ICS 35.100.70
L 79

中华人民共和国国家标准

GB/T 16264.2—2008/ISO/IEC 9594-2:2005
代替 GB/T 16264.2—1996

信息技术 开放系统互连 目录 第2部分:模型

Information technology—Open Systems Interconnection—The Directory—Part 2:Models

(ISO/IEC 9594-2:2005 Information technology—Open Systems Interconnection—The Directory:Models,IDT)

2008-08-06 发布　　2009-01-01 实施

中华人民共和国国家质量监督检验检疫总局
中国国家标准化管理委员会　发布

目　次

前　言

GB/T 16264《信息技术　开放系统互连　目录》包括以下10个部分：

——第1部分：概念、模型和服务的概述；

——第2部分：模型；

——第3部分：抽象服务定义；

——第4部分：分布式操作规程；

——第5部分：协议规范；

——第6部分：选定的属性类型；

——第7部分：选定的客体类；

——第8部分：公钥和属性证书框架；

——第9部分：复制(待发布)；

——第10部分：公用目录管理机构的系统管理用法(待发布)。

本部分是GB/T 16264的第2部分。

本部分等同采用ISO/IEC 9594-2:2005《信息技术　开放系统互连　目录　模型》，仅有编辑性修改。

本部分代替GB/T 16264.2—1996。

本部分与GB/T 16264.2—1996的差异在于增加了下列各项内容：

——目录管理模型；

——目录管理和操作信息模型；

——目录模式；

——目录服务管理；

——DSA模型；

——DSA信息模型；

——DSA操作框架；

——扩充了GB/T 16264.2—1996中各章条内容。

本部分的附录A～附录H是规范性附录，附录I～附录T是资料性附录。

本部分由中华人民共和国信息产业部提出。

本部分由全国信息技术标准化技术委员会归口。

本部分起草单位：中国电子技术标准化研究所。

本部分主要起草人：徐冬梅、冯惠、张翠、胡顺。

本部分于1996年首次发布，本次为第一次修订。

引　言

GB/T 16264 的本部分连同本标准其他部分是为方便信息处理系统之间的互连以提供目录服务而制定的。所有这些系统的集合，连同它们所拥有的目录信息可被视为一个整体，被称为“目录”。目录所拥有的信息，总称为目录信息库(DIB)，典型地被用于方便客体之间的通信、与客体的通信或有关客体的通信等，这些客体如应用实体、个人、终端和分布列表等。

目录在开放系统互连中扮演了重要角色，其目标是，在它们自身的互连标准之外做最少的技术约定的情况下，允许下述各种信息处理系统之间的互连：

——来自不同生产厂商；

——具有不同的管理；

——具有不同的复杂程度，以及

——有不同的年代。

本部分为目录提供一组不同的模型，作为其他部分参考的框架。这些模型包括总体(功能)模型、管理机构模型、提供关于目录信息的目录用户和管理用户视图的通用目录信息模型、通用目录系统代理(DSA)和 DSA 信息模型以及操作框架和安全模型。

例如，通用目录信息模型描述了客体的相关信息如何分组，形成该客体的目录条目以及那些信息如何为客体提供名(称)。

通用 DSA 和 DSA 信息模型以及操作框架为目录的分布提供了支持。

本部分提供了通用目录信息模型的专门化以支持对目录模式的管理。

本部分提供了一些基础框架，在此框架基础上，其他标准化组织和业界论坛可以定义工业配置集。在这些框架中定义为可选的许多特性，可通过配置集的说明，在某种环境下作为必选特性来使用。目前 ISO/IEC 9594 的第 5 版是原有国际标准第 4 版的修订和增强，但不是替代。在系统实现时仍可以声明为符合第 4 版。然而，在某些方面，将不再支持第 4 版(即不再消除一些报告上来的错误)。建议在系统实现时尽快符合第 5 版。

第 5 版详细定义了目录协议的第 1 版和第 2 版。

第 1 版和第 2 版仅定义了协议第 1 版。本版本(第 5 版)中定义的许多服务和协议被设计为可运行在第 1 版下。然而，一些增强的服务和协议，如署名错误，只有包含在操作中的所有的目录条目都协商支持协议第 2 版时才可运行。无论协商的是哪一版，第 5 版中所定义的服务之间的差异和协议之间的差异，除了那些特别分配给第 2 版的外，都可以使用 GB/T 16264.5—2008 中定义的扩展规则调节。

本部分使用术语“第 1 版系统”来指遵循国际标准第 1 版的所有系统，即 ISO/IEC 9594:1990 版本；本部分使用术语“第 2 版系统”来指遵循国际标准第 2 版本的所有系统，即 ISO/IEC 9594:1995 版本；本部分使用术语“第 3 版系统”来指遵循国际标准第 3 版的所有系统，即 ISO/IEC 9594:1998 版本；本部分使用术语“第 4 版系统”来指遵循国际标准第 4 版的所有系统，即 ISO/IEC 9594:2001 版本的第 1 部分到第 10 部分；本部分使用术语“第 5 版系统”来指遵循国际标准第 5 版的所有系统，即 ISO/IEC 9594:2005 版本。

GB/T 16264—1996 是参照 ISO/IEC 9594:1990 而制定的。我国没有制定与国际标准第 2 版、第 3 版、第 4 版对应的国家标准。本部分提到的版本号是指国际标准的版本号。

附录 A 是规范性附录，总结了本标准中 ASN.1 客体标识符的用法。

附录 B 是规范性附录，提供了 ASN.1 模块。

附录 C 是规范性附录，提供了子模式管理模式的 ASN.1 定义。

附录 D 是规范性附录，提供了服务管理的 ASN.1 模块定义。

附录 E 是规范性附录，提供了基本访问控制的 ASN.1 模块定义。

附录 F 是规范性附录，提供了一个 ASN.1 模块定义，该模块中包含了所有与 DSA 操作属性类型相关的定义。

附录 G 是规范性附录，提供了一个 ASN.1 模块定义，该模块中包含了与操作绑定管理操作相关的所有定义。

附录 H 是规范性附录，提供了一个 ASN.1 模块定义，该模块中包含了与增强的安全相关的所有定义。

附录 I 是资料性附录，对与树型结构相关的数学术语进行了概述。

附录 J 是资料性附录，描述了在设计名(称)时可以考虑的一些准则。

附录 K 是资料性附录，对模式的不同方面提供了一些示例。

附录 L 是资料性附录，提供了与基本访问控制许可相关的语义方面的概述。

附录 M 是资料性附录，提供了基本访问控制用法的一个扩展示例。

附录 N 是资料性附录，描述了一些 DSA 特定条目的组合。

附录 O 是资料性附录，提供了对知识建模的框架。

附录 P 是资料性附录，描述了一个名(称)是可替代辨别名还是主辨别名，它是否可以包括可替代值以及它是否可以包括上下文信息等的判断准则。

附录 Q 是资料性附录，描述了子过滤器的概念。

附录 R 是资料性附录，描述了如何对家族成员进行命名的建议和示例。

附录 S 是资料性附录，介绍了命名概念和相关的考虑。

附录 T 是资料性附录，以字母表顺序列出了本部分中定义的术语。

信息技术 开放系统互连 目录
第2部分:模型

第一篇:综述

1 范围

GB/T 16264的本部分中定义的模型为GB/T 16264的其他部分提供了一个概念框架和术语框架,这些部分规定了目录的各种特性。

功能模型和管理机构模型定义了目录进行功能分布和管理分布的方法。通用DSA和DSA信息模型以及操作框架也是为支持目录分布而提供的。

通用目录信息模型分别从目录用户和主管部门用户的角度描述了DIB的逻辑结构。事实上,但在这些模型中,目录是分布的而不是集中的,是不可见的。

本部分提供了通用目录信息模型的专门化以支持对目录模式的管理。

GB/T 16264—2008的其他部分使用了本部分中定义的概念,对通用信息和DSA模型进行专门化定义以提供特定的信息、特定的DSA和操作模型,用以支持特定的目录能力(如复制):

a) 目录提供的服务(在GB/T 16264.3—2008中定义)是根据信息框架的概念而描述的:这就允许所提供的服务在某种程度上独立于DIB的物理分布;

b) 规定了目录的分布式操作(在GB/T 16264.4—2008中定义),以此可以提供上述服务,并因此可以维护该逻辑信息结构,即使该DIB实际上是高度分布的;

c) 规定了目录的组成部分所提供的用以提高目录整体性能的复制能力(在ISO/IEC 9594-9中定义)。

安全模型为规范访问控制机制建立了一个框架。它为在DIT的特定部分内有效标识访问控制方案提供了一种机制,并且定义了三种灵活的、特定的访问控制方案,这些模式广泛适用于各种不同的应用以及使用风格。通过使用如密码和数字签名等机制,安全模型还为保护目录操作的保密性和完整性提供了一个框架,它使用了ISO/IEC 9594-8中定义的鉴别框架以及GB/T 18237.1—2000中定义的通用高层安全工具。

DSA模型为目录组件操作规范建立了一个框架,包括:

a) 目录功能模型描述了目录是如何表示为一个或多个组件(每个组件为一个DSA);

b) 目录分布模型描述了一些原则,按照这些原则,DIB条目和条目拷贝可以在DSA间分布;

c) DSA信息模型描述了目录用户以及DSA内存储的操作信息的结构;

d) DSA操作框架描述了为获得特定目标(例如影像),构造DSA之间特定合作形式定义的方法。

2 规范性引用文件

下列文件中的条款通过GB/T 16264的本部分的引用而成为本部分的条款。凡是注日期的引用文件,其随后所有的修改单(不包括勘误的内容)或修订版均不适用于本部分,然而,鼓励根据本部分达成协议的各方研究是否可使用这些文件的最新版本。凡是不注日期的引用文件,其最新版本适用于本部分。

GB/T 9387.1—1998 信息技术 开放系统互连 基本参考模型 第1部分:基本模型(idt ISO/IEC 7498-1:1994)

GB/T 9387.2—1995 信息处理系统 开放系统互连 基本参考模型 第2部分:安全体系结构(idt ISO 7498-2:1989)

GB/T 9387.3—1995 信息处理系统 开放系统互连 基本参考模型 第3部分:命名与编址(idt ISO 7498-3:1989)

GB/T 16262.1—2006 信息技术 抽象语法记法一(ASN.1) 第1部分:基本记法规范(ISO/IEC 8824-1:2002,IDT)

GB/T 16262.2—2006 信息技术 抽象语法记法一(ASN.1) 第2部分:信息客体规范(ISO/IEC 8824-2:2002,IDT)

GB/T 16262.3—2006 信息技术 抽象语法记法一(ASN.1) 第3部分:约束规范(ISO/IEC 8824-3:2002,IDT)

GB/T 16262.4—2006 信息技术 抽象语法记法一(ASN.1) 第4部分:ASN.1规范的参数化(ISO/IEC 8824-4:2002,IDT)

GB/T 16264.1—2008 信息技术 开放系统互连 目录 第1部分:概念、模型和服务的概述(ISO/IEC 9594-1:2005,IDT)

GB/T 16264.3—2008 信息技术 开放系统互连 目录 第3部分:抽象服务定义(ISO/IEC 9594-3:2005,IDT)

GB/T 16264.4—2008 信息技术 开放系统互连 目录 第4部分:分布式操作规程(ISO/IEC 9594-4:2005,IDT)

GB/T 16264.5—2008 信息技术 开放系统互连 目录 第5部分:协议规范(ISO/IEC 9594-5:2005,IDT)

GB/T 16264.6—2008 信息技术 开放系统互连 目录 第6部分:选定的属性类型(ISO/IEC 9594-6:2005,IDT)

GB/T 16264.7—2008 信息技术 开放系统互连 目录 第7部分:选定的客体类(ISO/IEC 9594-7:2005,IDT)

GB/T 17965—2000 信息技术 开放系统互连 高层安全模型(idt ISO/IEC 10745:1995)

GB/T 18237.1—2000 信息技术 开放系统互连 通用高层安全 第1部分:概述、模型和记法(idt ISO/IEC 11586-1:1996)

GB/T 18794.2—2002 信息技术 开放系统互连 开放系统安全框架 第2部分:鉴别框架(idt ISO/IEC 10181-2:1996)

GB/T 18794.3—2003 信息技术 开放系统互连 开放系统安全框架 第3部分:访问控制框架(ISO/IEC 10181-3:1996,IDT)

ISO/IEC 9594-8:2005 信息技术 开放系统互连 目录:公钥和属性证书框架

ISO/IEC 9594-9:2005 信息技术 开放系统互连 目录:复制

ISO/IEC 9594-10:2005 信息技术 开放系统互连 目录:公用目录管理机构的系统管理用法

ISO/IEC 9834-1:2005 信息技术 开放系统互连 OSI登记机构的操作规程:一般规程和ASN.1客体标识符树的顶级弧

ISO/IEC 10021-2 信息技术 消息处理系统(MHS):总体结构

ISO/IEC 10021-4:2003 信息技术 消息处理系统(MHS):消息传送系统 抽象服务定义和规程

CCITT 建议 X.800:1991 CCITT应用的开放系统互连安全体系结构

IETF RFC 3377:2002 轻量级目录访问协议(v3):技术规范

3 术语和定义

下列术语和定义适用于GB/T 16264的本部分。

3.1 通信定义

本部分使用 GB/T 16264.5 中定义的术语：

a) 应用实体 *application entity*；

b) 应用层 *application Layer*；

c) 应用进程 *application process*。

3.2 基本目录定义

本部分使用 GB/T 16264.1 中定义的术语：

a) 目录 *directory*；

b) 目录访问协议 *Directory Access Protocol*；

c) 目录信息库 *Directory Information Base*；

d) 目录操作绑定管理协议 *Directory Operational Binding Management Protocol*；

e) 目录系统协议 *Directory System Protocol*；

f) (目录)用户 *(Directory) user*。

3.3 分布式操作定义

本部分使用 GB/T 16264.4 中定义的术语：

a) 访问点 *access point*；

b) 分等级操作绑定 *hierarchical operational binding*；

c) 名(称)解析 *name resolution*；

d) 非特定分等级操作绑定 *non-specific hierarchical operational binding*；

e) 相关的分等级操作绑定 *relevant hierarchical operational binding*。

3.4 复制定义

下列术语在 ISO/IEC 9594-9 中定义：

a) 高速缓冲拷贝 *cache-copy*；

b) 使用者引用 *consumer reference*；

c) 条目拷贝 *entry-copy*；

d) 主 DSA *master DSA*；

e) 主影像 *primary shadowing*；

f) 复制区 *replicated area*；

g) 复制 *replication*；

h) 次影像 *secondary shadowing*；

i) 影像使用者 *shadow consumer*；

j) 影像提供者 *shadow supplier*；

k) 影像的 DSA 特定条目 *Shadowed DSA-Specific Entry*；

l) 影像 *shadowing*；

m) 提供者引用 *supplier reference*。

本部分定义的术语在每一章开头的适当位置。为便于参考，在本部分的附录 T 中提供了这些术语的索引。

4 缩略语

GB/T 16264 的本部分使用如下缩写词：

ACDF	访问控制决策功能	(Access Control Decision Function)
ACI	访问控制信息	(Access Control Information)
ACIA	访问控制内部区	(Access Control Inner Area)

ACSA	访问控制特定区	(Access Control Specific Area)
ADDMD	公共目录管理域	(Administration Directory Management Domain)
ASN.1	抽象语法记法一	(Abstract Syntax Notation One)
AVA	属性值断言	(Attribute Value Assertion)
BER	(ASN.1)基本编码规则	((ASN.1) Basic Encoding Rules)
DACD	目录访问控制域	(Directory Access Control Domain)
DAP	目录访问协议	(Directory Access Protoco)
DIB	目录信息库	(Directory Information Base)
DISP	目录信息影像协议	(Directory Information Shadowing Protocol)
DIT	目录信息树	(Directory Information Tree)
DMD	目录管理域	(Directory Management Domain)
DMO	域管理组织	(Domain Management Organization)
DOP	目录操作绑定管理协议	(Directory Operational Binding Management Protocol)
DSA	目录系统代理	(Directory System Agent)
DSE	DSA 特定条目	(DSA-Specific Entry)
DSP	目录系统协议	(Directory System Protocol)
DUA	目录用户代理	(Directory User Agent)
HOB	分等级操作绑定	(Hierarchical Operational Binding)
LDAP	轻量级目录访问协议	(Lightweight Directory Access Protocol)
NHOB	非特定分等级操作绑定	(Non-specific Hierarchical Operational Binding)
NSSR	非特定下级引用	(Non-Specific Subordinate Reference)
PRDMD	专用目录管理域	(Private Directory Management Domain)
RDN	相关可辨别名	(Relative Distinguished Name)
RHOB	相关的分等级操作绑定	(适当情况下,指 HOB 或 NHOB)
	(Relevant Hierarchical Operational Binding (a HOB or NHOB,as appropriate))	
SDSE	影像 DSE	(Shadowed DSE)

