y el método por el cual se compara con el **AuthenticationLevel** en la ACI es un asunto local.

NOTA 1 – Un nivel de autenticación asociado con una denegación implícita indica el nivel máximo al que se autenticará a un solicitante para que no se le niegue acceso. Por ejemplo, un **ACIItem** que niega el acceso a una determinada clase de usuario y requiera autenticación fuerte negará el acceso a todos los solicitantes que no puedan probar, por medio de una identidad autenticada de modo fuerte, que no están en esa clase de usuario.

NOTA 2 – El DSA puede basar el nivel de autenticación en factores que no sean valores recibidos en intercambios de protocolo.

18.4.2.4 Parámetros itemFirst y userFirst

Cada **ACIItem** contiene una posibilidad de elección entre **itemFirst** (**ítem primero**) o **userFirst** (**usuario primero**). Esta posibilidad de elección permite agrupar permisos atendiendo a que sea más conveniente agruparlos por clases de usuario o por ítems protegidos. **itemFirst** y **userFirst** son equivalentes en cuanto a que ambos captan la misma información de control de acceso; lo que los diferencia es la forma en que organizan esa información. La elección de una o la otra se basa en razones de conveniencia administrativa. A continuación se describen los parámetros utilizados en **itemFirst** o **userFirst**.

a) **ProtectedItems** definen los ítems a los que se aplican los controles de acceso especificados. Se define como un conjunto seleccionado entre lo siguiente:
   - **entry (inserción)** significa el contenido de toda la inserción. En el caso de un miembro de familia, significa también el contenido de la inserción de cada miembro de la familia subordinado dentro del mismo atributo compuesto. No incluye necesariamente la información en esas inserciones. Este elemento será pasado por alto si está presente el elemento **classes** ya que este elemento selecciona inserciones protegidas (y miembros de familia subordinados) en base a su clase de objeto.
   - **allUserAttributeTypes** (**todos los tipos de atributo de usuario**) significa toda la información de tipo de atributo de usuario asociada con la inserción, pero no los valores asociados con esos atributos.
   - **allUserAttributeTypesAndValues** (**todos los tipos y valores de atributo de usuario**) significa toda la información de atributo de usuario asociada con la inserción, incluidos todos los valores de todos los atributos de usuario.
   - **attributeType** (**tipo de atributo**) significa información de tipo de atributo relativa a atributos específicos pero no a valores asociados con el tipo.
   - **allAttributeValues** (**todos los valores de atributo**) significa toda la información de atributo relativa a atributos específicos.
   - **attributeValue** (**valor de atributo**) significa un valor específico de atributos específicos.
   - **selfValue** (**valor propio**) significa la aserción de valor de atributo que corresponde al solicitante actual. El ítem protegido **selfValue** se aplica solamente cuando los controles de acceso van a ser aplicados con respecto a un usuario autenticado específico. Sólo puede aplicarse en el caso concreto en que el atributo especificado es de sintaxis **DistinguishedName** (**sintaxis de nombre distinguido**) o **uniqueMember** (**miembro único**) y el valor de atributo dentro del atributo especificado concuerda con el nombre distinguido del originador de la operación.

NOTA 1 – **allUserAttributeTypes** y **allUserAttributeTypesAndValues** no incluyen atributos operacionales, los cuales deberán especificarse atributo por atributo, utilizando **attributeType**, **allAttributeValues**, o **attributeValue**.

– **rangeOfValues** significa cualquier valor de atributo que concuerda con el filtro especificado, es decir para el cual el filtro especificado evaluado sobre ese atributo devolvería el valor VERDADERO.

NOTA 2 – El filtro no se evalúa para cualquiera de las inserciones del DIB. Se evalúa utilizando la semántica definida en 7.8 de la Rec. UIT-T X.511 | ISO/CEI 9594-3, operando en una inserción ficticia que contiene solamente el único valor de atributo que es el elemento protegido.

Los siguientes elementos proporcionan limitaciones que pueden impedir la concesión de ciertos permisos a elementos protegidos de la misma SECUENCIA:

– **maxValueCount** limita el número máximo de valores de atributo permitidos para un tipo de atributo especificado. Se examina si el elemento protegido es un valor de atributo del tipo especificado y se añade el permiso solicitado. Se cuentan los valores del atributo de la inserción con respecto al contexto o al control de acceso como si la operación que adiciona los valores tuviera éxito. Si el número de valores del atributo excede **maxCount**, el elemento ACI se trata como si no autorizase el acceso de adición.

– **maxImmSub** limita el número máximo de subordinados inmediatos de la inserción superior a una entidad añadida o importada. Se examina si el elemento protegido es una inserción, el permiso solicitado es adición o importación y la inserción superior inmediata está en la misma DSA que la inserción que se añade o se importa. Se cuentan los subordinados inmediatos de la inserción superior sin tener en cuenta el contexto o el control de acceso como si tuvieran éxito la adición o la importación de la inserción. Si el número de subordinados rebasa **maxImmSub**, el elemento ACI se trata como si no autorizase el acceso de adición o importación.

– **restrictedBy** limita valores añadidos al tipo de atributo a aquellos valores ya presentes en la misma inserción como valores del atributo **valuesIn**. Se examina si el elemento protegido es un valor de atributo del tipo especificado y el permiso solicitado es adición. Los valores del atributo **valuesIn** se comprueban con respecto al contexto o al control de acceso, verificándose también si la operación que adiciona los valores tuvo éxito. Si el valor que debe agregarse no está presente en **valuesIn**, el elemento ACI se trata como si no garantizase el acceso de adición.

– **contexts** limita los valores agregados a la inserción que tienen listas de contexto que satisfacen todas ellas a las aserciones de contexto en **contexts**. Se examina si el elemento protegido es un valor de atributo y el permiso solicitado es *adición*. Si el valor que debe añadirse no satisface las aserciones de contexto, el elemento ACI se trata como si no autorizase el acceso de *adición*. Si no satisface ninguno de ellos el elemento ACI se trata como si no denegase el acceso de *adición*.

NOTA 3 – Esto únicamente es pertinente cuando el permiso solicitado es *adición* y deben satisfacerse todas las aserciones de contexto. No permite el uso general de contextos para distinguir elementos protegidos de otros permisos.

– **classes** significa el contenido de inserciones (posiblemente un miembro de familia) restringido a las que tienen valores de clase de objeto que satisfacen el predicado definido por **Refinement** (véase 12.3.5), junto con (en caso de un antepasado u otro miembro de la familia) el contenido de toda la inserción de cada inserción de miembro de familia subordinado; no necesariamente incluye la información de esas inserciones.

NOTA 4 – Por las reglas de **entry** y **classes**, todos los miembros de la familia heredan el control de acceso del antepasado o de miembros de la familia superiores dentro de la misma familia. Esto no impide que miembros de la familia estén sometidos a otras políticas de **entryACI** o **prescriptiveACI** que aumenten o disminuyan la protección.

b) **UserClasses** (**clases de usuario**) define un conjunto de ninguno o más usuarios a los que se aplican los permisos. El conjunto de usuarios se selecciona entre los siguientes:

– **allUsers** (**todos los usuarios**) significa todos los usuarios de directorio (con posibles requisitos de **nivel de autenticación**);

– **thisEntry** (**esta inserción**) significa el usuario con el mismo nombre distinguido que el de la inserción a la que se accede. o si la inserción es un miembro de una familia, además el usuario con el nombre distinguido del antepasado;

– **name** (**nombre**) es el usuario con el nombre distinguido especificado (con un identificador único facultativo);

– **userGroup** (**grupo de usuarios**) es el conjunto de usuarios que son miembros de la inserción **groupOfUniqueNames** (**grupo de nombres únicos**), identificados por el nombre distinguido especificado (con un identificador único facultativo). Los miembros de un grupo de nombres únicos son tratados como nombres de objeto individuales, y no como los nombres de otros grupos de nombres únicos. En 18.4.2.5 se describe cómo se determina la pertenencia a un grupo;

– **subtree** (**subárbol**) es el conjunto de usuarios cuyos nombres distinguidos están dentro de la definición del subárbol (no refinado).

Los nombres utilizados para especificar un usuario, grupo o subárbol serán nombres distinguidos primarios. No podrán incluirse contextos ni valores distinguidos alternativos. No se necesita la función de decisión de control de acceso para determinar el nombre distinguido primario para los nombres alternativos con los que éste se suministra.

NOTA 5 – Esto significa que si un solicitante ha proporcionado un nombre alternativo no subsiguientemente resuelto por el directorio como nombre distinguido primario, el control del acceso basado en nombres distinguidos primarios puede no reconocer al solicitante como perteneciente a la clase de usuario de acceso concedido o denegado.

c) **SubtreeSpecification** se utiliza para especificar un subárbol con relación a una inserción raíz denominado en **base**. **base** representa el nombre distinguido de la raíz del subárbol. El subárbol se extiende a las hojas del DIT a menos que se especifique otra cosa en **chop**. El uso de un componente **specificationFilter** no está permitido, si está presente, debe ignorarse.

NOTA 6 – **SubtreeSpecification** no permite refinamiento de subárbol porque un refinamiento podría requerir que un DSA utilizara una operación distribuida para determinar si un usuario dado pertenece a una determinada clase de usuario. Se designa control de acceso básico para evitar operaciones remotas en el curso de la toma de una decisión de control de acceso. Los miembros de un subárbol cuya definición incluye solamente **base** y **chop** pueden ser evaluados localmente, en tanto que los miembros de definiciones de subárbol que utilizan **specificationFilter** sólo pueden ser evaluados obteniendo información de la inserción de usuario que está potencialmente en otro DSA.

d) **ItemPermission** (**permiso de ítem**) contiene una colección de usuarios y sus permisos con respecto a **ProtectedItems** dentro de una especificación **itemFirst**. Los permisos se especifican en **grantsAndDenials** (**concesiones y denegaciones**) como se indica en el apartado f) de esta subcláusula. Se considera que cada uno de los permisos especificados en **grantsAndDenials** tiene el nivel de precedencia especificado en **precedence** a los efectos de evaluar información de control de acceso como se indica en 18.8. Si se omite **precedence** en **ItemPermission**, (el nivel de) precedence se toma de la **precedence** especificada para el **ACIItem** (véase 18.4.2.2).

e) **UserPermission** (**permiso de usuario**) contiene una colección de ítems protegidos y los permisos asociados con respecto a **userClasses** en una especificación **userFirst**. Los ítems protegidos se especifican en **protectedItems**, que se examinan en 18.4.2. Los permisos asociados se especifican en **grantsAndDenials** como se indicó en el apartado f) de esta subcláusula. Se considera que cada uno de los permisos especificados en **grantsAndDenials** tiene el nivel de precedencia especificado en **precedence** a los efectos de evaluar información de control de acceso como se indica en 18.8. Si se omite **precedence** en **userPermission**, (el nivel de) precedence se toma de la **precedence** especificada para el **ACIItem** (véase 18.4.2.2).

f) **GrantsAndDenials** especifica los derechos de acceso que son concedidos o denegados en la especificación **ACIItem**. Las semánticas precisas de estos permisos con respecto a cada ítem protegido se examinan en la Rec. UIT-T X.511 | ISO/CEI 9594-3.

g) **UniqueIdentifier** (**identificador único**) puede ser utilizado por el mecanismo de autenticación para distinguir entre caso de reutilización de nombre distinguido. El valor del identificador único es asignado por la autoridad de autenticación de acuerdo con su política y es proporcionado por el DSA que autentica. Si este campo está presente, entonces, para que un usuario accedente concuerde con la clase de usuario **name** de un **ACIItem** que conceda permisos, además de cumplirse el requisito de que el nombre distinguido del usuario concuerde con el nombre distinguido especificado, la autenticación del usuario dará un identificador único asociado, y ese valor concordará por igualdad con el valor especificado.

NOTA 7 – Cuando la autenticación se basa en **SecurityParameters (parámetros de seguridad)** suministrados, el identificador único asociado con el usuario se puede tomar del campo **subjectUniqueIdentifier** (**identificador único de sujeto**) de **Certificate** (**certificado**) del emisor en el **CertificationPath** (**trayecto de certificación**) facultativo.

18.4.2.5 Determinación de la pertenencia a un grupo

Para determinar si un solicitante dado es miembro de un grupo hay que verificar dos criterios. Además, la determinación puede verse afectada si la definición de grupo no es conocida localmente. Los criterios para establecer la calidad de miembro y para el tratamiento de grupos no locales se examinan a continuación.

a) Un DSA no está obligado a efectuar una operación remota para determinar si un solicitante pertenece a un determinado grupo, a los efectos del control de acceso básico. Si la pertenencia al grupo no puede ser evaluada, el DSA deberá suponer que el solicitante no pertenece al grupo si el ítem ACI concede el permiso solicitado, y que pertenece al grupo si niega el permiso solicitado.

NOTA 1 – Los administradores de control de acceso deberán estar prevenidos del riesgo de basar controles de acceso en la pertenencia a grupos no disponibles localmente o a grupos que sólo están disponibles por replicación (y que, por tanto, pueden estar anticuados).

NOTA 2 – Por razones de rendimiento, generalmente no es práctico extraer los miembros de un grupo de los DSA distantes como parte de la evaluación de controles de acceso. Sin embargo, en ciertas circunstancias puede ser práctico, y se permite a un DSA, por ejemplo, efectuar operaciones remotas para obtener o renovar una copia local de una inserción de grupo o utilizar la operación Compare para verificar la pertenencia al grupo antes de aplicar esta cláusula.

b) Para determinar si un solicitante es miembro de una clase de usuario **userGroup** se aplican los siguientes criterios:

- La inserción denominada por la especificación **userGroup** será un caso de la clase de objeto **groupOfNames** (**grupo de nombres**) o **groupOfUniqueNames**.
- El nombre del originador será un valor del atributo **member** (**miembro**) o **uniqueMember** de esa inserción.

NOTA 3 – Los valores del atributo **member** o **uniqueMember** que no concuerdan con el nombre del originador son ignorados incluso si representan los nombres de grupos de los que podría encontrarse que es miembro el originador. En consecuencia, no se soportan grupos jerarquizados cuando se evalúan controles de acceso.

NOTA 4 – Los nombres utilizados en **member** o **uniqueMember** serán nombres distinguidos primarios. No se incluirán contextos ni nombres alternativos con contexto.

18.5 Los atributos operacionales de ACI

La información de control de acceso se almacena en el directorio como un atributo operacional de inserciones y subinserciones. El atributo operacional es multivaluado, lo que permite representar la ACI como un conjunto de **ACIItem** (definido en 18.4).

18.5.1 Información de control de acceso prescriptivo

Los atributos de ACI prescriptiva se definen como atributos operacionales de subinserciones que contienen información de control de acceso aplicable a inserciones dentro del alcance de la subinserción correspondiente:

```
prescriptiveACI  ATTRIBUTE  ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-prescriptiveACI }
```

18.5.2 Información de control de acceso a inserción

Un atributo de ACI de inserción se define como atributos operacionales de una inserción. Contiene información de control de acceso aplicable a la inserción en la cual aparece, y el contenido de esa inserción:

```
entryACI  ATTRIBUTE  ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-entryACI }
```

18.5.3 ACI de subinserción

Los atributos de ACI de subinserción se definen como atributos operacionales de inserciones administrativas, y proporcionan información de control de acceso que se aplica a cada una de las subinserciones del punto administrativo correspondiente. La ACI prescriptiva dentro de las subinserciones de un punto administrativo particular no se aplica nunca a la misma subinserción o a cualquier otra de ese punto administrativo, pero puede ser aplicable a las subinserciones de puntos administrativos subordinados. Los atributos de ACI de subinserción están contenidos solamente en puntos administrativos y no afectan a ningún elemento del DIT distintos de las subinserciones subordinadas inmediatamente.

Al evaluar el control de acceso para una subinserción específica, la ACI que deberá considerarse es:

- **entryACI** (**ACI de inserción**) dentro de la propia subinserción (si la hubiere);
- **subentryACI** (**ACI de subinserción**) dentro de la inserción administrativa asociada (si la hubiere);
- **prescriptiveACI** asociada con otros puntos administrativos pertinentes dentro de la misma zona específica de control de acceso (si la hubiere).

```
subentryACI  ATTRIBUTE  ::= {
    WITH SYNTAX                 ACIItem
    EQUALITY MATCHING RULE      directoryStringFirstComponentMatch
    USAGE                       directoryOperation
    ID                          id-aca-subentryACI }
```

18.6 Protección de la ACI

Los atributos operacionales pueden estar sujetos al mismo mecanismo de protección que los atributos ordinarios. A continuación se señalan algunos puntos importantes en relación con esto:

a) **identificationTag** proporciona un identificador para cada **ACIItem**. Este rótulo puede utilizarse para suprimir un valor **ACIItem** específico, o para protegerlo mediante ACI prescriptiva o de inserción.

NOTA 1 – Las reglas de directorio aseguran que solamente un **ACIItem** por atributo de control de acceso posee un valor específico cualquiera de **identificationTag**.

b) La creación de subinserciones para una inserción administrativa puede ser controlada en el acceso por medio del atributo operacional **subentryACI** en la inserción administrativa.

NOTA 2 – El derecho a crear controles de acceso prescriptivo puede también ser gobernado directamente por la política de seguridad; se requiere esta disposición para crear controles de acceso en nuevas zonas administrativas autónomas.

18.7 Control de acceso y operaciones de directorio

Cada operación de directorio presupone que se tomen una serie de *decisiones de control de acceso* sobre los diversos ítems protegidos a que accede la operación.

Para algunas operaciones (por ejemplo, operaciones de modificar), cada una de esas decisiones de control de acceso deberá conceder acceso para que la operación tenga éxito; si se niega el acceso a cualquier ítem protegido, fracasa toda la operación. Para otras operaciones, los ítems protegidos a los que se niega el acceso son simplemente omitidos en la operación, y el procedimiento continúa.

Si se niega el acceso solicitado, puede que se necesiten ulteriores decisiones de control de acceso para determinar si el usuario tiene permisos **DiscloseOnError** para el ítem protegido. Sólo si se concede el permiso **DiscloseOnError** puede el directorio responder con un error que revela la existencia del ítem protegido; en todos los demás casos, el directorio actúa de modo que quede oculta la existencia del ítem protegido.

Las exigencias de control de acceso para cada operación, es decir, los ítems protegidos y el permiso de acceso requerido para acceder a cada ítem protegido, se especifican en la Rec. UIT-T X.511 | ISO/CEI 9594-3.

El algoritmo por el cual se toma cualquier decisión de control de acceso se especifica en 18.8.

18.8 Función de decisión de control de acceso

Esta subcláusula especifica cómo se toma una decisión de control de acceso para un ítem protegido cualquiera. Proporciona una descripción conceptual de la función de decisión de control de acceso (ACDF) para **basic-access-control** (**control de acceso básico**). Describe la manera de procesar ítems de ACI para decidir si se concede o se niega a un solicitante determinado un permiso especificado para un ítem protegido dado.

18.8.1 Entradas y salidas

Para cada invocación de la ACDF, las entradas son:

a) el nombre distinguido del solicitante (definido en 7.3 de la Rec. UIT-T X.511 | ISO/CEI 9594-3), identificador único, y nivel de autenticación, o tantos de éstos cuantos haya disponibles);

b) el ítem protegido (una inserción, un atributo o un valor de atributo) que se considera en el punto de decisión vigente para el cual se invocó la ACDF;

c) la categoría de permiso solicitada; categoría especificada para el punto de decisión vigente;

d) los ítems de ACI asociados con la inserción que contiene (o que es) el ítem protegido. Los ítems protegidos se describen en 18.4.2.4. El alcance de influencia para ítems de ACI dentro de un atributo **prescriptiveACI** se describe en 18.3.2 y 18.5.1. El alcance de influencia para ítems de ACI dentro de un atributo **entryACI** se describe en 18.3.2 y 18.5.2. El alcance de influencia para ítems de ACI dentro de un atributo **subentryACI** se describe en 18.5.3.

Cuando una inserción es miembro de una familia, también hereda el control de acceso del antepasado o del miembro de familia superior dentro de la misma familia. Esto no impide que miembros de la familia sean sometidos a otras políticas de **entryACI** o **prescriptiveACI** que aumentan o disminuyen la protección.

Además, si los elementos ACI incluyen alguna de las limitaciones de elemento protegido descritas en 18.4.2.4, pueden también necesitarse como entradas la inserción total y el número de subordinados inmediatos de su inserción superior.

La salida es una decisión de *conceder* o *denegar* acceso al ítem protegido.

En cualquier caso particular de toma de una decisión de control de acceso, la salida deberá ser la misma que si se hubiesen seguido los pasos indicados en 18.8.2 a 18.8.4.

18.8.2 Tuplas

Para cada valor de ACI en los ítems de ACI de 18.8.1 d), el valor se expande en un conjunto de *tuplas*, una tupla para cada elemento de los conjuntos **itemPermissions** y **userPermissions**. Se reúnen todas las tuplas de todos los valores de ACI y se forma un solo conjunto. Cada tupla contiene los siguientes ítems:

(userClasses, authenticationLevel, protectedItems, grantsAndDenials, precedence)

Toda tupla cuyos **grantsAndDenials** especifiquen las dos cosas, concesiones y denegaciones, se remplaza por dos tuplas, una que especifica sólo concesiones y otra que especifica sólo denegaciones.

18.8.3 Descarte de tuplas no pertinentes

El descarte de tuplas no pertinentes consiste en lo siguiente:

1) Se descartan todas las tuplas que no incluyen al solicitante en la **userClass** de la tupla [18.4.2.4 b)] de la manera siguiente:

   – En el caso de tuplas que conceden acceso, se descartan todas las tuplas que no incluyen la identidad del solicitante en el elemento **userClasses** de la tupla, teniendo en cuenta elementos **uniqueIdentifier**, si procede. Cuando una tupla especifica **uniqueIdentifier**, deberá haber una regla de concordancia en la identidad del solicitante si la tupla no ha de ser descartada. Se descartan las tuplas que especifican un nivel de autenticación más alto que el asociado con el solicitante de acuerdo con 18.4.2.3.

   – En el caso de tuplas que niegan el acceso, se retienen todas las tuplas que incluyen al solicitante en el elemento **userClasses** de la tupla, teniendo en cuenta elementos **uniqueIdentifier** si procede. Se retienen también todas las tuplas que niegan acceso y que especifican un nivel de autenticación más alto que el asociado con el solicitante de acuerdo con 18.4.2.3. Se descartan todas las demás tuplas que niegan acceso.

   NOTA 1 – El segundo requisito del segundo apartado (esto es, que se retenga toda tupla que niegue el acceso y también especifique un nivel de autenticación más alto que el asociado con el solicitante) refleja el hecho de que el solicitante no ha probado adecuadamente que no pertenece a la clase de usuario para la que se especificó la denegación.

2) Se descartan todas las tuplas que no incluyen el ítem protegido en **protectedItems** [18.4.2.4 a)].

3) Se examinan todas las tuplas que incluyen **maxValueCount**, **maxImmSub**, **restrictedBy** o **contexts**. Se descartan todas las tuplas que conceden acceso pero no satisfacen ninguna de esas limitaciones [18.4.2.4 a)].

4) Se descartan todas la tuplas que no incluyen el permiso solicitado como uno de los bits fijados en **grantsAndDenials** [18.4.1, 18.4.2.4 f)].

NOTA 2 – El orden en que se descartan las tuplas no pertinentes no cambia la salida de la ACDF.

18.8.4 Selección de las tuplas de más alta precedencia y más específicas

La selección de las tuplas de más alta precedencia y de más alta especificidad consiste en los siguientes pasos:

1) Se descartan todas las tuplas que tienen una **precedence** menor que la más alta precedencia restante.

2) Si queda más de una tupla, se eligen las tuplas con la *clase de usuario más específica*. Si hay algunas tuplas en las que el solicitante concuerda con el elemento **name** o **thisEntry** de **UserClasses**, se descartan todas las otras tuplas. En los demás casos, si hay algunas tuplas que concuerdan con **UserGroup** se descartan todas las demás tuplas. En los demás casos, si hay algunas tuplas que concuerdan con **subtree**, se descartan todas las demás tuplas.

3) Si queda más de una tupla, se eligen las tuplas con el *ítem protegido más específico*. Si el ítem protegido es un atributo y hay tuplas que especifican el tipo de atributo explícitamente, se descartan todas las demás tuplas. Si el ítem protegido es un valor de atributo y hay tuplas que especifican el valor de atributo explícitamente, se descartan todas las demás tuplas. Todo ítem protegido que sea un **rangeOfValues** se tratará como si especificase un valor de atributo explícitamente.

Se concede el acceso solamente si quedan una o más tuplas y todas conceden el acceso. De no ser así, se niega el acceso.

18.9 Control de acceso simplificado

18.9.1 Introducción

En esta subcláusula se describe la funcionalidad de un esquema de control de acceso, conocido como control de acceso simplificado, que se diseña para proporcionar un subconjunto de funcionalidad hallada en el control de acceso básico.

18.9.2 Definición de la funcionalidad de control de acceso simplificado

La funcionalidad de control de acceso simplificado se define como sigue:

a) Las decisiones de control de acceso serán tomadas solamente sobre la base de valores **ACIItem** de los atributos operacionales **prescriptiveACI** y **subentryACI**.

NOTA 1 – No se utilizará **entryACI**, si está presente, para tomar decisiones de control de acceso.

b) Se soportarán zonas administrativas específicas de control de acceso. No se utilizarán zonas administrativas interiores de control de acceso. Se tomarán decisiones de acceso particulares basándose en los valores de **ACIItem** obtenidos de un solo punto administrativo o de subinserciones de ese punto administrativo.

NOTA 2 – Para las decisiones de control de acceso no se utilizarán los valores de atributos de **prescriptiveACI** que aparecen en subinserciones de puntos administrativos que no contienen valores del atributo **administrativeRole id-ar-accessControlSpecificArea**.

c) Se definirán otras disposiciones para el control de acceso básico.

19 Control de acceso reglado

19.1 Alcance y aplicación

En esta cláusula se define un método de control de acceso específico (posiblemente de entre muchos) para el directorio. El método de control de acceso aquí definido se identifica con el atributo operacional **accessControlScheme** proporcionado el valor **rule-based-access-control** o, si se usa junto con los métodos de control de acceso básico o simplificado definidos en la cláusula 18, **rule-and-basic-access-control** o **rule-and-simple-access-control**. En 17.2.2 se describen las inserciones que contienen el atributo operacional **accessControlScheme**.

El método aquí definido se ocupa únicamente del control del acceso a la información de directorio dentro del DIB (incluyendo, potencialmente, la estructura del árbol e información de control de acceso). No se aplica al acceso de control para la comunicación con una entidad de aplicación DSA. El control del acceso a la información supone la prevención de una detección no autorizada, revelación o modificación de esa información.

19.2 Modelo de control de acceso reglado

Puede haber entornos en los que se utilice la información relativa a la liberación (en vez de la identidad) del solicitante para determinar si puede o no negarse el acceso a un valor de atributo. Esto se define como control de acceso reglado y utiliza reglas de política de control de acceso impuestas administrativamente para determinar cuando puede negarse el acceso a ciertos contenidos de directorio. Si se deniega un acceso en virtud del control de acceso reglado, no puede autorizarse bajo ningún otro método de control de acceso. El modelo de control de acceso reglado identifica la información utilizada para determinar si puede denegarse el acceso. Se aplica a cada operación. La decisión sobre el control de acceso supone lo siguiente:

a) Información de control de acceso asociada con los valores de atributo a los que se accede. Esta información de control de acceso se denomina etiqueta de seguridad.

b) Información de control de acceso asociada al usuario solicitante de la operación. Esta información de control de acceso se denomina liberación. El usuario solicitante de la operación se denomina peticionario.

c) Reglas que definen si se autoriza un acceso para una etiqueta de seguridad y una liberación determinadas, que se llaman políticas de seguridad.

Véase la figura 18.

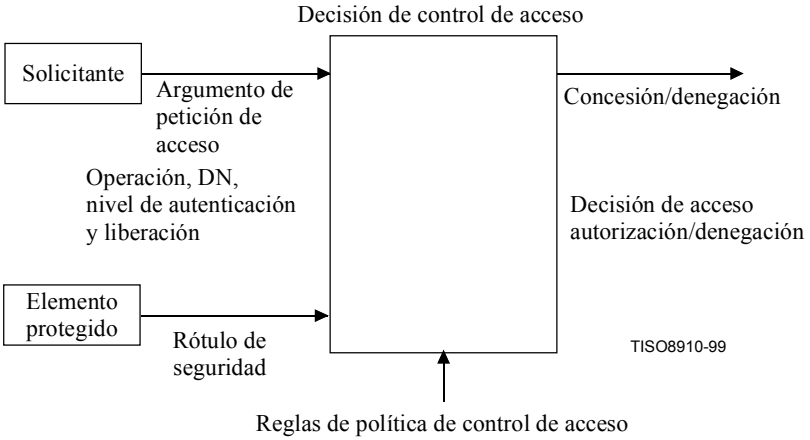


Figura 18 – Modelo de decisión de control de acceso reglado

Los rótulos de seguridad pueden asociarse con seguridad con valores de atributo vinculando la etiqueta con la información mediante el uso de una firma digital u otro mecanismo de integridad. Un rótulo de seguridad es una propiedad del valor de atributo y se asocia a ese valor en forma de contexto.

Se necesita la liberación para poder efectuar una comparación con el rótulo de seguridad. Puede asociarse la liberación al nombre distinguido del solicitante mediante un campo de extensión certificado (atributo de directorio sujeto) o a través de un certificado de atributo. Los medios seleccionados para proporcionar la liberación son objeto de la política de seguridad en vigor.

NOTA – El empleo de otra información de liberación (por ejemplo, la asociada con cualesquiera DSA intermedios que pueda tener encadenadas las operaciones) queda fuera del alcance de esta Especificación de directorio.

Las reglas de seguridad que deben aplicarse para efectuar una decisión de control de acceso se definen como parte de la política de seguridad. La política de seguridad se identifica en la etiqueta de seguridad o se define para el entorno que contiene el objeto etiquetado.

19.3 Zonas administrativas de control de acceso

De igual manera que para, el control de acceso básico (véase 18.3), el DIT se divide en zonas administrativas que incluyen zonas específicas de control de acceso. La inserción administrativa de una ACSA identifica las políticas de seguridad de etiquetado (reglas de acceso) aplicables a esa zona administrativa así como el método de control de acceso aplicable (**rule-based-access-control** o **rule-and-basic-access-control** o **rule-and-simple-access-control** o cualquier otro método de control de acceso).

19.4 Nivel de seguridad

19.4.1 Introducción

Pueden utilizarse rótulos de seguridad para asociar información relativa a la seguridad con atributos dentro de directorio.

Pueden asignarse rótulos de seguridad a un valor de atributo en línea con la política de seguridad en vigor para ese atributo. La política de seguridad puede también definir como se utilizan los rótulos de seguridad para reforzar esa política.

Un rótulo de seguridad comprende un conjunto de elementos que incluyen, facultativamente, un identificador de la política de seguridad, una clasificación de seguridad, una marca de privacidad y un conjunto de categorías de seguridad. El nivel de seguridad está vinculado al nivel de atributo que utiliza una firma digital u otro mecanismo de integridad.

19.4.2 Administración de los rótulos de seguridad

Se asigna un rótulo de seguridad a un valor de atributo mediante una función administrativa antes de situarla en el directorio.

La función administrativa es responsable de la asignación de rótulos de seguridad a valores de atributo en línea con la política de seguridad en vigor para la ACSA.

Se protege la vinculación de un rótulo de seguridad empleando una firma digital u otro mecanismo de integridad. Esta protección la aplica la función administrativa o el creador del valor de atributo.

19.4.3 Valores de atributo rotulados

Un contexto de rótulo de seguridad asocia un rótulo de seguridad a un valor de atributo. Con un valor de atributo únicamente puede asociarse un solo rótulo. Es decir, el contexto del rótulo de seguridad es monovaluado. Además, no se soportan las reglas de concordancia para el contexto de rótulos de seguridad.

NOTA – El concepto de contextos se presenta en 8.7.

```
attributeValueSecurityLabelContext CONTEXT ::= {
        WITH SYNTAX     SignedSecurityLabel     -- At most one security label context can be assigned to an
                                                -- attribute value
        ID              id-avc-attributeValueSecurityLabelContext }

SignedSecurityLabel  ::=  SIGNED {SEQUENCE  {
        attHash         HASH {AttributeTypeAndValue},
        issuer          Name            OPTIONAL,  -- name of labelling authority
        keyIdentifier   KeyIdentifier   OPTIONAL,
        securityLabel   SecurityLabel } }

SecurityLabel  ::=  SET {
        security-policy-identifier      SecurityPolicyIdentifier        OPTIONAL,
        security-classification         SecurityClassification          OPTIONAL,
        privacy-mark                    PrivacyMark                     OPTIONAL,
        security-categories             SecurityCategories              OPTIONAL }
                (ALL EXCEPT ( {-- none, at least one component shall be present -- } ) )

SecurityPolicyIdentifier  ::=  OBJECT IDENTIFIER

SecurityClassification ::= INTEGER {
        unmarked        (0),
        unclassified    (1),
        restricted      (2),
        confidential    (3),
        secret          (4),
        top-secret      (5) }

PrivacyMark  ::=  PrintableString (SIZE (1..ub-privacy-mark-length))

SecurityCategories ::= SET SIZE (1..MAX) OF SecurityCategory
```

Este contexto no se utiliza para filtrar o seleccionar atributos particulares, como en el caso de otros contextos. Tampoco se utilizan los mecanismos asociados con los contextos (repliegue, valores de contexto por defecto, etc.) para aplicar el control del acceso reglado.

El componente **attHash** contiene el valor resultante de la aplicación de un procedimiento de embrollo criptográfico a los octetos codificados de DER, como se define en la Rec. UIT-T X.509 | ISO/CEI 9594-8.

El componente **issuer** transporta el nombre de la autoridad etiquetante.

El componente **keyIdentifier** puede ser el identificador de una clave pública certificada situada en el campo de extensión del identificador de clave pública sujeto definido en la Rec. UIT-T X.509 | ISO/CEI 9594-8 o el identificador de una clave simétrica y de la información de control de seguridad asociada.

El componente **securityLabel** está constituido por un conjunto de elementos que incluyen, facultativamente, un identificador de política de seguridad, una clasificación de seguridad, una marca de privacidad y un conjunto de categorías como se definen en 8.5.9 de la Rec. UIT-T X.411 | ISO/CEI 10021-4.

19.5 Liberación

Un atributo liberación asocia una liberación con una entidad denominada que incluye DUA.

```
clearance  ATTRIBUTE  ::=  {
        WITH SYNTAX     Clearance
        ID              id-at-clearance }

Clearance  ::=  SEQUENCE  {
        policyId                 OBJECT IDENTIFIER,
        classList                ClassList                          DEFAULT {unclassified},
        securityCategories       SET SIZE (1..MAX) OF SecurityCategory   OPTIONAL }

ClassList  ::=  BIT STRING {
        unmarked         (0),
        unclassified     (1),
        restricted       (2),
        confidential     (3),
        secret           (4),
        topSecret        (5) }

SecurityCategory  ::=  SEQUENCE {
        type        [0]    SECURITY-CATEGORY.&id ({SecurityCategoriesTable}),
        value       [1]    EXPLICIT SECURITY-CATEGORY.&Type ({SecurityCategoriesTable} {@type}) }

SECURITY-CATEGORY  ::=  TYPE-IDENTIFIER

SecurityCategoriesTable SECURITY-CATEGORY ::= { ... }
```

El componente **policyId** transporta un identificador que puede emplearse para identificar la política de seguridad en vigor con la que están relacionadas **classList** y **securityCategories**.

El componente **classList** comprende una lista de clasificaciones asociadas con la entidad denominada.

El componente **securityCategories** (véase 8.5.9 de la Rec. UIT-T X.411 | ISO/CEI 10021-4), si está presente, proporciona restricciones ulteriores dentro del contexto de una **classList**.

NOTA – Una liberación se adhiere con seguridad a una entidad denominada utilizando un certificado de atributo (Rec. UIT-T X.509 | ISO/CEI 9594-8), un campo de extensión de certificado de clave pública (por ejemplo dentro de la extensión **SubjectDirectoryAttribute**) (Rec. UIT-T X.509 | ISO/CEI 9594-8) o mediante procedimientos que están fuera del alcance de esta Especificación de directorio.

19.6 Control de acceso y operaciones de directorio

Cada operación de directorio implica un conjunto de decisiones de control de acceso sobre los valores de atributo a los que accede la operación.

Para algunas operaciones (por ejemplo la operación Remove Entry) aunque pueda aparecer que la operación ha tenido éxito si se ha denegado el acceso a uno o más valores de atributo, pueden permanecer en el directorio los atributos ocultos. Para otras operaciones los elementos protegidos a los que se ha denegado el acceso se omiten simplemente del resultado de la operación y el procesamiento continúa.

Los requisitos de control de acceso para cada operación se especifican en la Rec. UIT-T X.511 | ISO/CEI 9594-3.

El algoritmo mediante el cual se ejecuta cualquier decisión de control de acceso determinada, se especifica como sigue:

- Si mediante **rule-based-access-control** se deniega el acceso a todos los valores de atributo de una inserción, queda también denegado el acceso a esa inserción para todas las operaciones.
- Si, mediante **rule-based-access-control**, se deniega el acceso a todos los valores atributo de un atributo, queda también denegado el acceso a ese atributo para todas las operaciones.
- El control de acceso reglado afecta a las operaciones relativas a la lectura de valores de atributo (por ejemplo lectura, búsqueda) en el sentido de que no es visible el valor de atributo (la operación se ejecuta como si no estuviera presente el valor de atributo) si se deniega el acceso al valor de atributo.

- El control de acceso reglado afecta a las operaciones que implican la supresión de una inserción (por ejemplo supresión de inserción) en el sentido de que no eliminan los valores de atributo a los que se deniega el acceso.
- El control de acceso reglado afecta a las operaciones que suponen la supresión de un tipo de atributo (por ejemplo modificar inserción-suprimir atributo) en el sentido de que no suprimen los valores de atributo a los que se deniega el acceso.
- El control de acceso reglado afecta a las operaciones que suponen la supresión de un valor de atributo (por ejemplo modificación de inserción-eliminación de un valor) en el sentido de que estas operaciones fracasan si se deniega el acceso al valor de atributo.

19.7 Función de decisión del control de acceso

En esta subcláusula se especifica como se realiza la decisión del control de acceso para cualquier valor de atributo particular. Proporciona una descripción conceptual de la función de decisión de control de acceso (ACDF) para el **rule-based-access-control**. Describe como se procesan la liberación y el nivel de seguridad para decidir si se autoriza o se deniega a un solicitante determinado un permiso específico a un valor de atributo dado. La función de decisión aplica las reglas de política de seguridad que establecen si se autoriza un acceso para un valor de atributo dados su etiqueta de seguridad y la liberación del peticionario. La definición de las reglas de seguridad queda fuera del alcance de estas Especificaciones de directorio. En M.10 se facilita un ejemplo simplificado de reglas de política de seguridad para **rule-based-access-control**.

Para cada invocación de la ACDF, las entradas son:

a) la liberación del solicitante (como se define en 19.5);

b) el valor de atributo considerado como punto de decisión vigente para el que se invocó la ACDF;

c) la política de seguridad en vigor para la zona específica de control de acceso;

d) el nivel de seguridad relacionado con el valor de atributo.

La salida es una decisión relativa a la denegación del acceso al valor de atributo.

Para cada caso particular de adopción de una decisión de control de acceso, la salida será la misma que si se ejecutaran las fases descritas en 19.6.

19.8 Utilización de los controles de acceso básico y reglado

Si están en vigor los controles de acceso básico y reglado, el orden el que se aplican es un asunto local, salvo cuando se niega el acceso a la inserción, en cuyo caso un mecanismo no autorizará un tipo de atributo o valor de atributo del otro mecanismo. A este respecto, el permiso *DiscloseOnError* (véanse 18.2.3 y 18.2.4) de **basic-access-control** es un permiso que no suprimirá la denegación de **rule-based-access-control**.

20 Protección criptográfica en el almacenamiento

20.1 Integridad de los datos en el almacenamiento

20.1.1 Introducción

En algunas situaciones, el directorio puede no proporcionar una seguridad suficiente de que los datos no se modificarán en el almacenamiento, independientemente de los controles de acceso. Puede validarse la integridad de los datos almacenados en el directorio empleando firmas digitales mantenidas como parte de la información de directorio. La firma digital de una inserción o atributos seleccionados de una inserción puede mantenerse como un atributo (véase 20.1.2) o la firma digital de un único valor de atributo puede mantenerse en un contexto (véase 20.1.3).

NOTA – El DSA mantendrá la codificación del atributo situado en el directorio para asegurar que la firma calculada en el resultado devuelto es correcta.

20.1.2 Protección de una inserción o tipos de atributo seleccionados

Se proporciona la integridad de los datos de atributos almacenados empleando firmas digitales que se mantienen junto con los atributos a los que protegen. Se protege la integridad de toda la inserción o de todos los valores de atributos seleccionados de una inserción, mediante un atributo que mantiene la firma digital de todos los atributos protegidos.

Esta firma digital la crea la autoridad o usuario de directorio responsable de colocar la información en la inserción de directorio. La firma digital puede validarse por cualquier usuario que lea los valores de atributo de la inserción. El servicio de directorio no interviene, por sí mismo, en la creación o validación de la firma digital mantenida en su atributo.

Este mecanismo de integridad protege la integridad de los atributos de directorio tanto en el almacenamiento como durante la transferencia entre componentes de directorio (DSA y DUA). Este mecanismo de integridad no depende de la seguridad del propio servicio de directorio.

Las firmas digitales aplicadas a toda la inserción no incluyen atributos operacionales, colectivos o el propio **attributeIntegrityInfo**. Se incluyen todos los contextos de valor de atributo.

A continuación se define un tipo de atributo para mantener una firma digital junto con la información de control asociada, que proporciona la integridad de una inserción completa o de todos los valores de tipos de atributo seleccionados.

```
attributeIntegrityInfo ATTRIBUTE  ::=  {
        WITH SYNTAX             AttributeIntegrityInfo
        ID                      id-at-attributeIntegrityInfo}

AttributeIntegrityInfo  ::=  SIGNED { SEQUENCE  {
        scope           Scope,                  -- Identifies the attributes protected
        signer          Signer      OPTIONAL,           -- Authority or data originators name
        attribsHash     AttribsHash } }         -- Hash value of protected attributes

Signer  ::=  CHOICE {
        thisEntry   [0]   EXPLICIT ThisEntry,
        thirdParty  [1]   SpecificallyIdentified }

ThisEntry  ::=  CHOICE  {
        onlyOne     NULL,
        specific    IssuerAndSerialNumber }

IssuerAndSerialNumber  ::=  SEQUENCE {
        issuer      Name,
        serial      CertificateSerialNumber }

SpecificallyIdentified  ::=  SEQUENCE  {
        name        GeneralName,
        issuer      GeneralName                 OPTIONAL,
        serial      CertificateSerialNumber     OPTIONAL }
        ( WITH COMPONENTS { ..., issuer PRESENT, serial PRESENT } |
        ( WITH COMPONENTS { ..., issuer ABSENT, serial ABSENT } ) )

Scope  ::=  CHOICE  {
        wholeEntry      [0]   NULL,             -- Signature protects all attribute values in this entry
        selectedTypes   [1]   SelectedTypes
                              -- Signature protects all attribute values of the selected attribute types
        }

SelectedTypes  ::=  SEQUENCE SIZE (1..MAX) OF AttributeType

AttribsHash  ::=  HASH { SEQUENCE SIZE (1..MAX) OF Attribute }
                  -- Attribute type and values with associated context values for the selected Scope
```

Un valor **AttributeIntegrityInfo** se puede crear de tres formas distintas:

a) Una autoridad administrativa puede crear y firmar el valor, y la clave pública para verificar la firma se obtiene por medios fuera de línea.

b) El propietario de la inserción, es decir, el objeto representado por la inserción, puede crear y firmar el valor. Si el propietario dispone de varios certificados o espera disponer de ellos en el futuro, el certificado ha de ser identificado por la autoridad de certificación (CA) que expide el certificado junto con el número de serie del certificado.

c) Un tercero puede crear y firmar el valor. Se exige el nombre del signatario, el nombre de la CA que expide el certificado y el número de certificado.

Si el alcance es **wholeEntry**, todos los atributos aplicables estarán ordenados según está especificado en el conjunto tipo del descrito en 6.1 de la Rec. UIT-T X.509 | ISO/CEI 9594-8. Si el alcance es **selectedTypes**, el orden será el mismo que el indicado en **SelectedTypes**.

NOTA – Si un usuario no recupera todos los atributos completos que están definidos dentro del tipo de datos **Scope**, no será posible para ese usuario verificar la integridad de los atributos.

20.1.3 Contexto para la protección de un valor de atributo único

A continuación se define un contexto para mantener una firma digital junto con la información de control asociada que proporciona la integridad para un valor de atributo único. Cualquier contexto de valor de atributo está incluido en la comprobación de integridad, salvo los contextos usados para el mantenimiento de las firmas.

```
attributeValueIntegrityInfoContext  CONTEXT  ::=  {
        WITH SYNTAX     AttributeValueIntegrityInfo
        ID              id-avc-attributeValueIntegrityInfoContext }

AttributeValueIntegrityInfo  ::=  SIGNED { SEQUENCE {
        signer          Signer      OPTIONAL,       -- Authority or data originators name
        aVIHash         AVIHash } }                 -- Hash value of protected attribute

AVIHash  ::=  HASH { AttributeTypeValueContexts }
                -- Attribute type and value with associated context values

AttributeTypeValueContexts ::= SEQUENCE {
        type            ATTRIBUTE.&id ({SupportedAttributes}),
        value           ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
        contextList     SET SIZE (1..MAX) OF Context OPTIONAL }
```

El **contextList** se ordenará como se especifica para el tipo conjunto definido en 6.1 de la Rec. UIT-T X.509 | ISO/CEI 9594-8.

20.2 Confidencialidad de los datos almacenados

Advertencia – Se sabe que esta subcláusula contiene especificaciones no válidas. Esta subcláusula es por tanto desaprobada. En la próxima edición se eliminarán o actualizarán las especificaciones desaprobadas.

20.2.1 Introducción

En algunas situaciones, el directorio puede no proporcionar una seguridad suficiente de que los datos se mantienen almacenados confidencialmente, independientemente de los controles de acceso. Se consigue la confidencialidad de los atributos en el almacenamiento mediante el uso de:

a) una plantilla para la definición de un tipo de atributo que es una variante cifrada de un tipo de atributo existente;

b) un atributo para la distribución de claves de confidencialidad.

Una vez que el directorio acepta una variante protegida de un valor de atributo, no pueden invocarse las reglas de concordancia del atributo original (clear text).

NOTA – Puede utilizarse cualquier mecanismo para distribuir las claves necesarias para la protección de atributos definidos utilizando la plantilla "atributo confidencial". En la ISO/CEI 11770: Information technology – Security techniques – Key Management puede encontrarse información adicional sobre técnicas de gestión de claves.

20.2.2 Plantilla de valor de atributo criptado

Si se requiere una variante criptada de un tipo de atributo de directorio existente, se utilizará la siguiente sintaxis:

```
EncryptedAttributeSyntax {AttributeSyntax}  ::=  SEQUENCE  {
        keyInfo     SEQUENCE OF KeyIdOrProtectedKey,
        encAlg      AlgorithmIdentifier,
        encValue    ENCRYPTED { AttributeSyntax } }

KeyIdOrProtectedKey  ::=  SEQUENCE  {
        keyIdentifier       [0]   KeyIdentifier     OPTIONAL,
        protectedKeys       [1]   ProtectedKey      OPTIONAL }
              -- At least one key identifier or protected key shall be present
```

```
ProtectedKey ::= SEQUENCE {
        authReaders     AuthReaders,      -- if absent, use attribute in authorized reader entry
        keyEncAlg       AlgorithmIdentifier  OPTIONAL,      --  algorithm to encrypt encAttrKey
        encAttKey       EncAttKey }
                                -- confidentiality key protected with authorized user's
                                -- protection mechanism

AuthReaders ::= SEQUENCE OF Name

EncAttKey ::= PROTECTED {SymmetricKey, keyProtection}

SymmetricKey ::= BIT STRING

keyProtection PROTECTION-MAPPING ::= {
        SECURITY-TRANSFORMATION {genEncryption} }
```

NOTA 1 – No es razonable el empleo de reglas de concordancia ordenadas o reglas de concordancia de subcadena para los atributos criptados. Se recomienda también que únicamente se definan atributos criptados para los atributos de usuario.

La **AttributeSyntax** es la sintaxis en texto claro del atributo y la **EncryptedAttributeSyntax** es la sintaxis del atributo confidencial equivalente.

El **keyIdentifier** puede ser el identificador de una clave pública certificada como se mantiene en el campo de extensión identificador de clave de sujeto público definido en la cláusula 8 de la Rec. UIT-T X.509 | ISO/CEI 9594-8, o el identificador de una clave simétrica y de la información de control de seguridad asociada.

El **authReaders** identifica los sujetos a los que se permite la recuperación de **ProtectedKey**.

El **keyEncAlg** es el identificador del algoritmo utilizado para criptar el **encAttKey** que es el identificador de la clave utilizada para aplicar el servicio de confidencialidad.

La **genEncryption SECURITY-TRANSFORMATION** define la transformación utilizada para la distribución de la clave. Esto se define ulteriormente en 17.3.1. Puede utilizarse esta transformación con cualquier algoritmo de criptación, incluyendo algoritmos de clave pública reversible y algoritmos de clave simétrica.

NOTA 2 – La gestión de la clave, que incluye la distribución de la clave, pertenece al ámbito de cierta política de seguridad controlada administrativamente. En consecuencia, esta especificación no limita la **genEncryption SECURITY-TRANSFORMATION** de forma que impida al gestor de seguridad del DMD la especificación de detalles precisos de la distribución de la clave. Además, esta especificación no requiere calidades de **genEncryption SECURITY-TRANSFORMATION** de forma que tales calidades impidan el uso de mecanismos de distribución de claves particulares.

El soporte de un atributo no implica el soporte del atributo criptado equivalente.

Cuando en este conjunto de especificaciones se defina un atributo de usuario, los identificadores de objeto para el criptado equivalente al de ese atributo se asignan en el anexo A a la Rec. UIT-T X.520 | ISO/CEI 9594-6.

20.2.3 Atributo para la clave de confidencialidad

Si es necesario distribuir información sobre la criptación que proporcione confidencialidad de los datos almacenados, se utilizarán los siguientes atributos:

```
confKeyInfo ATTRIBUTE ::= {
        WITH SYNTAX                     ConfKeyInfo
        EQUALITY MATCHING RULE          readerAndKeyIDMatch
        ID                              id-at-confKeyInfo }

ConfKeyInfo ::= SEQUENCE {
        keyIdentifier   KeyIdentifier,
        protectedKey    ProtectedKey }
```

20.2.4 Regla de concordancia del identificador de la clave y del lector

La regla de concordancia para una lista de lectores autorizados y sus claves asociadas, es la siguiente:

```
readerAndKeyIDMatch MATCHING-RULE ::= {
        SYNTAX  ReaderAndKeyIDAssertion
        ID      id-mr-readerAndKeyIDMatch }
```

```
ReaderAndKeyIDAssertion  ::=  SEQUENCE {
        keyIdentifier       KeyIdentifier,
        authReaders         AuthReaders OPTIONAL }
```

La regla de concordancia devuelve un valor de VERDADERO si todos los componentes existentes en el valor presentado concuerdan con los valores correspondientes del valor de atributo, como sigue:

a) **keyIdentifier** concuerda si es igual al componente **keyIdentifier** de la **ProtectedKey** en el valor del atributo almacenado;

b) si está presente, el **authReaders** concuerda si es igual a uno de los nombres del componente **authReaders** de la **ProtectedKey** en el valor del atributo almacenado.