5 约定

术语“目录规范(或本目录规范)”指的是 GB/T 16264.2—2008。术语“系列目录规范”指的是 GB/T 16264 的所有部分。

本目录规范使用术语“第 1 版系统”来指遵循系列目录规范第 1 版的所有系统,即 1988 年版本的 CCITT X.500 系列建议书和 GB/T 16264—1996 版本。本目录规范使用术语“第 2 版系统”来指遵循系列目录规范第 2 版本的所有系统,即 1993 版本的 ITU-T X.500 系列建议书和 ISO/IEC 9594:1995 版本。本目录规范使用术语“第 3 版系统”来指遵循系列目录规范第 3 版的所有系统,即 1997 版本的 ITU-T X.500 系列建议书和 ISO/IEC 9594:1998 版本。本目录规范使用术语“第 4 版系统”来指遵循系列目录规范第 4 版的所有系统,即 ISO/IEC 9594:2001 年版本的第 1 到第 10 部分。

本目录规范使用术语“第 5 版系统”来指遵循系列目录规范第 5 版的所有系统,即 GB/T 16264—2008 版本的第 1 到第 7 部分以及 ISO/IEC 9594:2005 年版本的第 8 到第 10 部分。

本目录规范使用粗体字体来表示 ASN.1 符号。若在常规文本中要表示 ASN.1 的类型和值时,为了区别于常规文本,使用了粗体字表示。为了表示过程的语义而引用过程名时,为了区别于常规文本,使用了粗体字表示。访问控制许可使用斜体字表示。

第二篇:目录模型概述

6 目录模型

6.1 定义

本部分使用下列术语和定义:

6.1.1

公共管理机构 administrative authority

域管理组织的一个代理机构,负责目录管理的不同方面。

6.1.2

公共目录管理域 administration directory management domain;ADDMD

由管理部门管理的目录管理区(DMD)。

注:术语"管理"指公共远程通信管理部门或提供公共远程通信服务的组织。

6.1.3

目录管理和操作信息 directory administrative and operational information

出于目录管理和操作的目的而使用的信息。

6.1.4

DIT 域 DIT domain

由 DSA 所拥有的构成一个 DMD 的全球 DIT 的一部分。

6.1.5

目录管理域 directory management domain;DMD

由一个单独的组织所管理的一个或多个 DSA 以及零个或多个 DUA 的集合。

6.1.6

域管理组织 domain management organization

管理一个 DMD(以及相应的 DIT 域)的组织。

6.1.7

目录用户信息 directory user information

关于用户及其应用的感兴趣的信息。

6.1.8

目录系统代理 directory system agent;DSA

一个 OSI 应用进程,是目录的一个组成部分。

6.1.9

(目录)用户 (directory) user

目录的端用户,即访问目录的实体或人员。

6.1.10

目录用户代理 directory user agent;DUA

OSI 的一个应用进程,它在访问目录过程中代表某一个用户。

注:DUA 可能还会提供一个本地范围的便利工具以帮助用户构成请求并解释响应。

6.1.11

客户机 LDAP client LDAP

一个应用进程,表示通过轻量级目录访问协议(LDAP)来访问目录的用户。

6.1.12

LDAP 请求者 LDAP requestor

一个 DSA,能够通过轻量级目录访问协议(LDAP)来发起请求,并且能够理解和处理 LDAP 的

响应。

6.1.13

LDAP 响应者　LDAP responder

一个 DSA，能够理解并响应轻量级目录访问协议(LDAP)的请求。

6.1.14

LDAP 服务器　LDAP server

一个组成目录的应用进程，它拥有 DIB 的一部分，并且能够通过轻量级目录访问协议(LDAP)对请求进行响应。

6.1.15

专用目录管理域　private directory management domain；PRDMD

由公共主管部门之外的组织所管理的一个目录管理域(DMD)。

6.2　目录及其用户

目录是一个信息仓库，这个仓库被称为目录信息库(DIB)。为用户提供的目录服务实际上是关于对这些信息的各种方式的访问。

目录提供的服务在 GB/T 16264.3—2008 中定义。

一个目录用户(如一个人或一个应用进程)通过访问目录而获得目录服务。更准确地说，是一个目录用户代理(DUA)或一个轻量级目录访问协议(LDAP)的客户机代表每个用户去真正地访问目录，并与目录交互以获取其服务。目录提供一个或多个访问可进行的访问点。上述概念如图 1 所示。

DUA 表现为一个应用进程。在任何一个通信实例中，每个 DUA 都确切地代表一个目录用户。

目录表现为一个或多个应用进程的集合，这些应用进程被称为目录系统代理(DSA)和/或轻量级目录访问协议(LDAP)服务器，每个 DSA 或 LDAP 服务器都提供零个、一个或多个访问点。关于 DSA 的更详细描述，见 21.2。

注 1：一些开放系统可能会为实际用户(如应用进程或人员等)获取信息提供一个集中式的 DUA 功能。这个对目录来说是透明的。

注 2：DUA 功能和一个 DSA 可以处于同一个开放系统中，并且可以在实现时选择是否让一个或多个 DUA 在 OSI 环境中作为可视的应用实体。

注 3：一个 DUA 可以有本地特性和结构，这些不在本目录规范的定义范围之内。例如，一个表示目录人类用户的 DUA 可能会提供一个本地范围的便利工具来帮助它的用户构成请求并解释响应。

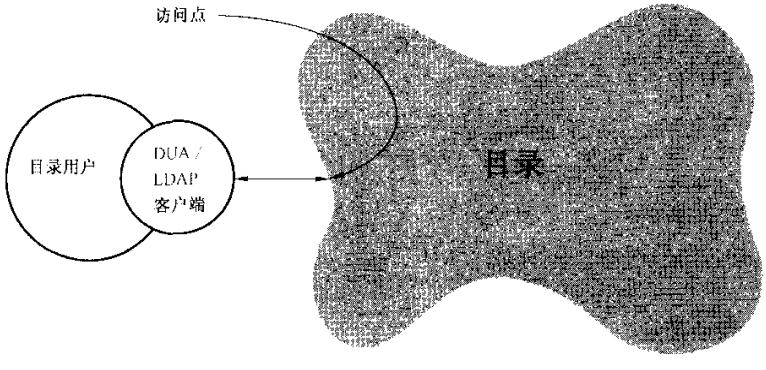


图 1　目录的访问

6.3　目录和 DSA 信息模型

6.3.1　通用模型

目录信息可能分成如下的类：

——用户信息：由用户置于目录内或者代表用户；随后被用户所管理或者代表用户。第 3 部分提供

了该信息的一个模型；或者

——管理和操作信息：由目录拥有，以适应各种不同的管理和操作需求。第5部分提供了该信息的一个模型。另外在第5部分还提供了一个关于用户信息模型、管理和操作信息模型之间关系的规范。

这些模型从不同方面表达了DIB视图，被认为是通用目录信息模型。

目录信息模型描述了目录作为一个整体如何来表示信息。作为一起操作的DSA集合的目录成分从该模型中得出。另一方面，DSA信息模型与DSA以及DSA应拥有的信息尤其相关，以使组成目录的DSA集合能够共同实现目录信息模型。DSA信息模型在本部分第22章到23章提供。

DSA信息模型是一个通用的模型，描述了DSA所拥有的信息以及这些信息与DIB和DIT之间的关系。

DSA信息模型所表示的一些信息可以通过目录抽象服务来访问，但不是全部信息。因此，如果对本系列目录规范中描述的全部信息都通过目录抽象服务来进行管理是不可能的。可以预见到的是，对DSA信息的管理最初应当是一个本地事物，但到最后应该会部署一些通用的系统管理服务来提供对DSA信息模型中描述的所有信息的访问。

6.3.2 特定的信息模型

对于作为整体的目录和它的组件，在通用模型制定以后，特定的信息模型需要对目录及其组件操作的特别方面进行标准化。

通用的目录信息模型为下述特定的信息模型建立了一个框架：

——访问控制信息模型；

——子模式信息模型；

——集合属性信息模型。

相应的，通用的DSA信息模型为下述特定的信息模型建立了一个框架：

——DSA分布知识的模型；

——DSA复制知识的模型。

6.4 目录管理机构模型

目录管理域(DMD)是由单个单独的组织所管理的一个或多个DSA以及零个或多个DUA的集合。

由(DSA组成的)目录管理域(DMD)所拥有的全球DIT的那部分被称为*DIT域*。在DMD和DIT域之间是一对一的对应关系。当提及对目录功能组件的管理时，使用术语“DMD”。当提及对目录信息的管理时，使用术语“DIT域”。与该术语相关的两个要点是：

——一个DIT域由一个或多个不相交的DIT子树组成(见11.5)。一个DIT域不得包含全球DIT的根；

——当管理的两个方面(目录功能组件管理和目录信息管理)放在一起考虑时，术语“DMD”作为一般术语使用。

管理某个DMD(以及关联的DIT域)的组织被称为一个*域管理组织(DMO)*。

注1：域管理组织可能是一个管理部门(即一个公共远程通信管理部门，或者其他提供公共远程通信服务的组织)，在这种情况下，被管理的DMD被称做公共目录管理域(ADDMD)；否则，它就是一个专用目录管理域(PRDMD)。应当认识到的是，关于ITU-T成员对专用目录系统的支持，这种指配属于国家法规的框架之内。因此，提供目录服务的主管部门可以提供所描述的技术可能性，也可不提供。专用目录管理域的内部操作和配置不在本目录规范的定义范围之内。

图2举例说明了DMO、DMD和DIT域之间的关系。

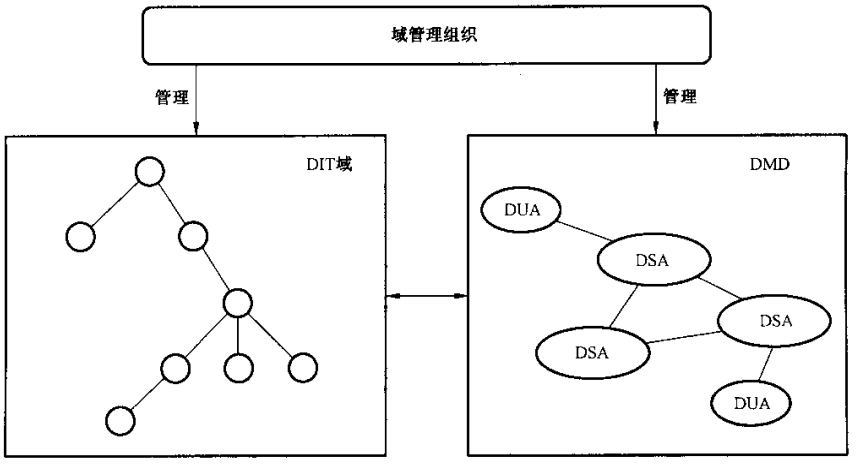


图 2 目录管理

由 DMO 对 DUA 进行管理意味着 DMO 对该 DUA 负有服务的责任，例如：维护，或在某些情况下被 DMO 拥有。DMO 可以选择或不选择使用本系列目录规范来管理 DMD 域内 DUA 和 DSA 之间的任何交互。

与目录管理的不同特性相关的域管理组织(DMO)的代理机构被称为*管理机构*。“管理权力”指的是由域管理组织授予某个管理机构的用以执行策略的权力。

注 2：目录管理机构模型在第四篇规定。

为便于引用，如在搜索规则中，可给 DMD 分配一个客体标识符(DMD-id)。

第三篇：目录用户信息模型

7 目录信息库

7.1 定义

本部分使用下列术语和定义：

7.1.1

别名条目 alias entry

一个包含用于为客体或者别名条目提供可替换名信息的“别名”类的项。

7.1.2

祖(条目) ancestor

组成一个复合条目的家族成员所形成的层次结构中的根条目。

7.1.3

复合条目 compound entry

表示一个客体，该客体由家族成员组成，且这些家族成员分层次地组织起来形成一个或多个条目的家族。

7.1.4

派生条目 derived entry

在搜索结果中的条目信息，其包含的属性值是通过从一个或多个目录条目中获取的原始数据进行结合而得到的。

7.1.5

直接上级类 direct superclass

相对于子类而言，直接派生子类的客体类。

7.1.6

目录信息库 directory information base;DIB

目录提供访问的完整信息集合，通过使用操作可阅读和操纵其中的信息段。

7.1.7

目录信息树 directory information tree;DIT

DIB 可被看作一棵树，除根以外的树的顶点都是目录条目。

注：只有在与信息的树型结构相关的上下文内，才使用术语“DIT”来替代“DIB”。

7.1.8

(目录)条目 (directory) entry

DIB 内的已命名的信息集合，DIB 由条目组成。

7.1.9

家族 family

复合条目内表示特定信息类的由家族成员条目所组成的具有层次结构的子集。复合条目内每个家族的根是一个祖先，但除了共享这一个祖先外，家族之间不共享公共成员。在一个复合条目内，每一个家族内直接包含在祖先下的成员都具有一个共同类(结构客体类)，以此将一个家族同其他家族区分开来。

7.1.10

家族成员 family member

组成一个复合条目的层次结构条目集合中的成员。

7.1.11

直接上级 immediate superior

相对于一个特定的条目或客体而言(它应当根据其所意向的上下文有明确定义)的直接上级条目或客体。

7.1.12

直接上级条目 immediately superior entry

相对于特定条目而言，直接上级条目是在 DIT 的一个弧的初始顶点上，而该特定条目是在弧的终结顶点上。

7.1.13

直接上级客体 immediately superior object

相对于特定客体而言，直接上级客体的客体条目是下一级客体的任何条目(客体条目或别名条目)的直接上级。

7.1.14

(关注的)客体 object (of interest)

“世界”中的任何事物，这里所说的“世界”通常是指远程通信和信息处理或是它们的某一部分，这些事物可被标识(能够被命名)，在 DIB 中保持的信息应是有意义的。

7.1.15

客体类 object class

共享某些特性的可被标识的客体(或可以设想的客体)族。

7.1.16

客体条目 object entry

DIB 中的一个条目，该条目是 DIB 中关于某个客体的主要信息集合，因此可以说，该条目在 DIB 中

代表了该客体。

7.1.17

相关条目 related entries

(目录)条目的一个集合,该集合中的每个条目都在DIB中拥有感兴趣的特定现实世界中的客体信息。该集合中的不同条目可能会包含关于现实世界中客体的不同类型的信息,甚至可能包含矛盾的信息。

注1:相关条目集合中信息的取值取决于每个条目的标识与现实世界之间的可靠度。

注2:相关条目处于不同的DIT中且拥有相同的标识名是可能的,但不是必要的。同样的,非相关条目拥有相同的标识名也是可能的;然而,建议相同的标识名仅用于相关条目。

7.1.18

子类 subclass

相对于一个或多个上级类而言——子类是从一个或多个上级类派生而来的。子类的成员共享其上级类的所有特性,并且拥有这些上级类成员所没有的附加特性。

7.1.19

下级 subordinate

上级的相反面。

7.1.20

上级类 superclass

相对于子类而言——指某个子类的直接上级类,或是其直接上级类的上级类(递归定义)。

7.1.21

上级 superior

(应用于条目或客体的)直接上级,或是其直接上级的上级(递归定义)。

7.2 客体

目录的目标是拥有某个现实"世界"中存在的*有意义的客体(或多个客体)*的信息,并提供对这些信息的访问。一个客体可以是该现实世界中可被标识(可被命名)的任何事物。

注1:一般来说,'世界'指的是远程通信和信息处理世界或该世界的某一部分。

注2:目录中的客体可能不是与现实世界中的"真实"事物有确切的对应关系。例如,现实世界中的人可能会被当做两个不同的客体:一个商务人士和一个小区居民,正如目录所关注的。在本目录规范中,并没有定义如何进行映射,映射工作由目录用户和提供者根据他们所应用的上下文进行。

*客体类*是共享某些特性的客体(或可想象到的客体)的家族,该家族可被标识。每个客体都至少属于一个客体类。客体类可能是其他客体类的*子类*,在这种情况下,子类的所有成员客体都可被认为是上级类的成员客体。子类还可以有子类,依此类推,可以形成任意深度。

7.3 目录条目

DIB由*(目录)条目*组成。一个条目是一个已命名的信息集合。

有4种类型的条目:

——*客体条目*:表示DIB中某个特定客体信息的主要集合。对于任何一个特定的客体,都有且仅有一个客体条目或复合条目(见8.10)来表示。可以说,一个客体条目代表了一个客体。一个客体条目或者是一个单个条目或者是一个由代表该客体的所有条目共同组成的复合条目。