SECCIÓN 9 – MODELOS DE DSA

21 Modelos de DSA

Esta cláusula trata de los modelos generales que describen distintos aspectos de los componentes que constituyen el directorio y los agentes de sistema de directorio (DSA). Las cláusulas siguientes tratan de otros modelos de DSA.

21.1 Definiciones

A los efectos de esta Especificación de directorio, se aplican las siguientes definiciones.

21.1.1 fragmento de base de información de directorio: La porción de la DIB contenida en un DSA maestro; está formada por uno o más contextos de denominación.

21.1.2 prefijo de contexto: La secuencia de los RDN que conduce desde la raíz del DIT al vértice inicial de un contexto de denominación; corresponde al nombre distinguido de ese vértice.

21.1.3 contexto de denominación: Un subárbol de inserciones contenidas en un solo DSA maestro.

21.2 Modelo funcional de directorio

El directorio se presenta como un conjunto de uno o más procesos de aplicación conocidos por *agentes de sistema de directorio (DSA)*, cada uno de los cuales proporciona ninguno, uno, o más puntos de acceso. Esto se ilustra en la figura 19. Cuando el directorio comprende más de un DSA, se dice que está *distribuido*. Los procedimientos para el funcionamiento de directorio cuando éste es distribuido se especifican en la Rec. UIT-T X.518 | ISO/CEI 9594-4.

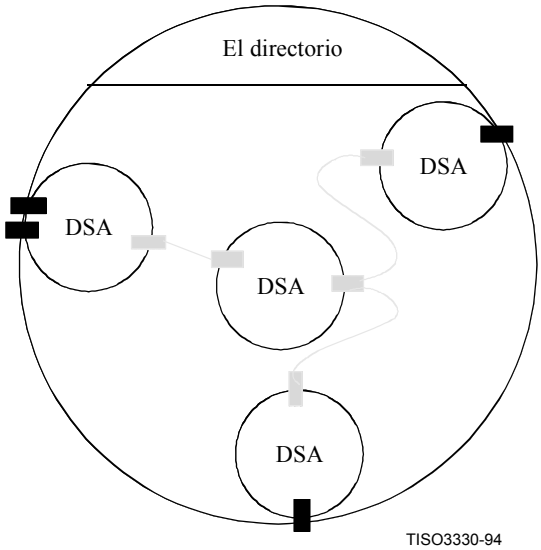


Figura 19 – Directorio proporcionado por múltiples DSA

NOTA – Un DSA presentará probablemente un comportamiento y una estructura locales que estén fuera del alcance de las Especificaciones de directorio contempladas. Por ejemplo, un DSA responsable de contener una parte o toda la información de la DIB normalmente lo hará por medio de una base de datos; la interfaz con una base de datos es un asunto local.

Un determinado par de procesos de aplicación que necesitan interactuar en el aprovisionamiento de servicios de directorio (sean un DUA y un DSA, o dos DSA) pueden estar situados en diferentes sistemas abiertos. Tal interacción se efectúa por medio de protocolos de directorio de OSI especificados en la Recomendación UIT-T X.519 | ISO/CEI 9594-5.

En la cláusula 23 se especifican los modelos que sirven de base para especificar los aspectos distribuidos de directorio. En las cláusulas 25 a 28 se proporciona un marco para la especificación de modelos operacionales concernientes a aspectos particulares de la operación de los componentes de directorio.

21.3 Modelo de distribución de directorio

Esta subcláusula define los principios con arreglo a los cuales la DIB puede ser distribuida.

Cada inserción en la DIB es administrada por un administrador solamente el administrador de DSA, del cual se dice que tiene autoridad administrativa para esa inserción. El mantenimiento y la gestión de una inserción se efectuará en un DSA administrado por la autoridad administrativa para la inserción. Este DSA se llama el *DSA maestro* para la inserción.

Cada DSA maestro en el directorio contiene un *fragmento* de la DIB. El fragmento de la DIB contenido por un DSA maestro se describe en términos del DIT y comprende uno o más contextos de denominación. Un *contexto de denominación* es un subárbol del DIT, todas cuyas inserciones tienen una autoridad administrativa común y están contenidas en el mismo DSA maestro. Un contexto de denominación comienza en un vértice del DIT (distinto de la raíz) y se extiende hacia abajo a vértices hoja y/o no-hoja. Estos vértices constituyen la frontera del contexto de denominación. El superior del vértice inicial de un contexto de denominación no está contenido en ese DSA maestro. Los subordinados de los vértices no-hoja pertenecientes a la frontera denotan el comienzo de ulteriores contextos de denominación.

NOTA 1 – El DIT está por tanto dividido en contextos de denominación disjuntos, cada uno de ellos bajo la autoridad administrativa de un solo DSA maestro.

NOTA 2 – Un contexto de denominación en sí mismo no es una zona administrativa que tiene un punto administrativo o una especificación de subárbol explícita, pero puede coincidir con una zona administrativa.

Una familia de inserciones residirá en un solo contexto de denominación.

Un administrador de DSA maestro puede tener responsabilidad administrativa sobre varios contextos de denominación disjuntos. Todo contexto de denominación sobre el cual un DSA maestro tiene autoridad administrativa contendrá lógicamente la secuencia de los RDN que conduce de la raíz del DIT al vértice inicial del subárbol que forma el contexto de denominación. Esta secuencia de los RDN se llama el *prefijo de contexto* del contexto de denominación.

NOTA 3 – Se utilizará como prefijo de contexto el nombre distinguido primario del contexto denominado. En los RDN pueden incluirse, facultativamente, contextos y valores alternativos con contexto.

Un administrador de DSA maestro puede delegar autoridad administrativa con respecto a cualesquiera subordinados inmediatos de cualquiera inserción contenida localmente, a otro DSA maestro. Un DSA que ha delegado autoridad se denomina *DSA superior* y el contexto que contiene la inserción superior de uno con respecto al cual la autoridad administrativa fue delegada se llama el *contexto de denominación superior*. La delegación de autoridad administrativa comienza por la raíz y continúa en sentido descendente en el DIT; por consiguiente sólo puede realizarse desde una inserción a sus subordinados.

La figura 20 ilustra un DIT hipotético dividido lógicamente en cinco contextos de denominación (denominados A, B, C, D y E), que están distribuidos físicamente en tres DSA (DSA1, DSA2 y DSA3).

En el ejemplo puede verse que los contextos de denominación contenidos en los DSA maestros particulares pueden ser configurados de modo que satisfagan una amplia gama de exigencias operacionales. Ciertos DSA maestros pueden ser configurados de modo que contengan las inserciones que representan dominios de denominación de nivel más alto dentro de algunas partes lógicas de la DIB, digamos, la estructura organizacional de una compañía grande, pero no necesariamente todas las inserciones subordinadas. Como otra posibilidad, se puede configurar los DSA maestros de modo que sólo contengan los contextos de denominación que representan esencialmente inserciones de hoja.

En base a las definiciones precedentes, el caso límite para un contexto de denominación puede ser una sola inserción o todo el DIT.

Si bien la correspondencia lógica a física del DIT en los DSA maestros pudiera ser arbitraria, la tarea de la localización y la gestión de la información se simplifica si los DSA maestros son configurados de suerte que contengan un pequeño número de contextos de denominación.

Los DSA pueden contener copias de inserción así como inserciones. Las inserciones sombreadas, el único género de copia de inserción considerado en las Especificaciones de directorio, son mantenidas por medio del servicio de sombreado descrito en la Rec. UIT-T X.525 | ISO/CEI 9594-9. Además de este género normalizado de información replicada, pueden encontrarse en el directorio otros dos géneros no normalizados de copia de inserciones:

- Pueden almacenarse copias de una inserción en otros DSA mediante acuerdo bilateral.
- Pueden adquirirse copias de una inserción almacenando (local y dinámicamente) una copia velada de una inserción resultante de una petición.

  NOTA 4 – Los medios para mantener y gestionar estas copias se definen en estas Especificaciones de directorio. Debido al tratamiento más preciso de características como el control de acceso, se recomienda que se utilice el servicio de replicación en vez de utilizar copias veladas.

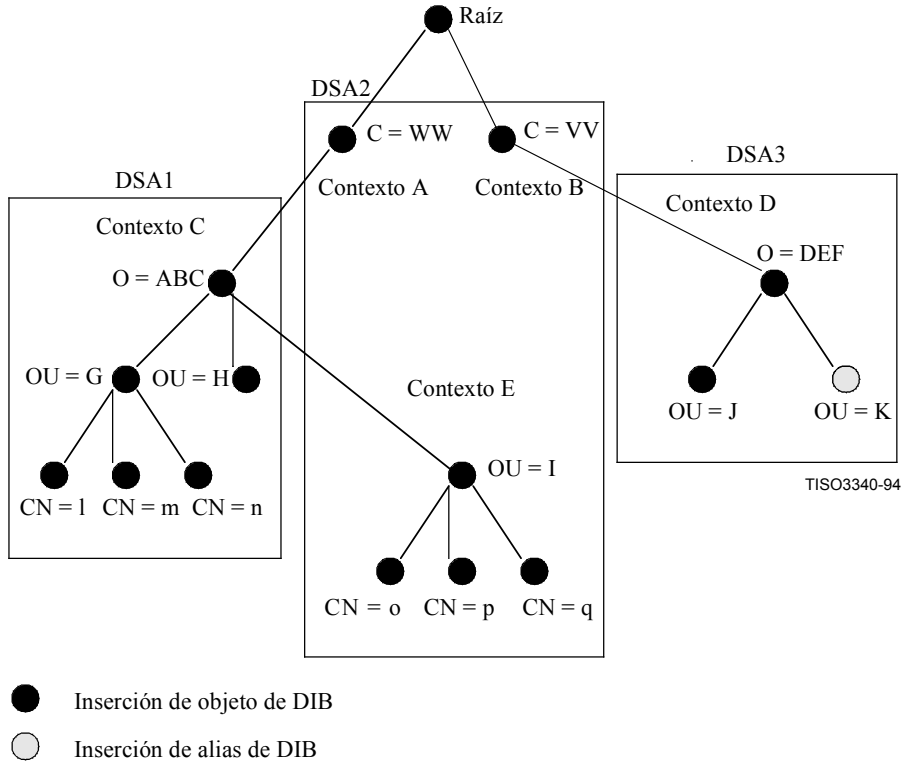


Figura 20 – DIT hipotético

Un DSA que mantiene una copia de inserción es un *DSA de sombra* para esa inserción. Un DSA de sombra puede mantener una copia de un contexto denominador o una porción del mismo. La especificación de la porción de un contexto de denominación que está sombreado se denomina una *unidad de replicación*.

Como se describe en 9.2 de la Rec. UIT-T X.525 | ISO/CEI 9594-9, se define una unidad de replicación definida dentro del modelo de información de directorio y se proporciona un mecanismo de especificación. El mecanismo de sombreado en el directorio se basa en la definición del subconjunto del DIT que será sombreado. Este subconjunto se denomina *unidad de replicación*. La unidad de replicación comprende una especificación de tres partes que define el alcance de la porción del DIT que ha de replicarse, los atributos que han de replicarse dentro de ese ámbito y los requisitos de conocimiento subordinado. La unidad de replicación hace también implícitamente que la información sombreada incluya información de política en forma de atributos operacionales mantenidos en inserciones y subinserciones (por ejemplo, información de control de acceso) que ha de utilizarse para realizar correctamente las operaciones de directorio. La información de prefijos que ha de incluirse comienza en un punto administrativo autónomo y se extiende a la inserción de base de replicación.

Al originador de una petición al directorio se le informa (mediante **fromEntry**) si la información devuelta en respuesta a una petición proviene de una copia de inserción o no. Se define un servicio de control, **dontUseCopy**, que autoriza al usuario a prohibir el uso de copias de inserción para responder a la petición (aunque pueda utilizarse información de copia en la resolución de nombres).

NOTA 5 – En algunos casos puede fracasar esa resolución de nombres para un nombre alternativo válido cuando se resuelva empleando una copia mantenida en un DSA de edición anterior a 1997 o en un DSA de una última edición que tiene una copia con una información de nombre incompleta, en la que un RDN incluye un tipo de atributo para el cual hay múltiples valores distinguidos diferenciados por el contexto.

Para poder comenzar a procesar una petición, un DUA deberá poseer alguna información, esencialmente la dirección de presentación, por lo menos sobre un DSA con el que puede ponerse en contacto inicialmente. El método que el DUA utiliza para adquirir y retener esta información es un asunto local.

Durante el proceso de modificación de inserciones es posible que el directorio se haga incoherente. Esta situación es particularmente probable que ocurra si la modificación incluye alias u objetos con alias que pueden estar en diferentes DSA. La incoherencia deberá ser corregida por una acción específica del administrador, por ejemplo para suprimir alias si los correspondientes objetos con alias han sido suprimidos. El directorio continúa funcionando durante este periodo de incoherencia.

SECCIÓN 10 – MODELO DE INFORMACIÓN DE DSA

22 Conocimiento

22.1 Definiciones

A los efectos de esta Especificación de directorio se aplican las siguientes definiciones:

22.1.1 categoría: Una característica de una referencia de conocimiento que la califica para identificar un DSA como maestro o sombra para un contexto de denominación.

22.1.2 comúnmente utilizable: Una característica de zona replicada que permite una distribución general del punto de acceso que la contiene; una zona replicada utilizable en común es normalmente una copia sombra completa de un contexto de denominación.

22.1.3 referencia cruzada: Una referencia de conocimiento que contiene información sobre el DSA que posee una inserción o una copia de inserción. Se utiliza para optimización. La inserción no necesita tener una relación de superior o subordinado con cualquier inserción en el DSA que contiene la referencia cruzada.

22.1.4 visión disjunta (del DIT): En ausencia de un trayecto de referencia, puede no resultar posible resolver un nombre distinguido (es decir, encontrar la ubicación de un DSA que contenga un nombre distinguido específico) desde el punto de vista de un DSA que contenga una porción determinada del DIT, porque no todas las partes del DIT pueden ser accesibles utilizando operaciones encadenadas. La porción del DIT contenida en ese tipo de DSA se le llama visión disjunta y se dice que el DSA contiene una visión disjunta. Una visión disjunta se produce cuando los DSA que contienen subárboles separados del DIT no interactúan a fin de soportar la accesibilidad del DIT completo.

22.1.5 referencia superior inmediata: Una referencia de conocimiento que contiene información sobre un DSA que contiene el contexto de denominación (o una zona replicada utilizable en común derivada a éste) que es inmediatamente superior a uno contenido en el DSA para el cual la referencia de conocimiento es pertinente.

22.1.6 (información de) conocimiento: Información operacional de DSA contenida por un DSA que la utiliza para localizar información de inserción o de copia de inserción remota.

22.1.7 referencia de conocimiento: Conocimiento que asocia, directa o indirectamente, una inserción o copia de inserción de DIT con el DSA en que está situada.

22.1.8 conocimiento maestro: Conocimiento del DSA maestro para un contexto de denominación.

22.1.9 referencia subordinada no específica: Una referencia de conocimiento que contiene información sobre el DSA que posee una o más inserciones o copias de inserciones subordinadas no específicas.

22.1.10 trayecto de referencia: Una secuencia continua de referencias de conocimiento.

22.1.11 conocimiento (de) sombra: Conocimiento de uno o más DSA sombra para un contexto de denominación (si el conocimiento es específico) o contextos de denominación (si no es específico).

22.1.12 referencia subordinada: Una referencia de conocimiento que contiene información sobre el DSA que tiene una inserción o copia de inserción subordinada específica.

22.1.13 referencia superior: Una referencia de conocimiento que contiene información sobre un DSA al que se considera capaz de resolver el DIT en su totalidad (es decir, hallar cualquier inserción contenida en el mismo).

22.2 Introducción

La DIB está distribuida a través de un gran número de DSA maestros, cada uno de los cuales contiene, un fragmento de la DIB del cual es responsable. Los principios que gobiernan la distribución se especifican en 21.3.

Además, estos y otros DSA pueden contener copias de porciones de la DIB.

Es un requisito de directorio que, para modos particulares de información de usuario, la distribución de directorio se haga transparente, con lo que se produce el efecto de que la DIB entera parece estar dentro de cada uno de los DSA.

Para soportar esta exigencia operacional, es necesario que cada DSA sea capaz de acceder a la información contenida en la DIB asociada a cualquier nombre (es decir, a cualquier nombre distinguido o de alias del objeto). Si el propio DSA no contiene una inserción del objeto o una copia de inserción del objeto asociadas con el nombre, deberá poder interactuar con un DSA que lo contenga, sea directa, o sea indirectamente por medio de interacciones directas y/o indirectas con otros DSA.

Cuando un usuario de directorio indica que no puede utilizarse información de copia de inserción para satisfacer su petición, el DSA que atiende la petición deberá poder acceder, directa o indirectamente al DSA maestro que contiene la información de inserción asociada con el nombre suministrado en la petición del usuario.

Esta cláusula define el conocimiento como la información operacional de DSA requerida para alcanzar estos objetivos técnicos. Las cláusulas siguientes especifican la representación de conocimiento en el contexto de un modelo general de información de DSA.

NOTA – Los enunciados precedentes representan objetivos técnicos de directorio. La realización de estos objetivos técnicos depende de otras cosas (por ejemplo, asuntos de política), además de una configuración coherente de conocimiento en los DSA. Las cláusulas 25 a 28 establecen un marco para tratar algunas de estas cuestiones.

El anexo O contiene una ilustración del modelado del conocimiento. La ilustración se basa en el DIT ficticio mostrado en la figura 20.

22.3 Referencias de conocimiento

Conocimiento es la información operacional contenida por un DSA y que representa una descripción parcial de la distribución de información de inserción y de copia de inserción contenida en otros DSA. Un DSA utiliza conocimiento para determinar con qué DSA apropiado ha de comunicar cuando una petición recibida de un DUA o de otro DSA no puede ser satisfecha con información mantenida localmente.

Un conocimiento consiste en referencias de conocimiento. Una *referencia de conocimiento* asocia directa o indirectamente el nombre de una inserción de directorio con un DSA que contiene la inserción o una copia de la inserción.

Los nombres utilizados en las referencias de conocimiento, sean prefijos de contexto, nombres de DSA o nombres de inserción, deberán ser nombres distinguidos primarios. En los RDN pueden también incluirse contextos y valores alternativos con contexto.

NOTA – La resolución de nombres puede fracasar para un nombre alternativo válido cuando las referencias de conocimiento se mantengan en DSA anteriores a 1997 que no reconocen múltiples valores distinguidos diferenciados por el contexto o en los DSA que no mantengan todos los nombres distinguidos alternativos en las referencias de conocimiento o copias de inserción.

22.3.1 Categorías de conocimiento

Hay dos categorías de referencia de conocimiento: referencias de conocimiento maestro y referencias de conocimiento sombra.

Conocimiento maestro es el conocimiento del punto de acceso del DSA maestro para un contexto de denominación.

Conocimiento sombra es el conocimiento de los DSA que poseen información de directorio replicada; puede ser distribuido por suministradores de sombra a consumidores de sombra por medio de los procedimientos de replicación descritos en la Rec. UIT-T X.525 | ISO/CEI 9594-9. Conocimiento sombra es el conocimiento del punto de acceso de un conjunto de uno o más DSA sombra para una zona replicada (un contexto de denominación o una porción del mismo).

Un DSA que es el objeto de conocimiento sombra mantendrá una zona replicada utilizable en común. Una forma de zona replicada que es utilizable en común es una copia sombra completa de un contexto de denominación. Una copia sombra incompleta de un contexto de denominación mantenido por un DSA puede ser utilizable en común si está suficientemente completa para satisfacer las peticiones de interrogación que los usuarios suelen hacer al DSA. Es responsabilidad de la autoridad administrativa que origina el conocimiento sombra de un DSA que mantiene una copia incompleta de un contexto de denominación que ha de distribuirse, que la zona replicada sea utilizable en común.

Un DSA dado puede contener conocimiento maestro y conocimiento sombra, implicando este último múltiples DSA sombra, con respecto a un determinado contexto de denominación. El conocimiento específico utilizado en el procesamiento de una petición recibida de un DUA o de otro DSA, por ejemplo en el proceso de resolución de nombre, viene determinado por el procedimiento de selección específico de DSA por el cual el DSA computa, basándose en cualesquiera criterios no normalizados, que la autoridad administrativa estima apropiados, un punto de acceso de un DSA capaz de hacer progresar la petición.

NOTA – Las Especificaciones de directorio no restringen la forma en que los DSA utilizan los conocimientos maestro y sombra (salvo en cuanto a constricciones relativas al comportamiento de los DSA con respecto a los controles de servicio **dontUseCopy** y **copyShallDo** especificados en la Rec. UIT-T X.511 | ISO/CEI 9594-3).

22.3.2 Tipos de referencia de conocimiento

El conocimiento poseído por un DSA se define en términos de un conjunto de una o más referencias de conocimiento, donde cada referencia asocia directa o indirectamente inserciones (o copias de inserción) de la DIB con DSA que contienen esas inserciones (o copias de inserciones).

Un DSA puede contener los siguientes tipos de referencia de conocimiento:

- una referencia superior,
- referencias superiores inmediatas,
- referencias subordinadas,
- referencias subordinadas no específicas, y
- referencias cruzadas.

Una referencia de conocimiento de un tipo particular será una referencia de conocimiento maestro, o una referencia de conocimiento sombra.

Además, un DSA que participa en sombreado como un suministrador y/o consumidor de sombra puede contener uno o más de los siguientes tipos de referencia de conocimiento:

- referencias de suministrador, y
- referencias de consumidor.

A continuación se describen estos tipos de referencia de conocimiento.

22.3.2.1 Referencia superior

Una referencia superior consiste en:

- el punto de acceso de un DSA.

Todo DSA que no es de primer nivel (véase 22.5) mantiene por lo menos una referencia superior. La referencia superior formará parte de un trayecto de referencia a la raíz. A menos que se emplee algún método no normalizado para asegurarlo, por ejemplo dentro de un DMD, esto se conseguirá refiriéndose a un DSA que tenga un contexto de denominación o zona replicada cuyo prefijo de contexto tenga menos RDN que el prefijo de contexto que menos RDN tenga, entre los contenidos por este DSA.

22.3.2.2 Referencias superiores inmediatas

Una referencia superior inmediata consiste en:

- el prefijo de contexto de un contexto de denominación que es inmediatamente superior a uno contenido (como inserciones o copias de inserciones), por el DSA que posee la referencia;
- el punto de acceso del DSA que posee ese contexto de denominación (como inserciones o copias de inserciones).

Las referencias superiores inmediatas son un tipo de referencia facultativa que sólo aparece cuando hay una vinculación operacional jerárquica al DSA referenciado (véase la cláusula 24 de la Rec. UIT-T X.518 | ISO/CEI 9594-4). En ausencia de tal vinculación operacional explícita, un contexto de denominación inmediatamente superior puede ser referenciado mediante una referencia cruzada.

22.3.2.3 Referencias subordinadas

Una *referencia subordinada* consiste en:

- un prefijo de contexto que corresponde a un contexto de denominación inmediatamente subordinado a uno contenido (como inserciones o copias de inserciones) por el DSA que contiene la referencia;
- el punto de acceso del DSA que tiene ese contexto de denominación (como inserciones o copias de inserciones).

Todos los contextos de denominación inmediatamente subordinados a contextos de denominación poseídos (como inserciones o copias de inserciones) por un DSA serán representados por referencias subordinadas (o por referencias subordinadas no específicas como se indica en 22.3.2.4).

Cuando un DSA mantiene copias de inserciones, los contextos de denominación subordinados pueden estar representados o no, según el acuerdo de sombra vigente.

22.3.2.4 Referencias subordinadas no específicas

Una *referencia subordinada no específica* consiste en:

- el punto de acceso de un DSA que contiene las inserciones (o copias de inserciones) de uno o más contextos de denominación inmediatamente subordinados.

Este tipo de referencia es facultativa, y se utiliza para tener en cuenta el caso en que se sabe que un DSA contiene algunas inserciones (o copias de inserciones) subordinadas, pero no se conocen los RDN específicos de esas inserciones (o copias de inserciones).

Para cada contexto de denominación que posee, sea como inserciones o como copias de inserciones, un DSA puede contener ninguna o varias referencias subordinadas no específicas. Los DSA accedidos por una referencia no específica deberán poder resolver la petición directamente (culmine ésta en éxito o fracaso). En el caso de fracaso, se devuelve al solicitante un **serviceError** que informa un problema de **unableToProceed**.

Cuando un DSA mantiene copias de inserciones, las referencias subordinadas no específicas pueden estar representadas o no, según el acuerdo de sombra vigente.

22.3.2.5 Referencias cruzadas

Una *referencia cruzada* consiste en:

- un prefijo de contexto;
- el punto de acceso de un DSA que contiene las inserciones o copias de inserciones para ese contexto de denominación.

Este tipo de referencia es facultativa y se emplea para optimizar la resolución de nombre. Un DSA puede contener un número cualquiera (incluido cero) de referencias cruzadas.

22.3.2.6 Referencias de suministrador

Una referencia de suministrador contenida por un DSA consumidor de sombra consiste en:

- el prefijo de contexto del contexto de denominación del que se derivó la zona replicada recibida del consumidor de sombra;
- el identificador del acuerdo de sombreado que el consumidor de sombra ha establecido con el suministrador de sombra;
- el punto de acceso del DSA suministrador de sombra;
- una indicación de si el suministrador de sombra de la zona replicada es o no el DSA maestro; y
- facultativamente, el punto de acceso del DSA maestro, si el suministrador no es el DSA maestro.

22.3.2.7 Referencias de consumidor

Una referencia de consumidor contenida por un DSA suministrador de sombra consiste en:

- el prefijo de contexto de un contexto de denominación del que se derivó la zona replicada contenida, proporcionada por el suministrador de sombra;
- el identificador del acuerdo de sombreado que el suministrador de sombra ha establecido con un consumidor; y
- el punto de acceso del DSA consumidor de sombra.

22.4 Conocimiento mínimo

Es una propiedad de directorio que se pueda acceder a cada inserción independientemente de donde se genera la petición.

Es también una propiedad de directorio que, para obtener niveles adecuados de rendimiento y disponibilidad, algunas peticiones pueden ser satisfechas utilizando una copia de una inserción, mientras que otras sólo pueden serlo utilizando la propia inserción (es decir, la información contenida en el DSA maestro para la inserción).

Para realizar estas propiedades de directorio, que son independientes de la ubicación, cada DSA mantendrá una cantidad mínima de conocimiento, que depende de su configuración particular.

El objetivo de estos requisitos mínimos es permitir que el proceso distribuido de resolución de nombre establezca un trayecto de referencia, como una secuencia continua de referencias de conocimiento maestro, a todos los contextos de denominación dentro de directorio.

Además de estos requisitos mínimos, puede emplearse conocimiento adicional para establecer otros trayectos de referencia a copias de contextos de denominación. Se puede emplear conocimiento (de maestro y de sombra) de referencia cruzada para establecer trayectos de referencia optimizados a contextos de denominación y copias de contextos de denominación.

Los requisitos de conocimiento mínimos para los DSA se especifican en 22.4.1 a 22.4.4.

22.4.1 Conocimiento superior

Todo DSA que no es de primer nivel mantendrá por lo menos una sola referencia superior. Puede mantener referencias superiores adicionales por motivos operacionales como trayectos alternativos a la raíz del DIT.

22.4.2 Conocimiento subordinado

Un DSA que es el DSA maestro de un contexto de denominación mantendrá referencias subordinadas, o referencias subordinadas no específicas, de la categoría conocimiento maestro, a cada DSA maestro que contenga (como maestro) un contexto de denominación inmediatamente subordinado.

22.4.3 Conocimiento de suministrador

Por cada DSA suministrador de sombra que le suministra una zona replicada, un DSA consumidor de sombra deberá mantener una referencia de suministrador. Si el conocimiento subordinado del consumidor de sombra para la copia del contexto de denominación está incompleto, utilizará su referencia de suministrador para establecer un trayecto de referencia a información subordinada. Este procedimiento se describe en la cláusula 20 de la Rec. UIT-T X.518 | ISO/CEI 9594-4.

22.4.4 Conocimiento de consumidor

Por cada DSA consumidor de sombra al que suministra una zona replicada, un DSA suministrador de sombra deberá mantener una referencia de consumidor.

22.5 DSA de primer nivel

El DSA referenciado por una referencia superior asume la carga de establecer un trayecto de referencia a todas las porciones del DIT que no son conocidas por el DSA referenciante. Un DSA referenciado por otros DSA puede, él mismo, mantener una o más referencias superiores. Este proceso recursivo de referimiento superior se detiene en un conjunto de *DSA de primer nivel* sobre los que recae la responsabilidad final del establecimiento de trayectos de referencia.

Un DSA de primer nivel se caracteriza por lo siguiente:

a) no contiene una referencia superior;

b) puede contener uno o más contextos de denominación inmediatamente subordinados a la raíz del DIT (como DSA maestro o sombra para el contexto de denominación); y

c) contiene una referencia subordinada (de categoría maestro y/o sombra) y referencias subordinadas no específicas (de categoría maestro y/o sombra) que tienen en cuenta todos los contextos de denominación inmediatamente subordinado a la raíz del DIT que él no contiene.

Las autoridades administrativas para los DSA de primer nivel son conjuntamente responsables de la administración de los subordinados inmediatos de la raíz del DIT. Los procedimientos que rigen esta administración conjunta se determinan por acuerdos multilaterales que están fuera del ámbito de estas Especificaciones de directorio.

Para limitar la cantidad de peticiones de interrogación que pudieran ser dirigidas a un DSA de primer nivel maestro (es decir, un DSA que es un maestro para un contexto de denominación inmediatamente subordinado a la raíz del DIT), es posible establecer los DSA de primer nivel sombra para ese DSA de primer nivel maestro. Estos DSA sombra contienen copias de las inserciones y referencias subordinadas inmediatamente subordinadas a la raíz contenida en su DSA de primer nivel maestro (o suministrador). Por tanto, pueden servir como una referencia superior para los DSA que no son de primer nivel.

23 Elementos básicos del modelo de información de DSA

23.1 Definiciones

A los efectos de esta Especificación de directorio se aplican las siguientes definiciones.

23.1.1 árbol de información (de) agente de sistema de directorio:El conjunto de todos las DSE contenidas por un DSA cuando son visualizados desde la perspectiva de sus nombres.

23.1.2 atributo compartido por agente de sistema de directorio: Un atributo operacional del modelo de información de DSA asociado con un nombre particular cuyo valor o valores, si están contenidos por varios DSA, son idénticos (excepto durante periodos de incoherencia transitoria).

23.1.3 atributo específico de agente de sistema de directorio: Un atributo operacional en el modelo de información DSA asociado con un nombre particular cuyo valor o valores, si están contenidos por varios DSA, no tienen que ser idénticos.

23.1.4 inserción específica de agente de sistema de directorio (DSE, *DSA-specific entry*): Información contenida por un DSA que está asociada con un nombre particular; DSE puede, pero no tiene necesariamente que contener la información asociada con la correspondiente inserción de directorio.

23.1.5 tipo de agente de sistema de directorio: Una indicación del propósito particular de una DSE; una DSE puede servir a múltiples propósitos, por lo que puede tener varios tipos.

23.2 Introducción

El modelo de información de directorio describe cómo el directorio, en su conjunto, representa información sobre objetos que tienen un nombre distinguido y facultativamente nombres de alias. En su descripción del DIT, las inserciones y los atributos, la composición de directorio se abstrae del modelo, como un conjunto de DSA potencialmente cooperantes.

El modelo de información de DSA, en cambio, se ocupa especialmente de los DSA y de la información que éstos deberán contener para que el conjunto de DSA que constituyen el directorio puedan realizar juntos el modelo de información de directorio. Este modelo se relaciona con:

- la correspondencia de la información de directorio (inserciones y subinserciones de objeto y de alias) con los DSA;
- cómo las copias de información de directorio pueden ser contenidas en los DSA;
- la información operacional requerida por los DSA para efectuar resoluciones de nombre y evaluaciones de operación; y
- la información operacional requerida por los DSA para efectuar sombreado y utilizar información sombreada.

El modelado de una representación de información operacional de DSA tal como conocimiento tiene por finalidad establecer el marco general para el acceso a información operacional de DSA con fines de gestión.

23.3 Inserciones específicas de DSA y sus nombres

En el modelo de información de DSA, los depósitos de información que contienen la información asociada con un nombre particular se llaman *inserciones específicas de DSA* (DSE, *DSA-specific entries*). Las inserciones de directorio existen en el modelo de información de DSA solamente como elementos de información de los cuales pueden estar compuestos las DSE. Los atributos operacionales específicos al modelo de información de DSA forman la otra variedad de elemento de información de la cual pueden estar compuestos las DSE.

Si un DSA contiene cualquier información que concierne a un nombre directamente (esto es, una información contenida en un depósito identificado por el nombre), se dice que conoce, o tiene conocimiento de ese nombre.

Para cada nombre conocido por un DSA, toda la información contenida por el DSA, y que está directamente asociada con el nombre pero no es el nombre mismo, se representa por una DSE. Esta última información (es decir, el RDN y su relación con el DIT) no se representa explícitamente como atributos en el modelo de información de DSA; el conjunto de nombres conocidos por un DSA constituye una textura implícita a la cual pueden considerarse adjuntados las DSE asociadas.

NOTA 1 – Una consecuencia de la manera en que el modelo de información de DSA trata los nombres es que, para las DSE que no son de tipo inserción, alias o subinserción, la o las AVA que expresan el RDN de la DSE no son modeladas como contenidas en uno o más atributos.

Cuando existan nombres alternativos debido a que los atributos de denominación posen múltiples valores distinguidos diferenciados por contexto, una DSE única representa toda la información mantenida por el DSA sobre todos los nombres alternativos. Ésta se modela en el modelo de la información del DSA como un nombre único con variantes específicas de contexto en vez de como nombres separados.

NOTA 2 – Para una resolución coherente de nombres y para el interfuncionamiento de los DSA anteriores a 1997, cada DSA deberá tener al menos información sobre los valores distinguidos primarios de todos los atributos que contribuyen a un nombre y, preferentemente, del mayor número de valores distinguidos alternativos posibles.

El conjunto de todos los nombres conocidos por un DSA, junto con la información asociada con cada nombre, cuando se ven desde la perspectiva de estos nombres, se llama el *árbol de información de DSA* para ese DSA. En la figura 21 se representa por un árbol de información de DSA.

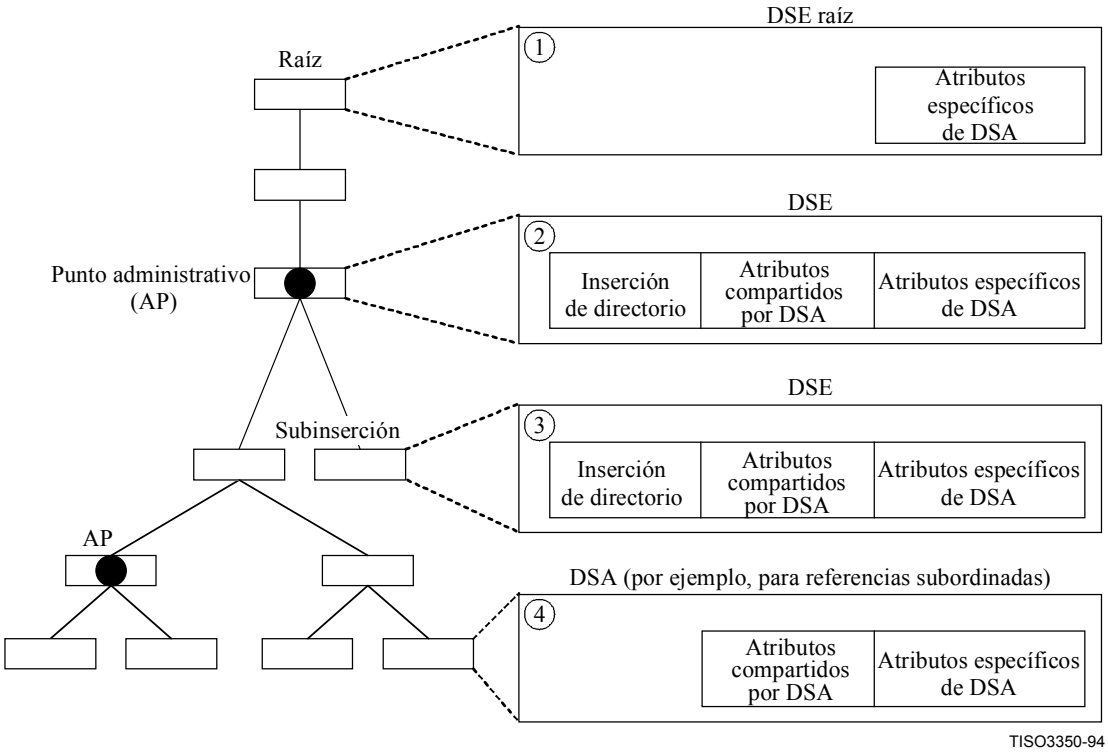


Figura 21 – Árbol de información de DSA

La información mínima que un DSA puede asociar con un nombre, conociendo de esa forma el nombre, consiste en una expresión del propósito para el cual se conoce el nombre (es decir, el papel desempeñado por el nombre en la operación del DSA que lo conoce). Este propósito se representa en el modelo de información de DSA por el atributo específico de DSA, **dseType**.

Además, un DSE puede contener otras informaciones asociadas con el nombre tales como una inserción o copia de inserción, atributos compartidos por DSA y atributos específicos de DSA.

Un DSE puede representar una inserción de directorio directamente, una porción de una inserción, o ninguna información de directorio. La información contenida en una DSE varía según su tipo o propósito. En general, los siguientes géneros de DSE pueden aparecer en los DSA:

- Una DSE que representa directamente una inserción de directorio contiene los atributos de usuario y operacionales correspondientes a esa inserción de directorio (como se representa en DSE 2 de la figura 21). La DSE puede también contener atributos compartidos por DSA y atributos específicos de DSA.
- Una DSE que representa una porción de una inserción (como resultado de sombreado) contiene algunos de los atributos de usuario y operacionales correspondientes a la inserción de directorio, atributos específicos de DSA, y puede también contener atributos compartidos por DSA.
- Una DSE de subinserción que representa, por ejemplo, ACI prescriptiva o atributos colectivos contiene los atributos de usuario y operacionales correspondientes a una subinserción de directorio (como se representa en DSE 3 de la figura 21). La DSE puede contener también atributos compartidos por DSA y atributos específicos de DSA.

– Una DSE que no representa ninguna información de inserción de directorio contiene solamente atributos compartidos por DSA y/o específicos de DSA (como se representa en los DSE 1 y 4 en la figura 21). Por ejemplo, una DSE que representa una referencia subordinada puede tener un atributo compartido por DSA que indique el punto de acceso maestro y un atributo específico de DSA que indique que la DSE es una referencia subordinada.

NOTA 3 – La DSE es una entidad conceptual que facilita la especificación y modelado de componentes de información de una manera coherente y conveniente. Aunque se dice que las DSE "contienen" o "almacenan" información, no se pretende con ello imponer a las realizaciones ninguna constricción o estructura de datos particulares.

23.4 Elementos básicos

Una DSE se compone de tres elementos básicos: el tipo de DSE, cierto número de atributos operacionales de DSA (el tipo de DSE es uno de ellos), y facultativamente una inserción o copia de inserción.

23.4.1 Atributos operacionales de DSA

En el modelo de información de DSA existen dos variedades de atributo operacional que no corresponden a información en inserciones de directorio, a saber: atributos compartidos por DSA y atributos específicos de DSA.

Un *atributo compartido por DSA* es un atributo operacional en el modelo de información de DSA asociado con un nombre particular cuyo valor o valores, si son contenidos por varios DSA, son idénticos (excepto durante periodos de incoherencia transitoria). Un DSA puede contener una copia sombra de un atributo compartido por DSA.

Un *atributo específico de DSA* es un atributo operacional en el modelo de información de DSA, cuyo valor o valores, si están contenidos por varios DSA, no tienen que ser idénticos. Un atributo específico de DSA representa información operacional que es específica del funcionamiento del DSA que la contiene. Un DSA no puede contener una copia sombra de un atributo específico de DSA.

NOTA – Si bien un DSA suministrador de sombra puede proporcionar a un DSA consumidor de sombra un atributo específico de DSA, éste no es, conceptualmente, una copia sombra de la información contenida por el suministrador, sino más bien una información producida por el suministrador para el consumidor, que el consumidor puede utilizar y modificar.

23.4.2 Tipos de DSE

El tipo de una DSE, representado en el modelo de información de DSA por el atributo operacional específico de DSA **dseType**, indica el propósito (o cometido) particular de una DSE. Este propósito se indica por los bits denominados del valor único del atributo **dseType**. Puesto que una DSE puede servir a varios propósitos, varios bits denominados del atributo **dseType** pueden ser fijados de modo que representen estos propósitos. En el anexo N se especifican varias combinaciones de bits denominados que pudieran aparecer.

La frase "una DSE de tipo **x**" se utiliza en las Especificaciones de directorio para indicar que el bit denominado **x** del atributo **dseType** del DSE ha sido fijado. Para una DSE de tipo **x**, otros bits denominados pueden o no ser fijados, según se requiera. Puede utilizarse también la frase alternativa "el tipo DSE incluye **x**".

La especificación sintáctica del atributo operacional **dseType** puede expresarse utilizando la siguiente notación de atributo:

```
dseType  ATTRIBUTE  ::=  {
        WITH SYNTAX                     DSEType
        EQUALITY MATCHING RULE          bitStringMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-dseType }
```

Este atributo operacional específico de DSA es gestionado por el propio DSA.

El tipo ASN.1 que representa la sintaxis de los posibles valores del atributo **dseType** es **DSEType**. Su definición es:

```
DSEType  ::=  BIT STRING {
        root            (0),    -- root DSE --
        glue            (1),    -- represents knowledge of a name only --
        cp              (2),    -- context prefix --
        entry           (3),    -- object entry --
        alias           (4),    -- alias entry --
        subr            (5),    -- subordinate reference --
        nssr            (6),    -- non-specific subordinate reference --
```

```
supr            (7),        -- superior reference --
xr              (8),        -- cross reference --
admPoint        (9),        -- administrative point --
subentry        (10),       -- subentry --
shadow          (11),       -- shadow copy --
immSupr         (13),       -- immediate superior reference --
rhob            (14),       -- rhob information --
sa              (15),       -- subordinate reference to alias entry --
dsSubentry      (16),       -- DSA Specific subentry --
familyMember    (17) }      -- family member --
```

Los valores de **DSEType** son:

a) **root** (**raíz**): La DSE raíz contiene atributos específicos de DSA, utilizados por el DSA, que caracterizan a ese DSA como un todo. El nombre correspondiente a la DSE raíz es el nombre degenerado constituido por una secuencia de cero RDN.

NOTA – La información que caracteriza a un DSA y que se va a poner a disposición a través del servicio abstracto de directorio está contenida en la inserción del DSA. Un DSA puede, pero no tiene necesariamente que contener su propia inserción o una copia de su propia inserción.

b) **glue** (**adhesivo**): Una DSE glue representa conocimiento de un nombre solamente. Un DSA que contiene una DSE de prefijo de contexto o una DSE de referencia cruzada puede contener las DSE glue para representar los nombres de los superiores de la DSE de prefijo de contexto o de referencia cruzada si ninguna otra información operacional (por ejemplo, conocimiento) está asociada con esos nombres. Esto se ilustra en la figura 22. Una DSE de tipo glue no tendrá fijado ningún otro bit **DSEType**.

c) **cp**: DSE que representa el prefijo de contexto de un contexto de denominación.

d) **entry**: DSE que contiene una inserción de objeto.

e) **alias**: DSE que contiene una inserción de alias.

f) **subr**: DSE que contiene un atributo de conocimiento específico para representar una referencia subordinada.

g) **nssr**: DSE que contiene un atributo de conocimiento no específico para representar una referencia subordinada no específica.

h) **supr**: DSE que contiene un atributo de conocimiento específico para representar referencias superiores de DSA.

i) **xr**: DSE que contiene un atributo de conocimiento específico para representar una referencia cruzada.

j) **admPoint**: DSE que corresponde a un punto administrativo.

k) **subentry**: Una DSE que contiene una subinserción.

l) **shadow** (**sombra**): Una DSE que contiene una copia sombra de una inserción (o de parte de una inserción) u otra información (por ejemplo, conocimiento) recibida de un suministrador de sombra; este bit denominado es fijado por el consumidor de sombra.

m) **immSupr**: DSE que contiene un atributo de conocimiento específico para representar una referencia superior inmediata.

n) **rhob**: DSE que mantiene información de punto administrativo y de subinserción recibida de un DSA superior en una vinculación operacional jerárquica pertinente (es decir, en una vinculación operacional jerárquica o en una vinculación jerárquica no específica como se describe en las cláusulas 24 y 25 de la Rec. UIT-T X.518 | ISO/CEI 9594-4).

o) **sa**: Un calificador de una DSE **subr** que indica que la inserción de contexto de denominación subordinado es un alias.

p) **dsSubentry**: una DSE que mantiene una subinserción específica de un DSA.

q) **familyMember**: una DSE que mantiene un miembro de familia.

La utilización de este atributo operacional para representar aspectos del modelo de información de DSA se describe en la cláusula 23.

24 Representación de información de DSA

Esta cláusula trata la representación de la información de DSA. Describe la representación de información operacional (conocimiento) de DSA, información de usuario de directorio e información operacional de directorio.

24.1 Representación de información de usuario y operacional de directorio

Esta cláusula especifica la representación de información de usuario y operacional de directorio en el modelo de información de DSA.

24.1.1 Inserción de objeto

Una inserción de objeto se representa por una DSE de tipo **entry** que contiene los atributos operacionales de usuario y de directorio asociados con la inserción de directorio. El nombre de la DSE es el nombre de la inserción de objeto (es decir, el nombre distinguido del objeto).

Si la DSE contiene una copia de la inserción, el tipo de la DSE incluye **shadow**.

Si el nombre de la inserción objeto incluye nombres distinguidos alternativos diferenciados por el contexto, el nombre de la DSE puede también incluir esos nombres distinguidos alternativos diferenciados por el contexto. En el caso de una DSE que mantenga una sombra de la inserción, el nombre de la DSE puede incluir un subconjunto de los nombres distinguidos alternativos. En el caso de que la DSE no sea una copia, su nombre incluirá todos los nombres distinguidos.

NOTA – Para coherencia e interfuncionamiento con los DSA anteriores a 1997, el nombre de una DSE que mantiene una copia deberá incluir, al menos, el valor distinguido primario de cualquier atributo de denominación. Así la copia tiene al menos el nombre distinguido primario de la inserción objeto. Se acentúa la resolución de nombres si está presente cada valor distinguido (y así cada nombre distinguido alternativo).

24.1.2 Inserción de alias

Una inserción de alias se representa por una DSE de tipo **alias** que contiene los atributos asociados con la inserción de alias (es decir, los atributos RDN y el atributo nombre de objeto con alias). El nombre de la DSE es el nombre de la inserción de alias.

Si la DSE contiene una copia de la inserción de alias, el tipo de la DSE incluye **shadow**.

Si el nombre de la inserción alias incluye nombres distinguidos alternativos diferenciados por el contexto, el nombre de la DSE puede también incluir esos nombres distinguidos alternativos diferenciados por el contexto. En el caso de una DSE que mantenga una sombra de la inserción, el nombre de la DSE puede incluir un subconjunto de los nombres distinguidos alternativos. En el caso de que la DSE no sea una copia, su nombre incluirá todos los nombres distinguidos.

NOTA – Para coherencia e interfuncionamiento con los DSA anteriores a 1997, el nombre de una DSE que mantiene una copia deberá incluir, al menos, el valor distinguido primario de cualquier atributo de denominación. Así la copia tiene al menos el nombre distinguido primario de la inserción objeto. Se acentúa la resolución de nombres si está presente cada valor distinguido (y así cada nombre distinguido alternativo).

24.1.3 Punto administrativo

Un punto administrativo se representa por una DSE de tipo **admPoint** que contiene los atributos asociados con el punto administrativo. El nombre de la DSE es el nombre del punto administrativo.

Si la DSE representa una inserción, el tipo de DSE incluye **entry**. Si la DSE mantiene una copia de la información del punto administrativo, el tipo de DSE incluye **shadow**.

Si el nombre de la inserción alias incluye nombres distinguidos alternativos diferenciados por el contexto, el nombre de la DSE puede también incluir esos nombres distinguidos alternativos diferenciados por el contexto. En el caso de una DSE que mantenga una sombra de la inserción, el nombre de la DSE puede incluir un subconjunto de los nombres distinguidos alternativos. En el caso de que la DSE no sea una copia, su nombre incluirá todos los nombres distinguidos.

NOTA – Para coherencia e interfuncionamiento con los DSA anteriores a 1997, el nombre de una DSE que mantiene una copia deberá incluir, al menos, el valor distinguido primario de cualquier atributo de denominación. Así la copia tiene al menos el nombre distinguido primario de la inserción objeto. Se acentúa la resolución de nombres si está presente cada valor distinguido (y así cada nombre distinguido alternativo).

24.1.4 Subinserción

Una subinserción se representa por una DSE de tipo **subentry** que contiene la información operacional asociada con la subinserción. El nombre de la DSE es el nombre de la subinserción.

Si la DSE contiene una copia de la subinserción, el tipo de la DSE es **subentry** y **shadow**.

24.1.5 Miembro de familia

Un miembro de familia (incluido el antepasado) es representado por una DSE de tipo **familyMember**. El antepasado es también una **entry** de tipo DSE; es el único miembro de la familia que puede tener este tipo de DSE.

24.2 Representación de referencias de conocimiento

Una referencia de conocimiento consiste en una DSE de un tipo apropiado que contiene un atributo operacional de DSA correspondientemente apropiado y que es identificado por un nombre que se encuentra en una relación definida con el contexto de denominación contenido por el DSA referenciado.

El nombre de esta DSE será el nombre distinguido primario y puede incluir nombres alternativos e información de contexto si está presente en el prefijo de contexto del contexto de denominación mantenido por el DSA referenciado. Para una DSE que mantenga una sombra, su nombre puede incluir un subconjunto de los nombres alternativos. Si la DSE no es una copia, su nombre incluirá todos los nombres distinguidos.

NOTA – Se acentúa la resolución de nombres si está presente cada valor distinguido (y así cada nombre distinguido alternativo).

24.2.1 Tipos de atributos de conocimiento

Los atributos operacionales de DSA se definen en el modelo de información de DSA de modo que representen lo siguiente de un DSA:

- conocimiento de su propio punto de acceso;
- conocimiento superior;
- conocimiento específico (sus referencias subordinadas);
- conocimiento no específico (referencias subordinadas no específicas);
- conocimiento de su(s) suministrador(es), incluyendo facultativamente el maestro, si es un consumidor de sombra;
- conocimiento de su(s) consumidor(es), si es un suministrador de sombra; y
- conocimiento de sombras secundarias, si es un suministrador de sombra.

En el anexo F se asignan valores de identificador de objeto para estos atributos operacionales.

24.2.1.1 Mi Punto de Acceso

El atributo operacional **myAccessPoint** (**mi punto de acceso**) lo utiliza un DSA para representar su propio punto de acceso. Es un atributo específico de DSA. Todos los DSA contendrán este atributo en su DSE raíz. Este atributo es univaluado y lo gestiona el propio DSA.

```
myAccessPoint  ATTRIBUTE  ::= {
        WITH SYNTAX                 AccessPoint
        EQUALITY MATCHING RULE      accessPointMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-myAccessPoint }
```

El tipo ASN.1 **AccessPoint** se define en la Rec. UIT-T X.518 | ISO/CEI 9594-4. Su especificación ASN.1 se reproduce para comodidad del lector.

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

NOTA – El **Name** de **ae-title** puede ser el nombre distinguido primario o un nombre distinguido alternativo. Sin embargo, se mejora la coherencia y el interfuncionamiento con los DSA anteriores a 1997 si se utiliza el nombre distinguido primario.

En las Especificaciones de directorio no se describe cómo el DSA obtiene la información contenida en **myAccessPoint**.

El tipo de atributo **myAccessPoint** está contenido en la DSE de tipo **root**.

La información contenida en **myAccessPoint** puede emplearse en el DOP cuando se establece o modifica una vinculación operacional.

24.2.1.2 Conocimiento superior

El tipo de atributo operacional **superiorKnowledge** (**conocimiento superior**) es utilizado por un DSA que no es de primer nivel para representar su referencia superior. Es un atributo específico de DSA. Todos los DSA de nivel diferente del primero contendrán este atributo en su DSE raíz. Este atributo es univaluado y lo gestiona el propio DSA.

```
superiorKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-superiorKnowledge }
```

Un DSA puede adquirir la información contenida en **superiorKnowledge** por medios no descritos en las Especificaciones de directorio. Podría también construirla a partir de sus referencias superiores inmediatas, es decir a partir de su referencia superior inmediata cuyo prefijo de contexto tenga el menor número de RDN en su nombre.

El tipo de atributo **superiorKnowledge** está contenido en una DSE de tipo **root**.

La información contenida en **superiorKnowledge** puede ser empleada por un DSA cuando construye una referencia de continuación retornada en un referimiento de DAP o DSP, o cuando realiza el encadenamiento.

24.2.1.3 Conocimiento específico

El conocimiento específico consiste en los puntos de acceso para el DSA maestro de un contexto de denominación y/o los DSA sombras para ese contexto de denominación. Es específico porque el prefijo de contexto se conoce y se asocia con la información de punto de acceso. El conocimiento específico se representa por el tipo de atributo operacional **specificKnowledge** (**conocimiento específico**). Es un atributo compartido por DSA, univaluado, y gestionado por el propio DSA.

```
specificKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation
        ID                              id-doa-specificKnowledge }
```

El tipo ASN.1 **MasterAndShadowAccessPoints** se define en la Rec. UIT-T X.518 | ISO/CEI 9594-4. Su especificación ASN.1 se produce aquí para comodidad del lector.

```
MasterAndShadowAccessPoints  ::=  SET OF MasterOrShadowAccessPoint

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

Un DSA puede adquirir la información contenida en **specificKnowledge** por medios no descritos en las Especificaciones de directorio. En el caso de una referencia cruzada (DSE de tipo **xr**), podría también construirla a partir de información recibida en el componente **crossReference** de **ChainingResults** de una respuesta DSP. En el caso de un subordinado (DSE de tipo **subr**), podría construirla a partir de información recibida en el DOP cuando establece o modifica una HOB.

El tipo de atributo **specificKnowledge** está contenido en una DSE de tipo **subr**, **immSupr** o **xr**. Lo utiliza un DSA para representar referencias subordinadas, superiores inmediatas y cruzadas.

La información contenida en **specificKnowledge** puede ser empleada por un DSA cuando construye una referencia de continuación retornada en un referimiento de DAP o DSP (o cuando realiza el encadenamiento) y cuando construye SDSE de tipo **subr**, **immSupr** o **xr** proporcionados en el DISP.

24.2.1.4 Conocimiento no específico

El conocimiento no específico consiste en los puntos de acceso para el DSA maestro de uno o más contextos de denominación y/o DSA sombras para el mismo o los mismos contextos de denominación. Es no específico porque el (los) prefijo(s) de contexto del (de los) contexto(s) de denominación no es (son) conocido(s). En cambio, el superior inmediato del (de los) contexto(s) de denominación es conocido, y la información de punto de acceso está asociada con su nombre. Conocimiento no específico se representa por el tipo de atributo operacional **nonSpecificKnowledge** (**conocimiento no específico**). Es un atributo compartido por DSA, multivaluado y gestionado por el propio DSA.

```
nonSpecificKnowledge ATTRIBUTE ::= {
    WITH SYNTAX                 MasterAndShadowAccessPoints
    EQUALITY MATCHING RULE      masterAndShadowAccessPointsMatch
    NO USER MODIFICATION        TRUE
    USAGE                       distributedOperation
    ID                          id-doa-nonSpecificKnowledge }
```

El valor **MasterAndShadowAccessPoints** consiste en un punto de acceso para un DSA maestro que mantiene uno o más contextos subordinados, y ninguno o más puntos de acceso de los DSA que contienen sombras de algunos o todos estos contextos de denominación.

Un DSA puede adquirir la información contenida en **nonSpecificKnowledge** por medios no descritos en las Especificaciones de directorio. En el caso de una referencia subordinada no específica (DSE de tipo **nssr**), podría también construirla a partir de información recibida en el DOP cuando establece o modifica una NHOB.

El tipo de atributo **nonSpecificKnowledge** está contenido en un DSE de tipo **nssr**. Es utilizado por un DSA para representar referencias subordinadas no específicas.

La información contenida en **nonSpecificKnowledge** puede ser empleada por un DSA cuando construye una referencia de continuación retornada en un referimiento de DAP o DSP (o cuando realiza el encadenamiento) y cuando construye SDSE de tipo **nssr** proporcionados en el DISP.

24.2.1.5 Conocimiento de suministrador

El suministrador de conocimiento de un DSA consumidor de sombra consiste en el (los) punto(s) de acceso e identificador(es) de acuerdo de sombreado para su(s) suministrador(es) de copia(s) de una zona replicada. Facultativamente, si el suministrador no es el maestro del contexto de denominación del que se derivó una zona replicada, el punto de acceso del maestro se puede incluir en conocimiento de suministrador. Conocimiento de suministrador se representa por el tipo de atributo operacional **supplierKnowledge** (**conocimiento de suministrador**). Es específico de DSA, multivaluado y gestionado por el propio DSA.

La sintaxis ASN.1 para un valor de **supplierKnowledge** es **SupplierInformation**. Un valor de este atributo está compuesto de un punto de acceso de un DSA suministrador de sombra y la ID del acuerdo de sombreado entre el DSA suministrador y el DSA consumidor que contiene el atributo específico de DSA (expresado como un valor del tipo **SupplierOrConsumer**), una indicación de si el suministrador de la zona replicada es o no el maestro del contexto de denominación del que se derivó, y, si no, facultativamente, el punto de acceso del DSA maestro.

```
SupplierOrConsumer ::= SET {
    COMPONENTS OF           AccessPoint,        -- supplier or consumer --
    agreementID       [3]   OperationalBindingID }

SupplierInformation ::= SET {
    COMPONENTS OF           SupplierOrConsumer, -- supplier --
    supplier-is-master  [4] BOOLEAN DEFAULT TRUE,
    non-supplying-master [5] AccessPoint OPTIONAL }

supplierKnowledge ATTRIBUTE ::= {
    WITH SYNTAX                 SupplierInformation
    EQUALITY MATCHING RULE      supplierOrConsumerInformationMatch
    NO USER MODIFICATION        TRUE
    USAGE                       dSAOperation
    ID                          id-doa-supplierKnowledge }
```

Un DSA puede adquirir la información contenida en **supplierKnowledge** por medios no descritos en las Especificaciones de directorio. Un DSA consumidor de sombra pudiera también construirla a partir de información recibida en el DOP cuando se establece o modifica un acuerdo de sombreado.

El tipo de atributo **supplierKnowledge** está contenido en una DSE de tipo **cp**. Se utiliza para representar una o más referencias de suministrador. Todos los DSA consumidores de sombra contendrán un valor de este atributo para cada acuerdo de sombreado que concierten como consumidor.

La información contenida en **supplierKnowledge** puede ser utilizada por un DSA cuando construye una referencia de continuación retornada en un referimiento de DAP o DSP. El componente **agreementID** (este tipo, **OperationalBindingID**, está definido en 28.2), de **supplierKnowledge** se requiere en las operaciones del DOP para gestionar un acuerdo de sombreado en todas las operaciones del DISP.

24.2.1.6 Conocimiento de consumidor

El conocimiento de consumidor de un DSA suministrador de sombra consiste en el (los) punto(s) de acceso e identificador(es) de acuerdo de sombreado para el (los) consumidor(es) de una o más copias de un contexto de denominación que les ha proporcionado el suministrador. El conocimiento de consumidor se representa por el tipo de atributo operacional **consumerKnowledge** (**conocimiento de consumidor**). Es específico de DSA, multivaluado y gestionado por el propio DSA.

La sintaxis ASN.1 para un valor de **consumerKnowledge** es **ConsumerInformation** (que tiene la misma sintaxis que **SupplierOrConsumer**, pero hace referencia a un punto de acceso de consumidor).

```
ConsumerInformation ::= SupplierOrConsumer      -- consumer --

consumerKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     ConsumerInformation
        EQUALITY MATCHING RULE          supplierOrConsumerInformationMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-consumerKnowledge }
```

Un DSA puede adquirir la información contenida en **consumerKnowledge** por medios no descritos en las Especificaciones de directorio. Un DSA suministrador de sombra pudiera también construirla a partir de información recibida en el DOP cuando se establecen o modifican acuerdos de sombreado.

El tipo de atributo **consumerKnowledge** está contenido en una DSE de tipo **cp**. Se utiliza para representar una o más referencias de consumidor. Todos los DSA suministradores de sombra contendrán un valor de este atributo para cada acuerdo de sombreado que concierten como suministrador.

El componente **agreementID** de **consumerKnowledge** se requiere en las operaciones del DOP para gestionar un acuerdo de sombreado en todas las operaciones del DISP.

24.2.1.7 Conocimiento de sombra secundaria

El conocimiento de sombra secundaria consiste en la información que un DSA suministrador (por ejemplo un DSA maestro) puede decidir mantener con relación a los DSA consumidores que intervienen en sombreado secundario desde su perspectiva. El conocimiento de sombra secundario se representa por el tipo de atributo operacional **secondaryShadows** (**sombras secundarias**). Es específico de DSA, multivaluado y gestionado por el propio DSA. La sintaxis ASN.1 para un valor de **secondaryShadows** es **SupplierAndConsumers**. Consiste en el punto de acceso de un suministrador de sombra y una lista de sus consumidores directos.

```
SupplierAndConsumers ::= SET {
        COMPONENTS OF           AccessPoint,        -- supplier --
        consumers       [3]     SET OF AccessPoint }

secondaryShadows ATTRIBUTE ::= {
        WITH SYNTAX                     SupplierAndConsumers
        EQUALITY MATCHING RULE          supplierAndConsumersMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-secondaryShadows }
```

El componente **consumers** de **SuppliersAndConsumers** sólo contiene puntos de acceso de los DSA que mantienen copias utilizables en común de una zona replicada.

Un DSA de suministrador puede obtener la información requerida para construir los valores de este atributo de un DSA consumidor aplicando el procedimiento descrito en 23.1.1 de la Rec. UIT-T X.518 | ISO/CEI 9594-4.

El tipo de atributo **secondaryShadows** está contenido en una DSE de tipo **cp**.

El soporte de conocimiento de sombra secundaria es facultativo.

24.2.1.8 Reglas de concordancia

A continuación se especifican cuatro reglas de concordancia de igualdad para los atributos de conocimiento antes mencionados. Estas reglas son aplicables a atributos con sintaxis de los tipos **AccessPoint**, **MasterAndShadowAccessPoints**, **SupplierInformation**, **ConsumerInformation** y **SuppliersAndConsumers**.

24.2.1.8.1 Concordancia de punto de acceso

La regla de concordancia de punto de acceso se especifica como sigue:

```
accessPointMatch  MATCHING-RULE  ::=  {
        SYNTAX      Name
        ID          id-kmr-accessPointMatch }
```

La regla de concordancia **accessPointMatch** se aplica a valores de atributo del tipo **AccessPoint**. Un valor de la sintaxis de aserción se deriva de un valor de la sintaxis de atributo utilizando el valor del componente (**Name**) del rótulo específico del contexto **[0]**. Se consideran dos valores para la concordancia de igualdad si el componente **Name** de cada uno concuerda utilizando el procedimiento de concordancia para los valores **DistinguishedName**.

24.2.1.8.2 Concordancia de puntos de acceso maestro y sombra

La regla de concordancia de igualdad de concordancia de los puntos de acceso maestro y sombra se especifica como:

```
masterAndShadowAccessPointsMatch  MATCHING-RULE  ::=  {
        SYNTAX      SET OF Name
        ID          id-kmr-masterShadowMatch }
```

La regla de concordancia **masterAndShadowAccessPointsMatch** se aplica a atributos del tipo **MasterAndShadowAccessPoints**. Se deriva un valor de la sintaxis de aserción de un valor de la sintaxis de atributo suprimiendo los componentes **category** y **address** de cada **SET** en el **SET OF MasterOrShadowAccessPoints**. Se consideran dos valores para concordar la igualdad si ambos valores tienen el mismo número de elementos **SET OF**, y, después de ordenar los elementos **SET OF** de cada uno en cualquier manera conveniente, el componente **ae-title** de cada par de elementos **SET OF** concuerda utilizando el procedimiento de concordancia para **distinguishedNameMatch**.

24.2.1.8.3 Concordancia de información de suministrador o consumidor

La regla de concordancia de información de suministrador o consumidor se especifica como:

```
supplierOrConsumerInformationMatch  MATCHING-RULE  ::=  {
        SYNTAX      SET {
                ae-title                [0]     Name,
                agreement-identifier    [2]     INTEGER }
        ID          id-kmr-supplierConsumerMatch }
```

La regla de concordancia **supplierOrConsumerInformationMatch** se aplica a valores de atributo del tipo **SupplierInformation** o **ConsumerInformation** (y a otros atributos que tienen valores compatibles con **SupplierInformation** o **ConsumerInformation**). Se deriva un valor de la sintaxis de aserción de un valor de la sintaxis de atributo seleccionando los componentes **SET** con rótulos que concuerdan con los componentes **SET** de la sintaxis aserción. Se considera que estos dos valores concuerdan en igualdad si el componente **ae-title** de cada uno (después de suprimir la información de rótulo **[0]** explícita) concuerda utilizando el procedimiento de concordancia para valores **DistinguishedName**, y el componente **identifier** contenido en el componente **agreement** para cada uno (después de suprimir la información de rótulo explícita **[2]** y **SEQUENCE** concuerda utilizando el procedimiento de concordancia para valores **INTEGER**.

24.2.1.8.4 Concordancia de suministradores y consumidores

La regla de concordancia de suministradores y consumidores se especifica como sigue:

```
supplierAndConsumersMatch  MATCHING-RULE  ::=  {
        SYNTAX      Name
        ID          id-kmr-supplierConsumersMatch }
```

La regla de concordancia de suministradores y consumidores se aplica a valores de atributo del tipo **SupplierAndConsumers** (teniendo los otros atributos valores compatibles con **SupplierAndConsumers**). Se considera que dos de estos valores concuerdan por igualdad si el componente **ae-title** de cada uno de ellos (después de retirar la información de rótulo **[0]** explícita) concuerda cuando se aplica el procedimiento de concordancia para valores **DistinguishedName**.

24.2.2 Tipos de referencia de conocimiento

Esta subcláusula especifica la representación de conocimiento en el modelo de información de DSA.

24.2.2.1 Referencia de sí mismo

Una referencia de sí mismo representa un conocimiento de DSA de su propio punto de acceso. Consiste en un valor del atributo **myAccessPoint** contenido en la DSE raíz del DSA, siendo la DSE de tipo **root**.

24.2.2.2 Referencia superior

Una referencia superior consiste en una DSE de tipo **supr** y **root** que contiene un atributo **superiorKnowledge**. Como un valor de atributo **superiorKnowledge** puede contener puntos de acceso de varios DSA, puede representar, por tanto, varias referencias superiores.

24.2.2.3 Referencia superior inmediata

Una referencia superior inmediata consiste en una DSE de tipo **immSupr** que contiene un atributo **specificKnowledge**. El nombre de la DSE que contiene el atributo corresponde al prefijo de contexto del contexto de denominación contenido por el DSA superior referenciado.

Como un valor de atributo **specificKnowledge** puede contener puntos de acceso de varios DSA, puede representar por tanto varias referencias superiores inmediatas, como máximo una de categoría **master** y ninguna o más de categoría **shadow**.

Si el DSE que contiene la referencia superior inmediata es, recibida de un suministrador de sombra; el tipo de DSE incluye **shadow**.

24.2.2.4 Referencia subordinada

Una referencia subordinada consiste en una DSE de tipo **subr** que contiene un atributo **specificKnowledge**. El nombre de la DSE que contiene el atributo corresponde al prefijo de contexto del contexto de denominación contenido por el DSA subordinado referenciado.

Como un valor de atributo **specificKnowledge** puede contener puntos de acceso de varios DSA, puede representar por tanto varias referencias superiores inmediatas, como máximo una de categoría **master** y ninguna o más de categoría **shadow**.

Si el DSE que contiene la referencia subordinada es información sombreada, recibida de un suministrador de sombra, el tipo del DSE incluye **shadow**.

La DSE puede incluir también **immSupr** en un DSA que posee dos contextos de denominación, uno superior al otro, que están separados por un tercer contexto de denominación constituido por una sola inserción y contenido en otro DSA. En el anexo O se presenta un ejemplo de esta situación.

24.2.2.5 Referencia subordinada no específica

Una referencia subordinada no específica consiste en una DSE de tipo **nssr** (y **entry**, normalmente) que contiene un atributo **nonSpecificKnowledge**. El nombre de la DSE que contiene el atributo corresponde al nombre formado al eliminar el RDN final del prefijo de contexto del contexto de denominación contenido por el DSA subordinado referenciado.

Como un valor de atributo **nonSpecificKnowledge** puede contener puntos de acceso de varios DSA, puede representar por tanto varias referencias subordinadas no específicas, como máximo una de categoría **master** y ninguna o más de categoría **shadow**. Cada valor de atributo **nonSpecificKnowledge** representa un conjunto conexo de referencias subordinadas no específicas – los DSA de categoría **shadow** mantienen una o más zonas replicadas derivadas del contexto o contextos de denominación mantenido por el DSA de categoría **master**.

Si el DSE que contiene la referencia subordinada no específica es información sombreada, recibida de un suministrador de sombra, el tipo de la DSE comprende **shadow**.

La DSE incluye **shadow** en el caso de un DSA sombra cuando el DSE corresponde a una la inserción para la cual el DSA maestro tiene conocimiento subordinado no específico y para la cual sólo el atributo **nonSpecificKnowledge** para la referencia subordinada no específica está sombreado.

La DSE incluye **cp** y **shadow** en el caso de un DSA cuya zona replicada no incluye la inserción de prefijo de contexto y el DSA maestro para el contexto de denominación tiene conocimiento subordinado no específico para el prefijo de contexto.

La DSE incluye **admPoint** y **shadow** en el caso de un DSA sombra cuando la DSE corresponde a un punto administrativo, la información de inserción para el punto administrativo no está sombreada, y el DSA maestro para el contexto de denominación tiene conocimiento subordinado no específico para el punto administrativo.

Cuando el punto administrativo coincide con un prefijo de contexto en los dos casos antes mencionados, la DSE puede incluir **admPoint**, **cp** y **shadow**.

24.2.2.6 Referencia cruzada

Una referencia cruzada consiste en una DSE de tipo **xr** que contiene un atributo **specificKnowledge**. El nombre de la DSE que contiene el atributo corresponde al prefijo de contexto del contexto de denominación contenido por el DSA referenciado.

Como un valor de atributo **specificKnowlegde** puede contener puntos de acceso de varios DSA, puede representar varias referencias cruzadas, como máximo una de categoría **master** y ninguna o más de categoría **shadow**.

24.2.2.7 Referencia de suministrador

Una referencia de suministrador consiste en una DSE de tipo **cp** que contiene un atributo **supplierKnowledge**. El nombre de la DSE que contiene el atributo corresponde al prefijo de contexto del contexto de denominación sombreado.

Como un atributo **supplierKnowledge** puede tener varios valores, puede representar varias referencias de suministrador. Cada valor de atributo representa una referencia de suministrador.

24.2.2.8 Referencia de consumidor

Una referencia de consumidor consiste en una DSE de tipo **cp** que contiene un atributo **consumerKnowledge**. El nombre de la DSE que contiene el atributo corresponde al prefijo de contexto del contexto de denominación sombreado.

Como un atributo **consumerKnowledge** puede tener varios valores, puede representar varias referencias de consumidor. Cada valor de atributo representa una referencia de consumidor.

24.3 Representación de nombres y de contextos de denominación

24.3.1 Nombres y DSE glue

Como se ha descrito en 23.3, la información mínima que un DSA puede asociar con un nombre es el propósito para el cual contiene el nombre, representado por una DSE que contiene un valor del atributo **dseType**. Cuando una DSE sólo contiene esta información mínima, el valor del atributo **dseType** será **glue**. En este caso DSE no contendrá una inserción o subinserción (ni una copia-sombra de una inserción o subinserción), ni un atributo compartido por DSA.

Las DSE adhesivas (DSE glue) aparecen en el modelo de información de DSA para representar nombres que son conocidos por un DSA porque contiene información asociada con otros nombres. Por ejemplo, considérese la referencia cruzada representada en la figura 22. El DSA que contiene esta referencia cruzada también "conoce" (en el sentido descrito en 23.3) los nombres que son superiores al nombre de prefijo de contexto asociado con la referencia cruzada. Cuando ninguna otra información está asociada con tales nombres de superior, éstos se representan en el modelo de información de DSA como DSE adhesivas.

24.3.2 Contextos de denominación

Un contexto de denominación consiste en un prefijo de contexto, un subárbol de ninguna o más inserciones subordinadas al prefijo de contexto (la raíz del subárbol) y, si hay contextos de denominación subordinados a él, referencias subordinadas y/o referencias subordinadas no específicas suficientes para constituir un conocimiento subordinado completo.

Un prefijo de contexto está representado por una DSE de tipo **cp**. Si el prefijo de contexto corresponde a una inserción, el tipo de DSE incluye **entry**. Si corresponde a un alias, el tipo de DSE incluye **alias**. Si el prefijo de contexto corresponde a un punto administrativo, el tipo de DSE incluye **admPoint**.

El subárbol de inserciones y subinserciones subordinadas al prefijo de contexto es representado por las DSE como se indica en 24.1.1 a 24.1.5.

La representación del conocimiento subordinado del contexto de denominación es representado por las DSE como se indica en 24.2.2.

Una zona replicada (una copia sombra de la totalidad o una parte del contexto de denominación) se representa en la forma antes mencionada con la salvedad de que el tipo de DSE incluye **shadow** en cada DSE para la cual se han recibido, del suministrador de sombra, atributos de usuario u operacionales. En el caso de zonas replicadas incompletas, pueden aparecer DSE de tipo **glue** para representar un puente entre las distintas piezas de la información sombreada. Estas DSE adhesivas no tienen asociado ningún atributo de usuario u operacional.

24.3.3 Ejemplo

La figura 22 presenta un ejemplo de la correspondencia de una porción del DIT (que corresponde a un contexto de denominación) con el árbol de información de un DSA. Además de la información de contexto de denominación propiamente dicha, se representa a la DSE raíz del DSA que contiene su referencia superior (ésta no es el árbol de información para un DSA de primer nivel), una DSE adhesiva y una DSE que representa una referencia (sea una referencia cruzada, sea una referencia superior inmediata) a un contexto de denominación inmediatamente superior.

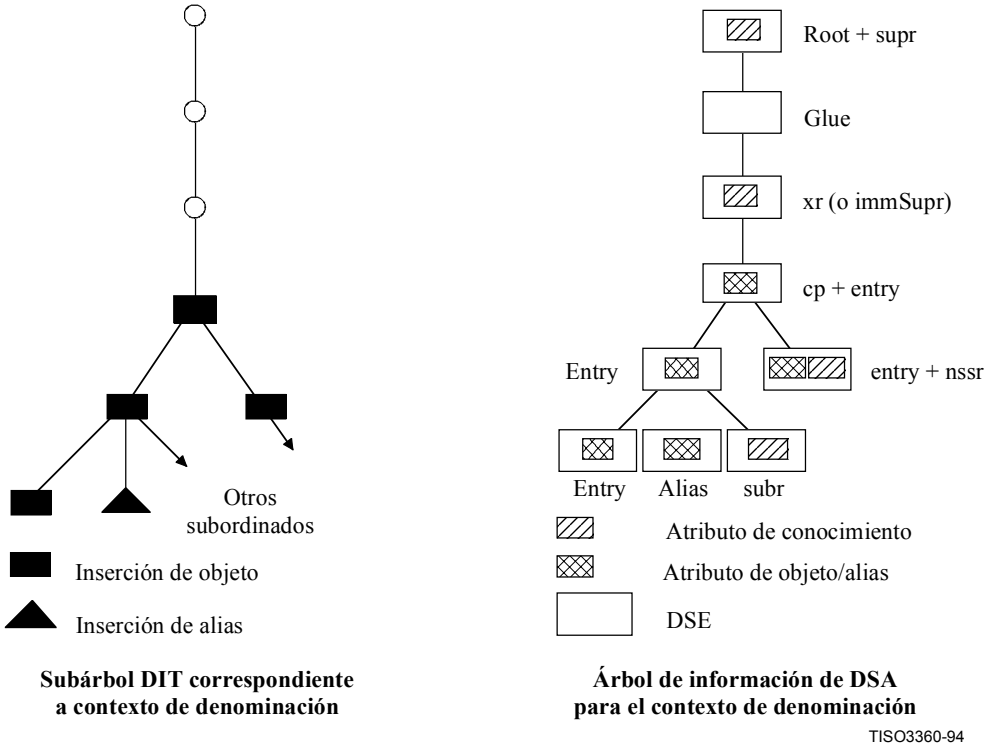


Figura 22 – DSE para un contexto de denominación

SECCIÓN 11 – MARCO OPERACIONAL DE DSA

25 Visión de conjunto

25.1 Definiciones

A los efectos de esta Especificación de directorio se aplican las siguientes definiciones.

25.1.1 estado cooperativo: Con respecto a un segundo DSA, el estado de un DSA para el cual un caso de vinculación operacional ha sido establecida y no ha sido terminada.

25.1.2 marco operacional de directorio: Marco operacional que, al ser aplicado, permite derivar modelos operacionales específicos referentes a aspectos particulares (por ejemplo, sombreado o creación de un contexto de aplicación) de la operación de los componentes de directorio (DSA). El marco operacional de directorio destaca los elementos comunes que están presentes en todas las interacciones entre componentes de directorio.

25.1.3 estado no cooperativo: Con respecto a un segundo DSA, el estado de un DSA antes del establecimiento o después de la terminación de un caso de vinculación operacional.

25.1.4 vinculación operacional: Un entendimiento mutuo entre dos DSA, que una vez establecido, expresa el "acuerdo" de estos dos DSA para participar subsiguientemente en algún género de interacción.

25.1.5 establecimiento de vinculación operacional: El proceso de establecer un caso de vinculación operacional.

25.1.6 ejemplar de vinculación operacional: Vinculación operacional de un tipo específico entre dos DSA.

25.1.7 gestión de vinculaciones operacionales: El proceso de establecer, terminar o modificar un caso de vinculación operacional. Esta gestión puede realizarse mediante intercambios de información definidos por Especificaciones de directorio, intercambios definidos en otras Especificaciones, o por otros medios.

25.1.8 modificación de vinculación operacional: El proceso de modificar un caso de vinculación operacional.

25.1.9 terminación de vinculación operacional: El proceso de terminar un caso de vinculación operacional.

25.1.10 tipo de vinculación operacional: Un tipo particular de vinculación operacional especificado para algún propósito concreto, que expresa el "acuerdo" de dos DSA para participar en tipos específicos de interacción (por ejemplo, sombreado).

25.2 Introducción

Las Especificaciones de directorio definen intercambios de información de protocolo de aplicación y procedimientos de DSA asociados que definen la operación distribuida de directorio. Las cláusulas 25 a 28 definen un marco operacional de DSA que modela ciertos elementos comunes en estos intercambios de información y procedimientos.

Dos DSA interactúan de una manera cooperativa porque, además de su capacidad técnica para intercambiar información y seguir procedimientos asociados con esos intercambios, cada uno de ellos ha sido configurado para aceptar ciertas interacciones con el otro.

En estas cláusulas se trata de la expresión de un marco común para la especificación de la estructura de los elementos de la cooperación entre dos DSA.

Un objetivo de este marco es ser lo suficientemente general para tener en cuenta que todas las formas de cooperación de DSA se definan en esta y en futuras versiones de las Especificaciones de directorio. El marco se utiliza en Especificaciones de directorio para definir tipos de sombreado y de vinculación operacional jerárquica.

26 Vinculaciones operacionales

26.1 Generalidades

En esta cláusula se trata de la definición de un marco general, el marco operacional de DSA, dentro del cual la especificación de la naturaleza de las interacciones cooperativas de componentes de directorio (DSA) puede ser estructurada para alcanzar un objetivo convenido en común.

El marco general destaca las prestaciones comunes que caracterizan todas las interacciones entre los DSA. Aplicando el marco operacional de DSA a aspectos específicos de interacción cooperativa entre los DSA, se obtendrán especificaciones que serán a la vez concisas y coherentes, con lo que se reducirá el número total de mecanismos que un DSA deba soportar.

El entendimiento mutuo entre dos DSA que, una vez establecido, expresa el "acuerdo", entre ambos, de participar en algún género de interacción se llama una *vinculación operacional*. Dos DSA pueden compartir tantas vinculaciones operacionales de un tipo específico cuantas se requieran.

El marco operacional de DSA proporciona un enfoque común de la definición de un *tipo de vinculación operacional*. Un tipo de vinculación operacional es un tipo particular de vinculación operacional especificado para cierto propósito individual que expresa el "acuerdo" de dos DSA de participar en tipos específicos de interacción (por ejemplo, sombreado). Esta interacción autoriza a que cualquiera de las dos partes que intervienen en el acuerdo invoquen operaciones entre las que forman un conjunto bien definido.

Dos DSA particulares que han concertado ese "acuerdo" comparten un caso de vinculación operacional de un tipo (de vinculación operacional) específico. De ellos se dice que están en el *estado cooperativo* de ese caso de vinculación operacional.

Antes del establecimiento o después de la terminación de un caso de vinculación operacional entre dos DSA se dice, que están en el *estado no cooperativo*.

La *gestión de vinculación operacional* es el proceso de establecer, modificar o terminar un caso de vinculación operacional. Esta gestión puede realizarse mediante intercambios de información definidos por Especificaciones de directorio, mediante intercambios definidos en otras Especificaciones, o por otros medios.

Estos conceptos generales se representan en la figura 23.

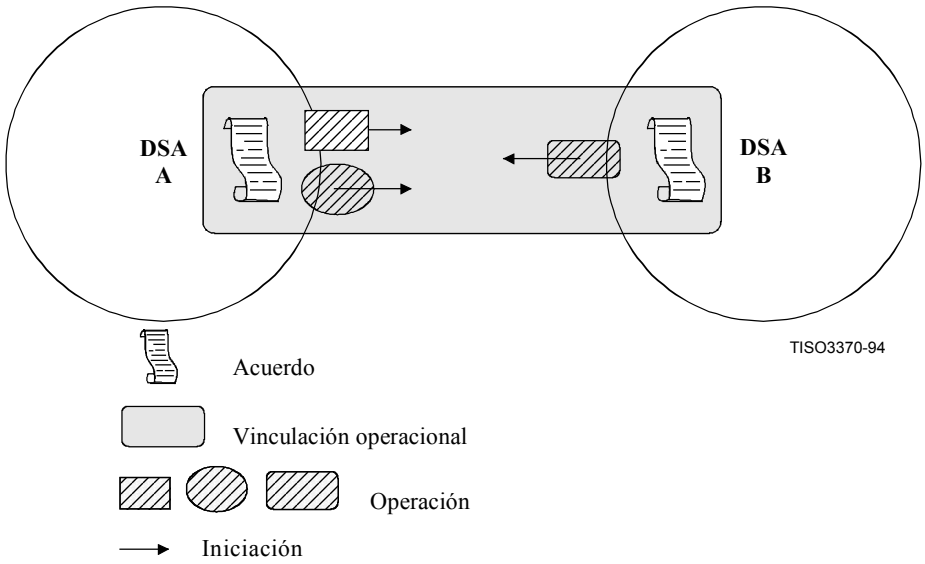


Figura 23 – Vinculación operacional

26.2 Aplicación del marco operacional

La aplicación del marco operacional DSA para definir un tipo de vinculación operacional concierne a los siguientes elementos básicos:

a) dos DSA;

b) un "acuerdo" sobre el servicio básico que un DSA proporcionará a otro DSA;

c) un conjunto de una o más operaciones, junto con los procedimientos que habrá de seguir un DSA, a través de los cuales el servicio puede ser realizado;

d) una especificación de las interacciones de DSA necesarias para la gestión del acuerdo.

La relación de estos elementos básicos se expresa por una vinculación operacional. Una vinculación operacional se compone del conjunto de estos elementos básicos que intervienen para representar el acuerdo abstracto en términos técnicos. Representa el entorno gobernado por un "acuerdo", según el cual un DSA proporciona a otro (y viceversa) un servicio definido.

26.2.1 Dos DSA

El marco operacional DSA proporciona una estructura dentro de la cual se puede especificar la interacción de un DSA con el otro y los procedimientos que ambos ejecutan consecuentemente.

Los dos DSA pueden desempeñar un cometido idéntico en la vinculación operacional, en cuyo caso ambos DSA pueden gestionar la vinculación operacional, ambos DSA pueden invocar las mismas operaciones con relación al otro, y ambos DSA están obligados a seguir el mismo conjunto de procedimientos. Se dice que ésta es una vinculación operacional simétrica.

Otra posibilidad es que cada uno de los DSA desempeñe un cometido diferente en la vinculación operacional, en cuyo caso a cada DSA es aplicable un conjunto diferente de operaciones y procedimientos. Uno de los dos, o ambos DSA, pueden intervenir en la gestión de una vinculación operacional. Se dice que ésta es una vinculación operacional asimétrica.

26.2.2 El acuerdo

Un "acuerdo" es un entendimiento mutuo a que llegan las autoridades administrativas de dos DSA sobre un servicio que deberá proporcionar uno de los DSA al otro (y/o viceversa). El "acuerdo" es negociado inicialmente por las autoridades administrativas de los DSA por medios que se sitúan fuera del alcance de las Especificaciones de directorio.

Los parámetros de este "acuerdo" pueden ser formalizados por un tipo de datos ASN.1 para uso en un intercambio de protocolo en la gestión de la vinculación operacional. De esta manera, ambos DSA llegan a un entendimiento mutuo sobre el servicio que cada uno ha de proporcionar al otro.

26.2.3 Operaciones

Las operaciones constituyen el medio básico de que disponen los DSA para interactuar. Un par de DSA se intercambiarán una o más operaciones para proporcionar el servicio convenido.

Si bien un DSA puede ser técnicamente capaz de soportar un gran número de operaciones, puede que sólo desee o esté dispuesto a cooperar con otro DSA en un pequeño número de esas operaciones, o en el procesamiento de operaciones que sólo tienen fijados valores particulares para ciertos parámetros.

La definición de un tipo de vinculación operacional requiere la enumeración de las operaciones que pueden ser intercambiadas. También autoriza a imponer restricciones a los valores de parámetros definidos dentro de las operaciones.

26.2.4 Gestión del acuerdo

El marco proporciona operaciones genéricas para gestionar un caso de vinculación operacional. Estas operaciones permiten el establecimiento, modificación y terminación de una vinculación operacional.

Para aplicar el marco a la especificación de un tipo particular de vinculación operacional es necesario que esté especificado el iniciador de cada una de las tres operaciones de gestión, y que estén definidos los procedimientos para el establecimiento, modificación y terminación. Cada vez que se aplique una operación de gestión a una vinculación operacional del tipo especificado, el DSA seguirá el procedimiento correspondiente.

26.3 Estado de cooperación

El modelo operacional genérico define dos estados de cooperación, regidos por un caso de un tipo de vinculación operacional particular, entre dos DSA, visto por un DSA con respecto al otro DSA y tres transiciones entre estos estados. Cada caso identificado de un caso de vinculación operacional compartida por dos DSA tiene sus propios estados de cooperación. Los estados de cooperación son:

a) *Estado no cooperativo*: No se ha establecido o se ha terminado un caso identificado particular de un caso de vinculación operacional entre los dos DSA. La interacción entre los dos DSA (con respecto al caso identificado de un caso de vinculación operacional) no se define. Un DSA que se ha puesto en contacto con otro con el cual está en un estado no cooperativo puede rehusar, por ejemplo, intervenir en una interacción, o puede estar preparado para atender la petición.

b) *Estado cooperativo*: Hay un caso de vinculación operacional del tipo en cuestión entre dos DSA. Su comportamiento cooperativo está gobernado por la definición del tipo de vinculación operacional y sus parámetros específicos y procedimientos asociados.

Las transiciones entre estos dos estados de cooperación pueden ser provocadas de dos modos: por interacciones de protocolo normalizadas, o por otros medios.

Las interacciones entre dos DSA para gestionar un caso de vinculación operacional (por ejemplo, para establecer y terminar un acuerdo de sombreado) son distintas de sus interacciones potenciales gobernadas por la vinculación (por ejemplo, la interacción para actualizar una unidad de replicación).

Las transiciones de estado son las siguientes:

a) La transición de *establecimiento* crea un caso de vinculación operacional de un tipo particular entre dos DSA, y como resultando de esto hay un paso del estado no cooperativo al estado cooperativo.

b) La transición de *terminación* destruye un caso de vinculación operacional de un tipo particular entre dos DSA, y como resultado de esto hay un paso del estado cooperativo al estado no cooperativo.

c) La transición de *modificación* modifica los parámetros de un caso de vinculación operacional entre dos DSA, y como resultado de esto hay un movimiento del estado cooperativo al estado no cooperativo.

Estos estados genéricos y transiciones se ilustran en la figura 24.

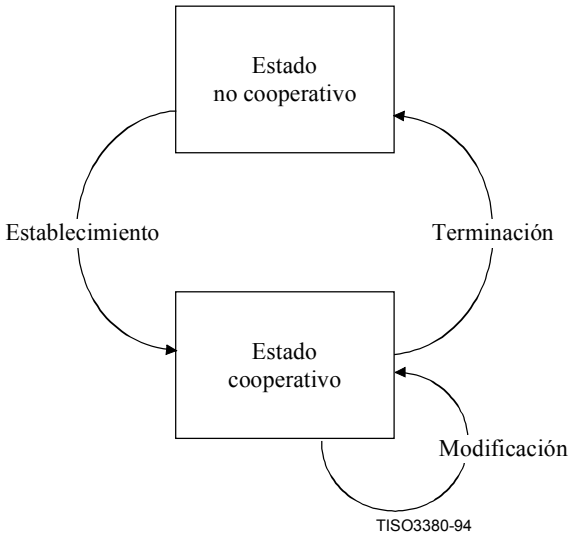


Figura 24 – Estados de cooperación

27 Especificación y gestión de vinculaciones operacionales

27.1 Especificación de tipo de vinculación operacional

Cuando se aplica el marco para definir la especificación de tipo de la vinculación operacional se especificarán las siguientes características del tipo:

a) *Simetría*

Una especificación de los cometidos respectivos de los DSA que intervienen en la vinculación operacional.

Las vinculaciones operacionales pueden ser simétricas, en cuyo caso el cometido de un DSA es intercambiable con el del otro y ambos DSA exhiben las mismas interacciones externas. Las vinculaciones operacionales pueden también ser asimétricas, en cuyo caso cada DSA desempeña un cometido distinto y ambos DSA exhiben interacciones externas diferentes. En este último caso, el marco operacional de directorio distingue los dos cometidos abstractos como "COMETIDO-A" y "COMETIDO-B".

Cada uno de los cometidos abstractos "COMETIDO-A" y "COMETIDO-B" tienen que ser asociados con un cometido concreto con semántica definida (por ejemplo, "COMETIDO-A" como suministrador de sombra, "COMETIDO-B" como consumidor de sombra).

b) *Acuerdo*

Una definición de la semántica y la representación de los componentes de "acuerdo". Esta información parametriza el caso específico de una vinculación operacional entre dos DSA.

c) *Iniciador*

Una definición de cuál de los dos cometidos abstractos "COMETIDO-A" y "COMETIDO-B" está autorizado a iniciar el establecimiento, modificación o terminación de una vinculación operacional de este tipo.

d) *Procedimientos de gestión*

Un conjunto de procedimientos que deberá seguir un DSA cuando la vinculación operacional de este tipo es establecida, modificada o terminada.

e) *Identificación de tipo*

Identifica el tipo de interacción de DSA que es determinado por la vinculación operacional. Estos identificadores son valores de identificador de objeto.

f) *Contextos de aplicación, operaciones y procedimientos*

Identifican el conjunto de contextos de aplicación cuyas operaciones o (un subconjunto de ellas) pueden ser empleadas durante la fase cooperativa de la vinculación operacional.

Para cada operación referenciada por el tipo de vinculación operacional se requiere una descripción de los procedimientos que ha de seguir un DSA si la operación es invocada en la forma requerida (esto puede hacerse por referencia a otra parte de estas Especificaciones de directorio).

Para las vinculaciones operacionales que no son gestionadas utilizando las operaciones genéricas de gestión de vinculaciones operacionales proporcionadas en esta cláusula, el tipo de vinculación deberá especificarse utilizando las tres clases de objeto de información **OPERATIONAL-BINDING**, **OP-BINDING-COOP** y **OP-BIND-ROLE** definidas en esta cláusula.

27.2 Gestión de vinculación operacional

En general, la gestión de una vinculación operacional requiere inicialmente el establecimiento de un ejemplar de vinculación operacional. Ésta puede facultativamente ir seguida de una o más modificaciones de algunos o todos los parámetros del acuerdo inicial, y finalmente puede entrañar la terminación del ejemplar de vinculación operacional. Los detalles precisos de cómo puede ser gestionado un ejemplar, se definen durante la definición del tipo de vinculación operacional. Esta definición de tipo requiere la especificación de:

a) el iniciador de cada una de las operaciones de gestión (éste puede ser uno, ambos, o ninguno de los dos DSA);

b) los parámetros para cada una de las operaciones de gestión; y

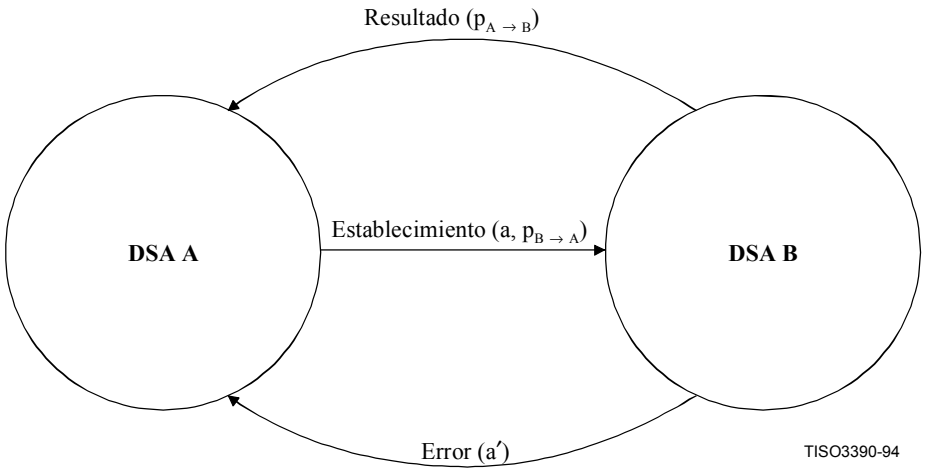
c) los procedimientos que cada DSA seguirá para cada una de las operaciones de gestión.

Durante el establecimiento de un ejemplar de vinculación operacional se crea un ejemplar de identificador de vinculación operacional (la id de vinculación). Este identificador, que es un entero, cuando es combinado con los nombres distinguidos de los dos DSA que intervienen en la vinculación operacional, forma un ejemplar de identificador único para la vinculación. Todas las operaciones de gestión subsiguientes al establecimiento de un ejemplar de vinculación operacional utilizarán la id de vinculación para identificar el ejemplar de vinculación operacional que está siendo modificado o terminado.

El iniciador de la operación de establecimiento transfiere siempre los parámetros del "acuerdo" al segundo DSA. El iniciador puede transferir, además, algunos parámetros de establecimiento que son específicos a su cometido en la vinculación operacional. Si el DSA respondedor está dispuesto a entrar en la vinculación operacional, puede retornar en el resultado parámetros de establecimiento que son específicos de su cometido. Si el DSA respondedor no está dispuesto a entrar en la vinculación operacional, devolverá un error, que puede facultativamente contener un acuerdo con un conjunto revisado de parámetros. Esto se representa en la figura 25 para el caso en que el cometido A es el iniciador de la operación de establecer, y en la figura 26 para el caso en que lo es el cometido B.

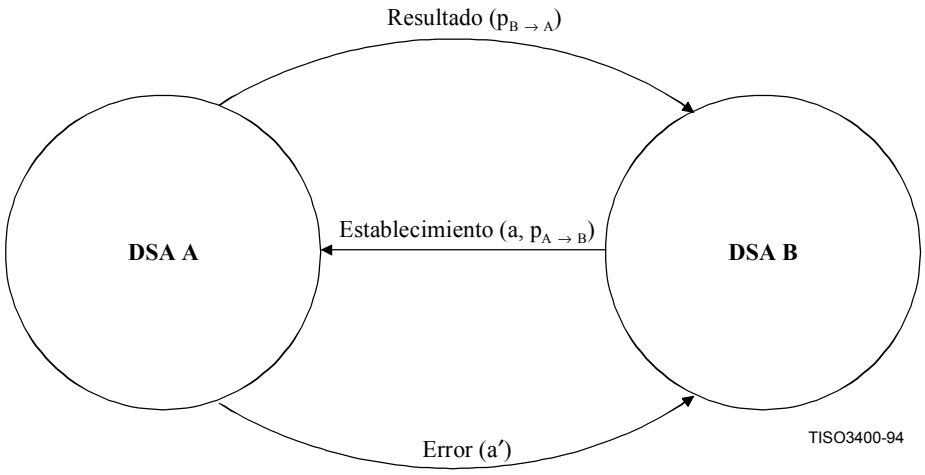
27.3 Plantillas para especificación de vinculación operacional

Para la definición de un tipo específico de vinculación operacional pueden utilizarse como plantillas las tres siguientes clases de objeto de información ASN.1. Estas plantillas permiten especificar mediante ASN.1 las partes del tipo de vinculación operacional que pueden ser formalizadas. Otros aspectos de la vinculación operacional, como son los procedimientos que ha de seguir un DSA cuando una vinculación operacional es establecida o terminada, tienen que ser especificados por otros medios (esto puede hacerse de una manera similar a la descripción informal de los procedimientos de DSA durante el proceso de resolución de nombre descrito en la Rec. UIT-T X.518 | ISO/CEI 9594-4).



a Acuerdo
p Parámetro de establecimiento

Figura 25 – Establecimiento iniciado por DSA con cometido A



a Acuerdo
p Parámetro de establecimiento

Figura 26 – Establecimiento iniciado por DSA con cometido B

27.3.1 Clase de objeto de información vinculación operacional

```
OPERATIONAL-BINDING ::= CLASS {
        &Agreement,
        &Cooperation        OP-BINDING-COOP,
        &both               OP-BIND-ROLE OPTIONAL,
        &roleA              OP-BIND-ROLE OPTIONAL,
        &roleB              OP-BIND-ROLE OPTIONAL,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        AGREEMENT                   &Agreement
        APPLICATION CONTEXTS        &Cooperation
        [ SYMMETRIC                 &both ]
        [ ASYMMETRIC
                [ ROLE-A            &roleA ]
                [ ROLE-B            &roleB ] ]
        ID                          &id }
```

La clase de objeto de información **OPERATIONAL-BINDING** sirve de plantilla de especificación para un tipo de vinculación operacional. Para facilitar su utilización como una plantilla se define una notación variable para esta clase. La correspondencia entre la definición de un tipo de vinculación operacional y los campos de la notación variable es la siguiente:

a) El tipo ASN.1 del parámetro de acuerdo que se utiliza para este tipo de vinculación operacional es el referenciado por el campo **AGREEMENT**.

b) Los contextos de aplicación y las operaciones de estos contextos de aplicación que se emplean en la fase de cooperación de un caso de vinculación operacional del tipo definido son los enumerados a continuación del campo **APPLICATION-CONTEXTS**. Todas las operaciones del contexto de aplicación listado son seleccionadas, a menos que el campo facultativo **APPLIES TO** esté presente y vaya seguido de una lista de referencias a operaciones que son seleccionadas entre el contexto de aplicación. Esta lista es un conjunto de clases de objeto compuesto de ejemplares de la clase de objeto de información **OPERATION**.

c) La clase de la vinculación operacional se define por los campos **SYMMETRIC** o **ASYMMETRIC**. En el caso de una vinculación operacional simétrica, el término **SYMMETRIC** va seguido de un solo objeto de información de la clase **OP-BIND-ROLE** que es válido para ambos cometidos de la vinculación operacional. En el caso de una vinculación operacional asimétrica, el término **ASYMMETRIC** va seguido de dos objetos de información de la clase **OP-BIND-ROLE**, uno referenciado por el subcampo **ROLE-A** y el otro por **ROLE-B**.

d) El valor de identificador de objeto que sirve para identificar el tipo de vinculación operacional que está definido es el referenciado por el campo **ID**.

27.3.2 Clase de objeto de información cooperación de vinculación operacional

```
OP-BINDING-COOP  ::=  CLASS {
      &applContext      APPLICATION-CONTEXT,
      &Operations       OPERATION OPTIONAL }
WITH SYNTAX {
      &applContext
      [ APPLIES TO      &Operations ] }
```

La clase de objeto de información **OP-BINDING-COOP** se emplea como una plantilla de especificación de las operaciones de un contexto de aplicación denominado, algunos de cuyos aspectos son determinados por la vinculación operacional. Un caso de esta clase sólo tiene sentido dentro del contexto de un tipo particular de vinculación operacional. Se define una notación variable para esta clase para facilitar su uso como una plantilla. La correspondencia entre la definición de un tipo de vinculación operacional y los campos de la notación variable es la siguiente:

a) El campo **applContext** (**contexto de aplicación**) identifica un contexto de aplicación, algunas o todas las operaciones del cual son determinadas de alguna manera por una vinculación operacional.

b) el campo **APPLIES TO (SE APLICA A)**, si está presente, identifica las operaciones particulares a que se aplica la vinculación operacional. Si el campo está ausente, la vinculación operacional se aplica a todas las operaciones del contexto de aplicación.

27.3.3 Clase de objeto de información cometido de vinculación operacional

```
OP-BIND-ROLE  ::=  CLASS {
      &establish                  BOOLEAN  DEFAULT FALSE,
      &EstablishParam                      OPTIONAL,
      &modify                     BOOLEAN  DEFAULT FALSE,
      &ModifyParam                         OPTIONAL,
      &terminate                  BOOLEAN  DEFAULT FALSE,
      &TerminateParam                      OPTIONAL }
WITH SYNTAX {
      [ ESTABLISHMENT-INITIATOR     &establish ]
      [ ESTABLISHMENT-PARAMETER     &EstablishParam ]
      [ MODIFICATION-INITIATOR      &modify ]
      [ MODIFICATION-PARAMETER      &ModifyParam ]
      [ TERMINATION-INITIATOR       &terminate ]
      [ TERMINATION-PARAMETER       &TerminateParam ] }
```

La clase de objeto de información **OP-BIND-ROLE** se emplea como una plantilla de especificación para cometidos de un tipo de vinculación operacional. Un caso de esta clase tiene sentido solamente en el contexto de un tipo particular de vinculación operacional. Para esta clase se ha definido una notación variable con el fin de simplificar su utilización como una plantilla. La correspondencia entre la definición de un cometido de vinculación operacional y los campos de la notación variable es la siguiente:

a) El campo **ESTABLISHMENT INITIATOR (INICIADOR DE ESTABLECIMIENTO)** indica si el DSA que asume el cometido definido puede iniciar el establecimiento de una vinculación operacional de un tipo particular.

b) El campo **ESTABLISHMENT PARAMETER (PARÁMETRO DE ESTABLECIMIENTO)** define el tipo ASN.1 intercambiado por un DSA que asume el cometido definido cuando se establece un caso del tipo de vinculación operacional.

c) El campo **MODIFICATION INITIATOR (INICIADOR DE MODIFICACIÓN)** indica si el DSA que asume el cometido definido puede iniciar la modificación de una vinculación operacional de un tipo particular.

d) El campo **MODIFICATION PARAMETER (PARÁMETRO DE MODIFICACIÓN)** define el tipo ASN.1 intercambiado por un DSA que asume el cometido definido cuando se modifica un caso del tipo de vinculación operacional.

e) El campo **TERMINATION INITIATOR (INICIADOR DE TERMINACIÓN)** indica si el DSA que asume el cometido definido puede terminar el establecimiento de una vinculación operacional de un tipo particular).

f) El campo **TERMINATION PARAMETER (PARÁMETRO DE TERMINACIÓN)** define el tipo ASN.1 intercambiado por un DSA que asume el cometido definido cuando se termina un caso del tipo de vinculación operacional.

28 Operaciones para la gestión de vinculaciones operacionales

Esta cláusula define el conjunto de operaciones que pueden utilizarse para establecer, modificar y terminar vinculaciones operacionales de diversos tipos. Estas operaciones son genéricas en cuanto a la forma en que pueden utilizarse para gestionar vinculaciones operacionales de cualquier tipo. Para la especificación de estas operaciones se utilizan las definiciones proporcionadas para un cierto tipo de vinculación operacional mediante la aplicación de la plantilla de la clase de objeto de información **OPERATIONAL-BINDING**.

NOTA – Con esta facilidad se pueden gestionar tipos arbitrarios de vinculaciones operacionales. Estas operaciones (junto con el contexto de aplicación asociado) proporcionan un medio de extensibilidad en lo tocante a interacciones de DSA. En el futuro podrán definirse nuevos tipos de vinculaciones operacionales que extiendan la funcionalidad que se proporciona entre los DSA.

28.1 Definición de contexto de aplicación

El conjunto de operaciones para gestionar ejemplares de vinculaciones operacionales puede utilizarse para la definición de un contexto de aplicación de las dos maneras siguientes:

1) Un contexto de aplicación puede construirse de modo que sólo contenga las operaciones para gestión de vinculaciones operacionales. Un contexto de aplicación para gestión de vinculaciones operacionales genéricas se define en la Rec. UIT-T X.519 | ISO/CEI 9594-5.

   Las operaciones que pueden ser intercambiadas durante la fase cooperativa de la vinculación operacional forman un contexto de aplicación aparte.

2) El conjunto de operaciones puede ser importado al módulo utilizado para definir un contexto de aplicación específico. Las operaciones de gestión de vinculaciones operacionales pueden entonces utilizarse, junto con las operaciones de la fase cooperativa, dentro de un contexto solo de aplicación.

   NOTA – La primera solución es útil en el caso en que un componente especializado de un DSA quiere utilizar una asociación únicamente para gestionar el conjunto de vinculaciones operacionales de ese DSA, y no está preparado para aceptar cualesquiera de las operaciones definidas para la fase cooperativa (por ejemplo, updateShadow).

28.2 Operación establecer vinculación operacional

La operación establecer vinculación operacional permite el establecimiento de un caso de vinculación operacional de un tipo predefinido entre dos DSA. Esto se efectúa mediante la transferencia de los parámetros de establecimiento y los términos del acuerdo que fueron definidos en la definición del tipo de vinculación operacional. El solicitante puede firmar, criptar o firmar y criptar los argumentos de la operación (véase 17.3). Si así se solicita, el respondedor puede firmar, criptar o firmar y criptar los resultados.

En el caso de una vinculación operacional simétrica, cualquiera de los dos DSA puede tomar la iniciativa de establecer un ejemplar de vinculación operacional del tipo predefinido.

En el caso de una vinculación operacional asimétrica, cualquiera de los dos DSA que asume "COMETIDO-A" o "COMETIDO-B" establece la vinculación operacional, según la definición específica del tipo de vinculación operacional.