——*别名条目*:用来为一个客体条目(可能是一个复合条目的祖先,但不会是孩子家族成员)提供替代名(称)的条目。

——*子条目*:代表DIB内的用来满足目录管理和操作需求的信息集合。子条目在第5部分讨论。

——*家族成员*:是用来组成复合条目的一种特殊的条目。复合条目的祖先也是一个家族成员。

目录条目的用户视图结构在图3中表示，并在8.2中描述。

每个条目都包含一个客体类指示以及该条目所属的上级类。

有一些客体条目是出于目录管理的目的而特别设计的。这些条目被称为管理条目。目录用户一般不关心这些条目，并且将这些条目与其他客体条目一同看待。

7.4 目录信息树(DIT)

为了满足对一个大型DIB进行分布和管理的需求，并且能够确保这些条目被无二义性地命名并被快速查找到，扁平式结构多半是不可行的。因此，采用了一种客体间通常存在的一种层次式关系(例如，一个人工作在某个部门，这个部门属于一个组织，而这个组织又在一个国家设立了总部等)，将条目组织为一棵树，这棵树被称为*目录信息树(DIT)*。

注：关于树型结构的概念和术语的介绍见附录I。

DIT的组成部分解释如下：

a) 顶点(vertices)是条目。客体条目可能是叶顶点或非叶顶点，而别名条目总是叶顶点。而树的根不是这样的条目，但如果在方便的时候(例如，在下面的b)和c)的定义中)，也可以将根看做是一个空的客体条目(见下面的d))。

b) 弧(arcs)定义了顶点(也即条目)间的关系。从顶点A到顶点B的弧意味着A上的条目是B上条目的*直接上级条目(直接上级)*，相反的，B上的条目是A上条目的*直接下级条目(直接下级)*。一个特定条目的*上级条目(上级)*是其直接上级以及直接上级的上级(递归定义)，一个特定条目的*下级条目(下级)*是其直接下级以及直接下级的下级(递归定义)。

c) 由条目所表示的客体具有其下级的命名机构，或者与其下级的命名权有密切关系(见第8章)。

d) 根代表DIB命名权的最高级别。

客体间的上级/下级关系可以从客体条目间的上下级关系中推派生。若一个客体是另一个客体的*直接上级客体(直接上级)*，当且仅当第一个客体的客体条目是第二个客体的任意客体条目的直接上级的情况下才成立。*直接下级客体*、*直接下级*、*上级*和*下级*等都有类似的含义。

客体间所允许的上级/下级关系由DIT的结构定义(见13.7)所决定。

除了与目录条目相关的信息外，目录还维护了一些与目录条目的集合相关的附加信息。这样的集合可形成(DIT的)子树或是子树精选(当不是一个真正的树型结构时)，见本部分第12章。

8 目录条目

8.1 定义

本目录规范使用下列术语和定义：

8.1.1

锚属性 anchor attribute

一个在相关的子模式中定义的具有友人的用户属性。可以使用一个锚属性将友人属性包含在所选择的属性集合中，或者匹配一个搜索操作，而不需要该友人属性真正出现在条目中。

8.1.2

属性 attribute

一种特定类型的信息。条目由属性构成。

8.1.3

用户属性 user attribute

一种表示用户信息的属性。

8.1.4

属性层次 attribute hierarchy

属性的一种特性，可以允许某个用户属性类型从另一个更通用的用户属性类型中派生出来。这两

个属性类型定义之间的关系即为层次性的(决定了与这些属性类型相应的属性的某些行为)。

8.1.5

属性子类型(子类型) attribute subtype (subtype)

一个属性类型 A 与另一个属性类型 B 相关,可以基于以下两种事实,一种是 A 从 B 派生而来,在这种情况下,A 是 B 的直接子类型;另一种是 A 从作为 B 的子类型的某个属性类型派生而来,在这种情况下,A 是 B 的间接子类型。

8.1.6

属性的上级类型(上级类型) attribute supertype (supertype)

一个属性类型 B 与另一个属性类型 A 相关,可以基于以下两种事实,一种是 A 从 B 派生而来,在这种情况下,B 是 A 的直接上级类型;另一种是 A 从作为 B 的子类型的某个属性类型派生而来,在这种情况下,B 是 A 的间接上级类型。

8.1.7

属性类型 attribute type

是属性的一个组成部分,指示了该属性所赋予的信息的类。

8.1.8

属性值 attribute value

由一个属性类型所指示的信息类的一个具体实例。

8.1.9

属性值断言 attribute value assertion

关于条目中存在的一个特定类型的属性值的一个命题,根据该属性类型所规定的匹配规则,该命题可能是正确的、错误的或是未定义的。

8.1.10

辅助客体类 auxiliary object class

一个描述了条目或条目类的客体类,且该客体类不用于 DIT 的结构规格说明。

8.1.11

集合属性 collective attribute

一种用户属性,该属性的值在一个条目集合内的每个成员中都是相同的。

8.1.12

上下文 context

与用户属性值相关联的一种特性,该特性规定了用来决定属性值适用性的信息。

8.1.13

上下文断言 context assertion

一个决定属性值适用性的命题,根据上下文类型和该类型的特定上下文值,该命题可能是正确的或错误的。

8.1.14

上下文类型 context type

上下文的一个组成部分,指示了上下文的类型或目的。

8.1.15

上下文列表 context list

与一个属性值相关的上下文的集合。

8.1.16

上下文值 context value

由上下文类型所指示的特性的一个具体实例。

8.1.17

派生属性 derived attribute

一个属性,其值或值集的全部或部分是计算出的,而不是直接存储的。

8.1.18

派生客体类值 derived object class value

一个客体类的值,该值的出现不受用户的管理,而是计算出的。派生客体类值被归类为抽象的。

8.1.19

直接属性引用 direct attribute reference

(在目录和DSA抽象服务中)表示对一个或多个属性值的引用,是通过使用属性类型标识符来实现的。

8.1.20

可辨别值 distinguished value

一个条目中的属性值,可出现在该条目的相关可辨别名中。

8.1.21

哑属性 dummy attribute

被定义为用户属性的一个属性,但是该属性永远不会出现在一个条目中。只有锚属性可以是一个哑属性。

8.1.22

条目集合 entry collection

属于一个明确定义的DIT子树或子树精选中的条目的集合。

8.1.23

友人属性 friend attributes

与某个特定用户属性(已知为一个锚属性)相关联的用户属性的集合,由管理机构将其关联起来,当指定了锚属性时,友人属性将包含在返回的属性集中,或者当某个谓词中包含了关于锚属性的条件时,则友人属性可潜在地用来匹配该谓词。

8.1.24

间接属性引用 indirect attribute reference

(在目录和DSA抽象服务中)表示对一个或多个属性值的引用,是通过使用相应属性类型的上级类型标识符来实现的。

8.1.25

匹配规则 matching rule

一种规则,组成了目录模式的一部分,该规则通过一个特定的与条目的属性值相关的声明(一个匹配规则断言)允许这些条目被选择。

8.1.26

匹配规则断言 matching rule assertion

关于条目中存在的属性值是否与匹配规则所定义的条件相匹配的一个命题,该命题可能是正确的、错误的或是未定义的。

8.1.27

操作属性 operational attribute

一种表示操作和/或管理信息的属性。

8.1.28

结构客体类 structural object class

一种用于规范DIT结构的客体类。

8.1.29

条目的结构客体类　structural object class of an entry

条目的结构客体类与某个特定条目相关，单个结构客体类用于确定应用到该条目上的 DIT 内容规则和 DIT 结构规则。该客体类由操作属性structuralObjectClass 所指示。该客体类是条目的结构客体类上级类链中的最下级客体类。

8.2　总体结构

如图 3 所示，一个条目由一系列的属性组成。

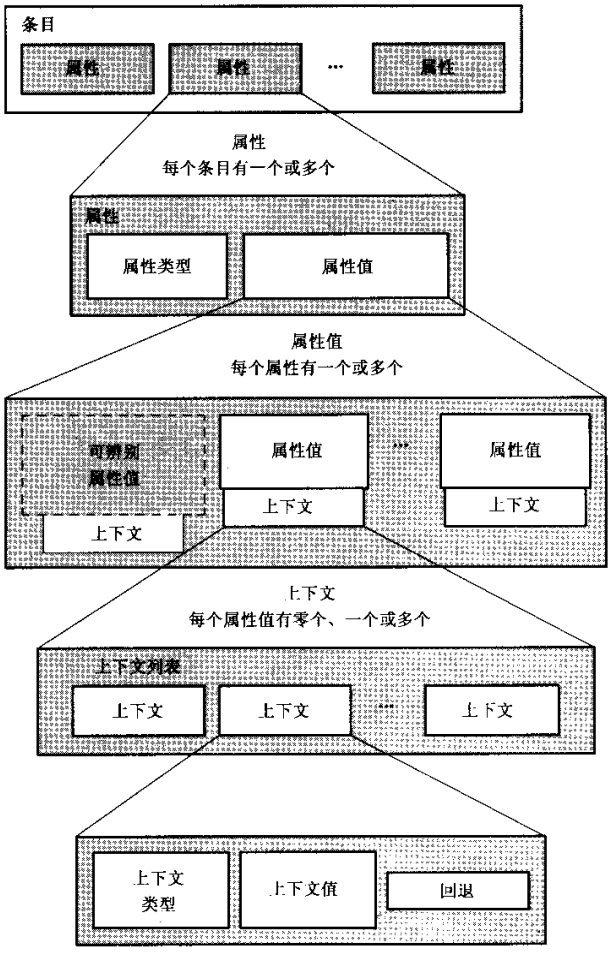


图 3　条目的结构

每个属性都提供了关于该条目相应客体的一部分信息，或者描述了该客体的某方面特性。

注 1：可能出现在条目中的属性示例包括：命名信息如客体的人名，地址信息如电话号码等。一个属性由属性类型和相应的属性值组成，属性类型描述了该属性所表示的信息类，属性值是出现在条目中的该信息类的具体实例。一个用户属性值在其上下文列表中，可能有零个、一个或多个与该值相关联的上下文。操作属性值不得有上下文。

注 2：属性类型、属性值和上下文分别在本部分 8.4、8.5 和 8.8 中描述。操作属性在第 12 章描述。

```
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 }
```

一个属性可能被设计为单值或多值。目录必须确保单值的属性仅有一个单独的取值，该值可能拥有一个上下文列表，将某些特性与该属性值相关联起来。已存储的属性必须至少有一个值，但是在准备存储或从存储处获取的过程中，可能会出现没有值的情况（例如由于属性值被访问控制隐藏）。

8.3 客体类

客体类用于目录中出于如下几种目的：

——对客体以及与这些客体相应的条目进行描述和分类；

——适当的时候，可以控制目录的操作；

——与 DIT 结构规则规范联合起来，可以调整条目在 DIT 中的位置；

——与 DIT 内容规则规范联合起来，可以调整条目中所包含的属性；

——标识条目的类，相应的管理机构将这些类与某个具体策略相关联起来。

某些类需要进行国际标准化，而其他类只需要由国内管理机构和/或专用组织定义即可。这就意味着将有许多各自独立的组织机构负责定义客体类，并无二义性地标识它们。这就需要在定义一个客体类的时候，使用一个客体标识符来标识每个客体类。用于该目的的表示法在本部分 13.3.3 中定义。

注 1：一个管理机构可能会使用一些客体类，这些客体类不同于在目录规范中已经定义并登记的有用客体类。管理机构可能会自己定义和登记客体类，如为了补充本目录规范中已经定义的那些客体类。

客体类（一个*子类*）可能从一个客体类（它的直接*上级类*）派生而来，而该上级类又是从另一个更通用的客体类派生而来。对于结构客体类，这个过程在最高层的通用客体类top 处终止。一个客体类的直到最上级客体类的上级类的有序集合，是该客体类的*上级类链*。

客体类可能从两个或多个直接上级类（这些上级类不是同一个上级类链上的部分）中派生而来。这种派生子类的特性被称为“*多继承*”。

条目客体类或家族成员客体类的规范标识了某个属性是必选的，还是可选的；该规范也可应用于其子类。可以说，子类*继承*了它的上级类的必选和可选属性的规范。一个子类的规范可能会指示其上级类中的某个可选属性在该子类中是必选的。

如果客体类定义了一个具有友人属性的锚属性，该锚属性是可选的或必选的，则该客体类自动地包含友人属性作为其可选属性，而不必在任何客体类定义或任何内容规则中再进行说明。

如果某个哑属性是一个锚属性的话，则一个客体类可能会定义该哑属性作为必选或可选属性。如果一个客体类将某个虚拟锚属性类型定义为必选或可选属性，则该锚属性不得出现在该客体类的条目中，但是，如果定义为必选属性，则必须至少有它的一个友人属性出现。然而，如果客体类将某个非虚拟的锚属性类型定义为必选属性类型时，则该锚属性类型的一个属性必须出现。

如果内容规则中排斥的话，则友人属性类型不得出现。

有 3 种类型的客体类：

——抽象客体类；

——结构客体类；和

——辅助客体类。

注 2：本目录规范不限制子类的定义一定是与上级通用于同一类型（即抽象、结构或辅助）；然而，管理者应当注意到，在某些情况下，与 LDAP 服务器的交互可能会受到不利的影响，尤其是当使用辅助客体的子类是结构客体，或者相反时，影响最明显。

每个客体类是且仅是上述类型中的一种，并且无论在目录中遇到什么情况都保持同一种类型。每个客体类的定义都必须指定该客体类所属的类型。

所有的条目都必须是客体类top 和至少一个其他客体类的成员。

8.3.1 抽象客体类

一个抽象客体类主要用于派生其他客体类，提供这些客体类的一些公共特性。一个条目不能仅属于某个抽象客体类。

top 是一个抽象客体类，是所有结构客体类的上级类。

除了用于派生其他客体类，一个抽象客体类的值还可以是一个派生值；也就是说，该值是由目录计算得出或推导得出的。例如，一个特定条目的parent 客体类值是通过从已存在的家族成员、辅助客体类 child 以及该条目的直接下级等进行计算或推导而得出的。

8.3.2 结构客体类

为了在 DIT 的结构规范中使用而定义的客体类被称为"*结构客体类*"。结构客体类用于与条目相应的客体名(称)结构的定义中。

一个客体条目或别名条目有且仅有一个结构客体类的上级类链，该链的最下级客体类是一个单独的结构客体类。该结构客体类被称为"*条目的结构客体类*"。

结构客体类与关联条目相关：

——遵从某个结构客体类的条目应当表示该客体类所限定的现实世界中的客体；

——DIT 的结构规则仅涉及结构客体类；条目的结构客体类用于定义该条目在 DIT 中的位置；

——条目的结构客体类，与相应的 DIT 内容规则一起，可用于控制一个条目的内容。

条目的结构客体类不得被修改。

8.3.3 辅助客体类

使用目录的某些特定应用将会经常发现定义一个*辅助客体类*是很有用的，辅助客体类可用于构造各种类型的条目。例如，消息处理系统使用辅助客体类"MHS 用户"(ISO/IEC 10021-2)来为条目类型定义一个包含必选和可选消息处理属性的包，而该条目类型的结构客体类是多样的，如组织人员或小区居民。

在某些环境下，有一种需求，能够为一个特定类(也可能是标准化的类)的条目所允许的属性列表中增加项或删除项。

上述需求可以通过定义并使用一个具有语义的、在本地团体中已知的并由本团体来维护的辅助客体类来实现，如果需要的话，该辅助客体类可以进行改变。

上述需求还可以通过使用 DIT 内容规则定义工具来动态地(即不需登记)从 DIT 特定点上的条目中增加或去除属性来实现(见 13.3.3)。

辅助客体类描述了条目或条目的类。

因此，一个条目除了是一个结构客体类的成员外，还可能是一个或多个辅助客体类的成员。

一个条目的辅助客体类可能会随着时间而改变。

注：未登记的客体类设施，为支持本款讨论的要求，曾在系列目录第 1 版存在。为支持 DIT 内容规则，该设施不再支持。

8.3.4 客体类定义与本目录规范的第 1 版

使用本目录规范第 1 版中的术语所定义的客体类，不能被分类为结构客体类、辅助客体类或抽象客体类。

使用本目录规范第 1 版中的术语所定义的别名客体类，可能被规定为或者抽象客体类，辅助客体类或者结构客体类，并且相应地被部署在子模式中。

8.4 属性类型

一些属性类型需要进行国际标准化，而其他属性类型只需要由国内管理机构和专用组织定义即可。这就意味着将有许多各自独立的组织机构负责定义属性类型，并无二义性地标识它们。这就需要在定义一个属性类型的时候，使用一个客体标识符来标识每个属性类型。可以使用在 13.4.8 中定义的ATTRIBUTE 信息客体类表示法。一个属性类型定义如下：