```
establishOperationalBinding  OPERATION  ::=  {
        ARGUMENT        EstablishOperationalBindingArgument
        RESULT          EstablishOperationalBindingResult
        ERRORS          { operationalBindingError | securityError | serviceError }
        CODE            id-op-establishOperationalBinding }

EstablishOperationalBindingArgument  ::=  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType             [0]   OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID               [1]   OperationalBindingID OPTIONAL,
        accessPoint             [2]   AccessPoint,
                -- symmetric, Role A initiates, or Role B initiates --
        initiator CHOICE {
                symmetric               [3]   OPERATIONAL-BINDING.&both.&EstablishParam
                                                ({OpBindingSet}{@bindingType}),
                roleA-initiates         [4]   OPERATIONAL-BINDING.&roleA.&EstablishParam
                                                ({OpBindingSet}{@bindingType}),
                roleB-initiates         [5]   OPERATIONAL-BINDING.&roleB.&EstablishParam
                                                ({OpBindingSet}{@bindingType}) } OPTIONAL,
        agreement               [6]   OPERATIONAL-BINDING.&Agreement
                                            ({OpBindingSet}{@bindingType}),
        valid                   [7]   Validity DEFAULT { },
        securityParameters      [8]   SecurityParameters OPTIONAL } }

OpBindingSet     OPERATIONAL-BINDING  ::=  {
        shadowOperationalBinding  |
        hierarchicalOperationalBinding  |
        nonSpecificHierarchicalOperationalBinding }

OperationalBindingID  ::=  SEQUENCE {
        identifier    INTEGER,
        version       INTEGER }
```

El componente **bindingType** indica el tipo de vinculación operacional que va a ser establecida. Los tipos de vinculación operacional se definen utilizando la plantilla para la clase de objeto de información **OPERATIONAL-BINDING** que asigna un valor de identificador de objeto al tipo de vinculación operacional. El **bindingType** se toma del campo **ID** de uno de los ejemplares de un tipo de vinculación operacional referenciado por **OpBindingSet**. Este conjunto es un parámetro de **EstablishOperationalBindingArgument**, un tipo parametrizado.

El DSA iniciador puede asignar una identificación al ejemplar de vinculación operacional mediante el componente **bindingID**. Si **bindingID** está ausente del argumento de la operación, el DSA respondedor asignará una ID al ejemplar de vinculación operacional y lo devolverá en el componente **bindingID** del **establishOperationalBindingResult**. Tanto en uno como en otro caso, cuando se establece una vinculación operacional el componente **identifier** y el componente **version** del valor **OperationalBindingID** serán asignados y emitidos por el DSA que hace la asignación.

El componente **accessPoint** especifica el punto de acceso del iniciador para interacciones subsiguientes.

El cometido asumido por el DSA que emite la operación establecer vinculación operacional se indica por el tipo **CHOICE** con las opciones **symmetric**, **roleA-initiates**, y **roleB-initiates**. La opción **CHOICE** gobierna los parámetros de establecimiento particulares empleados por los DSA iniciados y respondedor. Las semánticas de los cometidos se definen como parte de la definición del tipo de vinculación operacional. El tipo ASN.1 del **CHOICE** lo determina el **ESTABLISHMENT PARAMETER** de la plantilla de clase de objeto de información **OP-BIND-ROLE** del iniciador. El tipo **CHOICE** es omitido si el establecimiento del tipo de vinculación operacional no requiere que el iniciador introduzca un parámetro de establecimiento.

El componente **agreement** contiene los términos del acuerdo que gobierna el ejemplar de vinculación operacional. Su contenido efectivo depende del tipo de vinculación operacional que vaya a establecerse. El tipo ASN.1 para este parámetro lo define el campo **AGREEMENT** de la plantilla de clase de objeto de información **OPERATIONAL-BINDING** del tipo de vinculación operacional.

La duración del ejemplar de vinculación operacional se define en **valid**. La hora de comienzo de la existencia del ejemplar de vinculación operacional se especifica en **validFrom** y la hora (o tiempo) a la que se termina el ejemplar de vinculación operacional viene dada por **validUntil**.

```
Validity ::= SEQUENCE {
        validFrom   [0]   CHOICE {
                now         [0]   NULL,
                time        [1]   Time } DEFAULT now : NULL,
        validUntil  [1]   CHOICE {
                explicitTermination     [0]   NULL,
                time                    [1]   Time } DEFAULT explicitTermination : NULL }

Time ::= CHOICE {
        utcTime           UTCTime,
        generalizedTime   GeneralizedTime }
```

Antes de utilizar un valor de **Time** en cualquier operación de comparación y si la sintaxis de **Time** ha sido elegida como el tipo **UTCTime**, el valor del campo de año de dos cifras será racionalizado en un valor de año de cuatro cifras, como sigue:

- Si el valor de dos cifras es 00 a 49 inclusive, se añadirá 2000 al valor.
- Si el valor de dos cifras es 50 a 99 inclusive, se añadirá 1900 al valor

El uso de **GeneralizedTime** puede impedir el interfuncionamiento con implementaciones que desconocen la posibilidad de elegir **UTCTime** o **GeneralizedTime**. Es responsabilidad de los que especifican los dominios en los cuales se utilizará esta Especificación de directorio, por ejemplo, grupos de perfil, definir cuándo se puede usar **GeneralizedTime**. En ningún caso se utilizará **UTCTime** para representar fechas posteriores a 2049.

Si la operación establecer vinculación operacional tiene éxito, se devuelve el siguiente resultado, que puede ser firmado o criptado, o firmado y criptado por el respondedor (véase 17.3).

```
EstablishOperationalBindingResult ::= OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType     [0]   OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID       [1]   OperationalBindingID OPTIONAL,
        accessPoint     [2]   AccessPoint,
        -- symmetric, Role A replies , or Role B replies
        initiator CHOICE {
                symmetric       [3]   OPERATIONAL-BINDING.&both.&EstablishParam
                                      ({OpBindingSet}{@bindingType}),
                roleA-replies   [4]   OPERATIONAL-BINDING.&roleA.&EstablishParam
                                      ({OpBindingSet}{@bindingType}),
                roleB-replies   [5]   OPERATIONAL-BINDING.&roleB.&EstablishParam
                                      ({OpBindingSet}{@bindingType}) } OPTIONAL,
        COMPONENTS OF           CommonResultsSeq } }
```

El componente **bindingType** está contenido en el resultado para indicar el tipo de vinculación operacional para uso dentro del elemento **CHOICE**. Su valor es el mismo que el proporcionado por el iniciador del establecimiento y se toma del campo **ID** de uno de los ejemplares de un tipo de vinculación operacional referenciado por **OpBindingSet**. Este conjunto es un parámetro de **EstablishOperationalBindingResult**, un tipo parametrizado.

La identificación del ejemplar de vinculación operacional establecida puede ser devuelta en **bindingID**. Deberá utilizarse para identificar este ejemplar de vinculación operacional en cualquier operación subsiguiente de Modificar o Terminar Vinculación Operacional, y puede utilizarse en cualquier otra operación que se ejecute en la fase cooperativa del ejemplar de vinculación operacional establecida.

El componente **accessPoint** especifica el punto de acceso del respondedor para interacciones subsiguientes.

El DSA iniciador puede asignar una identificación al ejemplar de vinculación operacional mediante el componente **bindingID**. Si **bindingID** está ausente del argumento de la operación, el DSA respondedor asignará una ID al ejemplar de vinculación operacional y la devolverá en el componente **bindingID** del **establishOperationalBindingResult**.

El cometido asumido por el DSA que responde a la operación establecer vinculación operacional se indica por el tipo **CHOICE** con las opciones **symmetric**, **roleA-initiates** y **roleB-initiates**. Las semánticas de los cometidos se definen como parte de la definición del tipo de vinculación operacional. El tipo ASN.1 del **CHOICE** lo determina el **ESTABLISHMENT PARAMETER** de la plantilla de clase de objeto de información **OP-BIND-ROLE** del respondedor. El tipo **CHOICE** se omite si el establecimiento del tipo de vinculación operacional no requiere que el respondedor introduzca un parámetro de establecimiento.

28.3 Operación modificar vinculación operacional

La operación modificar vinculación operacional se utiliza para modificar una vinculación operacional establecida. El derecho a modificar se indica por el (los) campo(s) **MODIFICATION INITIATOR** en la definición del tipo de vinculación operacional utilizando las plantillas de clase de objeto de información **OP-BIND-ROLE** y **OPERATIONAL-BINDING**.

Los componentes de una vinculación operacional que pueden ser modificados son el contenido del acuerdo para la vinculación operacional y su periodo de validez. Además, el cometido iniciador puede especificar un parámetro de modificación. El solicitante puede firmar, criptar o firmar y criptar los argumentos de la operación (véase 17.3). Si así se solicita, el respondedor puede firmar, criptar o firmar y criptr el resultado.

```
modifyOperationalBinding  OPERATION  ::= {
        ARGUMENT        ModifyOperationalBindingArgument
        RESULT          ModifyOperationalBindingResult
        ERRORS          { operationalBindingError | securityError | serviceError }
        CODE            id-op-modifyOperationalBinding }

ModifyOperationalBindingArgument  ::=  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType     [0]   OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID       [1]   OperationalBindingID,
        accessPoint     [2]   AccessPoint OPTIONAL,
        -- symmetric, Role A initiates, or Role B initiates --
        initiator CHOICE {
                symmetric           [3]   OPERATIONAL-BINDING.&both.&ModifyParam
                                          ({OpBindingSet}{@bindingType}),
                roleA-initiates     [4]   OPERATIONAL-BINDING.&roleA.&ModifyParam
                                          ({OpBindingSet}{@bindingType}),
                roleB-initiates     [5]   OPERATIONAL-BINDING.&roleB.&ModifyParam
                                          ({OpBindingSet}{@bindingType}) } OPTIONAL,
        newBindingID    [6]   OperationalBindingID,
        newAgreement    [7]   OPERATIONAL-BINDING.&Agreement
                              ({OpBindingSet}{@bindingType})  OPTIONAL,
        valid           [8]   Validity OPTIONAL,
        securityParameters
                        [9]   SecurityParameters OPTIONAL } }
```

El componente **bindingType** indica el tipo de vinculación operacional que va a ser modificado. El **bindingType** se toma del campo **ID** de uno de los ejemplares de un tipo de vinculación operacional referenciado por **OpBindingSet**. Este conjunto es un parámetro de **ModifyOperationalBindingArgument**, un tipo parametrizado.

La identificación del ejemplar de vinculación operacional que va a ser modificada la da **bindingID**. El identificador revisado del ejemplar de vinculación operacional lo da **newBindingID**. El componente **version** de **newBindingID** deberá ser mayor que el de **bindingID**.

El componente facultativo **accessPoint** está presente si el punto de acceso del iniciador para interacciones subsiguientes va a ser cambiado.

El cometido asumido por el DSA que emite la operación modificar vinculación operacional es indicado por el tipo **CHOICE** con las opciones **symmetric**, **roleA-initiates**, y **roleB-initiates**. Las semánticas de los cometidos se definen como parte de la definición del tipo de vinculación operacional. El tipo ASN.1 del **CHOICE** lo determina el **MODIFICATION PARAMETER** de la plantilla de clase de objeto de información **OP-BIND-ROLE** del iniciador. El tipo **CHOICE** es omitido si el establecimiento del tipo de vinculación operacional no requiere un parámetro de establecimiento introducido por el iniciador.

El componente **newAgreement**, si está presente, contiene los términos modificados del acuerdo que gobierna el ejemplar de vinculación operacional. El tipo ASN.1 para este parámetro lo define el campo **AGREEMENT** de la plantilla de clase de objeto de información **OPERATIONAL-BINDING** del tipo de vinculación operacional. Si **newAgreement** no está presente los parámetros del acuerdo no son modificados por la operación.

El componente facultativo **valid** puede utilizarse para indicar un periodo revisado de validez para el acuerdo cambiado. Si el componente **valid** está ausente, se presupone que el componente **validFrom** tiene el valor **now** y que el componente **validUntil** no ha cambiado. Si el componente **validFrom** está presente y hace referencia a un instante de tiempo en el futuro, el acuerdo actual sigue en vigor hasta ese instante.

Si la operación modificar vinculación tiene éxito, se devuelve el siguiente resultado que puede ser firmado o criptado, o firmado y criptado por el respondedor (véase 17.3).

```
ModifyOperationalBindingResult ::= CHOICE {
    null            [0]  NULL,
    protected       [1]  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        newBindingID            OperationalBindingID,
        bindingType             OPERATIONAL-BINDING.&id ({OpBindingSet}),
        newAgreement            OPERATIONAL-BINDING.&Agreement
                                    ({OpBindingSet}{@bindingType}),
        valid                   Validity OPTIONAL,
        COMPONENTS OF           CommonResultsSeq } }
```

No es posible que el DSA respondedor devuelva el parámetro de modificación definido para su cometido, al iniciador de la terminación.

28.4 Operación terminar vinculación operacional

La operación terminar vinculación operacional se utiliza para pedir la terminación de un ejemplar de vinculación operacional establecida. El derecho a pedir la terminación es indicado por el (los) campo(s) **TERMINATION INITIATOR** en la definición del tipo de vinculación operacional utilizando las plantillas de clase de objeto de información **OP-BIND-ROLE** y **OPERATIONAL-BINDING**. El solicitante puede firmar, criptar o firmar y criptar los argumentos de la operación (véase 17.3). Si así se solicita, el respondedor puede firmar, criptar o firmar y criptar el resultado.

```
terminateOperationalBinding  OPERATION  ::= {
    ARGUMENT        TerminateOperationalBindingArgument
    RESULT          TerminateOperationalBindingResult
    ERRORS          { operationalBindingError | securityError | serviceError }
    CODE            id-op-terminateOperationalBinding }

TerminateOperationalBindingArgument  ::=  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
    bindingType             [0]  OPERATIONAL-BINDING.&id ({OpBindingSet}),
    bindingID               [1]  OperationalBindingID,
    -- symmetric, Role A initiates, or Role B initiates --
    initiator CHOICE {
        symmetric               [2]  OPERATIONAL-BINDING.&both.&TerminateParam
                                         ({OpBindingSet}{@bindingType}),
        roleA-initiates         [3]  OPERATIONAL-BINDING.&roleA.&TerminateParam
                                         ({OpBindingSet}{@bindingType}),
        roleB-initiates         [4]  OPERATIONAL-BINDING.&roleB.&TerminateParam
                                         ({OpBindingSet}{@bindingType})}  OPTIONAL,
    terminateAt             [5]  Time OPTIONAL,
    securityParameters      [6]  SecurityParameters OPTIONAL } }
```

El componente **bindingType** indica qué tipo de vinculación operacional va a ser terminado. El **bindingType** se toma del campo **ID** de ejemplares de un tipo de vinculación operacional referenciado por **OpBindingSet.** Este conjunto es un parámetro de **TerminateOperationalBindingArgument**, un tipo parametrizado.

La identificación del ejemplar de vinculación operacional que va a ser terminada la da **bindingID**. Si el componente **version** está presente en la **bindingID**, se pasa por alto.

El cometido asumido por el DSA que emite la operación terminar vinculación operacional se indica por el tipo **CHOICE** con las opciones **symmetric**, **roleA-initiates** y **roleB-initiates**. Las semánticas de los cometidos se definen como parte de la definición del tipo de vinculación operacional. El tipo ASN.1 del **CHOICE** lo determina el **TERMINATION PARAMETER** de la plantilla de clase de objeto de información **OP-BIND-ROLE** del iniciador. El tipo **CHOICE** se omite si la terminación del tipo de vinculación operacional no requiere un parámetro de terminación introducido por el iniciador.

Si la vinculación operacional no va a ser terminada inmediatamente, se puede definir una hora de terminación aplazada en **terminateAt**.

Si la operación terminar vinculación operacional tiene éxito, se devuelve el siguiente resultado que puede ser firmado o criptado, o firmado y criptado por el respondedor (véase 17.3).

```
TerminateOperationalBindingResult ::= CHOICE {
      null                [0]   NULL,
      protected           [1]   OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
                  bindingID               OperationalBindingID,
                  bindingType             OPERATIONAL-BINDING.&id ({OpBindingSet}),
                  terminateAt             GeneralizedTime OPTIONAL,
                  COMPONENTS OF           CommonResultsSeq } } }
```

No es posible que el DSA que responde devuelva el parámetro de terminación definido para su cometido, al iniciador de la terminación.

28.5 Error de vinculación operacional

Un error de vinculación operacional informa sobre un problema relacionado con la utilización de operaciones para la gestión de vinculaciones operacionales. El respondedor puede firmar, criptar o firmar y criptar el parámetro del *error* (véase 17.3).

```
operationalBindingError ERROR ::= {
      PARAMETER      OPTIONALLY-PROTECTED-SEQ {
                           OpBindingErrorParam }
      CODE           id-err-operationalBindingError }

OpBindingErrorParam ::= SEQUENCE {
      problem                 [0]   ENUMERATED {
                  invalidID                    (0),
                  duplicateID                  (1),
                  unsupportedBindingType       (2),
                  notAllowedForRole            (3),
                  parametersMissing            (4),
                  roleAssignment               (5),
                  invalidStartTime             (6),
                  invalidEndTime               (7),
                  invalidAgreement             (8),
                  currentlyNotDecidable        (9),
                  modificationNotAllowed       (10) },
      bindingType             [1]   OPERATIONAL-BINDING.&id ({OpBindingSet}) OPTIONAL,
      agreementProposal       [2]   OPERATIONAL-BINDING.&Agreement
                                    ({OpBindingSet}{@bindingType}) OPTIONAL,
      retryAt                 [3]   Time OPTIONAL,
      COMPONENTS OF                 CommonResultsSeq }
```

Los valores de **problem** tienen los siguientes significados:

a) **invalidID**: La ID de la vinculación operacional dada en la petición no es conocida por el DSA receptor o está en el estado erróneo para la petición solicitada.

b) **duplicatedID**: La ID de la vinculación operacional dada en la petición de establecimiento ya existe en el respondedor. Esto puede deberse a un anterior intento de establecer un ejemplar de vinculación operacional cuando, después de perdido el resultado, el iniciador repite la petición de establecimiento.

c) **unsupportedBindingType**: El tipo de vinculación operacional solicitado no es soportado por el DSA.

d) **notAllowedForRole**: En el caso de vinculación operacional se ha solicitado una operación de gestión que no está autorizada para el cometido que desempeña el solicitante (por ejemplo, una operación terminar vinculación operacional ha sido emitida por un DSA que asume un cometido que no está autorizado para iniciar la terminación del ejemplar de vinculación operacional).

e) **parametersMissing**: Faltan uno o más parámetros de establecimiento o de terminación requeridos, definidos para el tipo de vinculación operacional.

f) **roleAssignment**: La asignación de cometido solicitada para un ejemplar de vinculación operacional asimétrica no ha sido aceptada.

g) **invalidStartTime**: La hora de comienzo especificada para el ejemplar de vinculación operacional no ha sido aceptada.

h) **invalidEndTime**: La hora de terminación especificada para el ejemplar de vinculación operacional no ha sido aceptada.

i) **invalidAgreement**: Los términos del acuerdo para el ejemplar de vinculación operacional solicitada no han sido aceptados. Los términos del acuerdo que serían aceptados por el DSA respondedor pueden devolverse en **agreementProposal**.

j) **currentlyNotDecidable**: El DSA no puede decidir en-línea sobre el establecimiento del ejemplar de vinculación operacional solicitada. Se puede indicar en **retryAt** a qué hora se puede repetir la petición.

k) **modificationNotAllowed**: Se rechaza la operación modificar vinculación operacional porque no se permite modificación para este ejemplar de vinculación operacional.

El componente **bindingType** será igual al transmitido por el invocador de la operación de gestión de vinculación operacional fallida.

El componente **agreementProposal** sólo se utilizará en respuesta a una operación **EstablishOperationalBinding** para proponer un conjunto revisado de parámetros de acuerdo, según se describe en 28.2.

El componente **retryAt** se utilizará solamente junto con el valor **currentlyNotDecidable** de **problem** para indicar cuándo debe repetirse la operación **EstablishOperationalBinding** o **ModifyOperationalBinding**.

El componente **CommonResults** (véase 7.4 de la Rec. UIT-T X.511 | ISO/CEI 9594-3) incluye **SecurityParameters**. Si el respondedor debe firmar el parámetro de error, se incluirá el componente **SecurityParameters** (véase 7.10 de la Rec. UIT-T X.511 | ISO/CEI 9594-3) en los **CommonResultsSeq**.

28.6 Vincular y desvincular gestión de vinculación operacional

Las operaciones **DSAOperationalBindingManagementBind** (**vincular gestión de vinculaciones operacionales por DSA**) y **DSAOperationalBindingManagementUnbind** (**desvincular gestión de vinculaciones operacionales por DSA**), definidas en 28.6.1 y 28.6.2, son utilizadas por un DSA al comienzo y al final de un determinado periodo de actividad de gestión de vinculaciones operacionales.

La protección de **dSAOperationalBindingManagementBind** y **dSAOperationalBindingManagementUnbind** serán equivalentes a las protecciones aplicadas a las operaciones **DSAbind** y **DSAunbind**.

NOTA – El elemento de servicio de intercambio de seguridad (véase la Rec. UIT-T X.519 | ISO/CEI 9594-5) puede transportar las credenciales necesarias para la autenticación, en cuyo caso no estarán presentes en los argumentos o resultados de la vinculación.

28.6.1 Vincular gestión de vinculaciones operacionales por DSA

La operación **dSAOperationalBindingManagementBind** se utiliza al principio de un periodo de gestión de vinculaciones operacionales.

```
dSAOperationalBindingManagementBind OPERATION ::= directoryBind
```

Los componentes del **dSAOperationalManagementBind** son virtualmente idénticos a sus contrapartes en **directoryBind** (véase la Rec. UIT-T X.511 | ISO/CEI 9594-3); las diferencias se indican a continuación.

NOTA – El elemento de servicio de intercambio de seguridad (véase la Rec. UIT-T X.519 | ISO/CEI 9594-5) puede transportar las credenciales necesarias para la autenticación, en cuyo caso no estarán presentes en los argumentos o resultados de la vinculación.

28.6.1.1 Credenciales del iniciador

Las **Credentials** del **DirectoryBindArgument** autorizan a enviar el DSA respondedor información que identifica el Título-AE del DSA iniciador. El Título-AE tendrá la forma de un nombre distinguido de directorio.

28.6.1.2 Credenciales del respondedor

Las **Credentials** del **DirectoryBindResult** autorizan a enviar al DSA iniciador información que identifica el Título-AE del DSA respondedor. El Título-AE tendrá la forma de un nombre distinguido de directorio.

28.6.2 Desvincular gestión de vinculaciones operacionales por DSA

La operación **dSAOperationalManagementUnbind** se utiliza al final de un periodo de provisión de gestión de vinculaciones operacionales.

dSAOperationalBindingManagementUnbind OPERATION ::= directoryUnbind

No hay argumentos, resultados ni errores.

Anexo A

Utilización de identificadores de objeto

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo documenta los confines superiores del subárbol de identificador de objeto en el que residen todos los identificadores de objeto asignados en esta Especificación de directorio. Para ello, proporciona un módulo ASN.1 llamado **UsefulDefinitions (definiciones útiles)** en el cual se asignan nombres a todos los nodos no-hoja del subárbol.

```
UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

ID          ::=    OBJECT IDENTIFIER

ds    ID    ::=    {joint-iso-itu-t ds(5)}

-- categories of information object --

module                          ID    ::=    {ds 1}
serviceElement                  ID    ::=    {ds 2}
applicationContext              ID    ::=    {ds 3}
attributeType                   ID    ::=    {ds 4}
attributeSyntax                 ID    ::=    {ds 5}
objectClass                     ID    ::=    {ds 6}
-- attributeSet                 ID    ::=    {ds 7}
algorithm                       ID    ::=    {ds 8}
abstractSyntax                  ID    ::=    {ds 9}
-- object                       ID    ::=    {ds 10}
-- port                         ID    ::=    {ds 11}
dsaOperationalAttribute         ID    ::=    {ds 12}
matchingRule                    ID    ::=    {ds 13}
knowledgeMatchingRule           ID    ::=    {ds 14}
nameForm                        ID    ::=    {ds 15}
group                           ID    ::=    {ds 16}
subentry                        ID    ::=    {ds 17}
operationalAttributeType        ID    ::=    {ds 18}
operationalBinding              ID    ::=    {ds 19}
schemaObjectClass               ID    ::=    {ds 20}
schemaOperationalAttribute      ID    ::=    {ds 21}
administrativeRoles             ID    ::=    {ds 23}
accessControlAttribute          ID    ::=    {ds 24}
rosObject                       ID    ::=    {ds 25}
contract                        ID    ::=    {ds 26}
package                         ID    ::=    {ds 27}
accessControlSchemes            ID    ::=    {ds 28}
certificateExtension            ID    ::=    {ds 29}
managementObject                ID    ::=    {ds 30}
attributeValueContext           ID    ::=    {ds 31}
-- securityExchange             ID    ::=    {ds 32}
idmProtocol                     ID    ::=    {ds 33}
problem                         ID    ::=    {ds 34}
notification                    ID    ::=    {ds 35}
matchingRestriction             ID    ::=    {ds 36}  -- None are currently defined by this specification
controlAttributeType            ID    ::=    {ds 37}
```

```
-- modules --

usefulDefinitions                ID    ::=    {module usefulDefinitions(0) 4}
informationFramework             ID    ::=    {module informationFramework(1) 4}
directoryAbstractService         ID    ::=    {module directoryAbstractService(2) 4}
distributedOperations            ID    ::=    {module distributedOperations(3) 4}
protocolObjectIdentifiers        ID    ::=    {module protocolObjectIdentifiers(4) 4}
selectedAttributeTypes           ID    ::=    {module selectedAttributeTypes(5) 4}
selectedObjectClasses            ID    ::=    {module selectedObjectClasses(6) 4}
authenticationFramework          ID    ::=    {module authenticationFramework(7) 4}
algorithmObjectIdentifiers       ID    ::=    {module algorithmObjectIdentifiers(8) 4}
directoryObjectIdentifiers       ID    ::=    {module directoryObjectIdentifiers(9) 4}
upperBounds                      ID    ::=    {module upperBounds(10) 4}
dap                              ID    ::=    {module dap(11) 4}
dsp                              ID    ::=    {module dsp(12) 4}
distributedDirectoryOIDs         ID    ::=    {module distributedDirectoryOIDs(13) 4}
directoryShadowOIDs              ID    ::=    {module directoryShadowOIDs(14) 4}
directoryShadowAbstractService   ID    ::=    {module directoryShadowAbstractService(15) 4}
disp                             ID    ::=    {module disp(16) 4}
dop                              ID    ::=    {module dop(17) 4}
opBindingManagement              ID    ::=    {module opBindingManagement(18) 4}
opBindingOIDs                    ID    ::=    {module opBindingOIDs(19) 4}
hierarchicalOperationalBindings  ID    ::=    {module hierarchicalOperationalBindings(20) 4}
dsaOperationalAttributeTypes     ID    ::=    {module dsaOperationalAttributeTypes(22) 4}
schemaAdministration             ID    ::=    {module schemaAdministration(23) 4}
basicAccessControl               ID    ::=    {module basicAccessControl(24) 4}
directoryOperationalBindingTypes ID    ::=    {module directoryOperationalBindingTypes(25) 4}
certificateExtensions            ID    ::=    {module certificateExtensions(26) 4}
directoryManagement              ID    ::=    {module directoryManagement(27) 4}
enhancedSecurity                 ID    ::=    {module enhancedSecurity (28) 4}
-- directorySecurityExchanges    ID    ::=    {module directorySecurityExchanges (29) 4}
iDMProtocolSpecification         ID    ::=    {module iDMProtocolSpecification(30) 4 }
directoryIDMProtocols            ID    ::=    {module directoryIDMProtocols(31) 4 }
attributeCertificateDefinitions  ID    ::=    {module attributeCertificateDefinitions(32) 4}
serviceAdministration            ID    ::=    {module serviceAdministration(33) 4}
-- the following defintion is for a module that holds externally defined schema elements not defined
-- using formal ASN.1 notation (see latest version of Implementor's Guide)
externalDefinitions              ID    ::=    {module externalDefinitions(34) }

-- synonyms --

id-oc                            ID    ::=    objectClass
id-at                            ID    ::=    attributeType
id-as                            ID    ::=    abstractSyntax
id-mr                            ID    ::=    matchingRule
id-nf                            ID    ::=    nameForm
id-sc                            ID    ::=    subentry
id-oa                            ID    ::=    operationalAttributeType
id-ob                            ID    ::=    operationalBinding
id-doa                           ID    ::=    dsaOperationalAttribute
id-kmr                           ID    ::=    knowledgeMatchingRule
id-soc                           ID    ::=    schemaObjectClass
id-soa                           ID    ::=    schemaOperationalAttribute
id-ar                            ID    ::=    administrativeRoles
id-aca                           ID    ::=    accessControlAttribute
id-ac                            ID    ::=    applicationContext
id-rosObject                     ID    ::=    rosObject
id-contract                      ID    ::=    contract
id-package                       ID    ::=    package
id-acScheme                      ID    ::=    accessControlSchemes
id-ce                            ID    ::=    certificateExtension
id-mgt                           ID    ::=    managementObject
id-avc                           ID    ::=    attributeValueContext
-- id-se                         ID    ::=    securityExchange
id-idm                           ID    ::=    idmProtocol
id-pr                            ID    ::=    problem
id-not                           ID    ::=    notification
id-mre                           ID    ::=    matchingRestriction
id-cat                           ID    ::=    controlAttributeType
```

```
-- obsolete module identifiers --

--      usefulDefinition                    ID    ::=    {module 0}
--      informationFramework                ID    ::=    {module 1}
--      directoryAbstractService            ID    ::=    {module 2}
--      distributedOperations               ID    ::=    {module 3}
--      protocolObjectIdentifiers           ID    ::=    {module 4}
--      selectedAttributeTypes              ID    ::=    {module 5}
--      selectedObjectClasses               ID    ::=    {module 6}
--      authenticationFramework             ID    ::=    {module 7}
--      algorithmObjectIdentifiers          ID    ::=    {module 8}
--      directoryObjectIdentifiers          ID    ::=    {module 9}
--      upperBounds                         ID    ::=    {module 10}
--      dap                                 ID    ::=    {module 11}
--      dsp                                 ID    ::=    {module 12}
--      distributedDirectoryObjectIdentifiers ID  ::=    {module 13}

-- unused module identifiers --

--      directoryShadowOIDs                 ID    ::=    {module 14}
--      directoryShadowAbstractService      ID    ::=    {module 15}
--      disp                                ID    ::=    {module 16}
--      dop                                 ID    ::=    {module 17}
--      opBindingManagement                 ID    ::=    {module 18}
--      opBindingOIDs                       ID    ::=    {module 19}
--      hierarchicalOperationalBindings     ID    ::=    {module 20}
--      dsaOperationalAttributeTypes        ID    ::=    {module 22}
--      schemaAdministration                ID    ::=    {module 23}
--      basicAccessControl                  ID    ::=    {module 24}
--      operationalBindingOIDs              ID    ::=    {module 25}

END -- UsefulDefinitions
```

Anexo B

Marco de información en ASN.1

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo recapitula todas las definiciones ASN.1 de tipo, valor y macro contenidas en esta Especificación de directorio. Las definiciones forman el módulo ASN.1 **InformationFramework (marco de información)**.

```
InformationFramework {joint-iso-itu-t ds(5) module(1) informationFramework(1) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

        directoryAbstractService, id-ar, id-at, id-mr, id-nf, id-oa, id-oc, id-sc,
        selectedAttributeTypes, serviceAdministration, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

        SearchRule
                FROM ServiceAdministration serviceAdministration

-- from ITU-T Rec. X.511 | ISO/IEC 9594-3

        TypeAndContextAssertion
                FROM DirectoryAbstractService directoryAbstractService

-- from ITU-T Rec. X.520 | ISO/IEC 9594-6

        booleanMatch, commonName, DirectoryString {}, generalizedTimeMatch,
        generalizedTimeOrderingMatch, integerFirstComponentMatch, integerMatch,
        integerOrderingMatch, objectIdentifierFirstComponentMatch
                FROM SelectedAttributeTypes selectedAttributeTypes

        ub-search
                FROM UpperBounds upperBounds ;

-- attribute data types --

Attribute  ::=  SEQUENCE {
        type                    ATTRIBUTE.&id ({SupportedAttributes}),
        values                  SET SIZE (0 .. MAX) OF ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
        valuesWithContext       SET SIZE (1 .. MAX) OF SEQUENCE {
                value                   ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
                contextList             SET SIZE (1..MAX) OF Context } OPTIONAL }

AttributeType  ::=  ATTRIBUTE.&id

AttributeValue  ::=  ATTRIBUTE.&Type

Context  ::=  SEQUENCE {
        contextType     CONTEXT.&id ({SupportedContexts}),
        contextValues   SET SIZE (1..MAX) OF CONTEXT.&Type ({SupportedContexts}{@contextType}),
        fallback        BOOLEAN DEFAULT FALSE }
```

```
AttributeValueAssertion  ::=  SEQUENCE {
        type                    ATTRIBUTE.&id ({SupportedAttributes}),
        assertion               ATTRIBUTE.&equality-match.&AssertionType ({SupportedAttributes}{@type}),
        assertedContexts        CHOICE {
                allContexts             [0]  NULL,
                selectedContexts        [1]  SET SIZE (1..MAX) OF ContextAssertion } OPTIONAL }

ContextAssertion  ::=  SEQUENCE {
        contextType             CONTEXT.&id({SupportedContexts}),
        contextValues           SET SIZE (1..MAX) OF
                        CONTEXT.&Assertion ({SupportedContexts}{@contextType})}

AttributeTypeAssertion  ::=  SEQUENCE {
        type                    ATTRIBUTE.&id ({SupportedAttributes}),
        assertedContexts        SEQUENCE SIZE (1..MAX) OF ContextAssertion  OPTIONAL }


-- Definition of the following information object set is deferred, perhaps to standardized
-- profiles or to protocol implementation conformance statements. The set is required to
-- specify a table constraint on the values component of Attribute, the value component
-- of AttributeTypeAndValue, and the assertion component of AttributeValueAssertion.

SupportedAttributes  ATTRIBUTE  ::=  { objectClass | aliasedEntryName, ... }

-- Definition of the following information object set is deferred, perhaps to standardized
-- profiles or to protocol implementation conformance statements. The set is required to
-- specify a table constraint on the context specifications

SupportedContexts  CONTEXT  ::=  { ... }


-- naming data types --

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

RDNSequence  ::=  SEQUENCE OF RelativeDistinguishedName

DistinguishedName  ::=  RDNSequence

RelativeDistinguishedName  ::=  SET SIZE (1..MAX) OF AttributeTypeAndDistinguishedValue

AttributeTypeAndDistinguishedValue  ::=  SEQUENCE  {
        type                    ATTRIBUTE.&id ({SupportedAttributes}),
        value                   ATTRIBUTE.&Type({SupportedAttributes}{@type}),
        primaryDistinguished    BOOLEAN DEFAULT TRUE,
        valuesWithContext       SET SIZE (1..MAX) OF SEQUENCE {
                distingAttrValue        [0]     ATTRIBUTE.&Type ({SupportedAttributes}{@type}) OPTIONAL,
                contextList                     SET SIZE (1..MAX) OF Context } OPTIONAL }

-- subtree data types --

SubtreeSpecification  ::=  SEQUENCE {
        base                    [0]     LocalName DEFAULT { },
                                        COMPONENTS OF  ChopSpecification,
        specificationFilter     [4]     Refinement OPTIONAL }
        -- empty sequence specifies whole administrative area

LocalName  ::=  RDNSequence

ChopSpecification  ::=  SEQUENCE  {
        specificExclusions      [1]     SET SIZE (1..MAX) OF CHOICE {
                chopBefore              [0]     LocalName,
                chopAfter               [1]     LocalName } OPTIONAL,
        minimum                 [2]     BaseDistance DEFAULT 0,
        maximum                 [3]     BaseDistance OPTIONAL }

BaseDistance  ::=  INTEGER (0..MAX)
```

```
Refinement ::= CHOICE {
        item            [0]     OBJECT-CLASS.&id,
        and             [1]     SET OF Refinement,
        or              [2]     SET OF Refinement,
        not             [3]     Refinement }


-- OBJECT-CLASS information object class specification --

OBJECT-CLASS ::=    CLASS {
        &Superclasses           OBJECT-CLASS OPTIONAL,
        &kind                   ObjectClassKind DEFAULT structural,
        &MandatoryAttributes    ATTRIBUTE OPTIONAL,
        &OptionalAttributes     ATTRIBUTE OPTIONAL,
        &id                     OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        [ SUBCLASS OF           &Superclasses ]
        [ KIND                  &kind ]
        [ MUST CONTAIN          &MandatoryAttributes ]
        [ MAY CONTAIN           &OptionalAttributes ]
        ID                      &id }

ObjectClassKind ::= ENUMERATED {
        abstract    (0),
        structural  (1),
        auxiliary   (2) }


-- object classes --

top OBJECT-CLASS ::= {
        KIND                abstract
        MUST CONTAIN        { objectClass }
        ID                  id-oc-top }

alias OBJECT-CLASS ::= {
        SUBCLASS OF         { top }
        MUST CONTAIN        { aliasedEntryName }
        ID                  id-oc-alias }

parent OBJECT-CLASS ::= {
        KIND                abstract
        ID                  id-oc-parent }

child OBJECT-CLASS ::= {
        KIND                auxiliary
        ID                  id-oc-child }


-- ATTRIBUTE information object class specification --

ATTRIBUTE ::= CLASS {
        &derivation                     ATTRIBUTE OPTIONAL,
        &Type                           OPTIONAL, -- either &Type or &derivation required --
        &equality-match                 MATCHING-RULE OPTIONAL,
        &ordering-match                 MATCHING-RULE OPTIONAL,
        &substrings-match               MATCHING-RULE OPTIONAL,
        &single-valued                  BOOLEAN DEFAULT FALSE,
        &collective                     BOOLEAN DEFAULT FALSE,
        -- operational extensions --
        &no-user-modification           BOOLEAN DEFAULT FALSE,
        &usage                          AttributeUsage DEFAULT userApplications,
        &id                             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        [ SUBTYPE OF                    &derivation ]
        [ WITH SYNTAX                   &Type ]
        [ EQUALITY MATCHING RULE        &equality-match ]
        [ ORDERING MATCHING RULE        &ordering-match ]
        [ SUBSTRINGS MATCHING RULE      &substrings-match ]
        [ SINGLE VALUE                  &single-valued ]
```

```
        [ COLLECTIVE                    &collective ]
        [ NO USER MODIFICATION          &no-user-modification ]
        [ USAGE                         &usage ]
        ID                              &id }

AttributeUsage ::= ENUMERATED {
        userApplications        (0),
        directoryOperation      (1),
        distributedOperation    (2),
        dSAOperation            (3) }


-- attributes --

objectClass ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        ID                              id-at-objectClass }

aliasedEntryName ATTRIBUTE ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        ID                              id-at-aliasedEntryName }


-- MATCHING-RULE information object class specification --

MATCHING-RULE ::= CLASS {
        &ParentMatchingRules            MATCHING-RULE       OPTIONAL,
        &AssertionType                                      OPTIONAL,
        &uniqueMatchIndicator           ATTRIBUTE           OPTIONAL,
        &id                             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        [ PARENT                        &ParentMatchingRules ]
        [ SYNTAX                        &AssertionType ]
        [ UNIQUE-MATCH-INDICATOR        &uniqueMatchIndicator ]
        ID                              &id }


-- matching rules --

objectIdentifierMatch MATCHING-RULE ::= {
        SYNTAX    OBJECT IDENTIFIER
        ID        id-mr-objectIdentifierMatch }

distinguishedNameMatch MATCHING-RULE ::= {
        SYNTAX    DistinguishedName
        ID        id-mr-distinguishedNameMatch }

MAPPING-BASED-MATCHING
        { SelectedBy, BOOLEAN:combinable, MappingResult, OBJECT IDENTIFIER:matchingRule } ::=
CLASS {
        &selectBy               SelectedBy          OPTIONAL,
        &ApplicableTo           ATTRIBUTE,
        &subtypesIncluded       BOOLEAN             DEFAULT TRUE,
        &combinable             BOOLEAN             (combinable),
        &mappingResults         MappingResult       OPTIONAL,
        &userControl            BOOLEAN             DEFAULT FALSE,
        &exclusive              BOOLEAN             DEFAULT TRUE,
        &matching-rule          MATCHING-RULE       (matchingRule),
        &id                     OBJECT IDENTIFIER   UNIQUE }
WITH SYNTAX {
        [ SELECT BY             &selectBy ]
        APPLICABLE TO           &ApplicableTo
        [ SUBTYPES INCLUDED     &subtypesIncluded ]
        COMBINABLE              &combinable
        [ MAPPING RESULTS       &mappingResults ]
        [ USER CONTROL          &userControl ]
```

```
        [ EXCLUSIVE                 &exclusive ]
        MATCHING RULE               &matching-rule
        ID                          &id }


-- NAME-FORM information object class specification --

NAME-FORM ::= CLASS {
        &namedObjectClass     OBJECT-CLASS,
        &MandatoryAttributes  ATTRIBUTE,
        &OptionalAttributes   ATTRIBUTE OPTIONAL,
        &id                   OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        NAMES                 &namedObjectClass
        WITH ATTRIBUTES       &MandatoryAttributes
        [ AND OPTIONALLY      &OptionalAttributes ]
        ID                    &id }

-- STRUCTURE-RULE class and DIT structure rule data types --

STRUCTURE-RULE ::= CLASS {
        &nameForm             NAME-FORM,
        &SuperiorStructureRules  STRUCTURE-RULE OPTIONAL,
        &id                   RuleIdentifier }
WITH SYNTAX {
        NAME FORM             &nameForm
        [ SUPERIOR RULES      &SuperiorStructureRules ]
        ID                    &id }

DITStructureRule ::= SEQUENCE {
        ruleIdentifier        RuleIdentifier ,
                              -- shall be unique within the scope of the subschema
        nameForm              NAME-FORM.&id,
        superiorStructureRules  SET SIZE (1..MAX) OF RuleIdentifier OPTIONAL }

RuleIdentifier ::= INTEGER


-- CONTENT-RULE class and DIT content rule data types --

CONTENT-RULE ::= CLASS {
        &structuralClass              OBJECT-CLASS    UNIQUE,
        &Auxiliaries                  OBJECT-CLASS    OPTIONAL,
        &Mandatory                    ATTRIBUTE       OPTIONAL,
        &Optional                     ATTRIBUTE       OPTIONAL,
        &Precluded                    ATTRIBUTE       OPTIONAL }
WITH SYNTAX {
        STRUCTURAL OBJECT-CLASS       &structuralClass
        [ AUXILIARY OBJECT-CLASSES    &Auxiliaries ]
        [ MUST CONTAIN                &Mandatory ]
        [ MAY CONTAIN                 &Optional ]
        [ MUST-NOT CONTAIN            &Precluded ] }

DITContentRule ::= SEQUENCE {
        structuralObjectClass  OBJECT-CLASS.&id,
        auxiliaries                 SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
        mandatory           [1]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
        optional            [2]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
        precluded           [3]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL }

CONTEXT ::= CLASS {
        &Type,
        &Assertion      OPTIONAL,
        &id             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        WITH SYNTAX     &Type
        [ ASSERTED AS   &Assertion ]
        ID              &id }
```

```
DITContextUse ::= SEQUENCE {
        attributeType                   ATTRIBUTE.&id,
        mandatoryContexts       [1]     SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL,
        optionalContexts        [2]     SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL }

DIT-CONTEXT-USE-RULE ::= CLASS {
        &attributeType                  ATTRIBUTE.&id   UNIQUE,
        &Mandatory                      CONTEXT         OPTIONAL,
        &Optional                       CONTEXT         OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE                  &attributeType
        [ MANDATORY CONTEXTS            &Mandatory ]
        [ OPTIONAL CONTEXTS             &Optional ] }


-- system schema information objects --
-- object classes --

subentry OBJECT-CLASS ::= {
        SUBCLASS OF     { top }
        KIND            structural
        MUST CONTAIN    { commonName | subtreeSpecification }
        ID              id-sc-subentry }

subentryNameForm NAME-FORM ::= {
        NAMES                   subentry
        WITH ATTRIBUTES         { commonName }
        ID                      id-nf-subentryNameForm }

accessControlSubentry OBJECT-CLASS ::= {
        KIND        auxiliary
        ID          id-sc-accessControlSubentry }

collectiveAttributeSubentry OBJECT-CLASS ::= {
        KIND        auxiliary
        ID          id-sc-collectiveAttributeSubentry }

contextAssertionSubentry OBJECT-CLASS ::= {
        KIND            auxiliary
        MUST CONTAIN    {contextAssertionDefaults}
        ID              id-sc-contextAssertionSubentry }

serviceAdminSubentry OBJECT-CLASS ::= {
        KIND                    auxiliary
        MUST CONTAIN            { searchRules }
        ID                      id-sc-serviceAdminSubentry }


-- attributes --

subtreeSpecification ATTRIBUTE ::= {
        WITH SYNTAX     SubtreeSpecification
        USAGE           directoryOperation
        ID              id-oa-subtreeSpecification }

administrativeRole ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT-CLASS.&id
        EQUALITY MATCHING RULE          objectIdentifierMatch
        USAGE                           directoryOperation
        ID                              id-oa-administrativeRole }

createTimestamp ATTRIBUTE ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- as per 41.3 b) or c) of ITU-T Rec. X.680 | ISO/IEC 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-createTimestamp }
```

```
modifyTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- as per 41.3 b) or c) of ITU-T Rec. X.680 | ISO/IEC 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-modifyTimestamp }

subschemaTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- as per 41.3 b) or c) of ITU-T Rec. X.680 | ISO/IEC 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-subschemaTimestamp }

creatorsName  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-creatorsName }

modifiersName  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-modifiersName }

subschemaSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-subschemaSubentryList }

accessControlSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-accessControlSubentryList }

collectiveAttributeSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-collectiveAttributeSubentryList }

contextDefaultSubentryList  ATTRIBUTE  ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-contextDefaultSubentryList }
```

```
serviceAdminSubentryList  ATTRIBUTE  ::=  {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-serviceAdminSubentryList }

hasSubordinates ATTRIBUTE ::= {
        WITH SYNTAX                     BOOLEAN
        EQUALITY MATCHING RULE          booleanMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hasSubordinates }

collectiveExclusions  ATTRIBUTE  ::=  {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        USAGE                           directoryOperation
        ID                              id-oa-collectiveExclusions }

contextAssertionDefaults  ATTRIBUTE  ::=  {
        WITH SYNTAX                     TypeAndContextAssertion
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-oa-contextAssertionDefault }

searchRules  ATTRIBUTE ::= {
        WITH SYNTAX                SearchRuleDescription
        EQUALITY MATCHING RULE     integerFirstComponentMatch
        USAGE                      directoryOperation
        ID                         id-oa-searchRules }

SearchRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF              SearchRule,
        name                [28]   SET SIZE (1 .. MAX) OF DirectoryString { ub-search } OPTIONAL,
        description         [29]   DirectoryString { ub-search } OPTIONAL }

hierarchyLevel ATTRIBUTE  ::=  {
        WITH SYNTAX                     HierarchyLevel
        EQUALITY MATCHING RULE          integerMatch
        ORDERING MATCHING RULE          integerOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyLevel }

HierarchyLevel  ::=  INTEGER

hierarchyBelow ATTRIBUTE  ::=  {
        WITH SYNTAX                     HierarchyBelow
        EQUALITY MATCHING RULE          booleanMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyBelow }

HierarchyBelow  ::=  BOOLEAN

hierarchyParent ATTRIBUTE  ::=  {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameMatch
        SINGLE VALUE                    TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyParent }


-- object identifier assignments --
```

```
-- object classes --

id-oc-top                              OBJECT IDENTIFIER  ::=  {id-oc 0}
id-oc-alias                            OBJECT IDENTIFIER  ::=  {id-oc 1}
id-oc-parent                           OBJECT IDENTIFIER  ::=  {id-oc 28}
id-oc-child                            OBJECT IDENTIFIER  ::=  {id-oc 29}

-- attributes --

id-at-objectClass                      OBJECT IDENTIFIER  ::=  {id-at 0}
id-at-aliasedEntryName                 OBJECT IDENTIFIER  ::=  {id-at 1}

-- matching rules --

id-mr-objectIdentifierMatch            OBJECT IDENTIFIER  ::=  {id-mr 0}
id-mr-distinguishedNameMatch           OBJECT IDENTIFIER  ::=  {id-mr 1}


-- operational attributes --

id-oa-excludeAllCollectiveAttributes   OBJECT IDENTIFIER  ::=  {id-oa 0}
id-oa-createTimestamp                  OBJECT IDENTIFIER  ::=  {id-oa 1}
id-oa-modifyTimestamp                  OBJECT IDENTIFIER  ::=  {id-oa 2}
id-oa-creatorsName                     OBJECT IDENTIFIER  ::=  {id-oa 3}
id-oa-modifiersName                    OBJECT IDENTIFIER  ::=  {id-oa 4}
id-oa-administrativeRole               OBJECT IDENTIFIER  ::=  {id-oa 5}
id-oa-subtreeSpecification             OBJECT IDENTIFIER  ::=  {id-oa 6}
id-oa-collectiveExclusions             OBJECT IDENTIFIER  ::=  {id-oa 7}
id-oa-subschemaTimestamp               OBJECT IDENTIFIER  ::=  {id-oa 8}
id-oa-hasSubordinates                  OBJECT IDENTIFIER  ::=  {id-oa 9}
id-oa-subschemaSubentryList            OBJECT IDENTIFIER  ::=  {id-oa 10}
id-oa-accessControlSubentryList        OBJECT IDENTIFIER  ::=  {id-oa 11}
id-oa-collectiveAttributeSubentryList  OBJECT IDENTIFIER  ::=  {id-oa 12}
id-oa-contextDefaultSubentryList       OBJECT IDENTIFIER  ::=  {id-oa 13}
id-oa-contextAssertionDefault          OBJECT IDENTIFIER  ::=  {id-oa 14}
id-oa-serviceAdminSubentryList         OBJECT IDENTIFIER  ::=  {id-oa 15}
id-oa-searchRules                      OBJECT IDENTIFIER  ::=  {id-oa 16}
id-oa-hierarchyLevel                   OBJECT IDENTIFIER  ::=  {id-oa 17}
id-oa-hierarchyBelow                   OBJECT IDENTIFIER  ::=  {id-oa 18}
id-oa-hierarchyParent                  OBJECT IDENTIFIER  ::=  {id-oa 19}


-- subentry classes --

id-sc-subentry                         OBJECT IDENTIFIER  ::=  {id-sc 0}
id-sc-accessControlSubentry            OBJECT IDENTIFIER  ::=  {id-sc 1}
id-sc-collectiveAttributeSubentry      OBJECT IDENTIFIER  ::=  {id-sc 2}
id-sc-contextAssertionSubentry         OBJECT IDENTIFIER  ::=  {id-sc 3}
id-sc-serviceAdminSubentry             OBJECT IDENTIFIER  ::=  {id-sc 4}


--  Name forms --

id-nf-subentryNameForm                 OBJECT IDENTIFIER  ::=  {id-nf 16}

-- administrative roles --

id-ar-autonomousArea                   OBJECT IDENTIFIER  ::=  {id-ar 1}
id-ar-accessControlSpecificArea        OBJECT IDENTIFIER  ::=  {id-ar 2}
id-ar-accessControlInnerArea           OBJECT IDENTIFIER  ::=  {id-ar 3}
id-ar-subschemaAdminSpecificArea       OBJECT IDENTIFIER  ::=  {id-ar 4}
id-ar-collectiveAttributeSpecificArea  OBJECT IDENTIFIER  ::=  {id-ar 5}
id-ar-collectiveAttributeInnerArea     OBJECT IDENTIFIER  ::=  {id-ar 6}
id-ar-contextDefaultSpecificArea       OBJECT IDENTIFIER  ::=  {id-ar 7}
id-ar-serviceSpecificArea              OBJECT IDENTIFIER  ::=  {id-ar 8}

END  -- InformationFramework
```

Anexo C

Esquema de administración de subesquema en ASN.1

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo contiene definiciones de tipo, valor y objetos de información para la administración de subesquema según se especifica en la cláusula 15 en forma de un módulo ASN.1, **SchemaAdministration (administración de esquema)**.