AttributeType::=ATTRIBUTE.&id

一个条目中的所有属性应是可区分的属性类型。

某些属性可能不需要在条目中存储或被访问，但需要在操作中携带以传递信息（例如诊断信息），这些信息可以被表示为属性。其他属性，被称为“*控制属性*”，作为其定义的一部分，根据属性中的信息规定一个可被执行的特殊程序。一个控制属性可能会在操作中规定，并放置在条目中等等。可以参见GB/T 16264.6—2008的7.5.3中的示例。

有许多目录所了解并为其使用的属性类型，它们是：

a) objectClass——每个条目中都会具有该类型的属性，该属性类型指示了客体所属于的客体类和上级类。

b) aliasedEntryName——每个别名条目中都会具有该类型的属性，该属性类型保存着别名条目所引用的条目的名（称）（见8.5）。

这些属性在13.4.8中定义。

在客体条目或别名条目中宜出现或可能出现的用户属性的类型，由应用于该指定客体类的规则以及该条目的DIT内容规则（见13.8）所决定。子条目中可能出现的属性类型由系统模式规则所决定。

一些目录条目可能会包含一些特殊的，但一般来说对目录用户不可见的属性。这些属性被称为“操作属性”，用来满足对目录的管理和操作需求。操作属性的更详细讨论见本部分第5篇。

8.5 属性值

属性定义也包括规定该属性的每个值都应当遵守的句法和数据类型。使用13.4.8中定义的ATTRIBUTE信息客体类表示法，一个属性值被定义为：

AttributeValue::=ATTRIBUTE.&Type

属性值可能被指定为“可辨别值”，在这种情况下，该属性值可以组成此条目的相关可辨别名（见9.3）。正如在9.3中所描述的，拥有多个由上下文所区分的可辨别值是可能的。

客户端提供的属性值存储在目录中。比较值是短暂的，且不得影响所存储的值。

8.6 属性类型层次结构

在定义属性类型的时候，一些更通用的属性类型的特性可能被作为定义的基础。新的属性类型从更通用的属性类型派生而来，新的属性类型是更通用属性类型的“直接子类型”，更通用的属性类型被称为“上级类型”。

属性的层次结构可以允许以不同的粒度对DIB进行访问。这可以通过使用特定的属性类型标识符（一个指向该属性的直接引用）或更通用的属性类型标识符（间接引用）来访问属性的值组件而实现。

语义上相关的属性可能会置于一个层次关系中，更具体属性是更通用属性的下级。如果引用更通用的属性类型会让搜索或查询属性及属性值的工作更简单；这样规定的过滤项将对更具体类型和更通用类型做同样的评估；为更通用的属性类型所指定的上下文断言也会应用到更具体的类型。

若下级具体类型被选择作为搜索结果中的一部分要被返回，则这些类型在可用情况下必须返回。若更通用的类型被选择作为搜索结果中的一部分要被返回，则更通用类型和具体类型在可用情况下都必须返回。属性值必须总是作为其属性类型的值而返回。

对于一个包含了某个属性类型值的条目而言，如果该属性类型属于某个属性层次结构，则此类型宜明确包含在该条目所属的客体类的定义中，或者应用于该条目的DIT内容规则允许此属性类型。

出于管理条目以及用户对条目内容进行修改的目的，在属性层次结构中的所有属性类型都必须是不同的，而且是无关的。

存储在目录客体条目或别名条目中的属性值应当有且仅有一个属性类型。当属性值被初次加到条目中时，应规定其类型。

8.7 友人属性

友人属性是由管理机构指定的一种用户属性，这些属性在某些实际情况下，与一个特定的锚属性相

关。若某个锚属性被规定要在某个阅读或搜索操作的返回结果中返回，则这种特性将允许该锚属性的友人属性也被返回，且服从服务和管理控制（包括访问控制、搜索规则等）。类似的，在一个搜索谓词的过滤项中如果指定了某个锚属性，且如果针对友人属性的匹配规则与被提议的值一致时，则相应的友人属性可用来满足该谓词。

如果在某条目的客体类值的必选或可选列表中包含了一个锚属性，说明该锚属性允许包含在条目中，则相应的友人属性也可被允许包含在条目中，除非被内容规则所排斥。如果锚属性不是一个必选属性，则即使友人属性出现在条目中，它也可能不在条目中出现。

任何一个用户属性都能够被设计为子模式内的一个锚属性。

注 1：作为一个锚属性的示例，考虑一个假设的属性commsAddr，在某个特定的子模式中，该属性拥有友人属性为通信地址属性类型，如电话号码、E-mail 地址、URL 等。

锚-友人关系既不是可互换的，也不是可传递的：

—— 如果一个锚属性 A 具有友人 B，并不能推派生 A 是 B 的友人；

—— 如果一个锚属性 A 具有友人 B，B 具有友人 C，并不能推派生 C 是 A 的友人。

如果一个属性 A 是某个锚属性的友人属性，则 A 的所有子类型也是该锚属性的友人。然而，并不能推派生 A 的上级类型也是该锚属性的友人。

设计某个属性为友人属性，并没有赋予其特殊的访问控制或搜索规则保护，除非它与锚客体类的成员关联起来（这样，该友人属性就自动成为锚客体类的一个成员）。

注 2：目前，访问控制和搜索规则并没有专门为了特权或保护，而使用客体类作为定义属性集的方法。

8.8 上下文

信息模型可以通过与属性值的某些特性相关联而得到细化，这些特性被称为上下文。任何一个用户属性值都可能与一个上下文列表相关联，列表中的上下文提供了能够用来判断属性值适用范围的附加信息。

注 1：例如，可以使用上下文将一个属性值与某种具体的语言或地点相关联起来。

每个上下文都包括一个类型字段，一个由类型字段决定其句法的值字段以及一个fallback 标志。使用 13.9 定义的CONTEXT 信息客体类表示法，一个上下文定义如下：

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

contextType 是一个OBJECT IDENTIFIER，在 13.9 定义的CONTEXT 信息客体类进行规定，它规定了上下文所表示的特定特性。

contextValues 是contextType 所规定的特性的一个或多个值的集合，这些值与某个具体属性值相关联。

fallback 用来为与一个上下文类型相关的特定行为规定一个或多个属性值。一个属性值除了具有与其相关联的上下文类型相应的任何特定contextValues 外，如果该属性值的某个给定contextType 的fallback 值为TRUE，则该属性值：

- 被认为与给定的contextType 的任意值相关联，但是同一属性的其他值都不与其相关联。因此，在基于为contextValues 进行匹配的规则与该属性任意值的匹配中失败的该上下文类型的一个上下文断言，须与针对该上下文类型的fallback 为TRUE 的任意属性值相匹配。

注 2：例如，试图选择与某种特定语言相关的属性值，如果没有任何一个属性值与该选定的语言相关联，则应当遵循那些fallback 设定为 TRUE 的属性值。

- 被认为是需要在一个操作期间内保留的值，该操作对某个给定属性类型的属性值进行了重置。如果该属性类型所关联的上下文的fallback 被设置为FALSE 时，则修改操作（重置值）将删除

该选定属性类型的所有值。

注 3:值修改(值重置)在 GB/T 16264.3—2008 的 11.3.2 中描述。

如果属性值没有上下文,或者有上下文列表,但列表中没有包括某个具体类型的上下文时,则此属性值在该具体类型的所有上下文值中都可应用。

注 4:例如,基于语言上下文中的法语上下文值来选择时,如果一个属性值没有指定任何一个语言上下文与其相关联时,则该属性值会被选中(同样的,特定的具有法语上下文的属性值也会被选中)。

在一个属性值上下文列表中的所有上下文须具有不同的上下文类型。

与属性值相关联的上下文信息可以与属性值一起被获取(例如,为了区别这些属性值)。目录的用户也可能会在目录操作中使用上下文来细化选择和查询信息。

8.9 匹配规则

8.9.1 概述

对目录来说,极为重要的是应当具备一种能够根据与条目所具有的属性值相关的断言,从 DIB 中选择一个条目集的能力。

匹配规则就是通过构造一个与条目的属性值相关的特定断言来对条目进行选择。

断言的最基本类型就是属性值断言(AVA)。更复杂的断言可以使用匹配规则断言来支持。一个匹配规则断言是一个关于条目中存在的属性值是否与匹配规则所定义的条件相匹配的一个命题,该命题取值可能为真、假或未定义。

属性值断言或匹配规则断言可以根据与断言相关的匹配规则进行评估。

匹配规则的定义是通过对下述几个方面进行规定完成的:

——规则所支持的属性句法的范围;

——规则所支持的特定的匹配类型;

——表达每个特定匹配类型的声明所需的句法;

——需要时,从属性句法的值中派生一个断言句法值的规则。

注:对于为了支持某个具体应用而可能定义的匹配规则并没有限制。然而,为了支持某个具体应用而定义的规则可能不会被 DUA 和 DSA 广泛支持。在任何可能的时候,在 GB/T 16264.6—2008 中定义的匹配规则都应当比新定义的匹配规则优先使用。

有时,在匹配规则与所支持的匹配类型之间具有一对一的对应关系。例如,目录抽象服务支持一种"存在匹配规则"以检测条目中某个属性是否存在。

有时,在匹配规则与所支持的匹配类型之间具有多对多的对应关系。例如,目录抽象服务支持一种"通用排序规则"允许对匹配类型进行大于或等于和小于或等于排序。

8.9.2 属性值断言

属性值断言(AVA)是关于某个特定类型的属性值是否存在于条目中的一个命题,根据该属性类型所规定的匹配规则,该命题可能为真、假或未定义。它包括一个属性类型、一个声明的属性值以及一个可选的与该属性值相关的上下文断言。

```
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})}

AVA 中assertion 组件的句法由为该属性类型定义的相等匹配规则来决定，可能不同于属性本身的句法。

8.9.2.1 AVA 的评估

一个 AVA 取值是：

a) 未定义的，如果下述任何一项都符合：

1) 属性类型是未知的；

2) 属性值没有相等匹配规则；

3) 属性值不符合属性相等匹配规则中的声明句法所指示的数据类型；

注：一般来说，2)和 3)指示一个错误的 AVA；然而，1)项可能出现在本地情况下(例如没有配置一个特定的 DSA 来支持该特定的属性类型)。

b) 真，如果条目包含该类型的一个属性，该属性包含一个值，且该值包含与 8.9.2.2 所描述的assertedContexts 相匹配的一个上下文；

c) 假，与上述真的情况相反。

8.9.2.2 assertedContexts 或上下文断言缺省值的使用

在AttributeValueAssertion 中包含assertedContexts 字段是可选的。如果指定了assertedContexts 字段，则assertion 仅对使得assertedContexts 取值为真的那些属性值进行判断，如 8.9.2.3 所定义的。

如果 AttributeValueAssertion 中没有提供 assertedContexts 字段，则可能有一个缺省的上下文断言以一种同样的方式应用；即 assertion 仅对使得缺省上下文断言取值为真的那些属性值进行判断，如 8.9.2.3 所定义。有 3 个潜在的资源可用于缺省的上下文断言：在整体上为操作而规定，在 DIT 的子条目内可用以及在 DSA 本地可用。它们的应用如下所述：

1) 如果在一个AttributeValueAssertion 中没有提供assertedContexts 字段，且作为 GB/T 16264.3—2008 的 7.3 中定义的operationContexts 的一部分，提供了该给定类型的上下文断言，这些上下文断言是在整体上为操作而提供的，则须应用此上下文明；

2) 如果用户没有为 AVA 提供assertedContexts，且对于给定的属性类型没有任何上下文断言是整体上为操作而提供的，则须应用控制该条目的上下文断言子条目(如果有的话)内的给定属性类型的缺省上下文断言，如 14.7 所描述的；

3) 如果没有上述步骤 1)和 2)中的上下文断言，则 DSA 可能会为给定的属性类型应用一个本地定义的缺省上下文断言。这样的一个缺省值须典型地反映一些本地参数，如 DSA 所使用的语言或 DSA 所部属的位置，或者当前时间等，但 DSA 有可能会对它所响应的不同 DUA 做出不同的裁剪；

4) 如果上述资源中没有任何一个上下文断言可用，则assertion 须根据该属性的所有值进行评估。

8.9.2.3 assertedContexts 的评估

在下列情况下，assertedContexts 取值为真：

a) 规定了allContexts(这种方式允许上下文断言优先于任何缺省的上下文断言，否则如果在AttributeValueAssertion 中省略了assertedContexts 字段的话，就会应用这些缺省上下文断言)；或者

b) 如 8.9.2.4 中所描述的，selectedContexts 中的每个ContextAssertion 取值都为真。

8.9.2.4 ContextAssertion 的评估

在下述情况下，某个特定属性值的ContextAssertion 取值为真：

a) 属性值具有ContextAssertion 中相同contextType 的上下文，且根据对contextType 进行匹配的定义，该上下文任意存储的contextValues 与任意一个所声明的contextValues 相匹配；或者

b) 属性值没有包含所声明的contextType 的任何上下文;或者

c) 该属性没有其他的属性值可以满足上述 8.9.2.2 中 1)或 2)所描述的ContextAssertion ,但是该属性值确实包含所声明的contextType 的一个上下文,且fallback 设置为TRUE 。

8.9.3 属性类型声明

属性类型声明是一个命题,根据所关联的上下文,该命题取值可能为真、假或未定义。

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

8.9.3.1 属性类型声明的评估

属性类型声明的取值为:

a) 未定义,如果属性类型未知,或者条目中没有包含该属性;

b) 真:如果条目包含该类型的一个属性,该属性包含一个或多个属性值,且属性值包含与 8.9.3.2 所描述的assertedContexts 相匹配的一个上下文;

c) 假,与上述真的情况相反。

8.9.3.2 assertedContexts 或上下文断言缺省值的使用

在AttributeValueAssertion 中包含assertedContexts 字段是可选的。如果指定了assertedContexts 字段,则根据 8.9.2.4 描述的规则,至少对一个属性值而言,assertedContexts 须为真。

如果AttributeValueAssertion 中没有指定assertedContexts 字段,则可能有一个缺省的上下文断言以一种同样的方式应用;即根据 8.9.2.4 描述的规则,至少对一个属性值而言,缺省的上下文断言须为真。用于缺省上下文断言的潜在资源如 8.9.2.2 所述。

8.9.4 内嵌的匹配规则断言

目录能够理解的相关匹配规则的分类如下所述,这些规则的语义被广泛地理解,并应用到许多不同类型的属性值中:

——出现;

——相等;

——子串;

——排序;

——近似匹配。

与这些匹配规则分类相关的匹配类型的声明句法内嵌在目录抽象服务中:

——present 句法用于出现规则;

——equality 句法用于相等规则;

——greaterOrEqual 和lessOrEqual 句法用于排序规则;

——initial,any 和final 句法用于子串规则;

——approximateMatch 句法用于近似匹配规则。

present 句法可以用于任意类型的任意属性。可以对某个特定类型的某个任意值的出现执行出现匹配测试。

具体的相等匹配规则、子串匹配规则和排序匹配规则等在属性类型定义时,可以与属性类型相关联。当使用目录抽象服务内嵌的句法对相等、排序和子串规则的声明进行评估时,会用到这些具体规则。如果没有提供具体规则,则与这些属性相关的声明取值为"未定义"。

ApproximateMatch 句法支持近似匹配规则,它的定义属于 DSA 的内部问题。

8.9.5 匹配规则要求

为了使目录的行为方式是一致的和良好定义的,有必要对应当与内嵌入目录抽象服务中的句法一起使用的匹配规则进行某些限定。

对于一个相等匹配规则，如果该规则中的声明句法与该匹配规则所应用的属性句法不同，则须提供从属性句法的一个值如何推派生声明句法的一个值的规则。

用于命名属性的相等匹配规则须是可传递的和可交换的，并且其声明句法应与属性句法相一致。

一个可传递的匹配规则的特性是：如果一个值 a 与另一个值 b 相匹配；且如果该值 b 与第三个值 c 相匹配；则使用此规则，值 a 与值 c 也匹配。

一个可交换的匹配规则的特性是：如果一个值 a 与另一个值 b 相匹配，则该值 b 与值 a 也相匹配。属性presentationAddress 是一个示例，该属性所支持的属性句法的匹配规则是不可交换的。

对于某个特定的属性类型，相等规则和排序规则（如果两者都存在）须至少在以下方面总是相关的：两个值如果使用相等关系来判断是相等的，当且仅当它们使用排序关系来判断也是相等的。另外，排序关系应当是良好排序的，也就是说，对于所有的 x，y 和 z，如果根据排序关系，x 在 y 之前，y 在 z 之前，则 x 一定在 z 之前。

注：这些需求隐含着如果排序被定义的话，同时也定义了相等。

对于某个特定的属性类型，相等规则和子串规则（如果两者都存在的话）应当至少在以下方面总是相关：对于所有 x 和 y，如果根据相等关系来判断它们是匹配的，则对于所有的 z，根据子串关系来判断，与值 x 的声明评估结果和与值 y 的声明评估结果都应当是相等的。也就是说，如果使用相等匹配规则来判断两个值是不能区分的，则使用子串匹配规则来判断两个值也是不能区分的。

8.9.6 客体标识符和可辨别名的相等匹配规则

有很多相等匹配规则用于评估属性值断言，本标准已知这些匹配规则并且为其使用。这些规则包括：

——objectIdentifierMatch：该规则用于匹配具有ObjectIdentifier 句法的属性。

——distinguishedNameMatch：该规则用于匹配具有DistinguishedName 句法的属性。

8.10 条目集合

8.10.1 概述

由于某种共同的特性或相关客体的共享关系，客体条目和别名条目的集合可具有某种公共特性（如对于集合中的每个条目，某些属性具有相同的值）。这些条目的集合被称为一个“条目集合”。

条目集合可能会包含客体条目和别名条目，这些条目由于它们在 DIT 中的位置而相互关联。这些集合被规定为子树或子树精选，将在第五篇中描述。

根据第五篇定义的管理限制，一个条目可属于多个条目集合。

8.10.2 集合属性

当用户属性被条目集合中的条目所共享时，该用户属性被称为“*集合属性*”。

这种情况也是允许的，即同一个集合属性可以独立地与两个或多个条目集合相关联。在这种情况下，条目的集合属性有多个值。因此，集合属性应当总是被规定为多值的。

尽管集合属性对于目录查询操作的用户来说是一个条目属性，但在目录信息模型中，它们与其他条目属性是不同的。这种不同体现在，对于目录修改操作的用户来说，集合属性是不能通过它们所在的条目被管理的（即不能被修改），而应当通过它们相关的子条目来管理。

注：这些值的独立资源是不会显示给目录查询操作的用户的。

对于要出现在某个条目中的集合属性，该属性类型的出现务必得到管理该条目的 DIT 内容规则的许可。

条目可能会明确地排斥某个具体的集合属性。这可以通过使用属性collectiveExclusions 来实现，将在 12.7 描述，并在 14.6 定义。

8.11 复合条目和条目家族

一个复合条目是一种特殊的条目，由家族成员条目组成。这些家族成员构成了一个层次结构，因此提供了关于复合条目所代表的客体的层次化组织的信息。复合条目在 DIT 中由一个祖先家族成员所代表，该家族成员是包含所有家族成员的一棵树的根结点。

出于过滤或信息获取的目的，家族成员自身可以组织成一个或多个家族。每个家族是一棵子树；除

了共享的根结点(即祖先)外,不同的家族之间没有共同的家族成员。因此,一个家族包含一个祖先和一个下级家族成员的集合。

除祖先外,一个家族由属于同一结构客体类的所有直接下级家族成员所组成。它们的下级成员,如果存在的话,也是同一家族的组成部分,但不依赖于它们的结构客体类。

这些概念如图 4 所示。

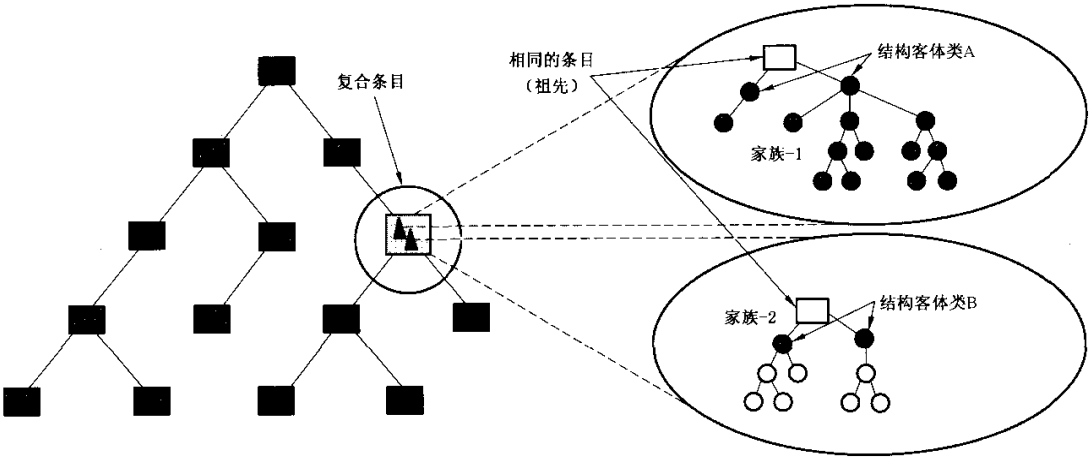


图 4 条目家族

家族树中是孩子的家族成员标记为辅助客体类child 。如果某个条目出现了child 客体类值,则其直接上级条目将自动使用抽象客体类值parent 来标记。一个在家族子树中既是parent 又是child 的条目将由两个客体类值同时标记。祖先是唯一一个不是孩子的家族成员。复合条目的构造是通过将条目标记为child 客体类值而完成的。

非祖先家族成员的每个下级本身都应当是一个家族成员,并且被标记为child 客体类值。

这些客体类的 ASN.1 定义见 13.3.3。

复合条目的所有的家族成员须处于与祖先一样的命名上下文中。家族成员不允许为别名条目。别名不得指向孩子家族成员。

9 名(称)

9.1 定义

本目录规范使用下列定义:

9.1.1

别名,别名名(称) alias,alias name

一个客体的可替代名(称),通过使用别名条目来提供。

9.1.2

(别名)解除引用 (alias) dereferencing

将一个客体的别名名(称)转换为客体的可辨别名的过程。

9.1.3

(条目的)可辨别名 distinguished name (of an entry)

每个客体条目、别名条目和子条目都至少有一个可辨别名。如果某个条目或其任何上级条目的任何一个 RDN 中包括一个属性,该属性拥有多个由上下文所区分(见 9.3 的描述)的可辨别值,则该条目须具有多个由上下文所区分的可辨别名。其中,*主辨别名* 是这样的一个可辨别名,即该可辨别名中每个 RDN 都是由属性的主可辨别值作为构造 RDN 的主要值。

9.1.4

(目录)名 (directory) name

将某个特定客体从其他客体中区分出来的结构。一个名应当是无二义性的(也就是说,仅能够标识一个客体);然而,它不是必须是唯一的(也就是说,它并不是能够无二义性地标识该客体的唯一的名)。

9.1.5

(条目)名 (entry) name

将某个特定条目从其他条目中区分出来的结构。

9.1.6

本地成员名 local member name

某个家族成员的名(称),通过从祖先一直到正在讨论的该成员的 RDN 顺次构造而成,但不包括祖先的 RDN。

9.1.7

命名机构 naming authority

负责 DIT 某个范围内的名(称)分配的组织机构。

9.1.8

声称名 purported name

一个结构,在句法上它是一个名(称),但还没有显示其是一个合法的名(称)。

9.1.9

相关可辨别名 relative distinguished name;RDN

一个或多个属性类型与属性值对的集合,每个属性类型和属性值对都与条目的一个不同的辨别属性值相匹配。

9.2 名(称)概述

*(目录的)名*是一个将某个特定客体从其他所有客体的集合中区分出来的结构。一个名(称)应当是无二义性的,也就是说,仅能够标识一个客体;然而,它不必是唯一的,也就是说,它并不是能够无二义性地标识该客体的唯一的一个名(称)。(目录的)名也可以标识一个条目。该条目或者是一个表示客体的客体条目,或者是一个别名条目,该别名条目中包含了可以帮助目录对表示了客体的条目进行定位的信息。

注 1:因此,一个客体的名(称)集合中包含了该客体的一系列别名以及该客体的可辨别名。

一个客体可以被分配一个可辨别名,但不一定要在目录中有一个条目来表示;但是如果在目录中有一个客体条目来表示该客体的话,则该可辨别名便是客体条目应当具有的名(称)。

按照句法的定义,每个客体或条目的名(称)是相关可辨别名的一个有序序列(见 9.3)。

```
Name ::=CHOICE{--目前仅有一种可能--rdnSequence  RDNSequence}
RDNSequence ::=SEQUENCE OF RelativeDistinguishedName
DistinguishedName ::=RDNSequence
```

注 2:以不同于本文所描述的其他方式构造的名(称),可能作为将来的扩展。

一个客体名(称)的每个初始子序列同时也是另一个客体的名(称)。在这样确定的客体序列中,即以根开始,以被命名的客体结束,每个客体都是序列中紧跟其后的客体的直接上级。

一个假设的名(称)也是指一个结构,其在句法上是一个名(称),但还没有显示其是一个合法的名(称)。

9.3 相关可辨别名

每个客体和条目都至少有一个相关可辨别名(RDN)。一个客体或别名条目的 RDN 由一个属性类型和值(以及可选的上下文列表)对的集合组成,使用相等匹配规则和可应用的上下文匹配规则,每个属性类型和值对都与条目的一个不同的辨别属性值相匹配。

任何用于 RDN 中的属性都可能有不止一个由上下文所区分的可辨别值，如下所述。这就为同一个客体提供了可替代的 RDN。在一个属性的可辨别值(由上下文所区分)集合中，有且仅有一个值被设计为主可辨别值。一个客体的*主相关可辨别名*由组成该 RDN 的属性集的主可辨别值的集合组成。当在协议中传送时，RDN 中的每个属性都显示了主可辨别值(如果存在的话)，并且可能还可选地包含与值相关的上下文以及符合该上下文的其他可替代属性值。在这种情况下，根据所应用的相等匹配规则和上下文匹配规则，每个属性值及其上下文都与条目中这种属性类型的一个特定的可辨别属性值相匹配。

注 1：能够使用相等匹配规则，是因为对于命名属性而言，属性句法与相等匹配规则的声明句法是相同的。

类似的，对于命名属性中用于区分可辨别值的上下文而言，其上下文句法和上下文断言句法也是相同的。

具有同一个特定直接上级的所有条目的 RDN 是不同的，与所关联的上下文列表无关。由条目的相关命名机构负责通过正确地分配辨别属性值来确保如此。RDN 的分配被认为是一个管理性事物，可能需要在相关的组织或管理部门之间进行协商，也可能不需要。本目录规范不提供这样的协商机制，并且不会假定它是如何执行的。