```
SchemaAdministration {joint-iso-itu-t ds(5) module(1) schemaAdministration(23) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

    id-soa, id-soc, informationFramework, selectedAttributeTypes, upperBounds
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

    ATTRIBUTE, AttributeUsage, CONTEXT, DITContentRule, DITStructureRule, MATCHING-RULE,
    NAME-FORM, OBJECT-CLASS, ObjectClassKind, objectIdentifierMatch
        FROM InformationFramework informationFramework

-- from ITU-T Rec. X.520 | ISO/IEC 9594-6

    DirectoryString {}, integerFirstComponentMatch, integerMatch,
    objectIdentifierFirstComponentMatch
        FROM SelectedAttributeTypes selectedAttributeTypes

    ub-schema
        FROM UpperBounds upperBounds;


-- types --

DITStructureRuleDescription  ::=  SEQUENCE {
    COMPONENTS OF       DITStructureRule,
    name                [1]   SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
    description                 DirectoryString { ub-schema } OPTIONAL,
    obsolete                    BOOLEAN DEFAULT FALSE }

DITContentRuleDescription  ::=  SEQUENCE {
    COMPONENTS OF               DITContentRule,
    name                [4]     SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
    description                 DirectoryString { ub-schema }       OPTIONAL,
    obsolete                    BOOLEAN DEFAULT FALSE }

MatchingRuleDescription  ::=  SEQUENCE {
    identifier                  MATCHING-RULE.&id,
    name                        SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
    description                 DirectoryString { ub-schema }       OPTIONAL,
    obsolete                    BOOLEAN                             DEFAULT FALSE,
    information         [0]     DirectoryString { ub-schema }       OPTIONAL }
                        -- describes the ASN.1 syntax
```

```
AttributeTypeDescription ::= SEQUENCE {
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information     [0]     AttributeTypeInformation }

AttributeTypeInformation ::= SEQUENCE {
        derivation              [0]     ATTRIBUTE.&id OPTIONAL,
        equalityMatch           [1]     MATCHING-RULE.&id           OPTIONAL,
        orderingMatch           [2]     MATCHING-RULE.&id           OPTIONAL,
        substringsMatch         [3]     MATCHING-RULE.&id           OPTIONAL,
        attributeSyntax         [4]     DirectoryString { ub-schema }OPTIONAL,
        multi-valued            [5]     BOOLEAN                     DEFAULT TRUE,
        collective              [6]     BOOLEAN                     DEFAULT FALSE,
        userModifiable          [7]     BOOLEAN                     DEFAULT TRUE,
        application                     AttributeUsage              DEFAULT  userApplications }

ObjectClassDescription ::= SEQUENCE {
        identifier              OBJECT-CLASS.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information     [0]     ObjectClassInformation }

ObjectClassInformation ::= SEQUENCE {
        subclassOf              SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
        kind                    ObjectClassKind                          DEFAULT structural,
        mandatories     [3]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
        optionals       [4]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL }

NameFormDescription ::= SEQUENCE {
        identifier              NAME-FORM.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description             DirectoryString { ub-schema } OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE,
        information     [0]     NameFormInformation }

NameFormInformation ::= SEQUENCE {
        subordinate             OBJECT-CLASS.&id,
        namingMandatories       SET OF ATTRIBUTE.&id,
        namingOptionals         SET SIZE (1..MAX) OF ATTRIBUTE.&id OPTIONAL }

MatchingRuleUseDescription ::= SEQUENCE {
        identifier              MATCHING-RULE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information     [0]     SET OF ATTRIBUTE.&id }

ContextDescription ::= SEQUENCE {
        identifier              CONTEXT.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString {ub-schema}   OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information     [0]     ContextInformation }

ContextInformation ::= SEQUENCE {
        syntax                  DirectoryString { ub-schema } ,
        assertionSyntax         DirectoryString { ub-schema } OPTIONAL }

DITContextUseDescription ::= SEQUENCE {
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description             DirectoryString { ub-schema }       OPTIONAL,
        obsolete                BOOLEAN                             DEFAULT FALSE,
        information     [0]     DITContextUseInformation }
```

```
DITContextUseInformation ::= SEQUENCE {
        mandatoryContexts       [1]     SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL,
        optionalContexts        [2]     SET SIZE (1..MAX) OF CONTEXT.&id  OPTIONAL }


-- object classes --

subschema  OBJECT-CLASS  ::= {
        KIND                    auxiliary
        MAY CONTAIN     {
                                dITStructureRules |
                                nameForms |
                                dITContentRules |
                                objectClasses |
                                attributeTypes |
                                contextTypes |
                                dITContextUse |
                                matchingRules |
                                matchingRuleUse }
        ID                      id-soc-subschema }


-- attributes --

dITStructureRules ATTRIBUTE  ::= {
        WITH SYNTAX                     DITStructureRuleDescription
        EQUALITY MATCHING RULE          integerFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITStructureRule }

dITContentRules  ATTRIBUTE  ::= {
        WITH SYNTAX                     DITContentRuleDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContentRules }

matchingRules  ATTRIBUTE  ::= {
        WITH SYNTAX                     MatchingRuleDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-matchingRules }

attributeTypes ATTRIBUTE ::= {
        WITH SYNTAX                     AttributeTypeDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-attributeTypes }

objectClasses ATTRIBUTE  ::= {
        WITH SYNTAX                     ObjectClassDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-objectClasses }

nameForms  ATTRIBUTE  ::= {
        WITH SYNTAX                     NameFormDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-nameForms }

matchingRuleUse ATTRIBUTE  ::= {
        WITH SYNTAX                     MatchingRuleUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-matchingRuleUse }
```

```
structuralObjectClass ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-soa-structuralObjectClass }

governingStructureRule ATTRIBUTE  ::= {
        WITH SYNTAX                     INTEGER
        EQUALITY MATCHING RULE          integerMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-soa-governingStructureRule }

contextTypes ATTRIBUTE  ::= {
        WITH SYNTAX                     ContextDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-contextTypes }

dITContextUse ATTRIBUTE  ::= {
        WITH SYNTAX                     DITContextUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContextUse }


-- object identifier assignments --

-- schema object classes --

id-soc-subschema                OBJECT IDENTIFIER   ::=   {id-soc 1}

-- schema operational attributes --

id-soa-dITStructureRule         OBJECT IDENTIFIER   ::=   {id-soa 1}
id-soa-dITContentRules          OBJECT IDENTIFIER   ::=   {id-soa 2}
id-soa-matchingRules            OBJECT IDENTIFIER   ::=   {id-soa 4}
id-soa-attributeTypes           OBJECT IDENTIFIER   ::=   {id-soa 5}
id-soa-objectClasses            OBJECT IDENTIFIER   ::=   {id-soa 6}
id-soa-nameForms                OBJECT IDENTIFIER   ::=   {id-soa 7}
id-soa-matchingRuleUse          OBJECT IDENTIFIER   ::=   {id-soa 8}
id-soa-structuralObjectClass    OBJECT IDENTIFIER   ::=   {id-soa 9}
id-soa-governingStructureRule   OBJECT IDENTIFIER   ::=   {id-soa 10}
id-soa-contextTypes             OBJECT IDENTIFIER   ::=   {id-soa 11}
id-soa-dITContextUse            OBJECT IDENTIFIER   ::=   {id-soa 12}

END  -- SchemaAdministration
```

Anexo D

Esquema de administración de servicio en ASN.1

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo contiene definiciones de tipo, valor y objetos de información para el subesquema de administración definido en la cláusula 16 en forma de un módulo ASN.1, **ServiceAdministration (administración de servicio)**.

```
ServiceAdministration {joint-iso-itu-t ds(5) module(1) serviceAdministration(33) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

    directoryAbstractService, informationFramework
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

    ATTRIBUTE, AttributeType, CONTEXT, MATCHING-RULE, OBJECT-CLASS,
    SupportedAttributes, SupportedContexts
        FROM InformationFramework informationFramework

-- from ITU-T Rec. X.511 | ISO/IEC 9594-3

    FamilyGrouping, FamilyReturn, HierarchySelections, SearchControlOptions,
    ServiceControlOptions
        FROM DirectoryAbstractService directoryAbstractService   ;


-- types --

SearchRule ::= SEQUENCE {
    COMPONENTS OF       SearchRuleId,
    serviceType         [1]  OBJECT IDENTIFIER                          OPTIONAL,
    userClass           [2]  INTEGER                                    OPTIONAL,
    inputAttributeTypes [3]  SEQUENCE SIZE (0..MAX) OF RequestAttribute OPTIONAL,
    attributeCombination [4] AttributeCombination                       DEFAULT and : { },
    outputAttributeTypes [5] SEQUENCE SIZE (1..MAX) OF ResultAttribute  OPTIONAL,
    defaultControls     [6]  ControlOptions                             OPTIONAL,
    mandatoryControls   [7]  ControlOptions                             OPTIONAL,
    searchRuleControls  [8]  ControlOptions                             OPTIONAL,
    familyGrouping      [9]  FamilyGrouping                             OPTIONAL,
    familyReturn        [10] FamilyReturn                               OPTIONAL,
    relaxation          [11] RelaxationPolicy                           OPTIONAL,
    additionalControl   [12] SEQUENCE SIZE (1..MAX) OF AttributeType    OPTIONAL,
    allowedSubset       [13] AllowedSubset                              DEFAULT '111'B,
    imposedSubset       [14] ImposedSubset                              OPTIONAL,
    entryLimit          [15] EntryLimit                                 OPTIONAL }

SearchRuleId ::= SEQUENCE {
    id                      INTEGER,
    dmdId               [0] OBJECT IDENTIFIER }

AllowedSubset ::= BIT STRING { baseObject (0), oneLevel (1), wholeSubtree (2) }

ImposedSubset ::= ENUMERATED { baseObject (0), oneLevel (1), wholeSubtree (2) }
```

```
RequestAttribute ::= SEQUENCE {
        attributeType                   ATTRIBUTE.&id ({ SupportedAttributes }),
        includeSubtypes         [0]     BOOLEAN                                         DEFAULT FALSE,
        selectedValues          [1]     SEQUENCE SIZE (0..MAX) OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType })     OPTIONAL,
        defaultValues           [2]     SEQUENCE SIZE (0..MAX) OF SEQUENCE {
                entryType                       OBJECT-CLASS.&id  OPTIONAL,
                values                          SEQUENCE OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType }) }   OPTIONAL,
        contexts                [3]     SEQUENCE SIZE (0..MAX) OF ContextProfile        OPTIONAL,
        contextCombination      [4]     ContextCombination                              DEFAULT and : { },
        matchingUse             [5]     SEQUENCE SIZE (1..MAX) OF MatchingUse           OPTIONAL }

ContextProfile ::= SEQUENCE {
        contextType         CONTEXT.&id({SupportedContexts}),
        contextValue        SEQUENCE SIZE (1..MAX) OF CONTEXT.&Assertion
                                ({SupportedContexts}{@contextType})  OPTIONAL }

ContextCombination ::= CHOICE {
        context                 [0]     CONTEXT.&id({SupportedContexts}),
        and                     [1]     SEQUENCE OF ContextCombination,
        or                      [2]     SEQUENCE OF ContextCombination,
        not                     [3]     ContextCombination }

MatchingUse ::= SEQUENCE {
        restrictionType         MATCHING-RESTRICTION.&id ({SupportedMatchingRestrictions}),
        restrictionValue        MATCHING-RESTRICTION.&Restriction
                                        ({SupportedMatchingRestrictions}{@restrictionType}) }

-- Definition of the following information object set is deferred, perhaps to standardized
-- profiles or to protocol implementation conformance statements. The set is required to
-- specify a table constraint on the components of SupportedMatchingRestrictions

SupportedMatchingRestrictions  MATCHING-RESTRICTION  ::= { ... }

AttributeCombination ::= CHOICE {
        attribute               [0]     AttributeType,
        and                     [1]     SEQUENCE OF AttributeCombination,
        or                      [2]     SEQUENCE OF AttributeCombination,
        not                     [3]     AttributeCombination }

ResultAttribute ::= SEQUENCE {
        attributeType           ATTRIBUTE.&id ({ SupportedAttributes }),
        outputValues            CHOICE {
                selectedValues          SEQUENCE OF ATTRIBUTE.&Type
                                                ({ SupportedAttributes }{ @attributeType }),
                matchedValuesOnly       NULL } OPTIONAL,
        contexts                [0] SEQUENCE SIZE (1..MAX) OF ContextProfile  OPTIONAL }

ControlOptions ::= SEQUENCE {
        serviceControls         [0]     ServiceControlOptions  DEFAULT { },
        searchOptions           [1]     SearchControlOptions  DEFAULT { searchAliases },
        hierarchyOptions        [2]     HierarchySelections     OPTIONAL }

EntryLimit ::= SEQUENCE {
        default             INTEGER,
        max                 INTEGER }

RelaxationPolicy ::= SEQUENCE {
        basic               [0]     MRMapping DEFAULT { },
        tightenings         [1]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        relaxations         [2]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        maximum             [3]     INTEGER OPTIONAL,    -- mandatory if tightenings is present
        minimum             [4]     INTEGER DEFAULT 1 }

MRMapping ::= SEQUENCE {
        mapping             [0]     SEQUENCE SIZE (1..MAX) OF Mapping           OPTIONAL,
        substitution        [1]     SEQUENCE SIZE (1..MAX) OF MRSubstitution    OPTIONAL }
```

```
Mapping ::= SEQUENCE {
        mappingFunction         OBJECT IDENTIFIER (CONSTRAINED BY { -- shall be an
                        -- object identifier of a mapping-based matching algorithm -- } ),
        level                   INTEGER  DEFAULT 0 }

MRSubstitution ::= SEQUENCE {
        attribute                       AttributeType,
        oldMatchingRule         [0]     MATCHING-RULE.&id  OPTIONAL,
        newMatchingRule         [1]     MATCHING-RULE.&id  OPTIONAL }


-- ASN.1 information object classes --

SEARCH-RULE ::= CLASS {
        &dmdId                          OBJECT IDENTIFIER,
        &serviceType                    OBJECT IDENTIFIER                       OPTIONAL,
        &userClass                      INTEGER                                 OPTIONAL,
        &InputAttributeTypes            REQUEST-ATTRIBUTE                       OPTIONAL,
        &combination                    AttributeCombination                    OPTIONAL,
        &OutputAttributeTypes           RESULT-ATTRIBUTE                        OPTIONAL,
        &defaultControls                ControlOptions                          OPTIONAL,
        &mandatoryControls              ControlOptions                          OPTIONAL,
        &searchRuleControls             ControlOptions                          OPTIONAL,
        &familyGrouping                 FamilyGrouping                          OPTIONAL,
        &familyReturn                   FamilyReturn                            OPTIONAL,
        &additionalControl              AttributeType                           OPTIONAL,
        &relaxation                     RelaxationPolicy                        OPTIONAL,
        &allowedSubset                  AllowedSubset                           DEFAULT '111'B,
        &imposedSubset                  ImposedSubset                           OPTIONAL,
        &entryLimit                     EntryLimit                              OPTIONAL,
        &id                             INTEGER UNIQUE }
WITH SYNTAX {
        DMD ID                          &dmdId
        [ SERVICE-TYPE          &serviceType ]
        [ USER-CLASS                    &userClass ]
        [ INPUT ATTRIBUTES              &InputAttributeTypes ]
        [ COMBINATION                   &combination ]
        [ OUTPUT ATTRIBUTES&OutputAttributeTypes ]
        [ DEFAULT CONTROL               &defaultControls ]
        [ MANDATORY CONTROL     &mandatoryControls ]
        [ SEARCH-RULE CONTROL   &searchRuleControls ]
        [ FAMILY-GROUPING               &familyGrouping ]
        [ FAMILY-RETURN                 &familyReturn ]
        [ ADDITIONAL CONTROL    &additionalControl ]
        [ RELAXATION                    &relaxation ]
        [ ALLOWED SUBSET                &allowedSubset ]
        [ IMPOSED SUBSET                &imposedSubset ]
        [ ENTRY LIMIT                   &entryLimit ]
        ID                              &id }

REQUEST-ATTRIBUTE ::= CLASS {
        &attributeType                  ATTRIBUTE.&id,
        &SelectedValues                 ATTRIBUTE.&Type                                 OPTIONAL,
        &DefaultValues                  SEQUENCE {
                                                entryType  OBJECT-CLASS.&id             OPTIONAL,
                                                values     SEQUENCE OF ATTRIBUTE.&Type } OPTIONAL,
        &contexts                       SEQUENCE OF ContextProfile                      OPTIONAL,
        &contextCombination             ContextCombination                              OPTIONAL,
        &MatchingUse                    MatchingUse                                     OPTIONAL,
        &includeSubtypes                BOOLEAN                                         DEFAULT  FALSE
}
WITH SYNTAX  {
        ATTRIBUTE TYPE                  &attributeType
        [ SELECTED VALUES               &SelectedValues ]
        [ DEFAULT VALUES                &DefaultValues ]
        [ CONTEXTS                      &contexts ]
        [ CONTEXT COMBINATION   &contextCombination ]
        [ MATCHING USE                  &MatchingUse ]
        [ INCLUDE SUBTYPES              &includeSubtypes ] }
```

```
RESULT-ATTRIBUTE  ::=  CLASS {
        &attributeType                  ATTRIBUTE.&id,
        &outputValues                   CHOICE {
                selectedValues          SEQUENCE OF ATTRIBUTE.&Type,
                matchedValuesOnly       NULL }                          OPTIONAL,
        &contexts                       ContextProfile                  OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE                  &attributeType
        [ OUTPUT VALUES                 &outputValues ]
        [ CONTEXTS                      &contexts ] }

MATCHING-RESTRICTION ::= CLASS {
        &Restriction,
        &Rules              MATCHING-RULE.&id,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
        RESTRICTION         &Restriction
        RULES               &Rules
        ID                  &id }

END  -- ServiceAdministration
```

Anexo E

Control de acceso básico en ASN.1

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo contiene un resumen de todas las definiciones de tipo y valores ASN.1 para el control de acceso básico. Las definiciones forman el módulo ASN.1 **BasicAccessControl (control de acceso básico)**.

```
BasicAccessControl {joint-iso-itu-t ds(5) module(1) basicAccessControl(24) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

    directoryAbstractService, id-aca, id-acScheme, informationFramework,
    selectedAttributeTypes, upperBounds
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

    ATTRIBUTE, AttributeType, ContextAssertion, DistinguishedName, MATCHING-RULE,
    objectIdentifierMatch, Refinement, SubtreeSpecification, SupportedAttributes
        FROM InformationFramework informationFramework

-- from ITU-T Rec. X.511 | ISO/IEC 9594-3

    Filter
        FROM DirectoryAbstractService directoryAbstractService

-- from ITU-T Rec. X.520 | ISO/IEC 9594-6

    DirectoryString{}, directoryStringFirstComponentMatch, NameAndOptionalUID,
    UniqueIdentifier
        FROM SelectedAttributeTypes selectedAttributeTypes

    ub-tag
        FROM UpperBounds upperBounds ;

-- types --

ACIItem ::= SEQUENCE {
    identificationTag           DirectoryString { ub-tag },
    precedence                  Precedence,
    authenticationLevel         AuthenticationLevel,
    itemOrUserFirst             CHOICE {
        itemFirst           [0] SEQUENCE {
            protectedItems      ProtectedItems,
            itemPermissions     SET OF ItemPermission },
        userFirst           [1] SEQUENCE {
            userClasses         UserClasses,
            userPermissions     SET OF UserPermission } } }

Precedence ::= INTEGER (0..255)

ProtectedItems ::= SEQUENCE {
    entry                       [0] NULL                                OPTIONAL,
    allUserAttributeTypes       [1] NULL                                OPTIONAL,
    attributeType               [2] SET SIZE (1..MAX) OF AttributeType  OPTIONAL,
    allAttributeValues          [3] SET SIZE (1..MAX) OF AttributeType  OPTIONAL,
```

```
        allUserAttributeTypesAndValues  [4]  NULL                                    OPTIONAL,
        attributeValue                  [5]  SET SIZE (1..MAX) OF AttributeTypeAndValue OPTIONAL,
        selfValue                       [6]  SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
        rangeOfValues                   [7]  Filter                                  OPTIONAL,
        maxValueCount                   [8]  SET SIZE (1..MAX) OF MaxValueCount      OPTIONAL,
        maxImmSub                       [9]  INTEGER                                 OPTIONAL,
        restrictedBy                    [10] SET SIZE (1..MAX) OF RestrictedValue    OPTIONAL,
        contexts                        [11] SET SIZE (1..MAX) OF ContextAssertion   OPTIONAL,
        classes                         [12] Refinement                              OPTIONAL }

MaxValueCount ::= SEQUENCE {
        type            AttributeType,
        maxCount        INTEGER }

RestrictedValue ::= SEQUENCE {
        type            AttributeType,
        valuesIn        AttributeType }

UserClasses ::= SEQUENCE {
        allUsers    [0] NULL                                    OPTIONAL,
        thisEntry   [1] NULL                                    OPTIONAL,
        name        [2] SET SIZE (1..MAX) OF NameAndOptionalUID OPTIONAL,
        userGroup   [3] SET SIZE (1..MAX) OF NameAndOptionalUID OPTIONAL,
                        -- dn component shall be the name of an
                        -- entry of GroupOfUniqueNames
        subtree     [4] SET SIZE (1..MAX) OF SubtreeSpecification OPTIONAL }

ItemPermission ::= SEQUENCE {
        precedence          Precedence OPTIONAL,
                        -- defaults to precedence in ACIItem
        userClasses         UserClasses,
        grantsAndDenials    GrantsAndDenials }

UserPermission ::= SEQUENCE {
        precedence          Precedence OPTIONAL,
                        -- defaults to precedence in ACIItem
        protectedItems      ProtectedItems,
        grantsAndDenials    GrantsAndDenials }

AuthenticationLevel ::= CHOICE {
        basicLevelsSEQUENCE {
            level           ENUMERATED { none (0), simple (1), strong (2) },
            localQualifier  INTEGER OPTIONAL,
            signed          BOOLEAN DEFAULT FALSE },
        other           EXTERNAL }

GrantsAndDenials ::= BIT STRING {
        -- permissions that may be used in conjunction
        -- with any component of ProtectedItems
        grantAdd            (0),
        denyAdd             (1),
        grantDiscloseOnError (2),
        denyDiscloseOnError (3),
        grantRead           (4),
        denyRead            (5),
        grantRemove         (6),
        denyRemove          (7),
        -- permissions that may be used only in conjunction
        -- with the entry component
        grantBrowse         (8),
        denyBrowse          (9),
        grantExport         (10),
        denyExport          (11),
        grantImport         (12),
        denyImport          (13),
        grantModify         (14),
        denyModify          (15),
        grantRename         (16),
        denyRename          (17),
```

```
        grantReturnDN           (18),
        denyReturnDN            (19),
        -- permissions that may be used in conjunction
        -- with any component, except entry, of ProtectedItems
        grantCompare            (20),
        denyCompare             (21),
        grantFilterMatch        (22),
        denyFilterMatch         (23),
        grantInvoke             (24),
        denyInvoke              (25) }

AttributeTypeAndValue ::= SEQUENCE {
        type        ATTRIBUTE.&id ({SupportedAttributes}),
        value       ATTRIBUTE.&Type({SupportedAttributes}{@type}) }

-- attributes --

accessControlScheme ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        USAGE                           directoryOperation
        ID                              id-aca-accessControlScheme }

prescriptiveACI ATTRIBUTE ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-prescriptiveACI }

entryACI ATTRIBUTE ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-entryACI }

subentryACI ATTRIBUTE ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-subentryACI }

-- object identifier assignments --

-- attributes --

id-aca-accessControlScheme          OBJECT IDENTIFIER   ::=   { id-aca 1 }
id-aca-prescriptiveACI              OBJECT IDENTIFIER   ::=   { id-aca 4 }
id-aca-entryACI                     OBJECT IDENTIFIER   ::=   { id-aca 5 }
id-aca-subentryACI                  OBJECT IDENTIFIER   ::=   { id-aca 6 }

-- access control schemes –

basicAccessControlScheme            OBJECT IDENTIFIER   ::=   { id-acScheme 1 }
simplifiedAccessControlScheme       OBJECT IDENTIFIER   ::=   { id-acScheme 2 }
rule-based-access-control           OBJECT IDENTIFIER   ::=   { id-acScheme 3 }
rule-and-basic-access-control       OBJECT IDENTIFIER   ::=   { id-acScheme 4 }
rule-and-simple-access-control      OBJECT IDENTIFIER   ::=   { id-acScheme 5 }

END -- BasicAccessControl
```

Anexo F

Tipos de atributos operacionales de DSA en ASN.1

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo comprende todas las definiciones de tipos y valores ASN.1 contenidas en las cláusulas 23 a 24 en forma de un módulo ASN.1, **DSAOperationalAttributeTypes (tipos de atributos operacionales de DSA)**.

```
DSAOperationalAttributeTypes {joint-iso-itu-t ds(5) module(1) dsaOperationalAttributeTypes(22) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

    distributedOperations, id-doa, id-kmr, informationFramework, opBindingManagement,
    selectedAttributeTypes, upperBounds
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4 }

    ATTRIBUTE, MATCHING-RULE, Name
        FROM InformationFramework informationFramework

    OperationalBindingID
        FROM OperationalBindingManagement opBindingManagement

-- from ITU-T Rec. X.518 | ISO/IEC 9594-4

    AccessPoint, MasterAndShadowAccessPoints
        FROM DistributedOperations distributedOperations

-- from ITU-T Rec. X.520 | ISO/IEC 9594-6

    bitStringMatch
        FROM SelectedAttributeTypes selectedAttributeTypes  ;

-- data types --

DSEType  ::=  BIT STRING {
    root            (0),    -- root DSE --
    glue            (1),    -- represents knowledge of a name only --
    cp              (2),    -- context prefix --
    entry           (3),    -- object entry --
    alias           (4),    -- alias entry --
    subr            (5),    -- subordinate reference --
    nssr            (6),    -- non-specific subordinate reference --
    supr            (7),    -- superior reference --
    xr              (8),    -- cross reference --
    admPoint        (9),    -- administrative point --
    subentry        (10),   -- subentry --
    shadow          (11),   -- shadow copy --
    immSupr         (13),   -- immediate superior reference --
    rhob            (14),   -- rhob information --
    sa              (15),   -- subordinate reference to alias entry --
    dsSubentry      (16),   -- DSA Specific subentry --
    familyMember    (17) }  -- family member --
```

```
SupplierOrConsumer ::= SET {
        COMPONENTS OF           AccessPoint,        -- supplier or consumer --
        agreementID         [3] OperationalBindingID }

SupplierInformation ::= SET {
        COMPONENTS OF           SupplierOrConsumer, -- supplier --
        supplier-is-master  [4] BOOLEAN DEFAULT TRUE,
        non-supplying-master [5] AccessPoint OPTIONAL }

ConsumerInformation ::= SupplierOrConsumer      -- consumer --

SupplierAndConsumers ::= SET {
        COMPONENTS OF           AccessPoint,        -- supplier --
        consumers           [3] SET OF AccessPoint }


-- attribute types --

dseType ATTRIBUTE ::= {
        WITH SYNTAX                     DSEType
        EQUALITY MATCHING RULE          bitStringMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-dseType }

myAccessPoint ATTRIBUTE ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-myAccessPoint }

superiorKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-superiorKnowledge }

specificKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation
        ID                              id-doa-specificKnowledge }

nonSpecificKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation
        ID                              id-doa-nonSpecificKnowledge }

supplierKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     SupplierInformation
        EQUALITY MATCHING RULE          supplierOrConsumerInformationMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-supplierKnowledge }

consumerKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     ConsumerInformation
        EQUALITY MATCHING RULE          supplierOrConsumerInformationMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-consumerKnowledge }
```

```
secondaryShadows  ATTRIBUTE  ::= {
        WITH SYNTAX                     SupplierAndConsumers
        EQUALITY MATCHING RULE          supplierAndConsumersMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-secondaryShadows }


-- matching rules --

accessPointMatch  MATCHING-RULE  ::= {
        SYNTAX      Name
        ID          id-kmr-accessPointMatch }

masterAndShadowAccessPointsMatch  MATCHING-RULE  ::= {
        SYNTAX      SET OF Name
        ID          id-kmr-masterShadowMatch }

supplierOrConsumerInformationMatch  MATCHING-RULE  ::= {
        SYNTAX      SET {
                ae-title                [0]     Name,
                agreement-identifier    [2]     INTEGER }
        ID          id-kmr-supplierConsumerMatch }

supplierAndConsumersMatch  MATCHING-RULE  ::= {
        SYNTAX      Name
        ID          id-kmr-supplierConsumersMatch }


-- object identifier assignments --

-- dsa operational attributes --

id-doa-dseType                  OBJECT IDENTIFIER   ::=   {id-doa 0}
id-doa-myAccessPoint            OBJECT IDENTIFIER   ::=   {id-doa 1}
id-doa-superiorKnowledge        OBJECT IDENTIFIER   ::=   {id-doa 2}
id-doa-specificKnowledge        OBJECT IDENTIFIER   ::=   {id-doa 3}
id-doa-nonSpecificKnowledge     OBJECT IDENTIFIER   ::=   {id-doa 4}
id-doa-supplierKnowledge        OBJECT IDENTIFIER   ::=   {id-doa 5}
id-doa-consumerKnowledge        OBJECT IDENTIFIER   ::=   {id-doa 6}
id-doa-secondaryShadows         OBJECT IDENTIFIER   ::=   {id-doa 7}

-- knowledge matching rules --

id-kmr-accessPointMatch         OBJECT IDENTIFIER   ::=   {id-kmr 0}
id-kmr-masterShadowMatch        OBJECT IDENTIFIER   ::=   {id-kmr 1}
id-kmr-supplierConsumerMatch    OBJECT IDENTIFIER   ::=   {id-kmr 2}
id-kmr-supplierConsumersMatch   OBJECT IDENTIFIER   ::=   {id-kmr 3}

END  -- DSAOperationalAttributeTypes
```

Anexo G

Gestión de vinculaciones operacionales en ASN.1

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Este anexo comprende todas las definiciones de tipos, valores y clases de objeto de información relativas a las vinculaciones operacionales pertinentes a esta Especificación de directorio en forma del módulo ASN.1 **OperationalBindingManagement (gestión de vinculaciones operacionales)**.

```
OperationalBindingManagement {joint-iso-itu-t ds(5) module(1) opBindingManagement(18) 4}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- The types and values defined in this module are exported for use in the other ASN.1 modules contained
-- within the Directory Specifications, and for the use of other applications which will use them to access
-- Directory services. Other applications may use them for their own purposes, but this will not constrain
-- extensions and modifications needed to maintain or improve the Directory service.

IMPORTS

-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

    directoryAbstractService, directoryShadowAbstractService, distributedOperations, dop,
    enhancedSecurity, hierarchicalOperationalBindings
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4}

    OPTIONALLY-PROTECTED-SEQ
        FROM EnhancedSecurity enhancedSecurity

    hierarchicalOperationalBinding, nonSpecificHierarchicalOperationalBinding
        FROM HierarchicalOperationalBindings hierarchicalOperationalBindings

-- from ITU-T Rec. X.511 | ISO/IEC 9594-3

    CommonResultsSeq, directoryBind, directoryUnbind, securityError, SecurityParameters
        FROM DirectoryAbstractService directoryAbstractService

-- from ITU-T Rec. X.518 | ISO/IEC 9594-4

    AccessPoint
        FROM DistributedOperations distributedOperations

-- from ITU-T Rec. X.519 | ISO/IEC 9594-5

    id-err-operationalBindingError, id-op-establishOperationalBinding,
    id-op-modifyOperationalBinding, id-op-terminateOperationalBinding
        FROM DirectoryOperationalBindingManagementProtocol dop

-- from ITU-T Rec. X.525 | ISO/IEC 9594-9

    shadowOperationalBinding
        FROM DirectoryShadowAbstractService directoryShadowAbstractService

-- from ITU-T Rec. X.880 | ISO/IEC 13712-1

    OPERATION, ERROR
        FROM Remote-Operations-Information-Objects
            {joint-iso-itu-t remote-operations(4) informationObjects(5) version1(0)}

-- from ITU-T Rec. X.881 | ISO/IEC 13712-2

    APPLICATION-CONTEXT
        FROM Remote-Operations-Information-Objects-extensions
            {joint-iso-itu-t remote-operations(4) informationObjects-extensions(8) version1(0)}  ;
```

```
-- bind and unbind --

dSAOperationalBindingManagementBind OPERATION      ::=    directoryBind

dSAOperationalBindingManagementUnbind OPERATION    ::=    directoryUnbind

-- operations, arguments and results --

establishOperationalBinding  OPERATION  ::=  {
      ARGUMENT       EstablishOperationalBindingArgument
      RESULT         EstablishOperationalBindingResult
      ERRORS         {operationalBindingError | securityError}
      CODE           id-op-establishOperationalBinding }

EstablishOperationalBindingArgument  ::= OPTIONALLY-PROTECTED-SEQ  { SEQUENCE {
      bindingType          [0]    OPERATIONAL-BINDING.&id ({OpBindingSet}),
      bindingID            [1]    OperationalBindingID OPTIONAL,
      accessPoint          [2]    AccessPoint,
            -- symmetric, Role A initiates, or Role B initiates --
      initiator CHOICE {
            symmetric              [3]    OPERATIONAL-BINDING.&both.&EstablishParam
                                              ({OpBindingSet}{@bindingType}),
            roleA-initiates        [4]    OPERATIONAL-BINDING.&roleA.&EstablishParam
                                              ({OpBindingSet}{@bindingType}),
            roleB-initiates        [5]    OPERATIONAL-BINDING.&roleB.&EstablishParam
                                              ({OpBindingSet}{@bindingType}) } OPTIONAL,
      agreement            [6]    OPERATIONAL-BINDING.&Agreement
                                        ({OpBindingSet}{@bindingType}),
      valid                [7]    Validity DEFAULT { },
      securityParameters   [8]    SecurityParameters OPTIONAL } }

OperationalBindingID  ::=  SEQUENCE {
      identifier    INTEGER,
      version       INTEGER }

Validity  ::=  SEQUENCE {
      validFrom   [0]    CHOICE {
            now           [0]    NULL,
            time          [1]    Time } DEFAULT now : NULL,
      validUntil  [1]    CHOICE {
            explicitTermination    [0]    NULL,
            time                   [1]    Time } DEFAULT explicitTermination : NULL }

Time  ::=  CHOICE {
      utcTime           UTCTime,
      generalizedTime   GeneralizedTime }

EstablishOperationalBindingResult  ::=  OPTIONALLY-PROTECTED-SEQ  { SEQUENCE {
      bindingType       [0]    OPERATIONAL-BINDING.&id ({OpBindingSet}),
      bindingID         [1]    OperationalBindingID OPTIONAL,
      accessPoint       [2]    AccessPoint,
      -- symmetric, Role A replies , or Role B replies --
      initiator CHOICE {
            symmetric         [3]    OPERATIONAL-BINDING.&both.&EstablishParam
                                     ({OpBindingSet}{@bindingType}),
            roleA-replies     [4]    OPERATIONAL-BINDING.&roleA.&EstablishParam
                                     ({OpBindingSet}{@bindingType}),
            roleB-replies     [5]    OPERATIONAL-BINDING.&roleB.&EstablishParam
                                     ({OpBindingSet}{@bindingType}) } OPTIONAL,
      COMPONENTS OF     CommonResultsSeq } }

modifyOperationalBinding  OPERATION  ::=  {
      ARGUMENT       ModifyOperationalBindingArgument
      RESULT         ModifyOperationalBindingResult
      ERRORS         { operationalBindingError | securityError }
      CODE           id-op-modifyOperationalBinding }
```

```
ModifyOperationalBindingArgument ::= OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType         [0]  OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID           [1]  OperationalBindingID,
        accessPoint         [2]  AccessPoint OPTIONAL,
        -- symmetric, Role A initiates, or Role B initiates --
        initiator CHOICE {
            symmetric       [3]  OPERATIONAL-BINDING.&both.&ModifyParam
                                 ({OpBindingSet}{@bindingType}),
            roleA-initiates [4]  OPERATIONAL-BINDING.&roleA.&ModifyParam
                                 ({OpBindingSet}{@bindingType}),
            roleB-initiates [5]  OPERATIONAL-BINDING.&roleB.&ModifyParam
                                 ({OpBindingSet}{@bindingType}) } OPTIONAL,
        newBindingID        [6]  OperationalBindingID,
        newAgreement        [7]  OPERATIONAL-BINDING.&Agreement
                                 ({OpBindingSet}{@bindingType}) OPTIONAL,
        valid               [8]  Validity OPTIONAL,
        securityParameters
                            [9]  SecurityParameters OPTIONAL } }

ModifyOperationalBindingResult ::= CHOICE {
        null            [0]  NULL,
        protected       [1]  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
            newBindingID          OperationalBindingID,
            bindingType           OPERATIONAL-BINDING.&id ({OpBindingSet}),
            newAgreement          OPERATIONAL-BINDING.&Agreement
                                      ({OpBindingSet}{@bindingType}),
            valid                 Validity OPTIONAL,
            COMPONENTS OF         CommonResultsSeq } } }

terminateOperationalBinding OPERATION ::= {
        ARGUMENT        TerminateOperationalBindingArgument
        RESULT          TerminateOperationalBindingResult
        ERRORS          {operationalBindingError | securityError}
        CODE            id-op-terminateOperationalBinding }

TerminateOperationalBindingArgument ::= OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType             [0]  OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID               [1]  OperationalBindingID,
        -- symmetric, Role A initiates, or Role B initiates --
        initiator CHOICE {
            symmetric           [2]  OPERATIONAL-BINDING.&both.&TerminateParam
                                     ({OpBindingSet}{@bindingType}),
            roleA-initiates     [3]  OPERATIONAL-BINDING.&roleA.&TerminateParam
                                     ({OpBindingSet}{@bindingType}),
            roleB-initiates     [4]  OPERATIONAL-BINDING.&roleB.&TerminateParam
                                     ({OpBindingSet}{@bindingType})} OPTIONAL,
        terminateAt             [5]  Time OPTIONAL,
        securityParameters      [6]  SecurityParameters OPTIONAL } }

TerminateOperationalBindingResult ::= CHOICE {
        null            [0]  NULL,
        protected       [1]  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
            bindingID             OperationalBindingID,
            bindingType           OPERATIONAL-BINDING.&id ({OpBindingSet}),
            terminateAt           GeneralizedTime OPTIONAL,
            COMPONENTS OF         CommonResultsSeq } } }

-- errors and parameters --

operationalBindingError ERROR ::= {
        PARAMETER       OPTIONALLY-PROTECTED-SEQ {
                            OpBindingErrorParam }
        CODE            id-err-operationalBindingError }

OpBindingErrorParam ::= SEQUENCE {
        problem             [0]  ENUMERATED {
            invalidID                   (0),
            duplicateID                 (1),
            unsupportedBindingType      (2),
            notAllowedForRole           (3),
```

```
                parametersMissing           (4),
                roleAssignment              (5),
                invalidStartTime            (6),
                invalidEndTime              (7),
                invalidAgreement            (8),
                currentlyNotDecidable       (9),
                modificationNotAllowed      (10) },
    bindingType             [1]  OPERATIONAL-BINDING.&id ({OpBindingSet}) OPTIONAL,
    agreementProposal       [2]  OPERATIONAL-BINDING.&Agreement
                                 ({OpBindingSet}{@bindingType}) OPTIONAL,
    retryAt                 [3]  Time OPTIONAL,
    COMPONENTS OF                CommonResultsSeq }


-- information object classes --

OPERATIONAL-BINDING ::= CLASS {
    &Agreement,
    &Cooperation    OP-BINDING-COOP,
    &both           OP-BIND-ROLE OPTIONAL,
    &roleA          OP-BIND-ROLE OPTIONAL,
    &roleB          OP-BIND-ROLE OPTIONAL,
    &id             OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
    AGREEMENT                   &Agreement
    APPLICATION CONTEXTS        &Cooperation
    [ SYMMETRIC                 &both ]
    [ ASYMMETRIC
        [ ROLE-A                &roleA ]
        [ ROLE-B                &roleB ] ]
    ID                          &id }

OP-BINDING-COOP ::= CLASS {
    &applContext    APPLICATION-CONTEXT,
    &Operations     OPERATION OPTIONAL }
WITH SYNTAX {
    &applContext
    [ APPLIES TO    &Operations ] }

OP-BIND-ROLE ::= CLASS {
    &establish                  BOOLEAN  DEFAULT FALSE,
    &EstablishParam                      OPTIONAL,
    &modify                     BOOLEAN  DEFAULT FALSE,
    &ModifyParam                         OPTIONAL,
    &terminate                  BOOLEAN  DEFAULT FALSE,
    &TerminateParam                      OPTIONAL }
WITH SYNTAX {
    [ ESTABLISHMENT-INITIATOR       &establish ]
    [ ESTABLISHMENT-PARAMETER       &EstablishParam ]
    [ MODIFICATION-INITIATOR        &modify ]
    [ MODIFICATION-PARAMETER        &ModifyParam ]
    [ TERMINATION-INITIATOR         &terminate ]
    [ TERMINATION-PARAMETER         &TerminateParam ] }

OpBindingSet    OPERATIONAL-BINDING ::= {
    shadowOperationalBinding |
    hierarchicalOperationalBinding |
    nonSpecificHierarchicalOperationalBinding }

END -- OperationalBindingManagement
```

Anexo H

Seguridad mejorada

(Este anexo es parte integrante de esta Recomendación | Norma Internacional)

Se sabe que este módulo contiene especificaciones no válidas. Una parte del mismo es, por tanto, desaprobada y se indica con ítems comentario ASN.1. En una edición futura se eliminarán o se actualizarán las especificaciones desaprobadas.

```
EnhancedSecurity { joint-iso-itu-t ds(5) modules(1) enhancedSecurity(28) 4 }
DEFINITIONS IMPLICIT TAGS  ::=
BEGIN

-- EXPORTS All --

IMPORTS


-- from ITU-T Rec. X.501 | ISO/IEC 9594-2

        authenticationFramework, basicAccessControl, certificateExtensions, id-at, id-avc, id-mr,
        informationFramework, upperBounds
                FROM UsefulDefinitions { joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 4 }

        Attribute, ATTRIBUTE, AttributeType, Context, CONTEXT, MATCHING-RULE, Name,
        objectIdentifierMatch, SupportedAttributes
                FROM InformationFramework informationFramework

        AttributeTypeAndValue
                FROM BasicAccessControl basicAccessControl

-- from ITU-T Rec. X.509 | ISO/IEC 9594-8

        AlgorithmIdentifier, CertificateSerialNumber, ENCRYPTED{}, HASH{}, SIGNED{}
                FROM AuthenticationFramework authenticationFramework

        GeneralName, KeyIdentifier
                FROM CertificateExtensions certificateExtensions

        ub-privacy-mark-length
                FROM UpperBounds upperBounds ;

-- from GULS

-- SECURITY-TRANSFORMATION, PROTECTION-MAPPING, PROTECTED
--      FROM Notation { joint-iso-ccitt genericULS (20) modules (1) notation (1) }

--dirSignedTransformation, KEY-INFORMATION
--      FROM GulsSecurityTransformations { joint-iso-ccitt genericULS (20) modules (1)
--              gulsSecurityTransformations (3) }

-- signed
--      FROM GulsSecurityTransformations { joint-iso-ccitt genericULS (20) modules (1)
--              dirProtectionMappings (4) };


-- The "signed" Protection Mapping and associated "dirSignedTransformations" imported
-- from the Generic Upper Layers Security specification (ITU-T Rec. X.830 | ISO/IEC 11586-1)
-- results in identical encoding as the same data type used with the SIGNED as defined in
-- ITU-T Rec. X.509 | ISO/IEC 9594-8

-- The three statements below are provided temporarily to allow signed operations to be supported as in edition 3.

OPTIONALLY-PROTECTED { Type }  ::=  CHOICE {
        unsigned        Type,
        signed          SIGNED {Type} }
```

```
OPTIONALLY-PROTECTED-SEQ { Type } ::= CHOICE {
        unsigned        Type,
        signed      [0] SIGNED { Type } }

-- The following out-commented ASN.1 specification are known to be erroneous and are therefore deprecated.