```
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}
```

在组成 RDN 的集合中，为条目中包含了可辨别值的每个属性都包含了且仅包含了一个AttributeTypeAndDistinguishedValue，也就是说，一个给定的属性类型不会在同一个 RDN 中出现两次。

被指定出现在 RDN 中的一个属性值被称为一个可辨别值。同一个属性可能还具有不是可辨别值的其他值，因此这些其他值可能不用于 RDN 中。只有当一个属性的可辨别值根据相关联的上下文有所区分时，属性才可能有多个可辨别值。这就允许一个客体可以具有多个由上下文所区分的可替代名(称)。这种情况仅会出现在一个属性可能拥有多个可辨别值的情况下。在那种情况下，这些可辨别值须具有包含同一个上下文类型的上下文列表，列表中的每个上下文值须规定对于给定的任何一个特定上下文，仅有一个可辨别值是可应用的。

对于某个给定的条目，其 RDN 的构造是通过从每个具有可辨别值的属性中选用一个可辨别值来完成的。最简单的情况是一个条目仅有一个可辨别值，因此就仅有一个使用该可辨别值构造的 RDN。一个条目中可以有不止一个属性能够用于构造 RDN。如果每个起作用的属性都仅有一个可辨别值，则该条目有一个唯一的由每个属性的可辨别值所构造的 RDN。如果任何一个起作用的属性都有多个由上下文所区分的可辨别值，则该条目会有多个 RDN，每个 RDN 是通过使用可能的组合之一构造而成的，为每个组成 RDN 的属性类型选择一个不同的可辨别值则形成一个可能的组合。

条目的每个 RDN 都应当为每个组成 RDN 的给定属性类型包含一个type 和value 对。primaryDistinguished 用于指示value 是该属性类型的主可辨别值。valuesWithContext 用于在需要时，为value 中的辨别属性值传送上下文列表。它还可用于在一个单独的 RDN 中，传送同一属性类型的某些或所有其他可辨别值。每个distingAttrValue 都伴随着它的contextList 。仅对于出现在value 中的可辨别值，distingAttrValue 字段才能够被忽略；这就是该值的上下文列表是如何出现在 RDN 中的。

对于条目中给定的属性类型有且仅有一个可辨别值被认为是该属性类型的"*主可辨别值*"。在构造客体的主相关可辨别名时，应当在AttributeTypeAndDistinguishedValue 中的value 中使用该值(见 9.8 和9.6)。主相关可辨别名是一个 RDN，在该 RDN 中，RDN 中每个属性的主可辨别值都出现在 RDN 的每个AttributeTypeAndDistinguishedValue 的value 组件中。上下文和可替代可辨别值可能会出现在每个AttributeTypeAndDistinguishedValue 的valuesWithContext 组件中。

必要时，可通过完全地替换起作用的所有属性的所有可辨别值来对 RDN 进行修改。

如同其他条目一样，家族成员也有 RDN。一个 RDN 能够由多个属性类型和值对组成。仅有主 RDN 可用。一个家族成员的本地成员名(称)是从祖先开始一直向下直到该成员的 RDN 的序列。祖先的本地成员名(称)为一个空序列。

注 2：RDN 被设计为长期的，因此目录的用户可以存储客体的可辨别名(例如，在目录内)而不必考虑其是否被废弃。因此对 RDN 的修改应当非常谨慎。

注 3：修改一个非叶条目的 RDN，将自动修改其下级条目的名(称)。

注 4：组成 RDN 的一个特定属性类型和值的上下文是否可用，与 RDN 中的其他部分以及可辨别名中的其他 RDN 的相关上下文之间是相互独立的。

注 5：例如，一个条目的合法可辨别名的构造，可以通过将该条目的 RDN"Language = French"与其上级条目的 DN "Language=English"组合而成。

9.4 名(称)匹配

在目录的操作中，经常需要判断两个名(称)是否匹配。这就要求相应的 RDN 要匹配。这里将描述通用的名(称)匹配方法；关于名(称)匹配特定用途的特别方法在其他适当的地方进行描述。

如果一个声称的 RDN 中的每个 AttributeTypeAndDistinguishedValue 字段都与目标 RDN 中同一属性类型的 AttributeTypeAndDistinguishedValue 字段相匹配，则可以说该声称的 RDN 与目标 RDN 相匹配。如果声称的 value 或声称的 AttributeTypeAndDistinguishedValue 字段中的任何 distingAttrValue，都与目标 value 或目标 AttributeTypeAndDistinguishedValue 字段中的任何 distingAttrValue 相匹配，则称它们之间匹配。在声称的或目标的 AttributeTypeAndDistinguishedValue 字段中出现的 primaryDistinguished，将在匹配时被忽略。

注 1：能够使用相等匹配规则，是因为对于命名属性而言，属性句法与相等匹配规则的声明句法是相同的。

注 2：并不能够保证，一个给定命名属性的每个可辨别值都能出现在给定 RDN 的该属性类型的 Attribute-TypeAnd-DistinguishedValue 字段中。同一个客体可以使用同一属性类型的不同可辨别值(由上下文区别)来构造两个 RDN。如果对于每个给定的属性，其所使用的可辨别值的集合之间没有交集的话，则即使声称的 RDN 和目标 RDN 是同一客体的两个可替代 RDN，它们之间也不匹配。这种情况如何发生以及影响如何，取决于使用名(称)匹配的原因(例如，名(称)解析、访问控制、过滤等)。

如果由于上述原因，而使得匹配的属性值找不到，则两个 RDN 不匹配。如果匹配属性值找到了，则如果这两个值存在相关上下文的话，在相关上下文间还应当有一个匹配，这样才可以认为这两个属性类型和值对之间是匹配的。在声称的属性值的上下文列表中的每个上下文都应当被认为是与匹配的目标属性值的上下文列表之间形成了一个上下文断言，并且应评估为真(如 8.9.2.4 中的描述)，这样的上下文才被认为是匹配的。在构造上下文断言时，声称的上下文中的 fallback 字段被忽略。

注 3：能够以这种方式使用声称的上下文作为上下文断言，是因为上下文断言句法和可能伴随可辨别值一起使用的上下文类型的上下文句法是相同的。

如果在声称的 RDN 中没有出现 valuesWithContext 字段，则应当应用在操作中提供的上下文断言，或者应用准备在操作中使用的缺省的上下文断言，如 8.9.2.2 所述。有一个例外情况是在目录操作中进行名(称)解析的过程中使用名(称)匹配，在那种情况下，如果在 valuesWithContext 中没有上下文断言可用时，则不必应用上下文断言。

9.5 操作中返回的名(称)

许多目录操作都返回条目的名(称)。当一个操作返回一个或多个条目的名(称)时，它须为每个条目都返回主辨别名，并可能返回其他的可替代辨别名的信息和上下文信息(见 GB/T 16264.3—2008 中 7.7)。

9.6 作为属性值存储或作为参数使用的名(称)

当名(称)作为其他一些属性的属性值被存储时，或者作为属性值在某些交换中被传递时(如一个别名指针)，总会有这样的问题：即该名(称)可以是一个可替代辨别名还是必须是主辨别名，该名(称)是否可以包含可替代可辨别值以及该名(称)是否可以包含上下文信息等。本系列目录规范在必要的时候会指出这些特殊的限制。

注：附录 C 中包含了如何提高与前 3 版系统交互性的建议以及如何确保与名(称)上下文使用相关的可预知行为的建议。

9.7 可辨别名

一个给定客体的可辨别名是由表示客体的条目及其所有上级条目的 RDN 序列(以递减顺序排列)组成的。由于客体与客体条目间具有一对一关系,因此客体的可辨别名便是客体条目的可辨别名。

注 1:建议人类需要处理的客体可辨别名,应当是用户友好的。

注 2:GB/T 9387.3 -1995 定义了一个主名(称)的概念。一个可辨别名可作为其所标识的客体的主名(称)使用。

注 3:由于仅包含了客体条目及其上级,因此可辨别名中永远不会包含别名条目。

别名条目同样也具有可辨别名,然而,该可辨别名不能作为一个客体的可辨别名。如果需要区分这两个概念,则我们使用全名"别名条目的可辨别名"来表示。如同客体条目的可辨别名,别名条目的可辨别名也是由别名条目及其所有上级条目的 RDN 序列(以递减顺序排列)组成的。

用这种方法定义根的可辨别名也是很方便的,尽管该名(称)永远都不会是一个客体的可辨别名。根的可辨别名被定义为一个空序列。

如果在一个客体的可辨别名内,当用于 RDN 的任意一个属性具有多个由上下文所区分的可辨别值时,则该客体将具有多个可辨别名。每个可辨别名都无二义性地标识了该客体。其中,主辨别名是每个 RDN 都是主 RDN 的可辨别名。在协议中传递时,主辨别名是这样构造的:在构成名(称)的每个 RDN 内,为每个属性的 AttributeTypeAndDistinguishedValue 字段中的 value 赋值为主可辨别值。可替代辨别名是这样构造的:在一个或多个 RDN 内,为属性赋值为可替代可辨别值。在某些情况下使用名(称)时,必须使用主辨别名。而另外一些情况下,也可以使用可替代辨别名。由于 RDN 内的 AttributeTypeAndDistinguishedValue 字段可能会在 valuesWithContext 组件中包含可替代可辨别值,因此,任何可辨别名在其 RDN 内可能包含可替代值。

注 4:当可辨别名内的一个 RDN 所使用的任意属性包含多个可辨别值时,则可认为该可辨别名包含可替代名。

一个可辨别名可能会包含上下文信息,该上下文信息存储在任意 RDN 的 valuesWithContext 组件内。在本系列目录规范中,无论名(称)在何种情况下使用,都应当指定名(称)是否应当为主辨别名,名(称)是否可能包含可替代值以及名(称)是否可能包含上下文信息等。如果没有显式的声明,则隐含着可以使用可替代辨别名,名(称)可能包含可替代值和/或上下文信息等。

注 5:在协议中使用主辨别名而非使用可替代辨别名的任何要求不必反映给终端用户。

图 5 示出了 RDN 和可辨别名概念的一个示例。

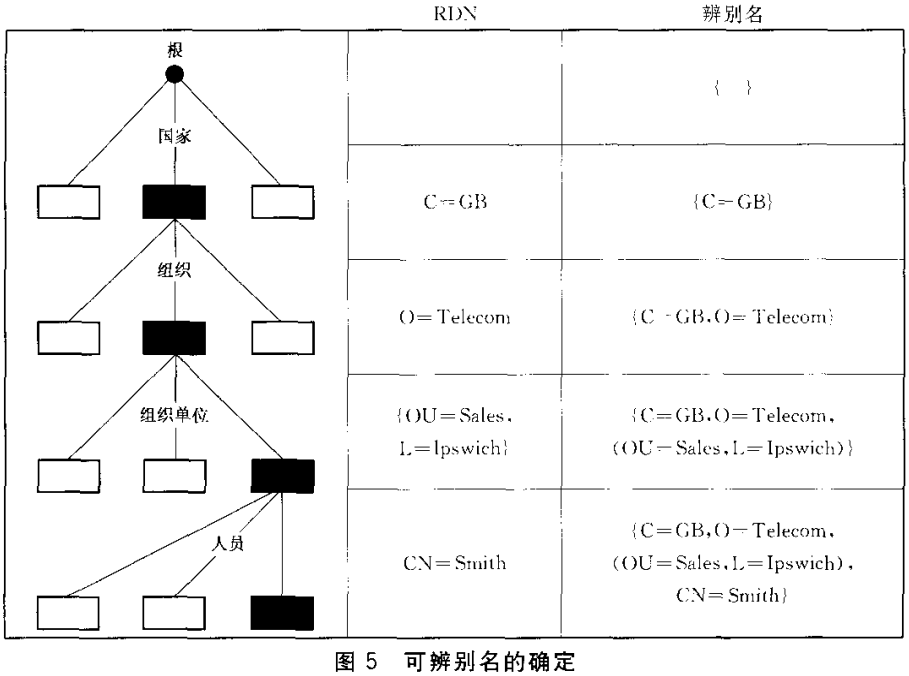


图 5 可辨别名的确定

9.8 别名名(称)

一个客体的别名,或称为别名名(称),是别名条目所指向的客体或客体条目的一个可替代名(称)。

每个别名条目的aliasedEntryName 属性内,都包含某个客体的名(称)。因此,别名条目的可辨别名也是该客体的一个名(称)。

注 1:存储在aliasedEntryName 属性内的名(称)指的是由别名指向的名(称)。它不一定必须是条目的可辨别名。

注 2:AliasedEntryName 属性的取值可能是主辨别名或任意可替代辨别名(如果存在的话)。如果没有使用主辨别名,则会影响到与前 3 版的 DSA 的一致性和互操作性。

一个别名名(称)到客体名(称)的转换被称为(别名的)"解除引用",过程如下:将在声称名中发现的别名名(称),系统地替换为相应的aliasedEntryName 属性的值。该过程可能要求对多个别名条目进行检查。

DIT 中的任何一个具体条目都可能拥有零个或多个别名名(称)。因此,多个别名条目可能会指向同一个条目。一个别名条目可能指向一个非叶结点的条目,也可能指向另一个别名条目。

一个别名条目不得有下级,因此一个别名条目始终是叶条目。

每个别名条目须属于alias 客体类,在 13.3.3 中定义。

家族成员不允许是别名条目。

10 层次结构组

10.1 定义

本目录规范使用下列术语和定义:

10.1.1

层次结构的孩子 hierarchical child

对于一个条目而言,层次结构中的孩子是一个条目,而它是该条目的层次结构中的双亲。

10.1.2

层次结构组 hierarchical group

一个层次结构组是条目(含复合条目)的一个集合,该集合构成了一棵不一定与 DIT 相关的逻辑树。

10.1.3

层次结构的叶子 hierarchical leaf

层次结构组中的一个没有孩子的条目。

10.1.4

层次结构的级 hierarchical level

一个整数,给出了从层次结构中的条目到层次结构顶端之间的距离,该距离是由条目和层次结构顶端之间链的个数来表示的。

10.1.5

层次结构的链 hierarchical link

这是一个通用术语,表示在层次结构中具有直接双亲/直接孩子关系的两个条目间的逻辑关系。

10.1.6

层次结构的双亲 hierarchical parent

对于一个条目而言,层次结构中的双亲指其直接层次结构双亲以及直接层次结构双亲的直接层次结构双亲,依次类推,递归定义直到包含了层次结构中的顶端。

10.1.7

层次结构的兄弟 hierarchical sibling

对于一个条目而言,层次结构中的兄弟是与其自身具有相同直接层次结构双亲的条目。

9.8 别名名(称)

一个客体的别名,或称为别名名(称),是别名条目所指向的客体或客体条目的一个可替代名(称)。

每个别名条目的aliasedEntryName 属性内,都包含某个客体的名(称)。因此,别名条目的可辨别名也是该客体的一个名(称)。

注 1:存储在aliasedEntryName 属性内的名(称)指的是由别名指向的名(称)。它不一定必须是条目的可辨别名。

注 2:AliasedEntryName 属性的取值可能是主辨别名或任意可替代辨别名(如果存在的话)。如果没有使用主辨别名,则会影响到与前 3 版的 DSA 的一致性和互操作性。

一个别名名(称)到客体名(称)的转换被称为(别名的)"解除引用",过程如下:将在声称名中发现的别名名(称),系统地替换为相应的aliasedEntryName 属性的值。该过程可能要求对多个别名条目进行检查。

DIT 中的任何一个具体条目都可能拥有零个或多个别名名(称)。因此,多个别名条目可能会指向同一个条目。一个别名条目可能指向一个非叶结点的条目,也可能指向另一个别名条目。

一个别名条目不得有下级,因此一个别名条目始终是叶条目。

每个别名条目须属于alias 客体类,在 13.3.3 中定义。

家族成员不允许是别名条目。

10 层次结构组

10.1 定义

本目录规范使用下列术语和定义:

10.1.1

层次结构的孩子 hierarchical child

对于一个条目而言,层次结构中的孩子是一个条目,而它是该条目的层次结构中的双亲。

10.1.2

层次结构组 hierarchical group

一个层次结构组是条目(含复合条目)的一个集合,该集合构成了一棵不一定与 DIT 相关的逻辑树。

10.1.3

层次结构的叶子 hierarchical leaf

层次结构组中的一个没有孩子的条目。

10.1.4

层次结构的级 hierarchical level

一个整数,给出了从层次结构中的条目到层次结构顶端之间的距离,该距离是由条目和层次结构顶端之间链的个数来表示的。

10.1.5

层次结构的链 hierarchical link

这是一个通用术语,表示在层次结构中具有直接双亲/直接孩子关系的两个条目间的逻辑关系。

10.1.6

层次结构的双亲 hierarchical parent

对于一个条目而言,层次结构中的双亲指其直接层次结构双亲以及直接层次结构双亲的直接层次结构双亲,依次类推,递归定义直到包含了层次结构中的顶端。

10.1.7

层次结构的兄弟 hierarchical sibling

对于一个条目而言,层次结构中的兄弟是与其自身具有相同直接层次结构双亲的条目。

10.1.8

层次结构的兄弟的孩子　hierarchical sibling-child

对于一个条目而言，它的层次结构中的兄弟孩子是其层次结构中的所有兄弟的所有较低层的孩子的完整集合。

10.1.9

层次结构的顶端　hierarchical top

是层次结构组中的一个条目，该条目是层次的根结点。层次结构的顶端没有直接层次结构双亲。

10.1.10

直接层次结构孩子　immediately hierarchical child

对于一个条目而言，其直接层次结构孩子是一个条目，而它是这个条目的直接层次结构双亲。这个直接层次结构孩子不必一定是 DIT 中的一个直接下级条目。

10.1.11

直接层次结构双亲　immediately hierarchical parent

对于一个条目而言，其直接层次结构双亲是它在层次结构组中的直接上级条目。这个直接层次结构双亲不必一定是 DIT 中的一个直接上级条目。

10.2　层次结构关系

目录条目之间有一种以 DIT 方式所表达的层次结构关系。然而条目之间还可能具有另一种 DIT 结构所没有反映出的层次结构关系。例如，一个动态的组织可能不想在 DI 中直接反映其当前的组织形式，因为这样可能会频繁地改变 DIT 的结构。因此，在目录中就有一种需求，要求能够独立于 DIT 结构来反映某种层次结构关系。层次结构组便构造了这种关系。一个*层次结构组*构成了一棵逻辑树，其根被称为“*层次结构顶端*”。

通过查找层次结构关系，在一个搜索操作中，可能不仅能够获取到一个给定条目的信息，还能够获取到同一层次结构组中的其他条目的信息。

一个复合条目在层次结构组的上下文中被认为是一个单独的条目。其孩子家族成员不能以自己的名义作为层次结构组中的一个独立部分。

注：层次结构组用来对具有逻辑信息关系的不同客体的集合进行建模，这些特定的关系可能是临时的。相反的，复合条目是对由子客体所组成的客体进行建模，这些子客体间被认为是有层次关系的。

为了描述如何在层次结构组中进行导航，一种便利的方法是为组内的某个给定条目与其他条目之间的关系定义一些术语。如 10.1 所述。这些术语中，表示直接关系的术语与 DIT 中为条目关系所定义的术语之间是平行的，如*直接层次结构孩子*、*层次结构中的孩子*、*直接层次结构双亲*和*层次结构中的双亲*等。然而，为更远的关系定义术语也是一种便利方法。在某些情况下，一个用户可能会希望获取*层次结构中的兄弟*的信息，甚至这些兄弟的孩子的信息(*层次结构中的兄弟孩子*)等。

一个条目在某一时间点，只能是单个层次结构组中的成员。

一个层次结构组中的条目拥有如 14.9 所定义的操作属性。这些操作属性反映了该条目与组内其他条目之间的关系，包括该条目在组内所在的“*层次结构的级*”。当一个复合条目是层次结构组的组成部分时，由祖先拥有这些操作属性。

一个层次结构组必须完全处于某个特定服务管理区(见 16.3)外，或者完全包含在一个特定服务管理区内。一个层次结构组应当仅限于一个单独的 DSA 内。目录服务须检测到打破这些规则的企图，并且阻止这些企图。

10.3　层次结构组的排序序列

在某些情况下，如在传送一个层次结构组时，需要一个排序序列规则。一个层次结构的序列顺序来自层次结构组中的所有束，如下所述：

a) 顶端条目是序列中的第一个条目，跟在它后面的是处于一个完整束中的其余条目，一个完整的束指的是从顶端一直到层次结构的某个叶结点。选择哪一束作为第一束属于本地事情；

b) 被选择的下一束是之前没有被选择的束，且与之前选择的束之间有最多数量的公共条目。如

果有多条束在这方面的条件相同,则选择哪一束属于本地事情。仅有之前未包含在内的那些条目才能包含在序列中;

c) 重复 b)的过程,知道所有的束都包括进去。

第四篇:目录管理模型

11 目录管理机构模型

11.1 定义

本目录规范使用下列术语和定义:

11.1.1

管理区 administrative area

从管理的角度看到的 DIT 中的一棵子树。

11.1.2

管理条目 administrative entry

处于某个管理点上的一个条目。

11.1.3

管理点 administrative point

一个管理区的根顶点。

11.1.4

管理用户 administrative user

管理机构的一个代表。管理用户概念的完整定义不在本目录规范的定义范围之内。

11.1.5

自治管理区 autonomous administrative area

DIT 的一棵子树,其上的条目都被同一个管理机构所管理。自治管理区之间是不能重叠的。

11.1.6

DIT 域管理机构 DIT domain administration authority

一个负责对 DIT 的一部分进行管理的管理机构。

11.1.7

DIT 域策略 DIT domain policy

为 DIT 域定义的通用目标和可接受的过程的一个表达式。

11.1.8

DMD 管理机构 DMD administration authority

一个负责对某个 DMD 进行管理的管理机构。

11.1.9

DMD 策略 DMD policy

管理 DMD 内所有 DSA 操作的策略。

11.1.10

DMO 策略 DMO policy

由 DMO 定义的策略,根据 DMD 和 DIT 域策略来表示。

11.1.11

内部管理区 inner administrative area

一个特定的管理区,其范围完全包含在同一类型的另一个特定管理区范围内。

11.1.12

策略 policy

由管理机构确定的通用目标和可接受的过程的一个表达式。

11.1.13

策略属性 policy attribute

表示策略的任意目录操作属性的一个一般术语。

11.1.14

策略客体 policy object

策略涉及到的一个实体。

11.1.15

策略过程 policy procedure

一个规则,定义应当如何对策略客体集进行考虑以及基于此考虑应当执行哪些动作。

11.1.16

策略参数 policy parameter

一个策略过程,该过程的特性由某些策略参数来表示,这些策略参数由管理机构所配置(即选择)。

11.1.17

特定管理区 specific administrative area

自治管理区的一个子集(以子树的形式存在),为管理的某个特定方面而定义:如访问控制,子模式或条目集合管理等。如果定义了特定管理区,则一个自治管理区被分割为不同的具有某种特定类型的特定管理区。

11.1.18

特定管理点 specific administrative point

一个特定管理区的根顶点。

11.2 概述

目录信息模型的基本目标是考虑一个具有良好定义的条目集合,使得它们可以被当做一个单元进行一致地管理。本条澄清了权力机构应对管理所负的职责特性和范围以及他们通过何种方式来执行权力。

11.3 定义的策略概念提供了一种机制,通过这种机制管理机构可以执行对目录的控制。

目录管理模型的某些方面需要目录管理和操作信息模型(见第 12 章)的支持。这就允许对规范目录用户信息以及其他管理目的所需的信息进行建模。

目录管理模型的其他方面需要支持管理和操作信息在目录的不同组件间(即 DSA)进行分布。第 22 到 24 章描述了支持该需求的 DSA 信息模型。

11.3 策略

*策略*是由作为 DMO 代理的管理机构所确定的关于通用目标和可接受的过程的一个表达式。一个策略通过规则和方面来进行定义,其中,规则将被适当的目录强制执行,而方面指的是在其内部,一个管理用户的动作和特定责任有某种程度的自由。

一个管理机构通过如下方面来表示 DMO 策略:

——DIT 域策略;

——DMD 策略。

这些策略可能被表示为策略属性。DIT 策略的模型在 11.6 定义。

注:第 14 章定义了为支持对集合属性的管理而必需的系统模式。第 15 章定义了一个支持子模式管理策略的框架。第 17 章定义了一个支持访问控制策略的框架。

DMD 策略与作为分布式目录组件的 DSA 具体相关。这些 DMD 策略在 11.7 描述,定义了一个 DSA 管理的模型。

最后,还有一些策略是与外部事物(如 DMO 间的双边商定)相关的,本部分不再进行详细描述。

*策略客体*是策略所涉及到的实体(如子模式管理区便是一个策略客体)。

*策略规程*是一个规则，定义了应当如何对策略客体集进行考虑以及基于此考虑应当(在哪种环境下)执行哪些动作(例如，第 15 章定义了子模式管理的策略过程)。

一个策略规程的特性由某些*策略参数*来表示，这些策略参数由管理机构来配置(即选择)。

操作属性用来表示策略参数。这些属性的值构成了它所表示的策略参数的某些或全部表达式。

11.4 特定的管理机构

对 DIT 域的管理包括执行涉及不同方面的 5 大功能：

——命名管理；

——子模式管理；

——安全管理；

——集合属性管理；

——服务管理。

一个*特定的管理机构*是一个管理机构，负责 DIT 域策略的这些特定方面之一。

术语*命名机构*(见第 9 章)标识了管理机构所起的作用，这些作用涉及到对名(称)的分配以及对这些名(称)结构的管理等。一个子模式机构的作用是在一个子模式内实现这些命名结构。

术语*子模式机构*标识了管理机构所起的作用，这些作用涉及到对子模式策略的建立、管理和执行，而子模式策略可以在一个 DIT 域内控制条目的命名和内容。第 15 章描述了目录对子模式管理的支持。

术语*安全机构*(ISO/IEC 9594-8)标识了管理机构所起的作用，这些作用涉及到对安全策略的建立、管理和执行，安全策略可以控制目录的与 DIT 域内条目相关的行为。

术语*集合属性机构*标识了管理机构所起的作用，这些作用涉及到对 DIT 域内的集合属性(见 12.7)的建立和管理。

术语*服务机构*标识了管理机构所起的作用，这些作用涉及到对服务约束和调整的建立和管理。

11.5 管理区和管理点

11.5.1 自治管理区

DIT 内的每个条目都由一个且仅由一个管理机构来管理(该管理机构可能具有不同的作用)。一个*自治管理区*是 DIT 的一棵子树，其上的所有条目都由同一个管理机构来管理。

DIT 域可被分割为一个或多个互不重叠的自治管理区。

如果一个 DMO 对一个或多个自治管理区的集合具有管理权力，则这些自治管理区是该 DMO 的 DIT 域，如图 6 所示。

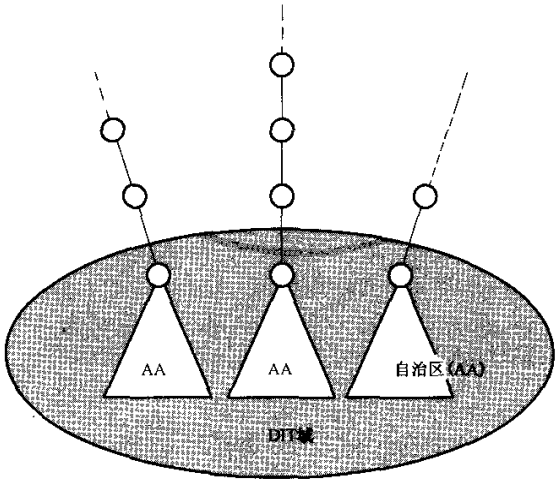


图 6 一个 DIT 域

11.5.2 特定管理区

以同样的方式，一个管理机构可能会起到某个特定的作用，而管理区内的条目可能会由于这种特定的管理功能而被考虑。在这种上下文中，管理区被称为**特定管理区**。有5种类型的特定管理区：

——子模式管理区；

——访问控制管理区；

——集合属性管理区；

——上下文缺省管理区；

——服务管理区。

一个自治管理区可能被认为是隐含地为管理的每个特定方面都定义了一个单独的特定管理区。在这种情况下，在每个特定管理区和自治管理区间有确切的对应关系。

另一种可替换方式是，对于管理的每个特定方面，自治管理区可能被分割为相互不重叠的特定管理区。

如果为管理的每个方面进行了如上分割，则自治管理区内的每个条目将被包含在一个且仅包含在一个该方面的特定管理区内。

每个特定管理区都由一个特定的管理机构来负责。对于一个特定的管理方面，如果一个自治管理区未被分割，则一个特定的管理机构将负责整个自治管理区的该方面的管理。

11.5.3 内部管理区

出于安全或集合属性管理的目的，需要定义这些特定管理区中的*内部(管理)区*：

a) 为了表示授权的一个限定格式；或者

b) 为了管理或操作的便利(例如，一棵子树的管理点所在的DSA，与拥有子树内条目的DSA不同，在这种情况下，该子树可能会被设计为一个内部区，以便允许通过本地DSA进行管理)。

一个内部管理区可能嵌套在另一个内部管理区内。

内部区表示一个具有有限自治能力的区域。内部区内的条目由它们所在的特定管理区内的特定管理机构进行管理，同时也由它们所在的内部区的管理机构进行管理。前一个管理机构对规范这些条目的策略具有全局的控制能力；而后一个管理机构对前者授予的权限具有有限的控制能力。

是否允许内部区进行嵌套的规则，应当作为包含它们的特定管理方面的定义中的一部分。

11.5.4 管理点

对自治管理区范围的规定是隐含的，它包含DIT中的一个点(自治管理区所在子树的根)，即一个*自治管理点*(AAP)，从该点开始一直往下直至遇到另一个自治管理点，这期间的范围即为一个管理区，而另一个自治管理点则是另一个自治区的开始。

注1：DIT中根的直接下级是一个自治管理点。

若自治管理区没有为了管理的某个特定方面而被分割，则该方面的管理区与自治管理区是一致的。在这种情况下，自治管理点同时也是该特定管理方面的*特定管理点*。

当自治管理区为了管理的某个特定方面而被分割，则对每个特定管理区范围的规定，包括特定管理区所在子树的根，即*特定管理点*(SAP)，从该点开始一直往下直至遇到另一个(同一个管理方面的)特定管理点，这期间的范围即为特定管理区，而另一个特定管理点则是另一个特定管理区的开始。

特定管理区总是由它们所分割的自治管理区来定界的。

一个管理点可能是一个自治管理区的根，也可能是一个或多个特定管理区的根。

对一个(特定管理区内的)内部管理区范围的规定，包括内部管理区所在子树的根，即一个*内部管理点*(IAP)。一个内部管理区由其所在的特定管理区所定界。

自治管理区的根相应的管理点，表示一个DIT域(和DSA)的边界。也就是说，在DIT中它的直接上级必须由另一个DMD的管理机构进行管理。

注2：这就意味着一个DMO不能随意地将DIT域分割为自治管理区。

在目录信息模型中，一个管理点由一个拥有administrativeRole属性的条目表示。该属性的值标识了管理点的类型，该属性在14.3定义。

第22到24章描述了管理区如何映射到DSA和DSA信息模型。

图7示例了一个自治管理区，为了某个特定的管理方面(如访问控制)，该域被分割为两个特定管理区。在一个特定管理区内，创建了一个内嵌的内部管理区(如：因为该子树由另一个不同的DSA所拥有，不同于该特定管理区的其余部分)。

图7使用了缩写AAP(自治管理点)、SAP(特定管理点)和IAP(内部管理点)。

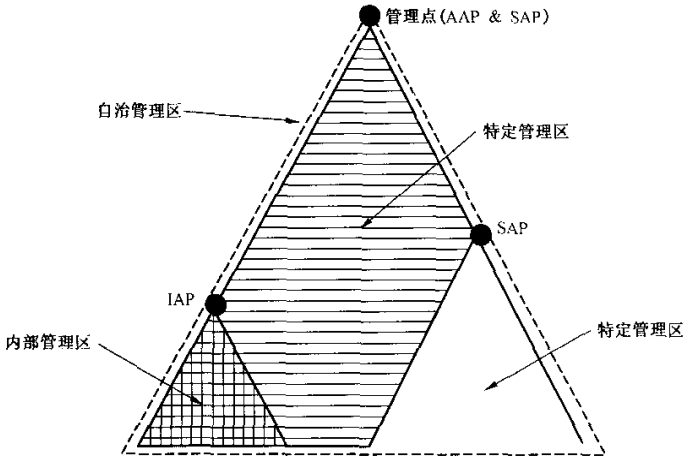


图7 管理点和管理区

11.5.5 管理条目

一个处于管理点的条目被称为"*管理条目*"。管理条目可能有一种特殊的条目作为它的直接下级，它们被称为"*子条目*"。管理条目及其相关的子条目可被用于对相关管理区内的条目进行控制。

当使用内部管理区时，这些区的范围可能会有重叠。

因此，对于管理机构的每个特定方面，若有条目可能包含在多个与该特定管理方面所定义的内部区相关的子树或子树精选内，则需要定义管理信息的组合方法。

注：对于一个管理点而言，不必表示管理机构的每个特定方面。例如，可能有一个管理点，是自治管理区根结点的直接下级，它仅用于访问控制的目的。

11.6 DIT域策略

一个DIT域策略具有如下组件：DIT策略客体、DIT策略规程和DIT策略参数。

一个表示DIT策略参数的操作属性被称为"*DIT策略属性*"(如第14章定义的子模式管理操作属性便是DIT域策略属性)。

对于一个特定的DSA，一个策略参数的可能取值与该组件动作的特定的、可实现的过程之间可能不是对应的。可能是这样一种情况，例如，DSA缺乏技术能力来执行策略过程的所有方面(如实现一个特定的访问控制方案)。为了良好地定义一个策略过程，在其定义中，应当考虑这些情况。

特定的DIT域策略客体和属性在第15章定义，以支持子模式管理。

11.7 DMD策略

一个*DMD策略*是一个策略，涉及到对DMD中的一个或多个DSA的操作。一个DMD策略可能以一种统一的方式应用于DMD中的所有DSA，或者应用于DMD中的一个DSA子集，或者应用于一个特定的DSA。

一种DMD策略是为了限制，或者控制目录以及由一个或多个DSA所提供的DSA抽象服务。

这些限制的示例如下：

a) 将提供给目录用户(非管理用户)的基本服务限制为仅提供查找服务,根据 CCITT F.500 建议书的定义;

b) 对提供给间接访问 DSA(而是通过一个链接访问 DSA)的用户的服务进行限制,包括根据用户的请求是否经过一个可靠的路径来对用户进行区分;

c) 对直接访问 DSA 的用户所发出的请求进行限制,如果这类请求所要链接的 DMD 内的 DSA,已知应当遵循上一条中指示的限制类型时,则应对这类请求进行限制;

d) 对某些用户可以执行的搜索类型进行限制,并对这些搜索的特性加以限制(如放宽策略)。

第五篇:目录管理和操作信息模型

12 目录管理和操作信息模型

12.1 定义

本目录规范使用下列术语和定义:

12.1.1

基 base

由子树规范评定产生的子树的根顶点或子树精选的根顶点。

12.1.2

切除(集) chop

与基的下级名(称)相关的断言集合。

12.1.3

目录操作属性 directory operational attribute

在目录管理和操作信息模型中定义并可见的一个操作属性。

12.1.4

目录信息模式 directory system schema

与操作属性和子条目相关的规则和限制的集合。

12.1.5

条目 *entry*

指一个目录条目或一个扩展的目录条目,取决于使用该术语时的上下文(或者是用户及其应用,或者是目录的管理和操作)。

12.1.6

子条目 subentry

目录所了解的一类特殊的条目,用于保持与子树或子树精选相关的信息。

12.1.7

子树 subtree

位于一棵树的顶点上的客体和别名条目的一个集合。该术语的前缀"子"强调了该树的基(或根)顶点常常是 DIT 根的下级。

12.1.8

子树精选 subtree refinement

在一棵子树上明确指定的一个条目子集,这些条目不是位于单个子树的顶点上。

12.1.9

子树规范 subtree specification

一棵子树或子树精选的明确的规范。子树规范包括零或多个规范元素:基、切除(集)(chop)和规范过滤器。该定义术语为"明确的"(与管理区的定义形成对比),是因为从基下级开始的包含在子树或子

树精选中的 DIT 部分是有明确规定的。

12.2 概述

从管理的角度来看，DIB 中所拥有的用户信息可以由管理和操作信息来补充，这些信息的表示包括：

——*操作属性*：表示用于控制目录操作的信息（如访问控制信息），或者是由目录使用，表示其操作的某些方面特性（如时戳信息）；以及

——*子条目*：将某个属性集（如集合属性）的值与子条目范围内的条目相关联起来。子条目的范围是一棵子树或子树精选。

如图 8 所示，这些信息可能由管理机构或 DSA 置于目录中，并且由目录在执行操作时使用。

目录抽象服务中，与上述视图的目录信息相关的两个机制是：

——EntryInformationSelection：已扩展为允许对条目中的操作属性进行选择；以及

——已增加swbentries 服务控制：允许对客体或别名条目或子条目应用列表和搜索操作。

对操作信息的访问，如同对用户信息的访问一样，可能被访问控制机制所限制。

通过目录抽象服务，可以使得条目对目录用户可见，但是它们与最终存储它们的 DSA 间的关系是不可见的。在第 22 章到第 24 章中描述的 DSA 信息模型，表示了这些条目到 DSA 信息存储器之间的映射关系。

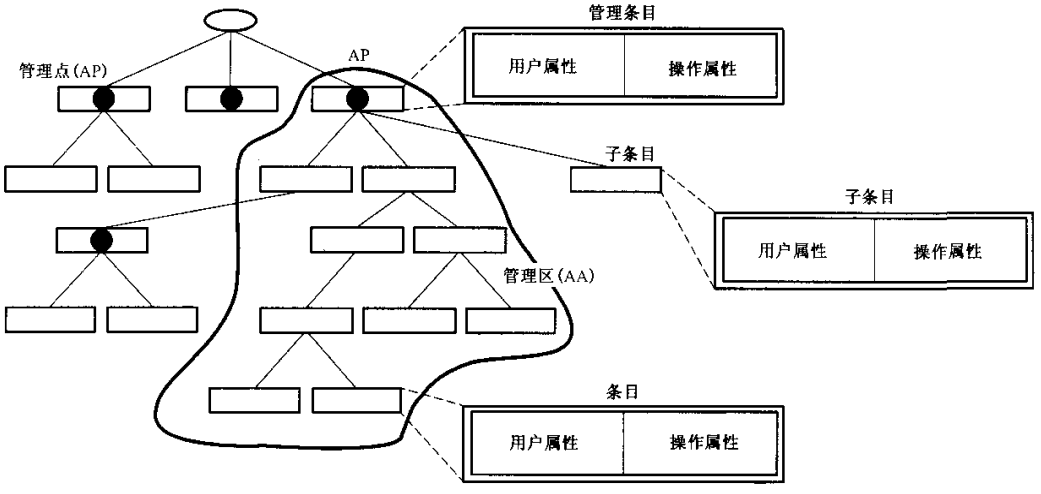


图 8 目录管理和操作信息模型

12.3 子树

12.3.1 概述

一棵子树是位于一棵树的顶点上的客体条目和别名条目的一个集合。子树不包含子条目。该术语的前缀“子”强调了该树的基（或根）顶点常常是 DIT 根的下级。

一棵子树从某个顶点开始，一直扩展到某个可标识的低层边界，可能扩展到叶结点。一棵子树总是在某个上下文内定义的，该上下文隐含地限定了该子树。例如，定义了一个复制区子树的顶点和低层边界由一个命名上下文所限定。类似的，定义了某个特定管理区的子树，其范围被限制到包围它的自治管理区的上下文中。

12.3.2 子树规范

子树规范是对处于某个特定顶点之下的条目子集的定义，该特定顶点构成了子树或子树精选的基。

可能会对子树的顶点和/或低层边界进行隐含定义，在这种情况下，它们由使用子树时的上下文所决定。

可能会使用本条所规定的机制对子树的顶点和/或低层边界进行明确定义。该机制还可能会用于规定非真正树型结构的子树精选。

注：在认识这些规范时，(子)树的拓扑概念是很有用的，尽管一个特定的规范可能决定的条目集并非是处于一个单独子树的顶点上。当集合中的条目不是处于一个单独子树的顶点上时，则使用术语子树精选。

一个子树的规范包括3个可选的规范元素，这些元素规定了子树的基客体，因此缩小了下级条目的集合。这3个规范元素为：

a) *基*：由子树规范评定所产生的子树或子树精选的根顶点；

b) *切除(集)*：与下级条目的名(称)相关的声明集合；以及

c) *规范过滤器*：一个应用于下级的过滤器所声明的能力的正确子集。

子树或子树精选的规范由下面的ASN.1类型表示：