-- genEncryptedTransform {KEY-INFORMATION: SupportedKIClasses }  SECURITY-TRANSFORMATION  ::=
--      {
--      IDENTIFIER                  { enhancedSecurity  gen-encrypted(2) }
--      INITIAL-ENCODING-RULES      { joint-iso-itu-t  asn1(1)  ber(1) }
                                    -- This default for initial encoding rules may be overridden
                                    -- using a static protected parameter (initEncRules).
--      XFORMED-DATA-TYPE           SEQUENCE {
--              initEncRules        OBJECT IDENTIFIER DEFAULT { joint-iso-itu-t  asn1(1)  ber(1) },
--              encAlgorithm        AlgorithmIdentifier OPTIONAL, --  -- Identifies the encryption algorithm,
--              keyInformation      SEQUENCE {
--                  kiClass         KEY-INFORMATION.&kiClass ({SupportedKIClasses}),
--                  keyInfo         KEY-INFORMATION.&KiType ({SupportedKIClasses} {@kiClass})
--                                  } OPTIONAL,
                    -- Key information may assume various formats, governed by supported members
                    -- of the KEY-INFORMATION information object class (defined in ITU-T
                    -- Rec. X.830 | ISO/IEC 11586-1)
--              encData     BIT STRING ( CONSTRAINED BY {
                    -- the encData value shall be generated following
                    -- the procedure specified in 17.3.1-- -- })
--              }
--      }

-- encrypted  PROTECTION-MAPPING  ::=  {
--      SECURITY-TRANSFORMATION { genEncryptedTransform } }

-- signedAndEncrypt  PROTECTION-MAPPING  ::=  {
--      SECURITY-TRANSFORMATION  { signedAndEncryptedTransform } }

-- signedAndEncryptedTransform {KEY-INFORMATION: SupportedKIClasses}
--      SECURITY-TRANSFORMATION  ::=  {
--          IDENTIFIER                  { enhancedSecurity  dir-encrypt-sign (1) }
--          INITIAL-ENCODING-RULES      { joint-iso-itu-t asn1 (1) ber-derived (2) distinguished-encoding (1) }
--          XFORMED-DATA-TYPE
--              PROTECTED
--              {
--                  PROTECTED
--                  {
--                  ABSTRACT-SYNTAX.&Type,
--                  signed
--                  },
--              encrypted
--              }
--      }

-- OPTIONALLY-PROTECTED {ToBeProtected, PROTECTION-MAPPING:generalProtection} ::=
--      CHOICE {
--          toBeProtected   ToBeProtected,
                                --no DIRQOP specified for operation
--          signed          PROTECTED {ToBeProtected, signed},
                                --DIRQOP is Signed
--          protected       [APPLICATION 0]
--                              PROTECTED { ToBeProtected, generalProtection } }
                                --DIRQOP is other than Signed

-- defaultDirQop ATTRIBUTE  ::=  {
--      WITH SYNTAX                 OBJECT IDENTIFIER
--      EQUALITY MATCHING RULE      objectIdentifierMatch
--      USAGE                       directoryOperation
--      ID                          id-at-defaultDirQop }
```

```
-- DIRQOP  ::=  CLASS
-- This information object class is used to define the quality of protection
-- required throughout directory operation.
-- The Quality Of Protection can be signed, encrypted, signedAndEncrypt
-- {
--      &dirqop-Id                                  OBJECT IDENTIFIER UNIQUE,
--      &dirBindError-QOP                           PROTECTION-MAPPING:protectionReqd,
--      &dirErrors-QOP                              PROTECTION-MAPPING:protectionReqd,
--      &dapReadArg-QOP                             PROTECTION-MAPPING:protectionReqd,
--      &dapReadRes-QOP                             PROTECTION-MAPPING:protectionReqd,
--      &dapCompareArg-QOP                          PROTECTION-MAPPING:protectionReqd,
--      &dapCompareRes-QOP                          PROTECTION-MAPPING:protectionReqd,
--      &dapListArg-QOP                             PROTECTION-MAPPING:protectionReqd,
--      &dapListRes-QOP                             PROTECTION-MAPPING:protectionReqd,
--      &dapSearchArg-QOP                           PROTECTION-MAPPING:protectionReqd,
--      &dapSearchRes-QOP                           PROTECTION-MAPPING:protectionReqd,
--      &dapAbandonArg-QOP                          PROTECTION-MAPPING:protectionReqd,
--      &dapAbandonRes-QOP                          PROTECTION-MAPPING:protectionReqd,
--      &dapAddEntryArg-QOP                         PROTECTION-MAPPING:protectionReqd,
--      &dapAddEntryRes-QOP                         PROTECTION-MAPPING:protectionReqd,
--      &dapRemoveEntryArg-QOP                      PROTECTION-MAPPING:protectionReqd,
--      &dapRemoveEntryRes-QOP                      PROTECTION-MAPPING:protectionReqd,
--      &dapModifyEntryArg-QOP                      PROTECTION-MAPPING:protectionReqd,
--      &dapModifyEntryRes-QOP                      PROTECTION-MAPPING:protectionReqd,
--      &dapModifyDNArg-QOP                         PROTECTION-MAPPING:protectionReqd,
--      &dapModifyDNRes-QOP                         PROTECTION-MAPPING:protectionReqd,
--      &dspChainedOp-QOP                           PROTECTION-MAPPING:protectionReqd,
--      &dispShadowAgreeInfo-QOP                    PROTECTION-MAPPING:protectionReqd,
--      &dispCoorShadowArg-QOP                      PROTECTION-MAPPING:protectionReqd,
--      &dispCoorShadowRes-QOP                      PROTECTION-MAPPING:protectionReqd,
--      &dispUpdateShadowArg-QOP                    PROTECTION-MAPPING:protectionReqd,
--      &dispUpdateShadowRes-QOP                    PROTECTION-MAPPING:protectionReqd,
--      &dispRequestShadowUpdateArg-QOP             PROTECTION-MAPPING:protectionReqd,
--      &dispRequestShadowUpdateRes-QOP             PROTECTION-MAPPING:protectionReqd,
--      &dopEstablishOpBindArg-QOP                  PROTECTION-MAPPING:protectionReqd,
--      &dopEstablishOpBindRes-QOP                  PROTECTION-MAPPING:protectionReqd,
--      &dopModifyOpBindArg-QOP                     PROTECTION-MAPPING:protectionReqd,
--      &dopModifyOpBindRes-QOP                     PROTECTION-MAPPING:protectionReqd,
--      &dopTermOpBindArg-QOP                       PROTECTION-MAPPING:protectionReqd,
--      &dopTermOpBindRes-QOP                       PROTECTION-MAPPING:protectionReqd
-- }
-- WITH SYNTAX
-- {
--      DIRQOP-ID                                   &dirqop-Id
--      DIRECTORYBINDERROR-QOP                      &dirBindError-QOP
--      DIRERRORS-QOP                               &dirErrors-QOP
--      DAPREADARG-QOP                              &dapReadArg-QOP
--      DAPREADRES-QOP                              &dapReadRes-QOP
--      DAPCOMPAREARG-QOP                           &dapCompareArg-QOP
--      DAPCOMPARERES-QOP                           &dapCompareRes-QOP
--      DAPLISTARG-QOP                              &dapListArg-QOP
--      DAPLISTRES-QOP                              &dapListRes-QOP
--      DAPSEARCHARG-QOP                            &dapSearchArg-QOP
--      DAPSEARCHRES-QOP                            &dapSearchRes-QOP
--      DAPABANDONARG-QOP                           &dapAbandonArg-QOP
--      DAPABANDONRES-QOP                           &dapAbandonRes-QOP
--      DAPADDENTRYARG-QOP                          &dapAddEntryArg-QOP
--      DAPADDENTRYRES-QOP                          &dapAddEntryRes-QOP
--      DAPREMOVEENTRYARG-QOP                       &dapRemoveEntryArg-QOP
--      DAPREMOVEENTRYRES-QOP                       &dapRemoveEntryRes-QOP
--      DAPMODIFYENTRYARG-QOP                       &dapModifyEntryArg-QOP
--      DAPMODIFYENTRYRES-QOP                       &dapModifyEntryRes-QOP
--      DAPMODIFYDNARG-QOP                          &dapModifyDNArg-QOP
--      DAPMODIFYDNRES-QOP                          &dapModifyDNRes-QOP
--      DSPCHAINEDOP-QOP                            &dspChainedOp-QOP
--      DISPSHADOWAGREEINFO-QOP                     &dispShadowAgreeInfo-QOP
--      DISPCOORSHADOWARG-QOP                       &dispCoorShadowArg-QOP
--      DISPCOORSHADOWRES-QOP                       &dispCoorShadowRes-QOP
--      DISPUPDATESHADOWARG-QOP                     &dispUpdateShadowArg-QOP
```

```
--      DISPUPDATESHADOWRES-QOP                 &dispUpdateShadowRes-QOP
--      DISPREQUESTSHADOWUPDATEARG-QOP          &dispRequestShadowUpdateArg-QOP
--      DISPREQUESTSHADOWUPDATERES-QOP          &dispRequestShadowUpdateRes-QOP
--      DOPESTABLISHOPBINDARG-QOP               &dopEstablishOpBindArg-QOP
--      DOPESTABLISHOPBINDRES-QOP               &dopEstablishOpBindRes-QOP
--      DOPMODIFYOPBINDARG-QOP                  &dopModifyOpBindArg-QOP
--      DOPMODIFYOPBINDRES-QOP                  &dopModifyOpBindRes-QOP
--      DOPTERMINATEOPBINDARG-QOP               &dopTermOpBindArg-QOP
--      DOPTERMINATEOPBINDRES-QOP               &dopTermOpBindRes-QOP
-- }

attributeValueSecurityLabelContext CONTEXT ::= {
        WITH SYNTAX     SignedSecurityLabel     -- At most one security label context can be assigned to an
                                                -- attribute value
        ID              id-avc-attributeValueSecurityLabelContext }

SignedSecurityLabel  ::=  SIGNED {SEQUENCE  {
        attHash         HASH {AttributeTypeAndValue},
        issuer          Name            OPTIONAL,  -- name of labelling authority
        keyIdentifier   KeyIdentifier   OPTIONAL,
        securityLabel   SecurityLabel } }

SecurityLabel  ::=  SET {
        security-policy-identifier      SecurityPolicyIdentifier        OPTIONAL,
        security-classification         SecurityClassification          OPTIONAL,
        privacy-mark                    PrivacyMark                     OPTIONAL,
        security-categories             SecurityCategories              OPTIONAL }
                (ALL EXCEPT ( {-- none, at least one component shall be present -- } ) )

SecurityPolicyIdentifier  ::=  OBJECT IDENTIFIER

SecurityClassification ::= INTEGER {
        unmarked        (0),
        unclassified    (1),
        restricted      (2),
        confidential    (3),
        secret          (4),
        top-secret      (5) }

PrivacyMark  ::=  PrintableString (SIZE (1..ub-privacy-mark-length))

SecurityCategories ::= SET SIZE (1..MAX) OF SecurityCategory

clearance  ATTRIBUTE  ::=  {
        WITH SYNTAX     Clearance
        ID              id-at-clearance }

Clearance  ::=  SEQUENCE  {
        policyId                OBJECT IDENTIFIER,
        classList               ClassList                       DEFAULT {unclassified},
        securityCategories      SET SIZE (1..MAX) OF SecurityCategory   OPTIONAL }

ClassList  ::=  BIT STRING {
        unmarked        (0),
        unclassified    (1),
        restricted      (2),
        confidential    (3),
        secret          (4),
        topSecret       (5) }

SecurityCategory  ::=  SEQUENCE {
        type    [0]     SECURITY-CATEGORY.&id ({SecurityCategoriesTable}),
        value   [1]     EXPLICIT SECURITY-CATEGORY.&Type ({SecurityCategoriesTable} {@type}) }

SECURITY-CATEGORY  ::=  TYPE-IDENTIFIER

SecurityCategoriesTable SECURITY-CATEGORY ::= { ... }
```

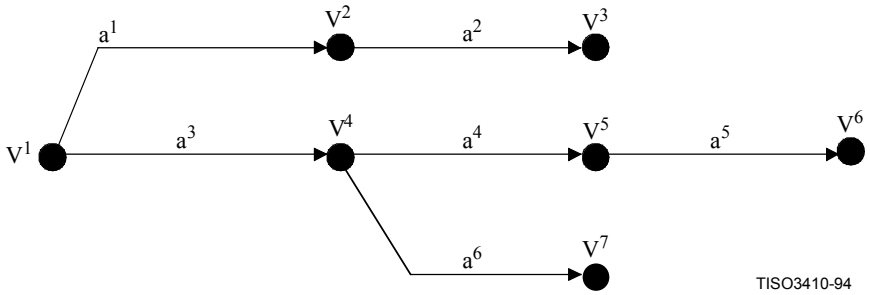
```
attributeIntegrityInfo ATTRIBUTE  ::=  {
        WITH SYNTAX                     AttributeIntegrityInfo
        ID                              id-at-attributeIntegrityInfo }

AttributeIntegrityInfo  ::=  SIGNED { SEQUENCE  {
        scope           Scope,                          -- Identifies the attributes protected
        signer          Signer      OPTIONAL,           -- Authority or data originators name
        attribsHash     AttribsHash } }                 -- Hash value of protected attributes

Signer  ::=  CHOICE {
        thisEntry   [0]   EXPLICIT ThisEntry,
        thirdParty  [1]   SpecificallyIdentified }

ThisEntry  ::=  CHOICE  {
        onlyOne     NULL,
        specific    IssuerAndSerialNumber }

IssuerAndSerialNumber  ::=  SEQUENCE {
        issuer      Name,
        serial      CertificateSerialNumber }

SpecificallyIdentified  ::=  SEQUENCE  {
        name        GeneralName,
        issuer      GeneralName                 OPTIONAL,
        serial      CertificateSerialNumber     OPTIONAL }
        ( WITH COMPONENTS { ..., issuer PRESENT, serial PRESENT } |
        ( WITH COMPONENTS { ..., issuer ABSENT, serial ABSENT } ) )

Scope  ::=  CHOICE  {
        wholeEntry      [0]   NULL,             -- Signature protects all attribute values in this entry
        selectedTypes   [1]   SelectedTypes
                              -- Signature protects all attribute values of the selected attribute types
        }

SelectedTypes  ::=  SEQUENCE SIZE (1..MAX) OF AttributeType

AttribsHash  ::=  HASH { SEQUENCE SIZE (1..MAX) OF Attribute }
                  -- Attribute type and values with associated context values for the selected Scope

attributeValueIntegrityInfoContext  CONTEXT  ::=  {
        WITH SYNTAX     AttributeValueIntegrityInfo
        ID              id-avc-attributeValueIntegrityInfoContext }

AttributeValueIntegrityInfo  ::=  SIGNED { SEQUENCE {
        signer          Signer      OPTIONAL,       -- Authority or data originators name
        aVIHash         AVIHash } }                 -- Hash value of protected attribute

AVIHash  ::=  HASH { AttributeTypeValueContexts }
                  -- Attribute type and value with associated context values

AttributeTypeValueContexts ::= SEQUENCE {
        type            ATTRIBUTE.&id ({SupportedAttributes}),
        value           ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
        contextList     SET SIZE (1..MAX) OF Context OPTIONAL }

-- The following out-commented ASN.1 specification are known to be erroneous and are therefore deprecated.

-- EncryptedAttributeSyntax {AttributeSyntax}  ::=  SEQUENCE  {
--      keyInfo     SEQUENCE OF KeyIdOrProtectedKey,
--      encAlg      AlgorithmIdentifier,
--      encValue    ENCRYPTED { AttributeSyntax } }

-- KeyIdOrProtectedKey  ::=  SEQUENCE  {
--      keyIdentifier       [0]   KeyIdentifier     OPTIONAL,
--      protectedKeys       [1]   ProtectedKey      OPTIONAL }
            -- At least one key identifier or protected key shall be present
```

```
-- ProtectedKey ::= SEQUENCE {
--     authReaders      AuthReaders,--    -- if absent, use attribute in authorized reader entry
--     keyEncAlg        AlgorithmIdentifier OPTIONAL, --  -- algorithm to encrypt encAttrKey
--     encAttKey        EncAttKey }
                              -- confidentiality key protected with authorized user's
                              -- protection mechanism

-- AuthReaders ::= SEQUENCE OF Name

-- EncAttKey ::= PROTECTED {SymmetricKey, keyProtection}

-- SymmetricKey ::= BIT STRING

-- keyProtection PROTECTION-MAPPING ::= {
--     SECURITY-TRANSFORMATION {genEncryption} }

-- confKeyInfo ATTRIBUTE ::= {
--     WITH SYNTAX                  ConfKeyInfo
--     EQUALITY MATCHING RULE       readerAndKeyIDMatch
--     ID                           id-at-confKeyInfo }

-- ConfKeyInfo ::= SEQUENCE {
--     keyIdentifier    KeyIdentifier,
--     protectedKey     ProtectedKey }

-- readerAndKeyIDMatch MATCHING-RULE ::= {
--     SYNTAX    ReaderAndKeyIDAssertion
--     ID        id-mr-readerAndKeyIDMatch }

-- ReaderAndKeyIDAssertion ::= SEQUENCE {
--     keyIdentifier    KeyIdentifier,
--     authReaders      AuthReaders OPTIONAL }

-- Object identifier assignments --
-- attributes --

id-at-clearance                          OBJECT IDENTIFIER   ::=  {id-at 55}
-- id-at-defaultDirQop                   OBJECT IDENTIFIER   ::=  {id-at 56}
id-at-attributeIntegrityInfo             OBJECT IDENTIFIER   ::=  {id-at 57}
-- id-at-confKeyInfo                     OBJECT IDENTIFIER   ::=  {id-at 60}

-- matching rules --

-- id-mr-readerAndKeyIDMatch             OBJECT IDENTIFIER   ::=  {id-mr 43}

-- contexts--

id-avc-attributeValueSecurityLabelContext  OBJECT IDENTIFIER  ::=  {id-avc 3}
id-avc-attributeValueIntegrityInfoContext  OBJECT IDENTIFIER  ::=  {id-avc 4}

END -- EnhancedSecurity
```

Anexo I

Matemática de árboles

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)



Un árbol es un conjunto de puntos, llamados *vértices*, y un conjunto de líneas dirigidas llamadas *arcos*; cada arco a conduce de un vértice V a un vértice V'. Por ejemplo, el árbol de la figura tiene siete vértices, señalados con V^1 a V^7, y seis arcos, señalados a^1 a a^6.

Los dos vértices V y V' se denominan vértice *inicial* y vértice *final*, respectivamente de un arco a de V a V'. Por ejemplo, V^2 y V^3 son los vértices inicial y final, respectivamente del arco a^2. Varios arcos diferentes pueden partir del mismo vértice inicial, pero no tendrán el mismo vértice final. Por ejemplo, el arco a^1 y a^3 tienen el mismo vértice inicial V^1, pero no hay dos arcos en la figura que tengan el mismo vértice final.

El vértice que no es el vértice final de ningún arco suele llamarse vértice *raíz*, o de manera aún más informal, la "raíz" del árbol. Por ejemplo, en la figura V^1 es la raíz.

Un vértice que no es el vértice inicial de ningún arco suele llamarse informalmente vértice *hoja*, o de manera aún más informal, una "hoja" de la gráfica del árbol. Por ejemplo, los vértices V^3, V^6 y V^7 son hojas.

Un *trayecto orientado* desde un vértice V a un vértice V' es un conjunto de arcos $(a^1, a^2, ..., a^n)$ $(n \geq 1)$ tal que V es el vértice inicial del arco a^1 y V' es el vértice final del arco a^n, y el vértice final del arco a^k es también el vértice inicial del arco a^{k+1} siendo $1 \leq k < n$. Por ejemplo, el trayecto orientado del vértice V^1 al vértice V^6 es el conjunto de arcos (a^3, a^4, a^5). Se entenderá que el término "trayecto" designa un trayecto orientado desde la raíz hacia un vértice.

Anexo J

Criterios de diseño de nombres

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

El marco de información es muy general y permite una diversidad arbitraria de inserciones y atributos en el DIT. Dado que, según lo aquí definido, los nombres guardan estrecha relación con los trayectos a través del DIT, es posible una diversidad arbitraria de nombres. En este anexo se exponen criterios que podrían considerarse para el diseño de nombres. Se han utilizado criterios adecuados para el diseño de las formas de nombre recomendadas que figuran en la Rec. UIT-T X.521 | ISO/CEI 9594-7. Se sugiere que estos criterios se sigan también, cuando proceda, en el diseño de nombres para objetos a los que no son aplicables las formas de nombre recomendadas.

Por el momento se sigue un solo criterio, el de la comodidad para el usuario.

NOTA – No todos los nombres tienen que ser fáciles para el usuario.

En el resto de este anexo se discute el concepto de comodidad de los usuarios con respecto a los nombres.

Los nombres con los que trabajan directamente las personas deben ser fáciles y cómodos para el usuario. Un nombre cómodo para el usuario es el que toma en consideración el punto de vista del usuario humano y no el del computador. Es un nombre fácil de deducir, recordar y comprender por las personas, y no uno que sea fácil de interpretar por los computadores.

El objetivo de la comodidad para el usuario puede exponerse con un poco más de precisión en base a los dos principios siguientes:

- Un individuo debe por lo general estar en condiciones de encontrar el nombre cómodo para el usuario de un objeto partiendo de la información que posee naturalmente acerca del objeto. Por ejemplo, debe poder adivinar el nombre de una persona dedicada a algún negocio sólo con la información que adquiera naturalmente sobre esa persona por una asociación normal en materia de negocios.
- Cuando un nombre de objeto se especifica de manera ambigua, el directorio debe reconocer esa situación en vez de llegar a la conclusión de que el nombre identifica a un objeto determinado. Por ejemplo, cuando dos personas tienen el mismo apellido, el apellido sólo deberá considerarse una identificación insuficiente para cada una de esas dos personas.

Del objetivo primordial de la comodidad para el usuario se desprenden los siguientes objetivos secundarios:

a) Los nombres no deben eliminar artificialmente las ambigüedades naturales. Por ejemplo, si dos personas poseen el mismo apellido "Jones", no se pedirá a ninguno de los dos que responda a "WJones " o "Jones2". En lugar de ello, el convenio de denominación deberá proporcionar un medio cómodo para el usuario de distinguir las entidades. Por ejemplo, puede requerirse el primer nombre y la inicial del segundo nombre, además del apellido.

b) Es preciso que los nombres admitan abreviaturas comunes y variantes comunes de su ortografía. Por ejemplo, si una persona está empleada por la Conway Steel Corporation y el nombre de la empresa acompaña al nombre de esa persona, cualquiera de los nombres "Conway Steel Corporation", "Conway Steel Corp.", "Conway Steel" y "CSC" debería bastar para identificar a la organización de que se trata.

c) En ciertos casos se pueden utilizar nombres de alias: para orientar la búsqueda de una inserción en particular, para una mayor comodidad para el usuario, o para reducir el ámbito de la búsqueda. El siguiente ejemplo muestra el uso de nombres de alias para tal efecto: como se ve en la figura J.1, la sucursal de Osaka también se puede identificar con el nombre {C = Japón, L = Osaka, O = ABC, OU = agencia de Osaka}.

d) En el caso de nombres compuestos, tanto el número de los componentes obligatorios como el de los facultativos deberá ser relativamente pequeño, con lo cual serán más fáciles de recordar.

e) Si los nombres tienen varios componentes, por lo general el orden preciso en que aparecen dichos componentes deberá ser intrascendente.

f) Los nombres cómodos para el usuario no deben incluir direcciones de computador.

g) En ciertos casos pueden utilizarse contextos para proporcionar nombres alternativos. Por ejemplo, como se muestra en la figura J.2, puede identificarse a la persona Jones mediante {O = "XYZ", OU = "Research", CN = "Jones"} cuando el contexto es lenguaje = inglés y {O = "XYZ", OU = "Recherche", CN = "Jones"} cuando el contexto es lenguaje = francés.

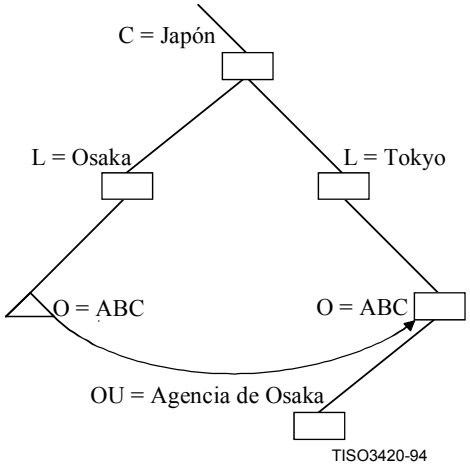


Figura J.1 – Ejemplo de utilización de alias

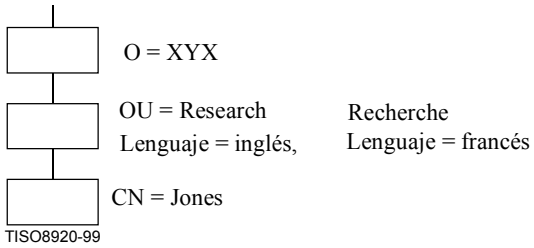


Figura J.2 – Ejemplo de variaciones de contexto de un nombre

Anexo K

Ejemplos de diversos aspectos de esquema

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

K.1 Ejemplo de una jerarquía de atributos

La figura K.1 muestra una jerarquía simple de valores de un atributo **telephoneNumber (número de teléfono)** genérico, unos valores del cual se representan como contenidos en el conjunto exterior. Del tipo genérico se han derivado dos tipos de atributo específicos: **workTelephoneNumber (número de teléfono del trabajo)** y **homeTelephoneNumber (número de teléfono de la casa)**. Los valores de estos tipos se representan como contenidos en los conjuntos interiores.

Un valor de tipo **homeTelephoneNumber** está contenido tanto en el conjunto interior que representa **homeTelephoneNumber** como en el conjunto exterior que representa **telephoneNumber**, pero no en el conjunto interior que representa **workTelephoneNumber**.

Podría definirse una regla de estructura del DIT que permitiera a las inserciones contener valores de los tres tipos mostrados en la figura K.1. Podría definirse otra regla que permitiera a las inserciones contener valores de tipo **telephoneNumber** solamente.

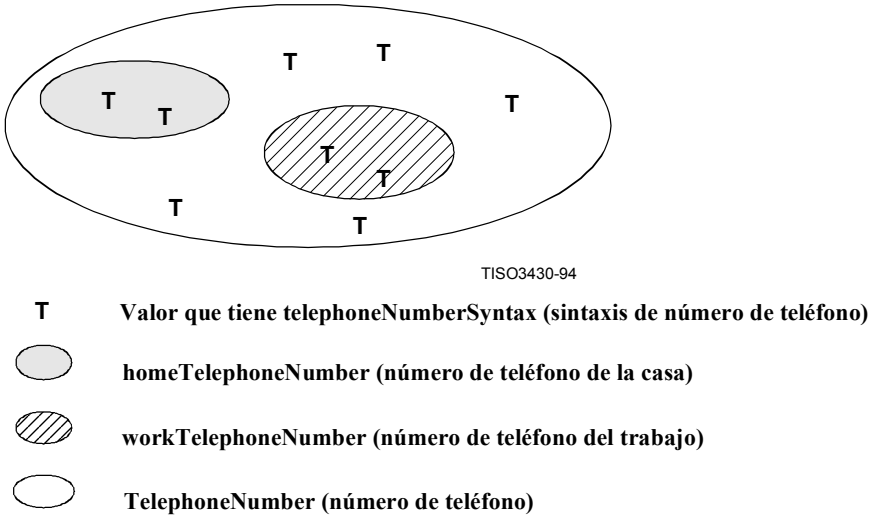


Figura K.1 – Jerarquía de valores del atributo número de teléfono

K.2 Ejemplo de una especificación de subárbol

El siguiente ejemplo ilustra la especificación de subárboles. Considérese la porción del DIT representada en la figura K.2.

El subárbol 1 y el subárbol 2 se especifican con respecto al punto administrativo de nombre a. Los identificadores b1, c2, d3, etc. representan valores de nombre locales con respecto al punto administrativo a.

El subárbol 1 puede especificarse como:

```
subtree1 SubtreeSpecification ::= {
                    specificExclusions { chopBefore b1 } }
```

El subárbol 2 puede especificarse como:

```
subtree2 SubtreeSpecification      ::= {
                    base b1 }
```

Supóngase que las inserciones identificadas en la figura con los nombres e1, e2, etc. representan inserciones de persona organizacional. Un refinamiento de subárbol que incluya todas estas inserciones en la zona administrativa podría especificarse como sigue:

```
subtree-refinement1 SubtreeSpecification ::= {
        specificationFilter
                item            id-oc-organizationalPerson }
```

Un refinamiento más restrictivo que sólo incluya personas organizacionales en el subárbol 2 podría especificarse de la forma siguiente:

```
subtree2-refinement SubtreeSpecification::=     {
        base                    b1,
        specificationFilter
                item            id-oc-organizationalPerson }
```

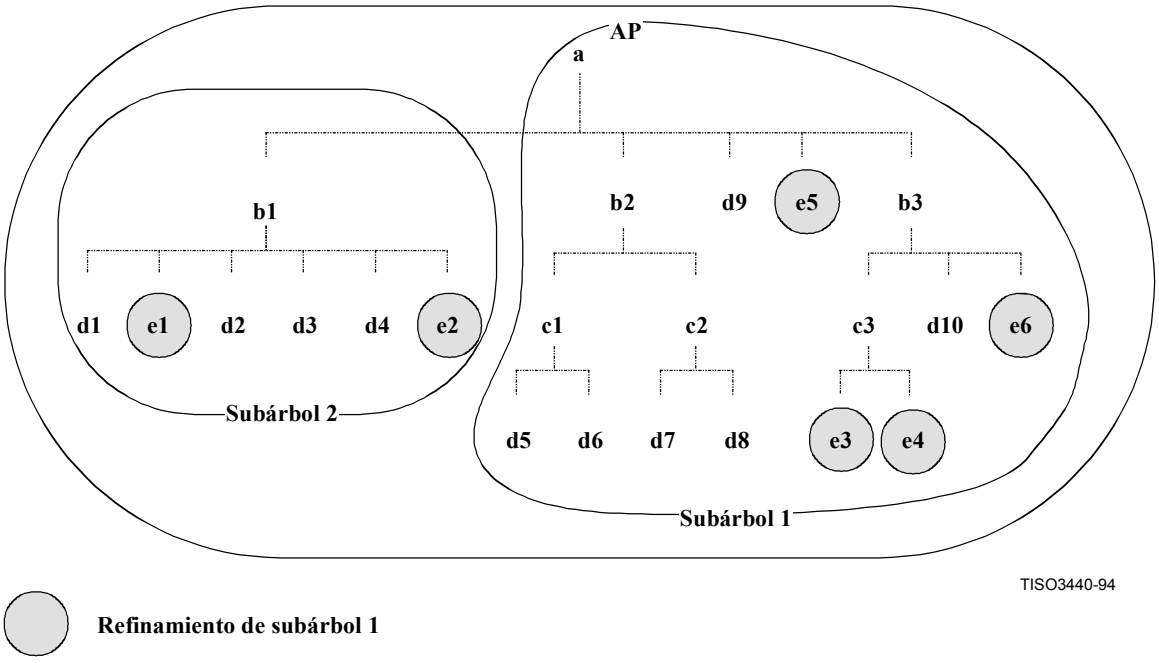


Figura K.2 – Ejemplo de especificación de subárbol

K.3 Especificación de esquema

K.3.1 Clases de objeto y formas de nombre

Las siguientes clases de objeto, definidas en la Rec. UIT-T X.521 | ISO/CEI 9594-7, se utilizan dentro de una zona administrativa de subesquema particular:

- **organization (organización)**;
- **organizationalUnit (unidad organizacional)**;
- **organizationalPerson (persona organizacional)**.

No se requiere una forma de nombre para la inserción administrativa, que será la única inserción en el subesquema de la clase de objeto **organization**. Las siguientes formas de nombre, definidas en la Rec. UIT-T X.521 | ISO/CEI 9594-7, se utilizan para incluir inserciones de clase **organizationalUnit** y **organizationalPerson**:

- **orgNameForm (forma de nombre de organización)**;
- **orgUnitNameForm (forma de nombre de unidad organizacional)**;
- **orgPersonNameForm (forma de nombre de persona organizacional)**.

K.3.2 Reglas de estructura del DIT

Se definen las siguientes reglas de estructura para especificar una estructura de árbol como la mostrada en la figura K.3. Esta figura ilustra la regla que puede utilizarse para añadir inserciones en los diversos puntos del DIT.

```
rule-0 STRUCTURE-RULE::= {
        NAME FORM                   orgNameForm
        ID                          0 }

rule-1 STRUCTURE-RULE::= {
        NAME FORM                   orgUnitNameForm
        SUPERIOR RULES              { rule-0 }
        ID                          1 }

rule-2 STRUCTURE-RULE::= {
        NAME FORM                   orgUniNameForm
        SUPERIOR RULES              { rule-1 }
        ID                          2 }

rule-3 STRUCTURE-RULE::= {
        NAME FORM                   orgUniNameForm
        SUPERIOR RULES              { rule-2 }
        ID                          3 }

rule-4 STRUCTURE-RULE::= {
        NAME FORM                   orgPersonNameForm
        SUPERIOR RULES              { rule-1, rule-2, rule-3 }
        ID                          4 }
```

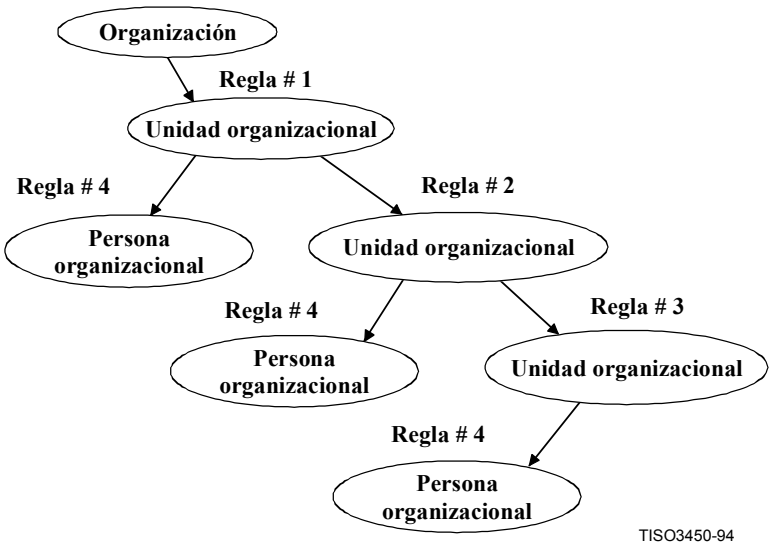


Figura K.3 – Ejemplo de subesquema

K.4 Reglas de contenido del DIT

El administrador de subesquema debe cumplir los dos requisitos siguientes para añadir información suplementaria a inserciones en la zona administrativa de subesquema:

- todas las inserciones **organizationalPerson** y **organizationalUnit** deben tener el atributo **organizationalTelephoneNumber**. Este atributo debe devolverse cuando se interroga el directorio para obtener números de teléfono;
- todas las inserciones **organizationalPerson** tendrán el nuevo gestor de atributo.

Para cumplir estos requisitos se definen los siguientes tipos de atributo:

```
manager ATTRIBUTE ::= {
        WITH SYNTAX                     BOOLEAN
        EQUALITY MATCHING RULE          booleanMatch
        SINGLE VALUE                    TRUE
        ID                              id-ex-managerAttribute }

organizationalTelephoneNumber ATTRIBUTE ::= {
        SUBTYPE OF          telephoneNumber
        COLLECTIVE          TRUE
        ID                  id-ex-organizationalTelephoneNumber }
```

Para cumplir estos requisitos se definen las siguientes reglas de contenido de DIT:

```
organizationRule CONTENT-RULE ::= {
        STRUCTURAL OBJECT CLASS     organization }

organizationalUnitRule CONTENT-RULE ::= {
        STRUCTURAL OBJECT CLASS     organizationalUnit
        MAY CONTAIN                 { organizationalTelephoneNumber } }

organizationalPersonRule CONTENT-RULE ::= {
        STRUCTURAL OBJECT CLASS     organizationalPerson
        MUST CONTAIN                { manager }
        MAY CONTAIN                 { organizationalTelephoneNumber } }
```

K.5 Uso del contexto del DIT

El administrador del subesquema debe desarrollar una política de organización internacional que obligue al empleo del contexto **locale** para distinguir entre distintos valores del título y tipos de atributo de descripción en la zona administrativa de la organización. Además, como la organización cambia sus obligaciones sobre una base regular, es conveniente utilizar en las inscripciones correspondientes a ciertas personas el contexto temporal con títulos.

Para cumplir estos requisitos se definen las siguientes reglas de contexto del DIT:

```
descriptionContextRule      DIT-CONTEXT-USE-RULE ::= {
        ATTRIBUTE TYPE          description
        MANDATORY CONTEXTS      { locale } }