```
SubtreeSpecification  ::=SEQUENCE{
    Base                    [0]  LocalName DEFAULT { },
                                 COMPONENTS OF ChopSpecification,
    specificationFilter     [4]  Refinement OPTIONAL }
```

——空序列规定整个管理区

该序列中的3个组件与上面所描述3个规范元素相对应。

当SubtreeSpecification的值标识了一个条目的集合，而这些条目都是位于一个单独子树的顶点上时，这样的集合被称为一棵“子树”；否则，这样的集合被称为一棵“子树精选”。

SubtreeSpecification类型为子树和子树精选的规范提供了一个具有通用目的的机制。该机制的任何具体应用都会精确地定义具体要规定的语义，并且可能会对SubtreeSpecification的组件加以限制或约束。

如果SubtreeSpecification中的每个组件都缺失的话(即类型SubtreeSpecification的值为一个空序列，{})，则这样指定的子树由SubtreeSpecification所使用时的上下文所隐含确定。

这些术语在图9中举例说明，这些子树都部署在管理区上下文内。

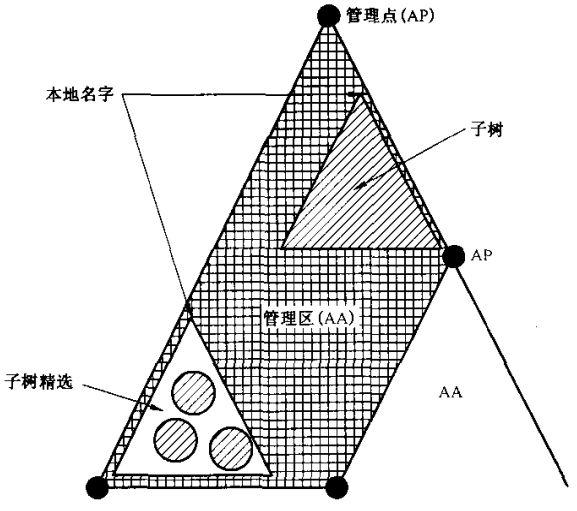


图9 管理区上下文内的子树和子树精选规范

12.3.3 基

SubtreeSpecification中的base组件表示子树或子树精选的根顶点。它可能是一个指定范围内的根顶点的下级条目，或者是一个指定范围的根顶点本身(缺省的情况)。

子树的根结点相对于所指定范围内的根结点的相对名(称)，是类型为LocalName的值：

LocalName ::=RDNSequence

注意当SubtreeSpecification 中省略了LocalName 字段时，则所指定范围内的根顶点和子树的根顶点是一致的。

用于对子树的根顶点命名的 RDN 须是主 RDN。

12.3.4 切除(集)(Chop)规范

ChopSpecification 组件包含了与基的下级的名(称)相关的声明集。它由类型为ChopSpecification 的值组成：