titleContextRule            DIT-CONTEXT-USE-RULE ::= {
        ATTRIBUTE TYPE          title
        MANDATORY CONTEXTS      { localeContext }
        OPTIONAL CONTEXTS       { temporalContext } }
```

Anexo L

Visión de conjunto de permisos de control de acceso básico

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

L.1 Introducción

Este anexo tiene carácter informativo y deberá ofrecer una visión de conjunto del significado de diversas combinaciones de operaciones, ítems protegidos, y categorías de permisos. En aquellos casos en que se adviertan diferencias entre esta visión de conjunto y la especificación que figura en el cuerpo de esta Especificación de directorio, prevalecerá el texto normativo de esta última.

El cuadro L.1 relaciona operaciones de directorio con los controles de acceso a inserciones y atributos para proporcionar una panorámica de las categorías de permiso que habrán de concederse para permitir la prosecución de las operaciones.

El cuadro L.2 presenta una visión de conjunto de las categorías de permiso **returnDN (devolver nombre distinguido)** y **discloseOnError (revelar error)** y la manera en que las concesiones y denegaciones se relacionan con diversos elementos de protocolo.

El cuadro L.3 da una visión de conjunto de las semánticas asociadas con las concesiones y denegaciones de controles de acceso a inserciones.

El cuadro L.4 da una visión de conjunto de las semánticas asociadas con las concesiones y denegaciones de controles de acceso a atributos.

L.2 Permisos requeridos para operaciones

Cuadro L.1 – Permisos para el acceso a información de directorio requeridos de acuerdo con el funcionamiento del directorio

Operación del directorio	Permisos de ítem protegido de inserción requeridos	Permisos de ítem protegido de atributo y de valor de atributo requeridos
Comparar	*Read*	*Compare* para el atributo que se compara *Compare* para el valor de atributo que se compara
Leer	*Read* y *ReturnDN* para nombre distinguido	*Read* para cualquier tipo de información de atributo devuelta *Read* para cualesquiera valores de atributo devueltos
Enumerar	*Browse* y *ReturnDN* para todas las inserciones subordinadas para las cuales se ha devuelto un RDN	Ninguno
Buscar	*Browse* para inserciones en el alcance de búsqueda que son posibles candidatas a selección; *ReturnDN* para cada nombre distinguido devuelto	*FilterMatch* para información de tipo y valor de atributo, si existe, utilizada para evaluar un ítem de filtro como TRUE o FALSE *Read* para cualquier tipo de información de atributo devuelta *Read* para cualesquiera valores de atributo devueltos
Añadir inserción	*Add*	*Add* para todos los tipos de atributo especificados *Add* para todos los valores de atributo especificados
Suprimir inserción	*Remove*	Ninguno
Modificar inserción	*Modify*	*Add* para todos los atributos que se añaden *Add* para todos los valores de atributo que se añaden *Remove* para todos los atributos que se suprimen *Remove* para todos los valores de atributo que se suprimen
Modificar nombre distinguido (DN)	*Rename* en la ubicación original si sólo se cambia el último RDN *Export* para desplazar un subárbol de la ubicación original *Import* para reubicar un subárbol en la ubicación de destino	Ninguno

L.3 Permisos que influyen en la devolución de error

Cuadro L.2 – Permisos que influyen en la devolución de error y nombre

	Protocol Elements Affected	Meaning
ReturnDN	**EntryInformation** **CompareResult** **ListResult** **SearchResult** **NameError** **ContinuationReference**	Si se concede, puede devolver el nombre distinguido real. Si se deniega, prohíbe devolver el nombre distinguido real. Por política local, se puede devolver en su lugar un nombre de alias válido.
DiscloseOnError	**NameError** **UpdateError** **AttributeError** **SecurityError**	Si se concede, permite devolver un error que puede revelar que el ítem protegido existe. Si se deniega, obliga al directorio a ocultar la existencia del ítem protegido.

L.4 Permisos a nivel de inserción

Cuadro L.3 – Permisos y significados a nivel de inserción

Permiso	Significado
Read	Si se concede, permite efectuar las operaciones de directorio de leer o comparar sobre la inserción, pero no autoriza, por sí misma, el retorno de ninguna información de atributo de la inserción. Si se deniega, impide las operaciones de leer o comparar sobre la inserción.
Browse	Si se concede, permite que la inserción participe como candidata a selección en el ámbito de una operación de enumerar o buscar. Si se deniega, excluye la inserción del ámbito de toda operación de enumerar o buscar.
Add	Si se concede, permite añadir la inserción, sin incluir sus atributos. Añadir sólo tiene sentido como ACI prescriptiva. Si se deniega, impide la adición de la inserción.
Modify	Si se concede, permite operaciones de modificar sobre la inserción. Si se deniega, evita que se efectúen operaciones de modificar sobre la inserción.
Remove	Si se concede, permite que la inserción sea suprimida, sin tener en cuenta consideraciones sobre los atributos. Si se deniega, impide la supresión de la inserción.
Rename	Si se concede, permite cambiar el RDN de la inserción, y facultativamente, suprimir un valor antiguo y añadir uno nuevo, sin tener en cuenta la protección de atributo o de valor de atributo de esa inserción, por medio de una operación ModifyDN a reserva de permisos de *Import* y *Export*, según proceda. Si se deniega, impide que se cambie el RDN de la inserción.
Import	Si se concede, permite que las inserciones, incluidas todas las subordinadas, sean reubicadas en la ubicación de destino, en el DIT, en una operación de ModifyDN. *Import* sólo tiene sentido como ACI prescriptiva. Si se deniega, impide la reubicación de una inserción con sus subordinadas en el punto indicado del DIT, utilizando una operación de ModifyDN.
Export	Si se concede, permite que una operación de ModifyDN reubique la inserción, incluyendo todas las subordinadas, en cualquier otro punto designado en el DIT. El solicitante debe tener permiso para *importar* a la ubicación de destino. Si se deniega, impide la reubicación de la inserción y sus subordinadas en una sola operación de ModifyDN.
ReturnDN	Si se concede, permite devolver el nombre distinguido de una inserción en un resultado de operación. Si se deniega, prohíbe devolver el nombre distinguido. Por política local, se puede devolver en su lugar un nombre de alias válido.
DiscloseOnError	Si se concede, permite devolver un error que puede revelar la existencia de la inserción. Si se deniega, obliga al directorio a ocultar la existencia de la inserción. *DiscloseOnError*, por sí mismo, no excluye la posibilidad de detectar la inserción por otros medios para los cuales se hayan concedido los permisos apropiados.

L.5 Permisos a nivel de atributo

Cuadro L.4 – Permisos y significados a nivel de atributos

Permiso	Categoría de ítem protegido	Significado
Read	Tipo de atributo	Si se concede, permite devolver información sobre el tipo de atributo en una operación leer o buscar. Aunque es un prerrequisito para la lectura de valores para ese atributo, no concede por sí mismo ningún derecho a valores de ese atributo. Si se deniega, impide devolver información sobre el tipo de atributo en operaciones de leer o buscar. En efecto, deniega también todos los valores.
Read	Valor de atributo	Si se concede, permite devolver uno o más valores designados de un atributo en una operación de leer o buscar. No concede ningún derecho al propio atributo. Para leer un valor también se requiere un permiso de *Read* para el tipo de atributo. Si se deniega, impide devolver valores designados de ese tipo de atributo en operaciones de leer y buscar.
Compare	Tipo de atributo	Si se concede, permite operaciones de comparar para hacer pruebas sobre el tipo de atributo. Aunque es un prerrequisito para comparar valores, no permite por sí mismo operaciones de comparar para cualesquiera valores. Si se deniega, impide que operaciones de comparar hagan pruebas sobre ese atributo. Se evitan así pruebas sobre todos los valores.
Compare	Valor de atributo	Si se concede, permite operaciones de comparar para hacer pruebas sobre valores designados del tipo designado. No concede derechos sobre el propio atributo. Para comparar un valor también se requiere permiso de *Compare* para el tipo de atributo. Si se deniega, impide que operaciones de comparar hagan pruebas sobre el valor designado.
FilterMatch	Tipo de atributo	Si se concede, permite utilizar el tipo de atributo en la evaluación de un ítem de filtro buscar. Es un prerrequisito para incluir valores de ese tipo en evaluaciones de filtros, pero no concede por sí mismo, derecho a ningún atributo. Si se deniega, impide que se utilice ese tipo de atributo, así como cualquiera de sus valores, en la evaluación de un ítem de filtro.
FilterMatch	Valor de atributo	Si se concede, permite que el valor o valores del atributo se utilicen en la evaluación de un ítem de filtro buscar. Para una evaluación exitosa se requiere también *FilterMatch* para el tipo de atributo. Si se deniega, impide el uso del valor o valores en la evaluación de un ítem de filtro.
Add	Tipo de atributo	Si se concede, permite añadir el tipo de atributo designado. No da derechos a añadir ningún valor de atributo. Si se deniega, impide que se añada el tipo de atributo designado, y en consecuencia, cualquier valor.
Add	Valor de atributo	Si se concede, permite añadir los valores de atributo designados. No concede por sí mismo derechos al tipo de atributo. A la inversa, para añadir un valor a un atributo existente no es necesario el derecho a añadir el tipo de atributo. Si se deniega, impide la adición de los valores de atributo designados.
Remove	Tipo de atributo	Si se concede, permite suprimir el tipo de atributo designado y todos sus valores en una operación modificar. No concede, por sí mismo el derecho a suprimir valores individuales. Si se deniega, impide que se suprima el tipo de atributo en una operación de modificar.
Remove	Valor de atributo	Si se concede, permite suprimir los valores de atributo designados en una operación de modificar. Para suprimir el último valor se necesita también el permiso de *Remove* para el tipo de atributo. Si se deniega, impide que se supriman los valores de atributo designados en una operación de modificar.
DiscloseOnError	Tipo de atributo	Si se concede, permite devolver un error que puede revelar la existencia del atributo. Si se deniega, obliga al directorio a ocultar la existencia del atributo. *DiscloseOnError*, por sí mismo no excluye la posibilidad de detectar el tipo de atributo por otros medios para los que se hayan concedido los permisos adecuados.
DiscloseOnError	Valor de atributo	Si se concede, permite devolver un error que puede revelar la existencia del valor de atributo. Si se deniega, obliga al directorio a ocultar la existencia del valor de atributo. *DiscloseOnError*, por sí mismo no niega la posibilidad de detectar el valor o valores de atributo por otros medios para los que se hayan concedido los permisos adecuados.

Anexo M

Ejemplos de control de acceso

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

M.1 Introducción

Este anexo es de carácter informativo y sus fines son exclusivamente didácticos. Trata tres temas fundamentales: los principios de diseño que son importantes para la arquitectura del mecanismo de control de acceso básico, un ejemplo ampliado de control de acceso básico y un ejemplo breve de control de acceso reglado. Una información detallada sobre el control de acceso básico y el control de acceso reglado se proporcionan en las cláusulas 18 y 19 de esta Especificación de directorio, así como en la Rec. UIT-T X.511 | ISO/CEI 9594-3.

M.2 Principios de diseño para el control de acceso básico

Esta subcláusula presenta varios de los principios de diseño más importantes utilizados en la arquitectura de control de acceso básico. Para facilitar las referencias, cada principio está etiquetado (por ejemplo, PR-1).

PR-1: Generalmente, los permisos asociados con **UserClasses (clases de usuario)** de mayor especificidad prevalecen sobre los permisos asociados con **UserClasses** de menor especificidad. Este principio se aplica cuando los permisos tienen el mismo nivel de precedencia. La especificidad, en este principio, mide cuán explícitamente un nombre de solicitante se relaciona con una especificación **UserClasses** particular; **allUsers (todos los usuarios)** tiene la más baja especificidad, mientras que **name (nombre)** es muy específico. Este principio se expone en 18.8.4 2) y facilita el tratamiento de situaciones en las que una política sobre los permisos por defecto (expresados en términos de **UserClasses** menos específicas) es anulada selectivamente por permisos asociados con una especificación **UserClasses** más específica.

PR-2: Generalmente, los permisos asociados con **ProtectedItems (ítems protegidos)** de una especificidad más alta prevalecen sobre permisos asociados con **ProtectedItems** de menor especificidad. Este principio se aplica cuando los permisos tienen el mismo nivel de precedencia y la misma especificidad **UserClasses**. Especificidad, en este principio, es una medida de cuán explícitamente la especificación **ProtectedItems** relaciona con el ítem exacto a se busca acceso. Por ejemplo, cuando el ítem protegido pretendido es un valor de atributo específico, **allAttributeValues (todos los valores de atributo)** y **allUserAttributeTypesAndValues (todos los tipos y valores de atributo de usuario)** son menos específicos que **attributeValue (valor de atributo)**. Este principio se expone en 18.8.4 3). Dicho principio facilita el tratamiento de situaciones en las que una política sobre permisos por defecto (expresados en términos de **ProtectedItems** menos específicos) es anulada selectivamente por permisos asociados con una especificación más **ProtectedItems** específica.

PR-3: El control de acceso básico es modelado como completamente independiente del proceso de resolución de nombre excepto en el caso de desreferenciación de alias. Con excepción de la desreferenciación de alias, decisiones de control de acceso sólo tienen lugar después de que el directorio ha localizado con éxito un DSA adecuado que contiene el ítem protegido pretendido. Un corolario de este principio es que el control de acceso básico no influye en la manera en que el directorio genera subpeticiones, ni en la manera en que el directorio efectúa la resolución de nombre asociada con subpeticiones (excepto en el caso de desreferenciación de alias).

PR-4: **Precedence (precedencia)** puede utilizarse para cumplir y hacer cumplir la relación entre una autoridad superior y una subordinada de modo que la superior pueda contraordenar controles establecidos por la subordinada. Por ejemplo: sea SE1 una subinserción de la inserción administrativa para una zona específica de control de acceso (ACSA), digamos ACSA-1; del mismo modo, sea SE2 una subinserción de la inserción administrativa para una zona interior de control de acceso (ACIA) dentro de ACSA-1. La autoridad de ACSA-1 puede especificar límites a la **Precedence** que se produce en SE2 de tal modo que **prescriptiveACI** en SE2 no pueda anular ACI prescriptiva en SE1. Asimismo, se pueden especificar límites a **Precedence** para **entryACI** (dentro de ACSA-1) de tal modo que **entryACI** no pueda contraordenar controles prescriptivos establecidos en SE1. Este principio facilita la implementación de delegación parcial de autoridad.

NOTA – La Especificación de directorio que se aplicará un método para limitar la precedencia para autoridades asociadas con zonas interiores. Sin embargo, la Especificación de directorio no define (ni describe) cómo ha de limitarse la precedencia.

PR-5: El control de acceso básico nunca concede acceso pasivamente; cada decisión de conceder acceso se basa en información de control de acceso explícitamente especificada. Un corolario de este principio es que la concesión de una forma de acceso nunca implica el permiso de efectuar otra forma de acceso. Estos principios son consecuentes con un principio de diseño de seguridad más general conocido por el *privilegio mínimo*.

PR-6: En ausencia de cualquier **prescriptiveACI**, **entryACI** o **subentryACI** en qué basar una decisión, la ACDF denegará el acceso. Si todos los demás parámetros de decisión permanecen iguales, las denegaciones prevalecen sobre las concesiones (por ejemplo, en una situación en que hay **ACIItems** que conceden y otros que deniegan (permiso de acceso), siendo iguales la **Precedence** y la especificidad, prevalece la denegación).

M.3 Introducción al ejemplo

La figura M.1 representa el subárbol DIT de una compañía ficticia, Z Computer Corporation (ZCC), que se utilizará en todo este ejemplo. La estructura de denominación de la figura M.1 sigue la sugerencia de la Rec. UIT-T X.521 | ISO/CEI 9594-7, anexo B. El nodo con nombre distinguido {C=US, O=ZCC} es una inserción administrativa y es el punto administrativo autónomo para ZCC; por tanto, define el comienzo de una zona administrativa autónoma (AAA, *autonomous administrative area*). El contenido de un AAA es un subárbol definido implícitamente que comienza en el punto administrativo autónomo y termina en nodos hoja, o cuando se encuentra otro punto administrativo autónomo. Puesto que no hay otros puntos administrativos autónomos por debajo de {C=US, O=ZCC}, la AAA contiene todos los nodos por debajo de {C=US} en la figura M.1. La clase de objeto estructural para {C=US, O=ZCC} es **organization**; tiene también una clase de objeto auxiliar de **certificationAuthority (autoridad de certificación)**. La clase de objeto auxiliar está presente para ayudar a soportar autorización fuerte donde se necesite.

Por debajo del punto administrativo autónomo hay tres subárboles: Administración (Admin); Investigación y Desarrollo (R&D, *Research & Development*); y Ventas (*Sales*). La raíz de cada subárbol es una inserción con una clase de objeto estructural **organizationalUnit** y una clase de objeto auxiliar **certificationAuthority**. El subárbol R&D contiene inserciones de clase de objeto estructural **organizationalUnit** que corresponden a sitios distantes, bajo los cuales aparecen objetos hoja de clase estructural **organizationalPerson**. Sólo unos pocos objetos representativos de la clase **organizationalPerson** se han hecho figurar. Todos los objetos de clase estructural **organizationalUnit** tienen una clase de objeto auxiliar de **certificationAuthority**. Todos los objetos de clase estructural **organizationalPerson** tienen una clase de objeto auxiliar de **strongAuthenticationUser**. Estas clases de objeto auxiliar ayudan a soportar autenticación fuerte donde sea necesario.

El objeto con nombre distinguido {C=US, O=ZCC, OU=Admin, CN=Ops} es de clase de objeto estructural **groupOfUniqueNames**; sus valores de atributo **uniqueMember** incluyen administradores de espacio de nombre. Un nombre allí contenido es {C=US, O=ZCC, OU=Admin, CN=Cauchy}. Hay otros dos de estos objetos: {C=US, O=ZCC,OU=R&D, CN=Ops} tiene miembros encargados de mantener inserciones en el subárbol R&D; y {C=US, O=ZCC, CN=Ops} tienen miembros encargados de inserciones que están inmediatamente subordinadas a {C=US, O=ZCC}. El usuario con nombre distinguido {C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley} es miembro de los dos últimos grupos.

Los dos trapezoides en la figura M.1 representan subárboles parciales cuyos detalles no son importantes para el ejemplo.

M.4 Política que afecta a la definición de zonas específicas e interiores

Para soportar el control de acceso básico, pueden establecerse dos tipos de zonas administrativas dentro de una AAA: zona específica de control de acceso (ACSA), y zona interior de control de acceso (ACIA). Se establece una zona administrativa de cualquiera de los dos tipos asignando el valor apropiado al atributo **administrative-role** en la inserción administrativa que ha de servir como vértice raíz para la zona. El contenido de una ACSA es un subárbol definido implícitamente que comienza en el vértice raíz y se extiende hacia abajo a los objetos hoja o hasta que se encuentra la raíz de otra ACSA. Asimismo, la frontera de una ACSA nunca se extiende más allá de la frontera más baja de la AAA contenedora. En el caso de una ACIA, la frontera más baja se presentará al encontrar una inserción de hoja o la frontera de la ACSA contenedora. Las ACIA jerarquizadas tienen la misma frontera más baja y esa frontera es igual que la frontera más baja de la ACSA contenedora.

ZCC tiene establecidas políticas que influyen en el número y los tipos de zonas administrativas que se necesitan dentro del AAA. La primera de esas políticas es que a la unidad organizacional conocida por Basic Research Consortium (BRC) le ha sido delegada la plena autoridad para establecer atributos operacionales de control de acceso para controlar inserciones en el subárbol con vértice raíz {C=US, O=ZCC, OU=R&D, OU=BRC). Para facilitar la implementación de la política, la raíz {C=US, O=ZCC, OU=R&D, OU=BRC} ha sido diseñada como una inserción administrativa con cometido administrativo **id-ar-accessControlSpecificArea**. La frontera inferior del ACSA resultante está definida implícitamente por la aparición de inserciones hoja.

NOTA – Un ACSA engloba el concepto de delegación completa de autoridad porque las decisiones de acceso dependen de ACI que aparece dentro del ACSA que contiene el ítem protegido pretendido y no son afectadas por ACI que aparece fuera del ACSA.

Además, el ACSA aquí descrita es el único ejemplar de delegación completa de autoridad de control de acceso dentro de ZCC. Sin embargo, una consecuencia del modelo administrativo del directorio es que cuando haya por lo menos un ACSA en un AAA, todos (y cada) uno de los objetos en AAA estarán contenidos en una (y solamente en una) ACSA. Esta exigencia se puede expresar más claramente en términos de en una la teoría de conjuntos, donde cada ACSA y la AAA asociada se perciben como conjuntos de inserciones: la intersección lógica de cada par de ACSA es un conjunto vacío y la unión lógica de todas las ACSA es igual a AAA. Por tanto, en el ejemplo, se necesita por lo menos un ACSA adicional para contener los objetos que están dentro del AAA pero fuera del subárbol BRC. Dado que sólo hay una situación de delegación completa dentro del AAA, la raíz de AAA es también el comienzo de una ACSA que contiene todas las inserciones en AAA excepto las pertenecientes al subárbol BRC.

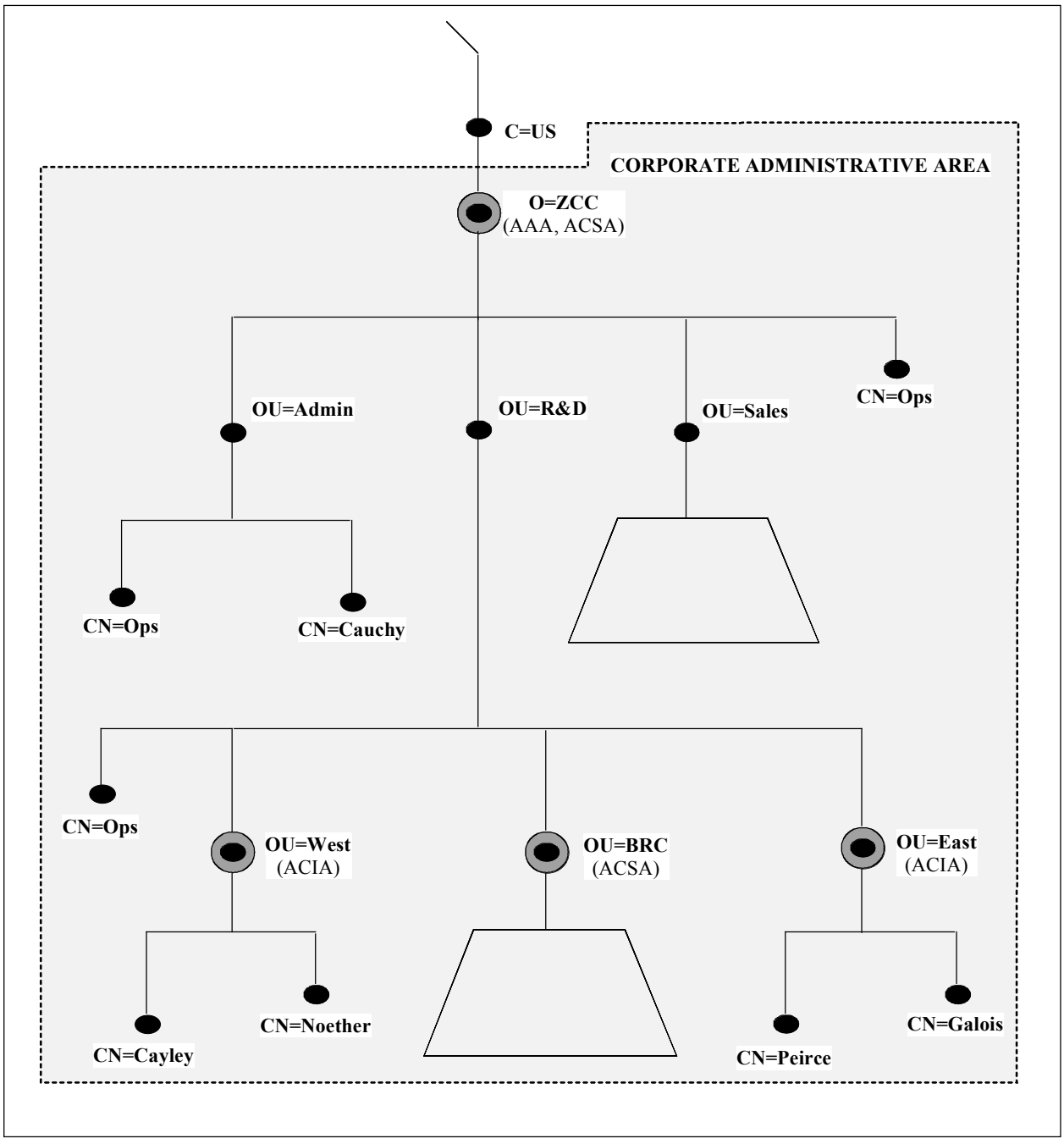


Figura M.1 – Rama del DIT para Z Computer Corporation (ZCC)

Las ACSA resultantes se representan como ACSA-1 y ACSA-2 en la figura M.2. Obsérvese también en la figura M.2 que, como las zonas administrativas son subárboles (definidos implícitamente), cada zona incluye su vértice raíz. El contexto de ACSA-1 se extiende hacia abajo desde su raíz hasta objetos hoja o hasta que se encuentra el vértice raíz de otra ACSA (como es el caso en {C=US, O=ZCC, OU=R&D, OU=BRC}). En este ejemplo no hay puntos administrativos autónomos por debajo de {C=US, O=ZCC}, por lo cual la frontera inferior del AAA está definida por objetos hoja. En la parte restante de este ejemplo se hará hincapié en el control de acceso dentro de ACSA-1 (no se hablará más del ACSA-2). Igualmente por razones de simplicidad, en este ejemplo no se examina el control subordinado por debajo de {C=US, O=ZCC, OU=Sales}.

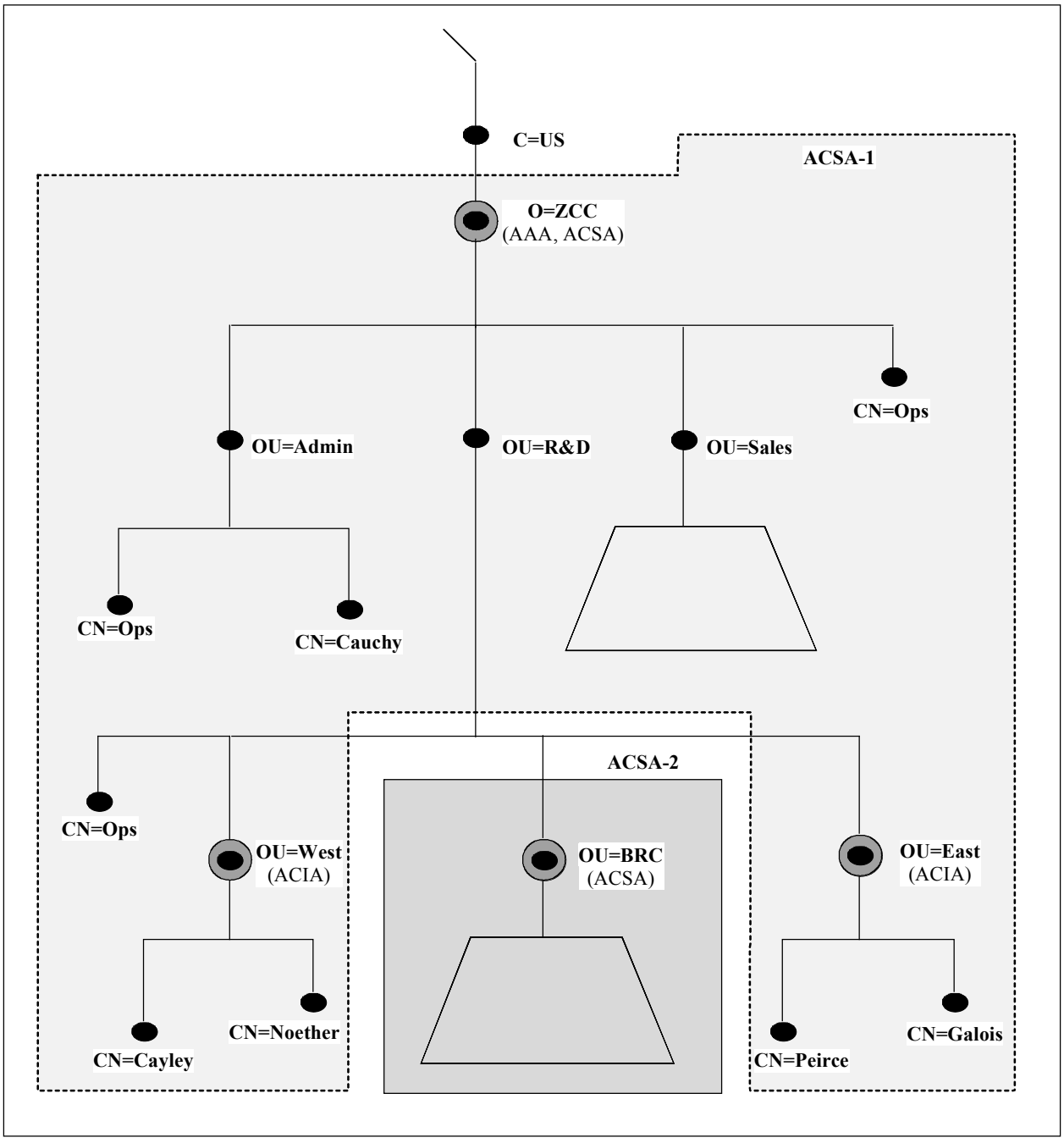


Figura M.2 – Zonas específicas de control de acceso

Otra política de ZCC que influye en la definición de zonas administrativas es que a la unidad organizacional R&D West le ha sido delegada autoridad parcial para atributos operacionales de control de acceso que afectan a las inserciones en el subárbol con vértice raíz {C=US, O=ZCC, OU=R&D, OU=West}. La mejor manera de aplicar esta política es haciendo de la raíz del subárbol R&D West un punto administrativo con cometido administrativo **id-ar-accessControlInnerArea**. Esto significa que los controles de acceso para ese subárbol serán, en general, una combinación de controles definidos en las subinserciones de la raíz de ese árbol y controles definidos en las subinserciones de la raíz de la ACSA contenedora (ACSA-1). El contexto de la ACIA resultante es un subárbol implícitamente definido con raíz en {C=US, O=ZCC, OU=R&D, OU=West} y que se extiende hacia abajo hasta que se encuentren objetos hoja. Dado que un ACIA es un subárbol, su contenido incluye el vértice raíz de ese subárbol.

Una política similar se sigue para la unidad organizacional R&D East. La ACIA correspondiente tiene el vértice raíz en {C=US, O=ZCC, OU=R&D, OU=East}. La figura M.3 representa las dos ACIA dentro de ACSA-1. ACIA para R&D West está etiquetada ACIA-1; la correspondiente a R&D East está etiquetada ACIA-2.

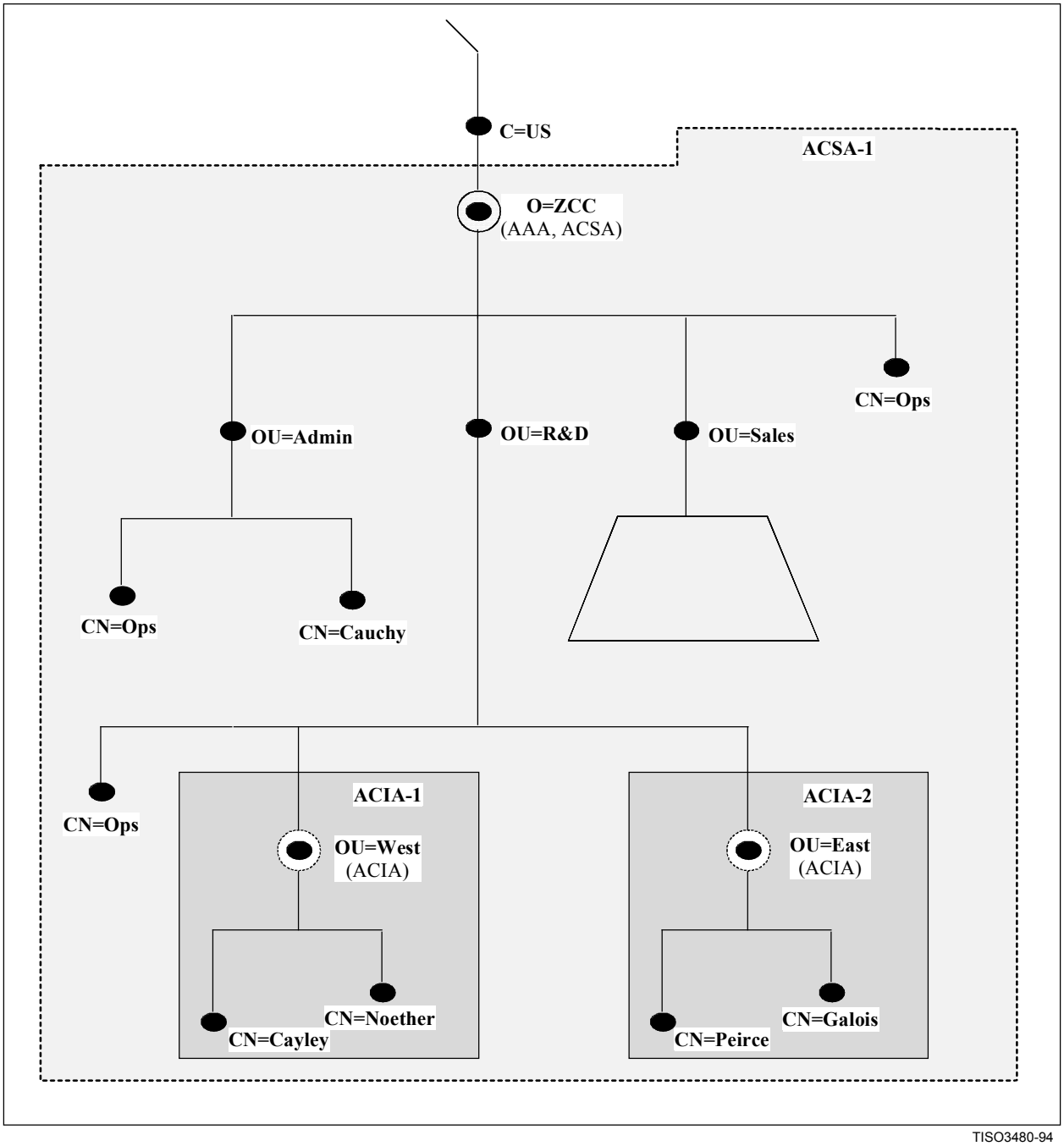


Figura M.3 – Interior del control de acceso

M.5 Política que influye en la definición de DACD

Los controles de acceso prescriptivos se definen en subinserciones (con la clase de objeto **accessControlSubentry**) de inserciones administrativas de control de acceso. Cada una de estas subinserciones tiene asociado un atributo **subtreeSpecification** que define el conjunto de inserciones en el alcance de la subinserción. Las inserciones contenidas en el alcance pueden formar un subárbol o un refinamiento de subárbol. En el contexto de control de acceso básico, el alcance de una subinserción de control de acceso se denomina dominio de control de acceso de directorio (DACD). Las autoridades de seguridad que utilizan el control de acceso básico deben tener cuidado de no confundir el concepto de zona administrativa con el concepto de DACD. Esta subcláusula comienza por un examen de las diferencias y relaciones entre zonas administrativas y DACD, después de lo cual se examina la política de ZCC que origina DACD individuales.

Las diferencias básicas entre las zonas administrativas y los DACD pueden resumirse como sigue:

- Una zona administrativa es un subárbol *definido implícitamente* que es la raíz de una inserción administrativa y se extiende hacia abajo como se describe en M.4. Se dice que tal zona está definida implícitamente porque no hay en el directorio un atributo normalizado que especifica su frontera; el DIT es examinado lógicamente para determinar la frontera de una zona administrativa. Una zona administrativa nunca es un refinamiento de subárbol.

  NOTA 1 – Una consecuencia de la manera de definir las zonas administrativas es que para cada inserción afectada por control de acceso básico deberá haber exactamente una ACSA que contenga la inserción (incluso si la inserción no está incluida en ningún DACD dentro de la ACSA).

- Un DACD es un árbol o refinamiento de subárbol definido explícitamente en el atributo **subtreeSpecification** de una subinserción con clase de objeto **accessControlSubentry**.

- La ACDF utiliza las ACSA y ACIA para determinar qué controles de acceso prescriptivos (es decir, qué subinserciones de control de acceso) podrían influir en el resultado de una decisión de control de acceso dada. Se utilizan las ACSA para aplicar la plena delegación de autoridad para control de acceso. Se utilizan las ACIA para aplicar la delegación parcial de autoridad para control de acceso.

- Se utiliza un DACD para especificar exactamente qué inserciones son afectadas por la subinserción de control de acceso asociada.

A continuación se hacen algunas observaciones sobre otros aspectos importantes de las zonas administrativas y los DACD, así como las relaciones entre ambos.

- Cada DACD está definido en una subinserción de una inserción administrativa particular que es, a su vez, el vértice raíz de alguna zona administrativa. Esta asociación entre un DACD, una subinserción, una inserción administrativa y una zona administrativa permite la determinación, para un DACD dado, de la *zona administrativa asociada* (véase M.5.1). El conjunto de inserciones contenidas en el DACD puede ser un conjunto propio o un conjunto impropio de las inserciones contenidas en la zona administrativa asociada.

  NOTA 2 – Los términos *subconjunto propio* y *subconjunto impropio* se han tomado de la teoría de conjuntos en matemáticas. El conjunto A es un subconjunto propio de B solamente si cada elemento de A es también un elemento de B y hay por lo menos un elemento de B que no es un elemento de A. El conjunto A es un subconjunto impropio de B solamente si ambos conjuntos contienen exactamente los mismos elementos.

- Cuando el conjunto de inserciones en el DACD es un subconjunto impropio de las inserciones en la zona administrativa asociada, se dice que el DACD y la zona administrativa son *congruentes*. No obstante, aunque se dé esa congruencia, el DACD y la zona administrativa continúan teniendo propósitos fundamentalmente diferentes (las zonas determinan qué subinserciones están autorizadas a afectar potencialmente el resultado de una decisión de control de acceso específica, mientras que cada DACD especifica exactamente qué inserciones son afectadas por los controles prescriptivos en una subinserción dada).

- El DACD nunca debe contener inserciones que están fuera de la zona administrativa asociada.

- La ACDF está diseñada de modo que sea robusta en el sentido de que si, la **subtreeSpecification** que define un DACD tiene dentro de su alcance inserciones que estén fuera de la zona administrativa asociada, las decisiones de control de acceso relativas a esas inserciones no serán afectadas. Este aspecto de robustez se manifiesta en el procedimiento de la ACDF para determinar qué subinserciones afectan potencialmente a una decisión dada [véanse 18.3.2 y 18.8.1 d)].

- Los DACD definidos en subinserciones de la misma inserción administrativa pueden superponerse libremente con la zona administrativa asociada común.

- Las ACSA nunca se superponen; cada ACIA está *debidamente jerarquizada* dentro de una ACSA. Por esto ha de entenderse que las inserciones en una zona circundada forman un subconjunto propio de las inserciones en la zona contenedora. Además, una ACIA puede contener una o más ACIA debidamente jerarquizadas.

– Cuando las zonas administrativas están jerarquizadas, los DACD asociados con una zona contenedora pueden superponerse libremente con los DACD asociados con cualquier zona contenida. La zona contenedora puede ser una ACSA o una ACIA, mientras que la zona contenida es siempre una ACIA.

Cada DACD está asociado con un aspecto de política que afecta a una o más inserciones o posibles inserciones. Las inserciones que son afectadas por un aspecto particular de política forman un DACD. El DACD para un aspecto particular de política debe estar asociado con la zona administrativa controlada por la autoridad responsable del cumplimiento de ese aspecto de política.

En el ejemplo hay varios aspectos de política de cuyo cumplimiento se encarga la autoridad que controla ACSA-1. Por ejemplo, hay controles "por defecto" que se aplican a objetos en la totalidad del ACSA-1. Se asigna a estos controles una precedencia y un nivel de especificidad que les permiten ser anulados fácilmente por otros controles prescriptivos o atributos **entryACI**. Hay también aspectos de política que se aplican solamente a subordinados inmediatos de {C=US, O=ZCC} (dentro de ZCC, tales inserciones se conocen por inserciones de *nivel administrativo*). Existen también otros aspectos de política que se aplican solamente a las inserciones que tienen la clase de objeto estructural **organizationalPerson**.

Todas las inserciones en ACSA-1 se incluyen en el DACD asociado con controles por defecto. El DACD se define por consiguiente de modo que sea un subárbol con vértice **base** en {C=US, O=ZCC} y una especificación **chop** que excluye el subárbol con raíz en {C=US, O=ZCC, OU=R&D, OU=BRC}. El DACD resultante es congruente con ACSA-1 y se muestra como el DACD-1 en la figura M.4.

NOTA 3 – Para el significado de *congruente* en este contexto, véase 18.3.2 g).

Asimismo, dentro de ACSA-1, el DACD para controlar inserciones de **organizationalPerson** es un refinamiento de subárbol con vértice **base** en {C=US, O=ZCC} y un **specificationFilter** que sólo incluye inserciones con **objectClass** de **organizationalPerson** (véase **subtree-refinement1** en K.2). Este DACD se muestra como DACD-2 en la figura M.4.

Un tercer DACD dentro de ACSA-1 se relaciona con el control de inserciones de nivel administrativo (es decir, subordinados inmediatos, que no son subinserciones, de la inserción raíz de organización). Este DACD es un subárbol (parte) con vértice **base** en {C=US, O=ZCC} y una especificación **chop** que incluye solamente los subordinados inmediatos, que no son subinserciones, de {C=US, O=ZCC}. Este DACD se muestra como DACD-5 en la figura M.4.

Para ACIA-1, un DACD deberá tratar un aspecto de política que ha sido delegado a la autoridad que controla la zona interior. La autoridad delegada influye solamente en subordinados de {C=US, O=ZCC, OU=R&D, OU=West}, por lo cual el DACD no es congruente con ACIA-1. El DACD se ha designado como DACD-3 en la figura M.4.

Para ACIA-2 sólo se requiere un DACD; sin embargo, la autoridad delegada influye en todas las inserciones en ACIA-2, por lo que el DACD es congruente con ACIA-2. El DACD se ha designado por DACD-4 en la figura M.4.

M.5.1 Zona administrativa asociada con cada DACD

Cada subinserción utilizada en el ejemplo se muestra en la figura M.4. Esta subcláusula recapitula la ubicación de cada subinserción e indica también la zona administrativa asociada con cada DACD.

DACD-1, DACD-2 y DACD-5 están definidos en subinserciones de {C=US, O=ZCC} que es la inserción administrativa que define el vértice raíz de ACSA-1. Por tanto, de estos tres DACD se dice que están *asociados con* ACSA-1. El nombre de la subinserción que define DACD-1 es {C=US, C=ZCC, CN=SE_DACD1}. Las otras subinserciones tienen nombres similares que indican el DACD que definen.

DACD-3 se define en una subinserción de {C=US, C=ZCC, OU=R&D, OU=West} que es la inserción administrativa que es el vértice raíz de ACIA-1. Por tanto, DACD-4 está asociado con ACIA-1.

DACD-4 está definido en una subinserción de {C=US, O=ZCC, OU=R&D, OU=East} que es la inserción administrativa que define el vértice raíz de ACIA-2. Por tanto, DACD-4 está asociado con ACIA-2.

M.6 Política expresada en atributos prescriptiveACI

Esta subcláusula contiene una descripción detallada de política de control de acceso aplicable a cada DACD en ACSA-1. La política examinada en este ejemplo debe considerarse una política parcial, simplificada para facilitar su presentación. En particular, no se examina la forma de controlar contraseñas, ya que, en general, las contraseñas constituyen un caso especial de control de acceso; tampoco se examinan los permisos *DiscloseOnError* ni *ReturnDN*.

La política examinada en esta subcláusula se presenta en términos de *fragmentos de política* que ayudan a comprender cómo utilizar atributos **prescriptiveACI** para hacer cumplir colectivamente la política global. A cada fragmento se da una etiqueta de referencia que se utiliza en subcláusulas posteriores; las etiquetas tienen la forma PF-*n* siendo *n* un entero secuencial. Para cada DACD hay también una indicación de la forma en que fragmentos de política aplicables podrían expresarse en términos de uno o más subinserciones (que contienen atributos **prescriptiveACI**).

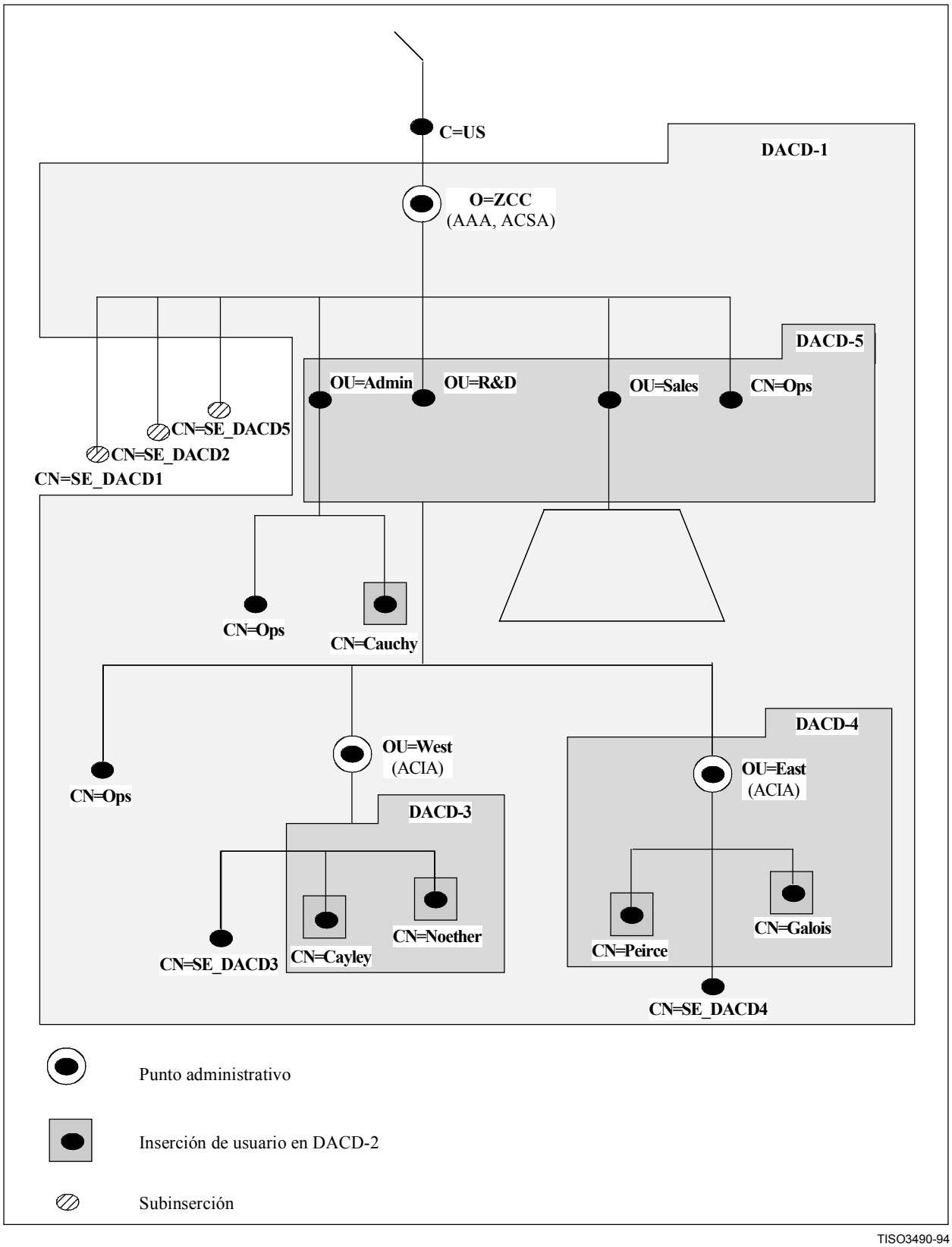


Figura M.4 – Dominios de control de acceso del directorio

M.6.1 ACI prescriptiva para DACD-1

Una de las finalidades principales de DACD-1 es hacer cumplir fragmentos de política relativos a un control de acceso "por defecto". Estos fragmentos de política proporcionan controles de resguardo que se aplican cuando no hay otro control que sea más alto en precedencia o especificidad. La especificidad se examina en los principios de diseño PR-1 y PR-2 en M.2.

ZCC ha formulado su política con respecto al acceso por el público en base a reglas de política por defecto que pueden ser anuladas para ciertas inserciones que necesitan un control más restrictivo. La política por defecto se enuncia en PF-1 y PF-2. Obsérvese que, de acuerdo con la política de ZCC, quienes aplican la política son responsables de asegurar que toda situación que se aparte de las reglas por defecto se verá sometida a reglas más restrictivas que éstas.

PF-1: Deberá distinguirse entre empleados y el público en general. Los derechos del público, en general, deberán limitarse de acuerdo con los apartados a) y b) que siguen; sin embargo, el acceso por el público puede ser restringido aún más con respecto a ciertas inserciones (el acceso nunca es menos restringido).

a) Las inserciones pueden ser localizadas por nombre común. La búsqueda por nombre común se permite para poder recurrir a una concordancia aproximada y nombres alternos. En particular, la búsqueda basada en número de teléfono no está autorizada para el público en general, pero se permite para quienes pertenecen a la organización. Los resultados de una búsqueda pueden revelar todos los valores de **commonName**.

b) Los únicos atributos públicos son **commonName**, **telephoneNumber**, componentes del **postalAttributeSet**, y **facsimileTelephoneNumber**.

PF-2: El acceso al público en general puede no ser autenticado, pero una persona del público en general deberá presentar una identidad.

ZCC utiliza también reglas de política por defecto para expresar su política general con respecto al acceso de empleado. Una situación que se aparte de las reglas de política por defecto puede verse sometida a reglas que sean más restrictivas o menos restrictivas que aquéllas. La política por defecto está formulada en PF-3 y PF-4.

PF-3: Los empleados, por lo general, tienen acceso a la mayor parte de los atributos de la mayor parte de las inserciones para leer y buscar.

PF-4: Se requiere autenticación simple para acceso de empleado que no modifica (de ninguna forma) el contenido de ACSA-1.

Hay también algunos fragmentos de política aplicables a DACD-1 que son tratados como reglas por defecto. En PF-5 y PF-6 se presentan dos ejemplos de esos fragmentos, relacionados con la administración de inserciones.

PF-5: {C=US, O=ZCC, CN=Cauchy} es "superusuario" autorizado para acceder a todos los datos y efectuar todas las operaciones necesarias.

PF-6: Se requiere autenticación fuerte para hacer cualquier modificación del contenido de ACSA-1.

Puede utilizarse una o más subinserciones de {C=US, O=ZCC} para aplicar los fragmentos de política para DACD-1. Cada subinserción tendría la misma **subtreeSpecification** con **base** de {C=US, O=ZCC} y una especificación **chop** para excluir el subárbol OU=BRC. Cada una de esas subinserciones contendrá también un atributo **prescriptiveACI** que aplica algún subconjunto de los fragmentos de política para DACD-1. A los fines del ejemplo, se supone que se utiliza una sola subinserción para captar todos los controles prescriptivos asociados con DACD-1 (no hay una razón técnica que obligue a usar más de una). Para facilitar las referencias, esta subinserción se designa por SE_DACD1. El atributo **prescriptiveACI** en SE_DACD1 tiene varios valores; el diseño de cada valor se examina en el resto de esta subcláusula.

El número de valores que aparecen en un atributo **prescriptiveACI** depende en parte del modo en que están agrupados los fragmentos de política, por razones de conveniencia, en valores **itemFirst** y **userFirst** (cualquiera de los dos estilos puede utilizarse en cualquier situación dada); depende también de cómo se va a tratar el control de acceso para los controles prescriptivos propiamente dichos.

Por ejemplo, para una parte de la implementación de PF-1 es necesario que a los usuarios públicos (esto es **allUsers**) se les conceda permiso para todo lo que se expresa a continuación:

a) *Browse* (*hojear*) para el ítem protegido **entry**;

b) *FilterMatch* (*concordar por filtro*) y *Read* (*leer*) para el ítem protegido **attributeType {commonName}**;

c) *FilterMatch* y *Read* para el ítem protegido **allAttributeValues {commonName}**.

Estos permisos son necesarios (pero no suficientes, véase la nota 1) para aplicar PF-1. Al haber tres ítems protegidos (**entry**, **attributeType** y **allAttributeValues**) y solamente una clase de usuarios (**allUsers**), lo más natural parece utilizar un solo **ACIItem** del estilo **userFirst**, pero podría utilizarse en su lugar el estilo **itemFirst**.

NOTA 1 – Los permisos examinados anteriormente serían también suficientes para permitir una búsqueda por **commonName** si se cumplen simultáneamente las dos condiciones siguientes:

a) no hay otros **ACIItems** pertinentes con una precedencia o especificidad más elevada que denieguen cualesquiera de los permisos para *Browse* o *FilterMatch* antes mencionados; y

b) no hay otros valores para el atributo **prescriptiveACI** en SE_DACD1 que denieguen cualquiera de los permisos para *BrowseRead* o *FilterMatch* antes mencionados.

Como otra posibilidad, podrían utilizarse tres **ACIItems** distintos: uno para cada uno de los ítems protegidos. Esta solución autoriza a cada **ACIItem** a tener un control de acceso aparte; cada uno tiene un **identificationTag** (**rótulo de identificación**) que es único (con respecto a los otros **identificationTag** para otros valores en el mismo atributo **prescriptiveACI**) y que puede ser referenciado en otro **ACIItem** donde el ítem protegido es **attributeValue** y la aserción de valor de atributo asociada especifica el **identificationTag** del valor que va a ser protegido. Obsérvese que utilizando **attributeValue** de esta forma se aprovecha la regla de concordancia por igualdad particular definida para atributos **prescriptiveACI** (véase 18.5.1). En una parte ulterior del ejemplo se examinan detalladamente ejemplos de ACI de protección.

A los efectos del ejemplo, se utilizan seis valores para el atributo **prescriptiveACI** en SE_DACD1, con el fin de implementar los fragmentos de política PF-1 a PF-4. A continuación se resume el diseño de cada uno de los tres valores.

NOTA 2 – Cada ítem protegido en los resúmenes de diseño que se presentan a continuación está provisto de una etiqueta para facilitar las referencias. La etiqueta se escribe entre paréntesis en cursiva (por ejemplo, *A1*, *A2*, *B1*).

NOTA 3 – En el ejemplo se utilizan cuatro niveles de precedencia: 10, 20, 30 y 40.

identificationTag: "Public Access – Enable entry access for List and Search on common name"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*A1*) **entry }**
grantsAndDenials: **{ grantBrowse }**
---

identificationTag: "Public Access – Enable filter access for Search"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*B1*) **attributeType { commonName },**
(*B2*) **allAttributeValues { commonName },**
(*B3*) **attributeType { objectClass },**
(*B4*) **allAttributeValues { objectClass } }**
grantsAndDenials: **{ grantFilterMatch }**
---

identificationTag: "Public Access – Enable entry access for Read and Compare operations"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*C1*) **entry }**
grantsAndDenials: **{ grantRead }**
---

identificationTag: "Public Access – Enable attribute access for interrogation operations"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*D1*) **attributeType { commonName,**
postalAttributeSet,
telephoneNumber,
facsimileTelephoneNumber } ,
(*D2*) **allAttributeValues { commonName,**
postalAttributeSet,
telephoneNumber,
facsimileTelephoneNumber } }
grantsAndDenials: **{ grantRead, grantCompare }**
---

identificationTag:	"Employee Access – Enable attribute access for interrogation operations"
Precedence:	10
UserClasses:	**subtree** with **base {** C=US, O=ZCC **}** and **chop** to exclude O=BRC subtree
authenticationLevel:	**simple**
ProtectedItems:	**{** (*E1*) **allUserAttributeTypesAndValues }**
grantsAndDenials:	**{ grantRead, grantCompare }**
	----------
identificationTag:	"Employee Access – Enable filter access for Search"
Precedence:	10
UserClasses:	**subtree** with **base {** C=US, O=ZCC **}** and **chop to** exclude O=BRC subtree
authenticationLevel:	**simple**
ProtectedItems:	**{** (*F1*) **allUserAttributeTypesAndValues }**
grantsAndDenials:	**{ grantFilterMatch }**

NOTA 4 – Los permisos para empleados son la unión de los permisos para el público y los permisos específicos a empleados. Los mencionados valores **ACIItem** para acceso de empleado están estrechamente acoplados a valores asociados con el acceso del público. Este estrecho acoplamiento podría evitarse, si fuera necesario, repitiendo cada uno de los valores para acceso del público (cada valor repetido tendría un nuevo **UserClasses** que especifica empleados solamente).

Hay otros dos valores del atributo que están relacionados con aplicación de la política en cuanto a la forma de administrar las inserciones (PF-5 y PF-6). Para simplificar la exposición, en este ejemplo se supone que los atributos de control de acceso son los únicos atributos operacionales presentes en la AAA. A continuación se resume el diseño de los dos valores.

identificationTag:	"Cauchy is superuser (Part 1)"
Precedence:	40
UserClasses:	**user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **}** **uniqueIdentifier** = 12345
authenticationLevel:	**strong**
ProtectedItems:	**{** (*G1*) **entry }**
grantsAndDenials:	**{ grantAdd, grantRead, grantRemove, grantBrowse, grantModify, grantRename}**
	----------
identificationTag:	"Cauchy is superuser (Part 2)"
Precedence:	40
UserClasses:	**user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **}** **uniqueIdentifier = 12345**
authenticationLevel:	**strong**
ProtectedItems:	**{** (*H1*) **allUserAttributeTypesAndValues,** (*H2*) **attributeType { entryACI },** (*H3*) **allAttributeValues { entryACI } }**
grantsAndDenials:	**{ grantAdd, grantRead, grantRemove, grantCompare, grantFilterMatch }**

Obsérvese que los dos valores anteriores son necesarios, pero no lo suficiente para que Cauchy sea un superusuario. No son suficientes porque no permiten el control de Cauchy en subinserciones del punto administrativo para ACSA-1; hay dos razones por las cuales esto es verdadero. En primer lugar, la ACI prescriptiva no se aplica a la subinserción en la cual aparece. En segundo lugar, la ACI prescriptiva colocada en una subinserción, digamos subinserción-1, no puede utilizarse para controlar subinserciones que son hermanas de la subinserción 1. Por tanto, es necesario colocar **subentryACI** en la inserción correspondiente al punto administrativo para ACSA-1, de modo que Cauchy pueda administrar su autoridad en las subinserciones del punto administrativo. La **subentryACI** necesaria se examina en M.7.

Obsérvese también que la autoridad concedida en los dos valores anteriores de ACI descriptiva permiten a Cauchy administrar el control completo de las subinserciones asociadas con los puntos administrativos que son subordinados del punto administrativo para ACSA-1.

M.6.2 ACI prescriptiva para DACD-2

DACD-2 está definido en una subinserción de la inserción administrativa para ACSA-1. DACD-2 se encarga de las inserciones de control con clase de objeto **organizationalPerson**. El siguiente fragmento de política es pertinente.

PF-7: Sólo miembros del grupo de administración de espacio de nombre {C=US, O=ZCC, 0U=Admin, CN=Ops}, pueden añadir, suprimir, o redenominar inserciones de usuario. Sin embargo, sólo se les permite añadir atributos obligatorios a una nueva inserción (una inserción que sólo contiene atributos obligatorios se llama una *inserción mínima*).

Los siguientes dos valores en el atributo **prescriptiveACI** de SE_DACD2 aplican PF-7.

NOTA – Por redenominación de inserciones, en contexto de PF-7, ha de entenderse una redenominación sin cambiar el superior inmediato. Para simplificar la exposición, en este ejemplo no se trata el caso más complicado en el que la redenominación comprende el cambio del superior inmediato de la inserción redenominada (y de sus subordinadas); en este caso hay que considerar permisos de *Import* y *Export*.

identificationTag:	"Minimal leaf entry administration (Part 1)"
Precedence:	20
UserClasses:	**userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel:	**strong**
ProtectedItems:	**{** (*J1*) **entry,** (*J2*) **attributeType {commonName, surname },** (*J3*) **allAttributeValues {commonName, surname } }**
grantsAndDenials:	**{ grantAdd, grantRemove }** **----------**

identificationTag:	"Minimal leaf entry administration (Part 2)"
Precedence:	20
UserClasses:	**userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel:	**strong**
ProtectedItems:	**{** (*K1*) **entry }**
grantsAndDenials:	**{ grantRename }** **----------**

M.6.3 ACI prescriptiva para DACD-3

DACD-3 se define en una subinserción de la inserción administrativa para ACIA-1 e implementa fragmentos de política con respecto a la política que ha sido parcialmente delegada a ACIA-1. Un ejemplo de esto es que la política para ACIA-1 con relación a **telephoneNumber** es diferente de la que se proporciona como política por defecto en DACD-1. En DACD-3, no se considera que **telephoneNumber** es un ítem de acceso por el público. Esto se refleja en el siguiente fragmento de política.

PF-8: Los únicos atributos públicos dentro de ACIA-1 son **commonName**, componentes de **postalAttributeSet**, y **facsimileTelephoneNumber**.

El siguiente valor en el atributo **prescriptiveACI** de la subinserción {C=US, O=ZCC, OU=R&D, OU=West, CN=SE DACD-3} implementa PF-8.

identificationTag:	"Delegated control of public access"
Precedence:	10
UserClasses:	**{ allUsers }**
authenticationLevel:	**none**
ProtectedItems:	**{** (*L1*)**attributeType { telephoneNumber } }**
grantsAndDenials:	**{ denyRead, denyCompare, denyFilterMatch }** **----------**

La organización R&D West tiene también autoridad delegada para aplicar una autoadministración de inserciones de clase de objeto **organizationalPerson**. Esta política se refleja en el siguiente fragmento.

PF-9: Los empleados de R&D West pueden administrar valores dentro de su propia inserción de directorio para los siguientes tipos de atributo: **telephoneNumber**, **commonName**, y **facsimileNumber**; sin embargo, no pueden modificar ni suprimir el valor de número de teléfono suministrado por la administración.

La primera parte de PF-9 se refleja en los dos **ACIItems** que siguen. La restricción en cuanto a la supresión de un valor particular de **telephoneNumber** se aplica utilizando **entryACI** como se describe en M.8.

identificationTag:	"Self-Administration of R&D West employee entries (Part 1)"	
Precedence:	20	
UserClasses:	**thisEntry**	
authenticationLevel:	**strong**	
ProtectedItems:	**{** (*M1*) **entry }**	
grantsAndDenials:	**{ grantModify }**	
	----------	
identificationTag:	"Self-Administration of R&D West employee entries (Part 2)"	
Precedence:	20	
UserClasses:	**thisEntry**	
authenticationLevel:	**strong**	
ProtectedItems:	**{** (*N1*) **attributeType {**	**commonName,**
		postalAttributeSet,
		telephoneNumber,
		facsimileTelephoneNumber },
	(*N2*)**allAttributeValues {**	**commonName,**
		postalAttributeSet,
		telephoneNumber,
		facsimileTelephoneNumber } }
grantsAndDenials:	**{ grantAdd, grantRemove }**	

PF-10: El grupo con miembros identificados en {C=US, O=ZCC, OU=R&D, CN=Ops} es responsable del mantenimiento general de atributos de usuario para inserciones en ACIA-1; sin embargo, no pueden modificar subinserciones ubicadas en ACIA-1.

La primera parte de esta política se refleja en el siguiente **ACIItem**:

identificationTag:	"R&D general administration (Part 1)"
Precedence:	20
UserClasses:	**userGroup {** C=US, O=ZCC, OU=R&D, CN=Ops **}**
authenticationLevel:	**strong**
ProtectedItems:	**{** (*P1*)**entry }**
grantsAndDenials:	**{ grantModify, grantAdd, grantRemove, grantBrowse,**
	grantRead, grantRename }
	----------
identificationTag:	"R&D general administration (Part 2)"
Precedence:	20
UserClasses:	**userGroup {** C=US, O=ZCC, OU=R&D, CN=Ops **}**
authenticationLevel:	**strong**
ProtectedItems:	**{** (*Q1*)**allUserAttributeTypesAndValues }**
grantsAndDenials:	**{ grantAdd, grantRemove, grantRead, grantFilterMatch,**
	grantCompare}

La restricción con respecto a subinserciones se trata no incluyendo valores **subentryACI**, en la inserción administrativa para ACIA-1 que permitan el acceso.

M.6.4 ACI prescriptiva para DACD-4

DACD-4 se define en una subinserción de la inserción administrativa para ACIA-2. Como tal, implementa fragmentos de política relativos a la política que ha sido parcialmente delegada a ACIA-2.

Para simplificar la exposición, DACD-4 no se tratará en lo sucesivo.

M.6.5 ACI prescriptiva para DACD-5

DACD-5 se define en una inserción en el punto administrativo para ACSA-1. Este DACD se utiliza para controlar el acceso para todos los subordinados inmediatos, que no son subinserciones, de la raíz operacional. En particular, se aplica la siguiente política.

PF-11: El grupo operaciones {C=US, O=ZCC, CN=Ops} es responsable de la administración de todas las inserciones que están inmediatamente subordinadas a {C=US, O=ZCC}.

PF-11 se expresa en los siguientes valores de **ACIItem**:

identificationTag: "Control of administrative level entries (Part 1)"
Precedence: 40
UserClasses: **userGroup {** C=US, O=ZCC, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*R1*) **entry }**
grantsAndDenials: **{ grantRead, grantBrowse, grantRemove, grantAdd, grantRename, grantModify }**
---

identificationTag: "Control of administrative level entries (Part 2)"
Precedence: 40
UserClasses: **userGroup {** C=US, O=ZCC, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*S1*) **allUserAttributeTypesAndValues,**
(*S2*) **attributeType { entryACI },**
(*S3*) **allAttributeValues { entryACI } }**
grantsAndDenials: **{ grantRead, grantRemove, grantAdd, grantCompare, grantFilterMatch }**

M.7 Política expresada en atributos subentryACI

M.7.1 subentryACI en la inserción administrativa para ACSA-1

PF-5 se manifiesta en una combinación de **prescriptiveACI** y **subentryACI** la **prescriptiveACI** asociada se ha descrito ya en M.6.1. Para que Cauchy pueda administrar las subinserciones del punto administrativo para ACSA-1 (y cualquiera subinserciones para puntos administrativos subordinados al punto administrativo para ACSA-1), es necesario colocar los siguientes valores de **subentryACI** en la inserción correspondiente al punto administrativo para ACSA-1.

identificationTag: "Cauchy is superuser (Part 3)"
Precedence: 40
UserClasses: **user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **}**
uniqueIdentifier = 12345
authenticationLevel: **strong**
ProtectedItems: **{** (*G1*)**entry }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantBrowse, grantModify, grantRename}**
---

identificationTag: "Cauchy is superuser (Part 4)"
Precedence: 40
UserClasses: **user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **}**
uniqueIdentifier = 12345
authenticationLevel: **strong**
ProtectedItems: **{** (*H1*) **allUserAttributeTypesAndValues,**
(*H2*) **attributeType { entryACI },**
(*H3*) **allAttributeValues { entryACI } }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantCompare, grantFilterMatch }**

M.7.2 subentryACI en la inserción administrativa para ACIA-1

Se coloca un atributo **subentryACI** en el vértice raíz de ACIA-1 para implementar el siguiente fragmento de política.

PF-12: El usuario con nombre común Cayley es responsable de la gestión de todas las **prescriptiveACI** definidas en ACIA-1.

Los dos valores siguientes en el atributo **prescriptiveACI** implementan PF-12.

identificationTag: "Cayley manages subentries in ACIA-1 (Part 1)"
Precedence: 20
UserClasses: **user {** C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*T1*) **entry }**
grantsAndDenials: **{ grantRead, grantBrowse, grantRemove, grantAdd, grantRename, grantModify }**

---

identificationTag: "Cayley manages subentries in ACIA-1 (Part 2)"
Precedence: 20
UserClasses: **user {** C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*U1*) **attributeType { prescriptiveACI },** (*U2*) **allAttributeValues { prescriptiveACI } }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantCompare, grantFilterMatch }**

M.8 Política expresada en atributos entryACI

PF-9 requiere que todos los empleados de R&D West estén autorizados a gestionar todos los valores de **telephoneNumber** en su inserción de directorio con la restricción de que no pueden modificar ni suprimir un valor particular suministrado por la administración. Para hacer cumplir la restricción, la administración añade un valor **entryACI** a cada inserción en el momento en que se añade a la inserción el número de teléfono restringido. El valor **entryACI** se resume como sigue:

identificationTag: "Restrict self-administration of telephone numbers"
Precedence: 30
UserClasses: **thisEntry**
authenticationLevel: **none**
ProtectedItems: **{** (*V1*) **attributeValue { telephoneNumber** = *value supplied by administration* **} }**
grantsAndDenials: **{ denyRemove }**

Obsérvese que como los usuarios no pueden modificar el atributo **entryACI** (por no ser parte de la autoadministración definida en PF-9), el control antes mencionado no puede ser anulado por un usuario.

El siguiente fragmento de política es un ejemplo de la utilización de **entryACI** para aplicar una autoadministración para una inserción de grupo.

PF-13: La inserción {C=US, O=ZCC, OU=Admin, CN=Ops} es una inserción de grupo "autoadministrada"; esto significa que cada miembro del grupo puede suprimir su nombre en el grupo o cambiar su nombre en el grupo. Los miembros no pueden suprimir ni redenominar el grupo como tal.

PF-13 se implementa mediante un atributo **entryACI** en la inserción {C=US, O=ZCC, OU=Admin, CN=Ops} con dos valores como se indica a continuación.

identificationTag: "self-administration of the Administrative Ops group (Part 1)"
Precedence: 30
UserClasses: **userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*W1*) **entry }**
grantsAndDenials: **{ grantModify }**

---

identificationTag: "self-administration of the Administrative Ops group (Part 2)"
Precedence: *medium*
UserClasses: **userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*X1*) **selfValue { uniqueMember } }**
grantsAndDenials: **{ grantRemove, grantAdd }**

M.9 Ejemplos de ACDF

M.9.1 Acceso del público a leer

Una persona que forma parte del público y cuyo nombre distinguido es {C=GB, O=XC, CN=Smith} intenta una operación de lectura pidiendo valores de número de teléfono para Cayley. Las decisiones de control de acceso para la operación se definen en la Rec. UIT-T X.511 | ISO/CEI 9594-3. Suponiendo que en la resolución de nombre no interviene una desreferenciación de alias, el primer punto de decisión es determinar si se concede permiso de *Leer* para la inserción pretendida; esta decisión se basa en las siguientes entradas a la ACDF:

- permiso solicitado: *Leer*;
- originador: {C=GB, O=XC, CN=Smith} sin identificador único;
- nivel de autenticación: ninguno;
- ítem protegido: inserción {C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley};
- tuplas indicadas en el cuadro M.1.

La inserción pretendida protegida está en el alcance de DACD-1, DACD-2, y DACD-3 (véase la figura M.4). No tiene **entryACI**. Los tres DACD aportan las tuplas (aplicables al originador especificado) mostradas en el cuadro M.1 al procedimiento ACDF descrito en 18.8.

La ACDF, después de descartar las filas no pertinentes termina considerando dos filas solamente: la fila 4 que concede *Read* sobre la inserción y la fila 13 que deniega *Read* sobre la inserción. En consecuencia, la ACDF niega el acceso.

NOTA – Para simplificar la exposición, en este ejemplo no se tratan permisos ni procedimientos asociados con condiciones de error. Sin embargo, en el anterior caso de acceso denegado, el comportamiento del DSA respondedor sería gobernado por 18.2.3 ó 18.4.1 e incluiría una nueva utilización de ACDF para determinar si se concede *DiscloseOnError* para la inserción pretendida.

Cuadro M.1

Nivel de autenticación	Ítem	Permiso	Conceder (G) o Denegar (D)	Precedencia	Nivel de autenticación
allUsers	(*A1*)**entry**	*Browse*	G	10	Ninguno
allUsers	(*B1*)**commonName** type	*FilterMatch*	G	10	Ninguno
allUsers	(*B2*)**commonName** values	*FilterMatch*	G	10	Ninguno
allUsers	(*B3*)**objectClass** type	*FilterMatch*	G	10	Ninguno
allUsers	(*B4*)**objectClass** values	*FilterMatch*	G	10	Ninguno
allUsers	(*C1*)**entry**	*Read*	G	10	Ninguno
allUsers	(*D1*)**commonName** type	*Read*	G	10	Ninguno
allUsers	(*D1*)**postalAttributeSet** type	*Read*	G	10	Ninguno
allUsers	(*D1*)**telephoneNumber** type	*Read*	G	10	Ninguno
allUsers	(*D1*)**facsimileTelephoneNo** type	*Read*	G	10	Ninguno
allUsers	(*D2*)**commonName** values	*Read*	G	10	Ninguno
allUsers	(*D2*)**postalAttributeSet** values	*Read*	G	10	Ninguno
allUsers	(*D2*)**telephoneNumber** values	*Read*	G	10	Ninguno
allUsers	(*D2*)**facsimileTelephoneNo** values	*Read*	G	10	Ninguno
allUsers	(*L1*)**telephoneNumber** type	*Read*	D	10	Ninguno
allUsers	(*L1*)**telephoneNumber** type	*Compare*	D	10	Ninguno
allUsers	(*L1*)**telephoneNumber** type	*FilterMatch*	D	10	Ninguno

M.9.2 Acceso del público a buscar

Una persona del público en general con el nombre distinguido {C=GB, O=XC, CN=Smith} intenta una operación de búsqueda pidiendo todos los valores de todos los atributos para todos los usuarios (**wholeSubtree**) subordinados al objeto base {C=US, O=ZCC, OU=R&D, OU=West}; el **filter** especifica **FilterItem equality**: **objectClass = organizationalPerson**. Los puntos de decisión de control de acceso para la operación se definen en 10.2.6 de la Rec. UIT-T X.511 | ISO/CEI 9594-3.

M.9.2.1 Comprobación de cada inserción en el ámbito de la búsqueda para comprobar la existencia de un permiso adecuado para la inserción

Para cada inserción en el ámbito de la búsqueda, suponiendo que en la resolución de nombre no haya una desreferenciación de alias, el primer punto de decisión es determinar si se ha concedido *Browse* para esa inserción. Para la primera inserción de esta naturaleza, las entradas a la ACDF son:

- permiso solicitado: *Browse*;
- originador: {C=GB, O=XC, CN=Smith};

- identificador único: ninguno;
- nivel de autenticación: ninguno;
- ítem protegido: inserción {C=US, O=ZCC, OU=R&D, OU=West};
- tuplas mostradas en el cuadro M.2.

Dado que la inserción que se comprueba está incluida en DACD-1 solamente, el conjunto inicial de tuplas reunidas por la ACDF es el mostrado en el cuadro M.2. Obsérvese que no hay que considerar ningún **entryACI**.

Como resultado del procedimiento de descarte de filas del cuadro M.2 seguido por la ACDF, la única fila retenida es la primera; en consecuencia, la ACDF concede el acceso.

De manera similar, la ACDF concederá *Browse* para cada inserción en el ámbito de la búsqueda.

Cuadro M.2

Usuario	Ítem	Permiso	Conceder (G) o Denegar (D)	Precedencia	Nivel de autenticación
allUsers	(*A1*)**entry**	*Browse*	G	10	Ninguno
allUsers	(*B1*)**commonName** type	*FilterMatch*	G	10	Ninguno
allUsers	(*B2*)**commonName** values	*FilterMatch*	G	10	Ninguno
allUsers	(*B3*)**objectClass** type	*FilterMatch*	G	10	Ninguno
allUsers	(*B4*)**objectClass** values	*FilterMatch*	G	10	Ninguno
allUsers	(*C1*)**entry**	*Read*	G	10	Ninguno
allUsers	(*D1*)**commonName** type	*Read*	G	10	Ninguno
allUsers	(*D1*)**postalAttributeSet** type	*Read*	G	10	Ninguno
allUsers	(*D1*)**telephoneNumber** type	*Read*	G	10	Ninguno
allUsers	(*D1*)**facsimileTelephoneNo** type	*Read*	G	10	Ninguno
allUsers	(*D2*)**commonName** values	*Read*	G	10	Ninguno
allUsers	(*D2*)**postalAttributeSet** values	*Read*	G	10	Ninguno
allUsers	(*D2*)**telephoneNumber** values	*Read*	G	10	Ninguno
allUsers	(*D2*)**facsimileTelephoneNo** values	*Read*	G	10	Ninguno

M.9.2.2 Comprobación de la aplicación del filtro

Para cada inserción en el alcance de la búsqueda para la que se concede *Browse*, el siguiente punto de decisión es determinar si se ha concedido *FilterMatch* sobre el atributo **objectClass**. Para la primera inserción de esta naturaleza, las entradas a la ACDF son:

- permiso solicitado: *Browse*;
- originador: {C=GB, O=XC, CN=Smith};
- identificador único: ninguno;
- nivel de autenticación: ninguno;
- ítem protegido: inserción {C=US, O=ZCC, OU=R&D, OU=West};
- tuplas mostradas en el cuadro M.2.

La ACDF descartará todas las filas del cuadro M.2 excepto la fila 4; en consecuencia, se concederá el acceso. Seguidamente, la operación de búsqueda comprobará si alguno de los valores del atributo **objectClass** son iguales a **organizationalPerson**. Puesto que la inserción que se está comprobando es una inserción administrativa, el **Filter (filtro)** evaluará a **FALSE**.

De manera similar, el **Filter** evaluará a **FALSE** para la inserción con CN=SE_DACD3.

Para las otras dos inserciones en el alcance de la búsqueda (CN=Cayley, CN=Noether), el **Filter** evaluará a **TRUE**. Para cada una de estas inserciones, la siguiente decisión de control de acceso es determinar si se concede *FilterItem* para el valor de atributo que hizo que **Filter** evaluara a **TRUE**. Dado que estas inserciones están incluidas en DACD-1 y DACD-2, y también en DACD-3, el conjunto inicial de tuplas introducidas en la ACDF viene dado por el cuadro M.1. La fila 5 del cuadro M.1 concede el acceso solicitado para ambas inserciones.

Por consiguiente, el resultado de buscar contiene información tomada de las inserciones para Cayley y Noether. Las demás decisiones de control de acceso para estas dos inserciones son esencialmente las mismas mostradas en el ejemplo de Leer por el público en M.9.1.

M.10 Control de acceso reglado

Para ilustrar el empleo del control de acceso reglado se incluye el siguiente ejemplo de utilización de reglas de seguridad (debe subrayarse de que se trata de un fin puramente ilustrativo que no representa necesariamente ninguna política real completa).

Los valores posibles de la etiqueta de seguridad forman un conjunto jerárquico: no marcados, no clasificados, limitados, confidenciales, secretos, muy secretos.

Los valores de la liberación son valores de clase máxima jerárquica: no marcados, no clasificados, limitados, confidenciales, secretos, muy secretos.

NOTA – Las comunidades pueden ampliar estas reglas para abarcar información privilegiada ulterior transportada con señales de privacidad o categorías de seguridad.

Las reglas de acceso son tales que:

a) Se concede el acceso si el nivel de liberación es mayor o igual que el nivel de la etiqueta.

b) Se deniega el acceso si el nivel de liberación es inferior al nivel de la etiqueta.

Anexo N

Combinaciones de tipos de DSE

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

El cuadro N.1 especifica un número de combinaciones de tipos de DSE (es decir, combinaciones de los bits denominados del atributo **dseType**) que probablemente ocurren cuando se aplica al DSA el modelo de información de DSA en ausencia de sombreado. Este cuadro se proporciona para aclarar el modelo de información de DSA. El soporte de estas u otras combinaciones de tipos de DSE no está prescrito como obligatorio en esta Especificación de directorio.

La primera columna del cuadro designa los tipos de DSE que no necesitan combinarse con cualquier o cualesquiera otros tipos de DSE para expresar la función de un DSE. Por ejemplo, puede encontrarse un DSE con el bit **entry** fijado únicamente. Las columnas de la segunda a la sexta indican, por una marca de control (✓), bits de tipo de DSE adicionales que pueden también ser fijados además del bit designado en la primera columna. Estos bits pueden ser fijados independientemente. Por ejemplo un DSE **entry** puede también tener fijados el bit **nssr**, los bits **cp** y **admPoint,** o varias otras combinaciones de los bits **admPoint**, **cp** y **nssr**. La última columna describe las diversas combinaciones de tipos de DSE indicadas en la fila correspondiente del cuadro.

El cuadro N.2 especifica un número de combinaciones adicionales de tipos de DSE que probablemente aparezcan cuando se produce sombreado. Como en el caso del cuadro anterior, la primera columna designa los tipos de DSE que no necesitan, en un DSA de sombra para el DSE, combinarse con cualquier o cualesquiera otros tipos de DSE para expresar la función de un DSE. Las columnas segunda a sexta indican, por una marca de control (✓), los bits de tipo de DSE que pueden ser también fijados, además de los designados en la primera columna. Estos bits pueden ser fijados independientemente.

Cuadro N.1 – Combinaciones de tipos de DSE definidas cuando no se emplea sombreado

Tipo de DSE	**admPoint**	**cp**	**supr**	**nssr**	**sa**	**Miembro de familia**	Comentarios
Root			✓	✓			DSE raíz para un DSA de primer nivel. DSA de primer nivel con nssr si el bit **nssr** está fijado. DSE raíz para un DSA que no es de primer nivel si el bit **supr** está fijado.
Glue							DSE adhesivo (glue).
Entry	✓	✓		✓		✓	DSE de inserción de objeto; es también un punto administrativo si el bit **admPoint** está fijado; es prefijo de contexto si el bit **cp** está fijado; es nssr si el bit **nssr** está fijado.
Alias		✓					DSE de inserción de alias.
Subentry					✓		DSE de subinserción.
subr							DSE de referencia subordinada; señala subinserciones subordinadas si el bit **sa** está fijado.
immSupr	✓					✓	Referencia superior inmediata.
xr							DSE de referencia cruzada.
NOTA – Los tipos DSE **subr** e **immSupr** pueden aparecer también (posiblemente con el bit adicional **admPoint**), pero no es conveniente representarlos en el cuadro. Las informaciones de subinserciones y de punto administrativo mantenidas por RHOBs se indican por la presencia del bit **rhob**.							

Cuadro N.2 – Combinaciones adicionales definidas de tipos de DSE cuando se emplea sombreado

Tipo de DSE	**admPoint**	**cp**	**supr**	**nssr**	**sa**	**Miembro de familia**	Comentarios
Root				✓			DSE raíz para un DSA de primer nivel de sombra con nssr.
Entry	✓	✓		✓		✓	DSE de inserción de objeto; es también un punto administrativo si el bit **admPoint** está fijado, un prefijo de contexto si el bit **cp** está fijado; nssr si el bit **nssr** está fijado.
Alias		✓					DSE de inserción de alias.
Subentry							DSE de subinserción.
subr					✓		DSE de referencia subordinada; señala subinserciones subordinadas si el bit **sa** está fijado.
immSupr	✓					✓	Referencia superior inmediata.
admPoint		✓		✓		✓	DSE de punto administrativo sin atributos de usuario (inserción no sombreada); es también: prefijo de contexto si el bit **cp** está fijado, **nssr** si el bit nssr está fijado.
Cp			✓	✓		✓	DSE de prefijo de contexto (inserción no sombreada); es también **nssr** si el bit nssr está fijado.
nssr						✓	DSE Nssr (inserción no sombreada)
NOTE **–** El bit **shadow** está fijado en todos los casos indicados en el cuadro (y por ello no se ha representado explícitamente). Al igual que en el cuadro N.1, los tipos de DSE **subr**, **immSupr** y **shadow** pueden aparecer también (posiblemente con el bit adicional **admPoint**). Por último, para DSE con los bits **subr** y/o **immSupr** fijados, los bits **entry** y **shadow** pueden aparecer también, ya que la información de inserción sombreada está superpuesta a la información de conocimiento mantenida por RHOB o sombreado.							

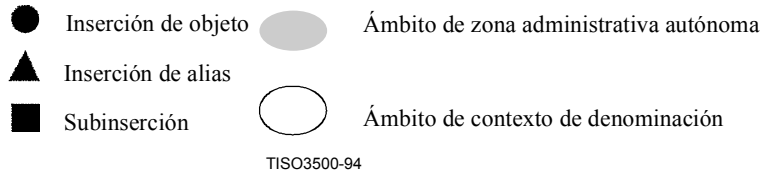
Anexo O

Modelado de conocimiento

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

El siguiente ejemplo ilustra un DIT hipotético, su posible correspondencia a tres DSA, y la información que los DSA tendrían que mantener (incluida la información de conocimiento) para sustentar la correspondencia.

En las figuras O.1 y O.2 que se insertan más adelante se emplean los siguientes símbolos.



La figura O.1 representa dicho DIT hipotético, que ha sido dividido en cuatro zonas administrativas autónomas: los casos degenerados de las inserciones individuales {C=WW} y {C=VV}, y los dos subárboles con raíces en {C=WW, O=ABC} y {C=VV, O=DEF}. Una inserción, {C=VV, O=DEF, OU=K}, es un alias de la inserción de objeto {C=WW, O=ABC, OU=I}.

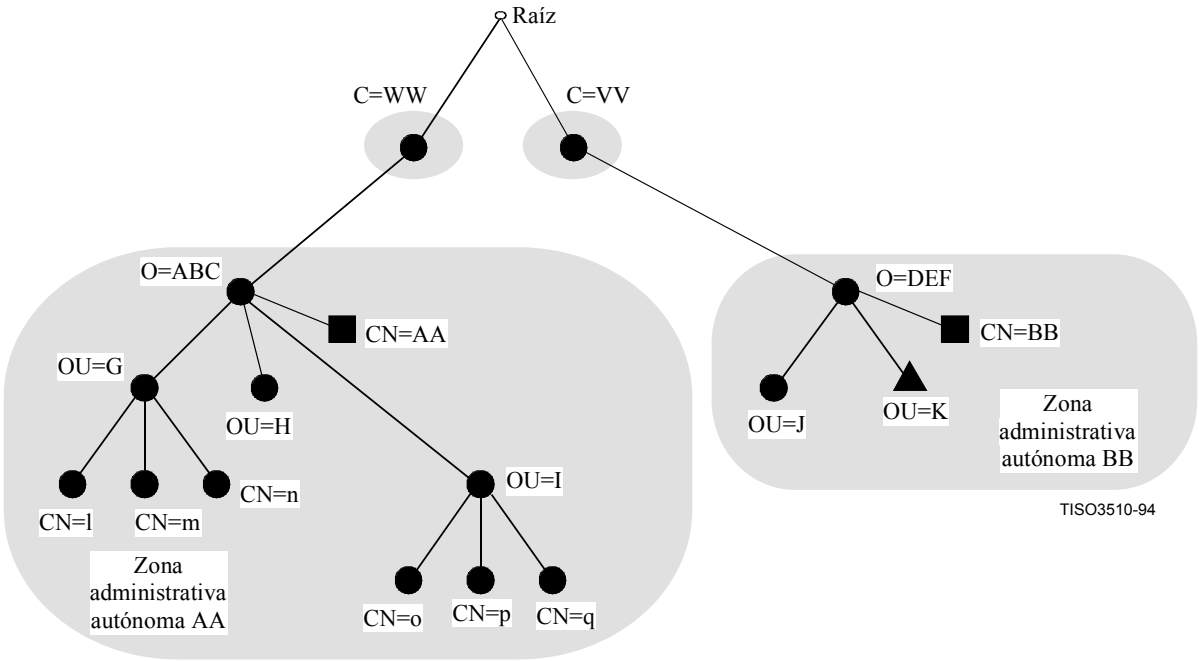


Figura O.1 – DIT hipotético

La figura O.2 representa la división del DIT hipotético en cinco contextos de denominación (A, B, C, D, y E) y su correspondencia con los tres DSA (DSA1, DSA2, y DSA3). En la figura, DSA1 contiene el contexto C, DSA2 contiene los contextos A, B, y E, y DSA3 contiene el contexto D.

El conocimiento contenido por los tres DSA es el siguiente: DSA1 emplea a DSA2 como su referencia superior y tiene una referencia subordinada no específica a DSA2 para información subordinada a {C=WW, O=ABC}. DSA2 es un DSA de primer nivel y mantiene una referencia subordinada a DSA1 para el contexto C y una referencia superior inmediata a éste para el contexto inmediatamente superior al contexto E. DSA2 mantiene una referencia subordinada a DSA3 para el contexto D. DSA3 también emplea a DSA2 como su referencia superior y tiene una referencia cruzada a DSA2 para el contexto E.

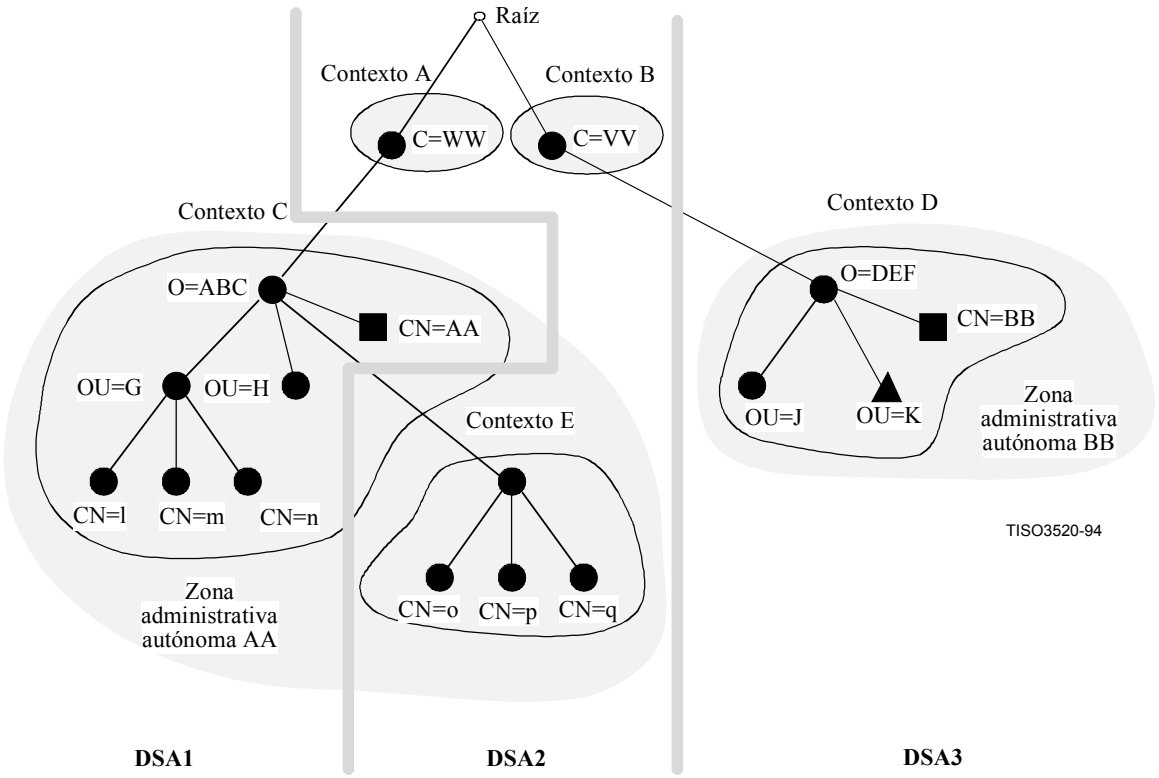
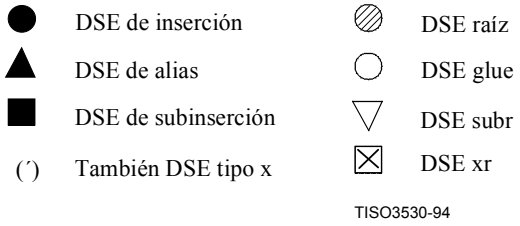


Figura O.2 – Correspondencia de DIT hipotético con tres DSA

Las figuras O.3 a O.6 representan la información contenida en cada uno de los DSA (es decir, el árbol de información (de) DSA para cada DSA) para soportar esta configuración. En estas figuras se emplean los siguientes símbolos.



La figura O.3 ilustra el árbol de información DSA de DSA1.

Dado que DSA1 no es un DSA de primer nivel, su DSE raíz contiene una referencia superior, que en este ejemplo, es el punto de acceso para DSA2. Este DSE es de tipo **root + supr**.

DSA1 contiene un DSE glue (adhesivo) para representar su conocimiento del nombre {C=WW}.

La zona administrativa autónoma AA está subdividida en dos contextos de denominación C y E, estando el contexto C contenido en DSA1. Para simplificar este ejemplo se ha supuesto que las zonas administrativas específicas relativas a información de control de acceso y de subesquema coinciden, y que hay un solo dominio de control de acceso y un solo subesquema para la totalidad de la zona administrativa autónoma. Una consecuencia de esto es que se requiere una sola inserción (de múltiples propósitos) para cada una de las zonas administrativas autónomas del ejemplo.

Para DSA1, el DSE en {C=WW, O=ABC}, que representa el punto administrativo para AA, el prefijo de contexto para C, y una referencia subordinada no específica para DSA2, es de tipo **entry + cp + admPoint + nssr**. La información operacional de la zona está contenida en la subinserción {C=WW, O=ABC, CN=AA}.

DSA1 tiene las siguientes inserciones contenidas en el contexto C: {C=WW, O=ABC, OU=G}, {C=WW, O=ABC, OU=H}, {C=WW, O=ABC, OU=G, CN=l}, {C=WW, O=ABC, OU=G, CN=m} y {C=WW, O=ABC, OU=G, CN=n}.

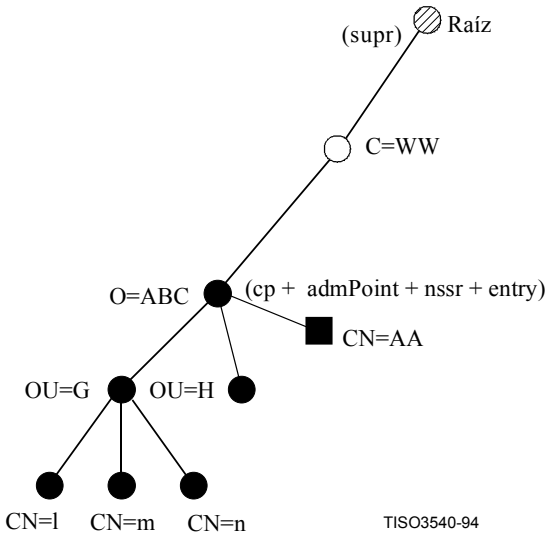


Figura O.3 – Árbol de información de DSA para DSA1

La figura O.4 ilustra un posible árbol de información DSA para DSA2.

En esta situación hipotética DSA2 es un DSA de primer nivel, por lo que su DSE raíz no contiene una referencia superior.

Las dos zonas administrativas autónomas degeneradas, {C=WW} y {C=VV}, se representan por las DSE de tipo **cp + entry + admPoint**.

El conocimiento de subordinado del DIT se representa por dos DSE de referencia subordinada, {C=WW, O=ABC} y {C=VV, O=DEF}. En el primer caso, esta DSE es de tipo **subr + admPoint + immSupr + rhob** por razones que se expondrán a continuación.

En la figura O.4, DSA2 se ha configurado suponiendo que una sola subinserción contiene la información operacional de zona referente a AA. Para esto es necesario que una copia de la subinserción esté presente en DSA2 (si se desea obtener un rendimiento razonable). Una forma de lograr esto es estableciendo NHOB entre DSA1 y DSA2 para mantener una copia de la subinserción. En este caso, la información operacional de zona está contenida en la DSE denominada {C=WW, O=ABC, CN=AA} que es de tipo **subentry + rhob**. El atributo **administrative-role** contenido en el DSE en {C=WW, O=ABC} es proporcionado a DSA2 desde DSA1 como parte de NHOB. Por esta razón, el DSE es de tipo **admPoint + rhob**.

Por último, el contexto de denominación E está contenido en el DSE de prefijo de contexto {C=WW, O=ABC, OU=I} que es de tipo **cp + entry** y los tres DSE de asiento {C=WW, O=ABC, OU=I, CN=o}, {C=WW, O=ABC, OU=I, CN=p} y {C=WW, O=ABC, OU=I, CN=q}.

En la figura O.5 se presenta otra posible manera de configurar el DSA2.

Esta configuración sólo se diferencia de la representada en la figura N.4 en cuanto al tratamiento de la información operacional de zona, lo que quizás se deba al deseo de no tener que mantener NHOB con DSA1.

La estrategia en este caso consiste en dividir AA (es decir, la información de control de acceso de dominio, y, de manera similar, la información de subesquema) en dos zonas administrativas autónomas que coinciden, una con el contexto C, y la otra con el contexto E.

En este caso el DSE de prefijo de contexto {C=WW, O=ABC, OU=I} deviene también un punto administrativo, siendo el DSE de tipo **cp + admPoint + entry**. La información operacional de zona reducida, en lugar de ser una subinserción sombreada suministrada por DSA1 como parte de NHOB, está contenida en la subinserción {C=WW, O=ABC, OU=I, CN=AA}.

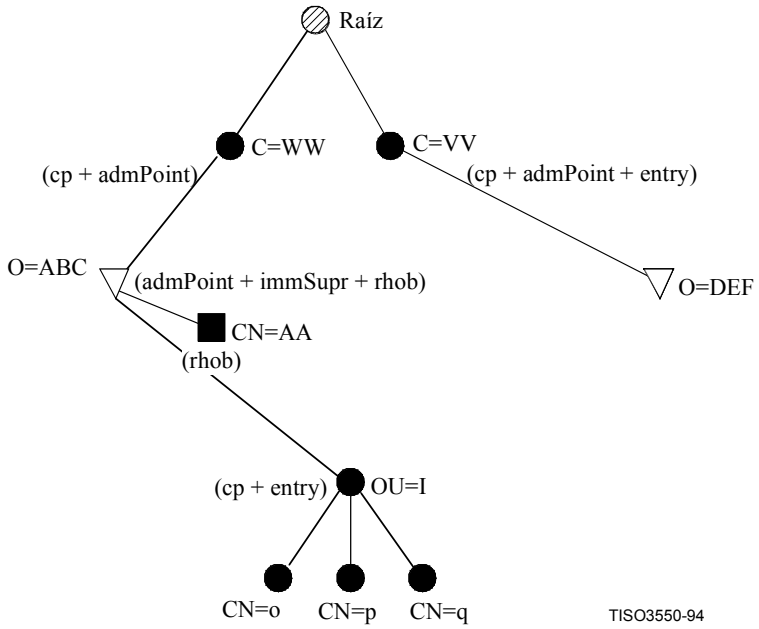


Figura O.4 – Árbol de información de DSA para DSA2

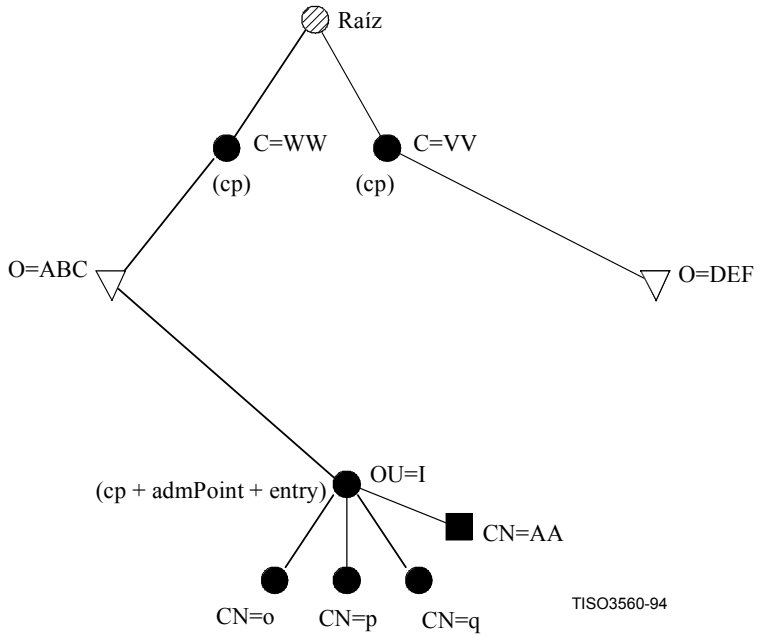


Figura O.5 – Árbol de información de DSA alternativo para DSA2

La figura O.6 ilustra el árbol de información de DSA para DSA3.

Como DSA1, DSA3 no es un DSA de primer nivel. Su DSE raíz contiene una referencia superior, que en este ejemplo es el punto de acceso para DSA2. Este DSE es de tipo **root + supr**.

DSA2 contiene una DSE glue (adhesivo) para representar su conocimiento del nombre {C=VV}.

La zona administrativa autónoma BB coincide con el contexto de denominación D. Para simplificar la exposición de este ejemplo, se ha supuesto, como en el caso de la zona administrativa autónoma AA, que las zonas administrativas específicas referentes a información de control de acceso y de subesquema coinciden y que hay un solo dominio de control de acceso y un solo subesquema para la totalidad de la zona administrativa autónoma. Por tanto, sólo se requiere una subinserción única (de múltiples propósitos) para cada una de las zonas administrativas autónomas del ejemplo.

Para DSA3 el DSE en {C=VV, O=DEF}, que representa el punto administrativo para BB y el prefijo de contexto para el contexto D, es de tipo **entry + cp + admPoint**. La información operacional de zona está contenida en la subinserción {C=VV, O=DEF, CN=BB}

DSA3 tiene una inserción de objeto y una inserción de alias contenidas en el contexto D: {C=VV, O=DEF, OU=J} (de tipo **entry**) y {C=VV, O=DEF, OU=K} (de tipo **alias** y que contiene un atributo **aliasedEntryName** que tiene el valor {C=WW, O=ABC, OU=I}).

Por último, DSA3 contiene una referencia cruzada al contexto E, un DSE de tipo **xr** con nombre {C=WW, O=ABC, OU=I}.

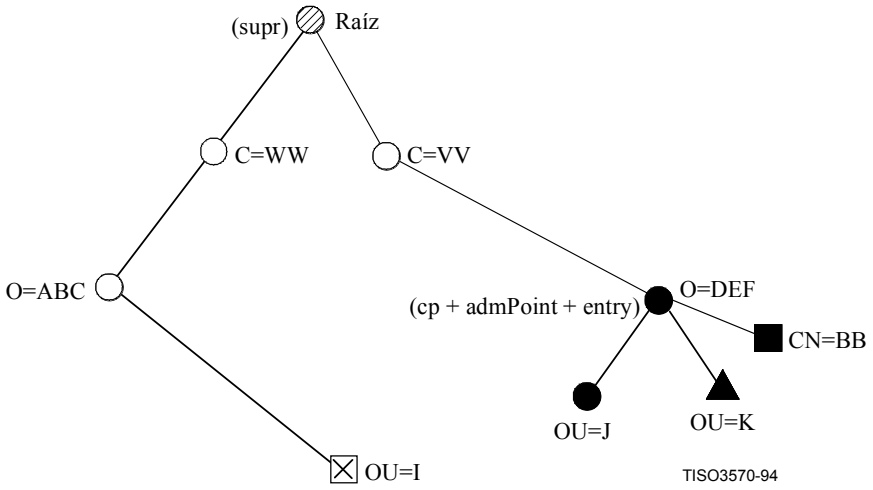


Figura O.6 – Árbol de información de DSA para DSA3

Anexo P

Nombres mantenidos como valores de atributo o usados como parámetros

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

Cuando un nombre se mantiene como un valor de atributo dentro de algún otro atributo o se traspasa como valor de atributo en algún intercambio (por ejemplo un puntero de alias), se plantea siempre la cuestión de si el nombre mantenido puede ser un nombre distinguido alternativo o el nombre distinguido primario, si puede contener valores alternativos y si puede incluir información de contexto. A lo largo de estas Especificaciones de directorio se mencionan, siempre que es necesario, limitaciones específicas. En general, no hay limitaciones en el nombre almacenado como valor de atributo. Sin embargo, para facilitar el interfuncionamiento con sistemas más antiguos y proporcionar resultados predecibles, se efectúan las siguientes sugerencias:

Cuando el valor de un atributo operacional sea el nombre de un objeto (por ejemplo **creatorsName**), el nombre deberá ser el nombre distinguido primario de ese objeto. Los valores alternativos y la información de contexto no son necesarios aunque pueden incluirse.

Cuando un tipo de atributo y un par de valores de un RDN dentro del nombre incluyan múltiples valores distinguidos empleando **valuesWithContext**, deberá utilizarse el valor distinguido primario como **value** en **AttributeTypeAndDistinguishedValue** de forma que sea predecible el interfuncionamiento con sistemas antiguos.

Cuando el valor de un atributo de usuario sea un nombre (por ejemplo miembro de un grupo de nombres, seeAlso), puede ser cualquier nombre distinguido alternativo, múltiples nombres alternativos o todos los nombres alternativos, aunque se recomienda la utilización del nombre distinguido primario de forma que sea posible el interfuncionamiento con sistemas más antiguos. Además, en general, no son útiles los contextos y valores alternativos si se incluyen en esos atributos de referenciación.

Cuando el atributo forme parte del árbol de información del DSA y se utilice en la resolución de nombres (por ejemplo referencias de conocimiento), deberá ser el nombre distinguido primario y cada RDN deberá transportar contextos y todos los valores distinguidos alternativos en el **AttributeTypeAndDistinguishedValue** para cada atributo, a fin de acentuar la resolución de nombres de forma que sea predecible el interfuncionamiento con sistemas más antiguos.

Anexo Q

Subfiltros

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

Un filtro puede convertirse en un conjunto de subfiltros por ampliación gradual utilizando las reglas de Morgan. (Estas reglas se aplican para la lógica de tres valores utilizada con el filtro.) Considérese que un filtro es un árbol en el que a cada **and**{}, **or**{}, **not**{}, corresponden nodos no-hoja y en el que cada nodo hoja es un elemento de filtro. Cada arco representa un elemento en **and**{}, **or**{}, **not**{}; en el caso de **not**{}, sólo puede haber un arco como ese.

En primer lugar, hacer que cada **not**{} avance hacia las hojas utilizando las reglas siguientes:

not{**and**{x,y,z}} es lo mismo que **or**{**not**{x}, **not**{y}, **not**{z}}
not{**or**{x,y,z}} es lo mismo que **and**{**not**{x}, **not**{y}, **not**{z}}
not{**not**{x}} es lo mismo que x

dejando que los **not** se apliquen directamente a los elementos de filtro.

A continuación, reducir el árbol combinando **and** y **or** y trasladar los **and** en la dirección de las hojas utilizando las reglas siguientes:

and{**and**{x,y,z}, p, q} es lo mismo que **and**{ x,y,z,p,q}
or{**or**{x,y,z}, p, q} es lo mismo que **or**{ x,y,z,p,q}
and{**or**{x,y,z}, p, q} es lo mismo que **or**{**and**{x,p,q}, **and**{y,p,q}, **and**{z,p,q}}
and{x,y,z} es lo mismo que **and**{cualquier orden de x,y,z}
or{x,y,z} es lo mismo que **or**{ cualquier orden de x,y,z}
and{} es VERDADERO, por lo que **or**{**and**{},x,y,z} es siempre VERDADERO y **and**{**and**{},x,y,z} es lo mismo que **and**{x,y,z}
or{} es FALSO, por lo que **and**{**or**{},x,y,z} es siempre FALSO y **or**{**or**{},x,y,z} es lo mismo que **or**{x,y,z}

NOTA – La notación {x,y,z} (etc.) aquí utilizada significa un conjunto de cero, uno o más miembros, tales como x, y y z.

Mediante la aplicación progresiva de estas reglas, el filtro llega a convertirse a una forma canónica:

or{**and**{p_1, p_2 … }, **and**{q_1, q_2 … } …}

donde cada p_i o q_i es un elemento de filtro F o un elemento de filtro negado **not**{F}.

Cada **and**{p_1, p_2 … } es entonces un subfiltro del filtro original.

Anexo R

Patrones de nombres de inserciones compuestas y su utilización

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

El concepto de nombre de miembro local se presenta en 9.3. Esta Especificación de directorio no impone ningún límite sobre la manera de atribuir los nombres más allá de lo determinado por las reglas de estructura. Sin embargo, el establecimiento en algunas situaciones de un patrón de denominación de los miembros de una familia es fundamental para conseguir el efecto deseado. En su forma sencilla, miembros de familia similares de inserciones compuestas diferentes podrían tener nombres de miembro local idénticos. Por ejemplo, un miembro de familia que posee un número telefónico y las características asociadas (utilización, tarifa, restricciones, etc.) podría tener el mismo nombre local en inserciones compuestas diferentes. Algo esencial cuando las inserciones compuestas son miembros de grupos jerárquicos (véase 7.13 de la Rec. UIT-T X.511 | ISO/CEI 9594-3). También se puede establecer un patrón haciendo que el RDN de un miembro de familia refleje cuál es la información que contiene ese miembro. Por ejemplo, una dirección para comunicaciones (un número telefónico o una dirección de correo electrónico) podría tener un RDN igual a { comAddressName = telephone1 } o { comAddressName = emailAddress3 }. Todos los, digamos, miembros de la familia del número telefónico se pueden localizar entonces efectuando una concordancia inicial de subcadenas en el RDN.

El ejemplo que sigue de utilización de patrón de nombres es también un ejemplo de la utilización de los atributos de control a los que hace referencia el componente regla de búsqueda **additionalControl** (véase 16.10.8). Ha de quedar claro que este ejemplo es justamente eso, un ejemplo, y no una especificación que pudiera implementarse o a la que puedan referirse de manera formal otras especificaciones. Se da únicamente para ilustrar cómo podría construirse un atributo de control y qué especificaciones podrían asociarse con dicho atributo.

Una regla de búsqueda controla el comportamiento de una búsqueda con una zona específica del DIT. Este servicio se adapta al usuario particular que accede. Sin embargo, los "propietarios" de las inserciones, por ejemplo, los abonados representados por inserciones de abonado, pueden tener sus propios requisitos, posiblemente con fundamento jurídico, respecto a cómo debería limitarse y ajustarse el servicio asociado con esa inserción particular. Esos requisitos individuales podrían ser como sigue:

a) La información de una inserción puede suministrarse en diferentes idiomas. Sin embargo, el propietario de la inserción puede pedir, por ejemplo, que la información de direccionamiento se devuelva en un idioma determinado con independencia del idioma que el usuario que accede emplea en la petición **search** y de lo que requiera dicho usuario. Esta función no puede ser proporcionada por la función de contexto.

b) El propietario de una inserción puede pedir que se devuelva una dirección falsa o alternativa incluso cuando el usuario que accede concuerde con la real.

c) Cuando el usuario que accede concuerde con un número telefónico, obtendrá la totalidad o una selección de los números telefónicos junto con la información asociada.

Esas constricciones y esos ajustes de tipo personal podrían aplicarse mediante el atributo de control **markingRules** de muestra. Lo previsto es que dicho atributo sea mantenido por una inserción o el antepasado de una inserción compuesta dentro de una zona administrativa específica de servicio. Su definición es como sigue:

```
markingRules ATTRIBUTE  ::= {
        WITH SYNTAX             MarkingRule
        USAGE                   directoryOperation
        ID                      id-oa-xx }

MarkingRule  ::=  SEQUENCE {
        searchRules             SEQUENCE SIZE (1 .. MAX) OF INTEGER  OPTIONAL,
        markingStrands          [0]  Filter                                DEFAULT  and : { },
        localName               [1]  SEQUENCE SIZE (1 .. MAX) OF FilterItem  OPTIONAL,
        explicitUnmark          [2]  Filter                                OPTIONAL }
```

Un valor del atributo de control **markingRules** representa una regla para marcar y desmarcar miembros de inserciones compuestas con los que se ha concordado durante la evaluación de búsqueda y para eliminar de la salida inserciones concordantes que no son de la familia.

El componente **searchRules** indica a qué reglas de búsqueda aplica el valor particular de este atributo. Si la regla de búsqueda directora tiene un **id** igual a uno de los valores de este componente, deberá aplicarse la remarcación especificada por este atributo de control. Una regla de búsqueda determinada se puede representar en varios valores de este tipo de atributo. Si falta el componente, se aplica la regla de marcación para todas las reglas de búsqueda.

El componente **markingStrands** sólo tiene importancia si el **familyGrouping** durante la concordancia fue **strand (ramal)** o **multiStrand (multiramal)**. Ello indica la condición que debe estar presente para una posible marcación de ramales. El filtro de este componente se evalúa con relación a cada ramal cuyos miembros han sido marcados, todos ellos, como miembros participantes como resultado de la concordancia del filtro de búsqueda. Evalúa a VERDADERO si al menos un ramal evalúa a VERDADERO. La concordancia sigue las mismas reglas que se especifican en 7.8 de la Rec. UIT-T X.511 | ISO/CEI 9594-3. Si este componente está ausente, pasa por defecto a un filtro que evalúa siempre a VERDADERO.

El componente **localName** sólo tiene importancia si la **familyGrouping** durante la concordancia fue **strand** o **multiStrand** y **markingStrands** evalúa a VERDADERO. A continuación indica qué ramales deberán tener sus miembros de familia marcados como miembros participantes seleccionando cero o más miembros de la familia. Un miembro de la familia se elige si su nombre de miembro local tiene el mismo número de RDN que el número de elemento de filtro en este componente y si cada elemento de filtro concuerda uno por uno con el RDN correspondiente. Un elemento de filtro concuerda con un RDN si concuerda con una AVA de ese RDN. Cualquier ramal que pase por un miembro de la familia seleccionado tiene todos sus miembros de familia marcados como participantes.

El componente **explicitUnmark** especifica un filtro que, si concuerda con una inserción o un miembro de familia, hace que esa inserción o ese miembro de familia sea desmarcado de manera explícita. La desmarcación explícita sólo tiene importancia en el caso de inserciones y miembros de familia que se han seleccionado para devolverlos en el resultado de una búsqueda. Si un miembro de familia es desmarcado de manera explícita y si la agrupación de la familia durante la concordancia del filtro de búsqueda no fue **entryOnly**, todas las inserciones de la familia subordinadas al miembro desmarcado explícitamente son también desmarcadas de manera explícita. La desmarcación explícita de una inserción ajena a la familia significa suprimir esa inserción del resultado como si no hubiera sido concordada. La desmarcación explícita de un miembro de la familia significa que ese miembro no se incluirá en el resultado.

La evaluación del atributo de control **markingRules** se lleva a cabo en un proceso de dos fases.

La primera sólo se efectúa si **familyGrouping** durante el proceso de concordancia fue **strand** o **multiStrand** y **familyReturn** en la selección de información de inserción no es **contributingEntriesOnly**.

En esa primera fase, se consideran únicamente las inserciones compuestas que han sido concordadas durante la evaluación del filtro de búsqueda y cumplen todas las condiciones siguientes:

a) el antepasado mantiene un atributo de control **markingRules**;

b) para la regla de búsqueda directora son aplicables uno o más valores, que incluyen el componente **localName**.

A continuación se marcan más miembros como miembros participantes según lo especificado más arriba.

En la segunda fase, se controlan todos los miembros de la familia marcados ahora como miembros participantes y todas las inserciones ajenas a la familia para verificar la presencia del tipo de atributo de control **markingRules**, y a continuación para verificar si el atributo tiene uno o más valores aplicables para la regla de búsqueda directora. Si tal es el caso, se evalúa el componente **explicitUnmark**, caso de que esté presente. Si evalúa a VERDADERO para un miembro de la familia, se desmarca explícitamente, es decir, no se marca como participante ni como contribuyente. Todos los miembros de la familia subordinados son también desmarcados explícitamente de manera similar. Si se trata de una inserción ajena a la familia, la desmarcación explícita tiene el mismo efecto que si la inserción no hubiera sido concordada por el filtro de búsqueda.

Anexo S

Índice alfabético de definiciones

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

En este anexo se indican en orden alfabético todos los términos definidos en esta Especificación de directorio con una referencia a la cláusula en que han sido definidos.

Anexo T

Enmiendas y correcciones

(Este anexo no es parte integrante de esta Recomendación | Norma Internacional)

Esta edición de la presente Especificación de directorio incluye las siguientes enmiendas:

- Enmienda 1 para mejoras destinadas a soportar la Recomendación UIT-T F.510 y la correspondencia de protocolos de directorio con TCP/IP.
- Enmienda 6 para inserciones conexas en el directorio.

La edición de esta Especificación de directorio abarca los siguientes corrigenda técnicos que subsanan defectos indicados en los siguientes Informes de Defectos: 173, 179, 189, 205, 211, 228, 229, 230, 242, 250, 255, 259,260, 267 y 269.

SERIES DE RECOMENDACIONES DEL UIT-T

Serie A	Organización del trabajo del UIT-T
Serie B	Medios de expresión: definiciones, símbolos, clasificación
Serie C	Estadísticas generales de telecomunicaciones
Serie D	Principios generales de tarificación
Serie E	Explotación general de la red, servicio telefónico, explotación del servicio y factores humanos
Serie F	Servicios de telecomunicación no telefónicos
Serie G	Sistemas y medios de transmisión, sistemas y redes digitales
Serie H	Sistemas audiovisuales y multimedios
Serie I	Red digital de servicios integrados
Serie J	Redes de cable y transmisión de programas radiofónicos y televisivos, y de otras señales multimedios
Serie K	Protección contra las interferencias
Serie L	Construcción, instalación y protección de los cables y otros elementos de planta exterior
Serie M	RGT y mantenimiento de redes: sistemas de transmisión, circuitos telefónicos, telegrafía, facsímil y circuitos arrendados internacionales
Serie N	Mantenimiento: circuitos internacionales para transmisiones radiofónicas y de televisión
Serie O	Especificaciones de los aparatos de medida
Serie P	Calidad de transmisión telefónica, instalaciones telefónicas y redes locales
Serie Q	Conmutación y señalización
Serie R	Transmisión telegráfica
Serie S	Equipos terminales para servicios de telegrafía
Serie T	Terminales para servicios de telemática
Serie U	Conmutación telegráfica
Serie V	Comunicación de datos por la red telefónica
Serie X	**Redes de datos y comunicación entre sistemas abiertos**
Serie Y	Infraestructura mundial de la información y aspectos del protocolo Internet
Serie Z	Lenguajes y aspectos generales de soporte lógico para sistemas de telecomunicación

Ginebra, 2002