```
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 }
```

该类型允许用两种方法对一个以基开始的树型结构(或其子集)进行规范化：特殊例外和基距离。

如果在chopBefore 或chopAfter 内的 RDN 的任意一个属性有多个通过上下文来区分的可辨别值时，则须使用主可辨别值作为LocalName 中的 RDN 的 value 的值。

12.3.4.1 特别例外

组件specificExclusions 有两种形式：chopBefore 和chopAfter，这两种形式可以单独使用或联合使用。

组件chopBefore 定义了一个例外列表，每个例外都定义了一些限制点，这些限制点与其下级一起，将被排斥在子树或子树精选之外。限制点是由相对于基的LocalName 所标识的一些条目。

组件chopAfter 定义了一个例外列表，每个例外都定义了一些限制点，其下级将被排斥在子树或子树精选之外。限制点是由相对于基的LocalName 所标识的一些条目。

12.3.4.2 最小值和最大值

这两个组件允许排除这样两类条目：在基之下，且 RDN 弧的个数为minimum 的条目的所有上级条目；在基之下，且 RND 弧的个数为maximum 的条目的所有下级条目。这种距离通过类型BaseDistance 的值来表示：

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

出于切除(集)规范的目的，一个复合条目被当做一个独立的条目来计数。在一个复合条目内，所有的家族成员都被认为具有与祖先相同的基距离，因为它们是同一个逻辑条目的所有部分。

若minimum 的值等于 0(缺省值)，则表示基。如果组件maximum 缺失则表示不对子树或子树精选的低限加以限制。

12.3.5 规范过滤器

组件specificationFilter 包含了应用于基下级的过滤器(GB/T 16264.3—2008)所声明的能力的一个正确子集。只有使得过滤器评估为真的条目才能够包含在最终的子树精选中。它由类型Refinement 的值组成：

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

如果Refinement 被评估为TRUE，则好像它是一个仅仅针对属性类型为objectClass 的值执行了一个相等声明的过滤器一样。

如果一个家族成员根据此规范被排斥在子树之外，则其所有下级家族成员都将被排斥在外。

12.4 操作属性

有3种不同的操作属性：目录操作属性、DSA共享的操作属性和DSA特定的操作属性。

*目录操作属性*出现在目录信息模型中，并被用于表示控制信息（如访问控制信息）或其他由目录提供的信息（如指示某个条目是一个叶条目还是非叶条目）。

*DSA共享的操作属性*仅出现在DSA信息模型中，并且在目录信息模型中是完全不可见的。

*DSA特定的操作属性*仅出现在DSA信息模型中，并且在目录信息模型中是完全不可见的。

注：这些信息模型在第23章到第24章中描述。

每种操作属性的定义和用法都需要在适当的目录规范中进行规范。

12.5 条目

12.5.1 概述

从管理的角度来看，一个条目所拥有的用户信息可以通过操作属性所表示的管理和操作信息来补充。

目录使用客体通用性和应用于某个条目的DIT内容规则来控制条目所需的用户属性和允许放入条目中的用户属性。一个条目的操作属性由应用于该条目的目录系统模式（见第14章）来支配。

12.5.2 操作属性的访问

尽管一般来说是不可见的，但对目录抽象服务的授权用户（如管理用户）来说，条目内的目录操作属性可能会变得可见。而某些操作属性（如entryACI或modifyTimestamp等）可能对普通用户也是可见的。

12.6 子条目

12.6.1 概述

子条目是一种特殊类型的条目，是某个管理点的直接下级。它包含的属性所涉及的内容是与管理点相关的一棵子树（或子树精选）。子条目与它的管理点是同一个命名上下文（见第21章）中的一部分。

一个单独的子条目可能会服务于管理机构的所有方面或部分方面。可选的，一个管理机构的某个特定方面也可能会由一个或多个其子条目来处理。一个子模式管理机构最多允许一个子条目。访问控制机构和集合属性机构可能会拥有多个子条目。

在列表和搜索操作中不考虑子条目，除非在请求中包含了subentries服务控制。

子条目不应有下级。

与某个管理点相对应的子条目的结构如图10所示。

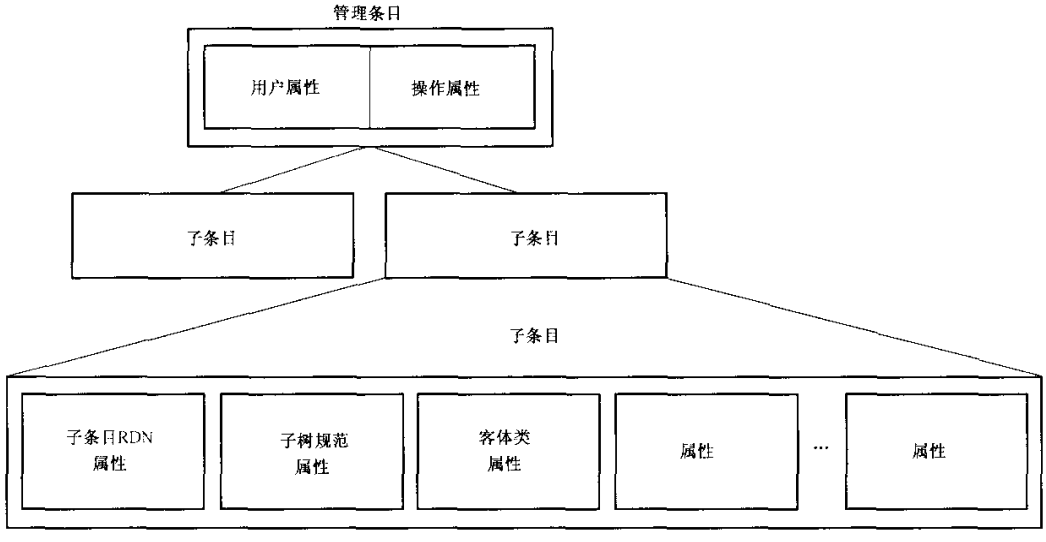


图10 子条目的结构

一个子条目由以下组成：

——一个commonName 属性，在 GB/T 16264.6—2008 中规定，包含了子条目的 RDN；

——一个subtreeSpecification 属性，在第 14 章规定；

——一个objectClass 属性，在第 13 章规定，指示了子条目在目录操作中的目的；

——其他取决于objectClass 属性值的属性。

子条目还可能包含具有正确语义的操作属性(见 12.6.4)。

12.6.2 子条目 RDN 属性

commonName 属性用做子树标识符，可用于区分定义为某个特定管理条目的直接下级的不同子条目。

注：该属性的值可能会被选中作为代表管理机构的记忆符号。

子条目的commonName 属性可能不包括由上下文所区分的多个可辨别值，仅允许使用一个单独的辨别值。

12.6.3 子树规范属性

subtreeSpecification 属性定义了该子树所涉及的管理区内的条目集合。

12.6.4 客体通用性的用法

子条目的内容由子条目的属性objectClass 的值所规定。

所有子条目的属性objectClass 都须包含值subentry 。客体类subentry 是一个结构客体类，在第 14 章定义，用于在所有子条目内包含commonName 、subtreeSpecification 和objectClass 属性。

为了规定其余的属性，须使用objectClass 属性的其他值，这些值代表了子条目所允许的辅助客体类。

对这些其余值之一的语义定义包括零个或多个属性类型的标识符和规范，当属性objectClass 取这些值时，这些属性类型应当或可能出现在子条目中。属性objectClass 的值的语义定义须包括：

——指示一个条目是否可以包含在与某个特定目的相关的多个子树或子树精选内(如在子模式情况下可能不允许，但为了访问控制可能会被允许)；且假如这样的话，

——相关的子条目属性的组合效果，如果存在的话。

如果属性administrativeRole 允许一个特定客体类的子条目做下级，则该子条目可能仅能作为一个管理条目的下级。

对于客体和别名条目，子条目所拥有的信息可以通过操作属性所表示的管理和操作信息来补充。例如，一个子条目允许包含条目 ACI，只有在该 ACI 被相应的访问控制特定点的accessControlScheme 属性的值所允许，且与该属性值一致的情况下才可以。类似的，一个子条目可能包含一个modifyTimestamp 。

12.6.5 其他子条目属性

子条目内的其他属性取决于objectClass 属性的值。例如，只要当objectClass 属性的值中包含subschema 时，一个子模式属性才可能包含在子条目中。

12.7 集合属性的信息模型

为了部署和管理集合属性，一个自治管理区可能会被指定为一个集合属性特定管理区。这就需要通过在相关管理条目的属性administrativeRole 中取值为id-ar-collectiveAttributeSpecificArea 来指示(此外，还应当取值autonomousArea 以及其他可能的值)。

为了在特定的部分内部署和管理集合属性，这样的一个自治管理区可能会被分割为不同的部分。在这种情况下，每个集合属性特定管理区的管理条目可以通过在这些条目的属性administrativeRole 中取值为id-at-collectiveAttributeSpecificArea 来指示。

如果这样的一个自治管理区不进行分割，则对于集合属性有一个单独的特定管理区，该管理区包含了整个自治管理区。

另外，为了集合属性管理而定义的特定管理区可进一步划分为具有相同目的的嵌套内部区。每个内部管理区管理条目的administrativeRole 属性应通过id-ar-collectiveAttributeInnerArea 值的存在来指示上述情况。

一个条目集合及其相关的集合属性在目录信息模型中通过子条目来表示，该子条目被称为"*集合属性子条目*"，其objectClass 属性取值为id-sc-collectiveAttributeSubentry，如第 14 章所描述。该类的一个子条目可能是某个管理条目的直接下级，该管理条目的属性administrativeRole 包含值id-ar-collectiveAttributeSpecificArea 或id-ar-collectiveAttributeInnerArea 。

若在一个给定的集合属性区内有不同的条目集合，则每个条目集合都须有自己的子条目。

条目集合自身是通过子条目的操作属性subtreeSpecification 的值来定义的。该值定义了集合属性子条目的范围。子条目的用户属性是该条目集合的集合属性。

注 1：由于子树精选是基于客体类的，因此集合属性与客体条目之间的关联可以通过对这些条目的模式进行自然地扩展而实现。例如，一个组织的organizationalPerson 条目可能会通过一个适合组织内所有成员的集合属性集进行扩展，其方法是创建一个子条目，其相关的子树被细化为仅包含organizationalPerson 条目，该条目包含了组织的集合属性集。此外，需要定义该条目的 DIT 内容规则，以便允许集合属性在条目内可见。

集合属性类型和非集合属性类型在语义上是不同的。一个能够表示集合语义的属性类型须在定义时便指定其为一个集合属性类型。

注 2：当一个集合属性类型的值是独立资源时，目录所部署的合并过程，在 GB/T 16264.3—2008 中描述。

通过使用第 14 章定义的属性collectiveExclusions，集合属性可能会不允许出现在某个特定条目内。

12.8 上下文缺省值的信息模型

为了部署和管理上下文缺省值，一个自治管理区可能会被指定为一个上下文缺省值特定管理区。这就需要通过在相关管理条目的属性administrativeRole 中取值为id-ar-contextDefaultSpecificArea 来指示（此外，还应当取值id-ar-autonomousArea 以及其他可能的值）。

为了在特定部分内部署和管理上下文缺省值，这样的一个自治管理区可能会被分割为不同的部分。在这种情况下，每个上下文缺省值特定管理区的管理条目，都应当通过在条目的属性administrativeRole 中取值为id-ar-contextDefaultSpecificArea 来指示。

如果这样的一个自治管理区不进行分割，则对于上下文缺省值有一个单独的特定管理区，该管理区包含了整个自治管理区。

上下文缺省值在目录信息模型中通过子条目来表示，该子条目被称为"*上下文缺省值子条目*"，其objectClass 属性具有值id-sc-contextAssertionSubentry，如 14.7 所描述的。该类的一个子条目可能是某管理条目的直接下级，该管理条目的属性administrativeRole 中包含值id-ar-contextDefaultSpecificArea 。

上下文缺省值子条目定义了一个上下文断言集，在访问由子条目的操作属性subtreeSpecification 所确定的 DIT 一部分时，如果用户没有指定上下文断言可以应用到某个给定的属性类型，则会应用该声明集中的一个声明。缺省的上下文断言的应用在 8.9.2.2 以及 GB/T 16264.3—2008 的 7.6.1 描述。

第六篇：目录模式

13 目录模式

13.1 定义

本目录规范使用下列术语和定义：

13.1.1

属性句法 attribute syntax

用于表示属性值的 ASN.1 数据类型。

13.1.2

目录模式 directory schema

决定 DIB 特性的 DIT 结构、DIT 内容、DIT 上下文的用法、客体类以及属性类型、句法和匹配规则的一系列规则和约束的集合。目录模式体现为一系列互不重叠的子模式，每个子模式控制着一个自治管理区(或其中的一个子模式特定部分)内的条目。目录模式仅与目录用户信息相关。

13.1.3

(目录)子模式 (directory) subschema

一个自治管理区(或其中的一个子模式特定部分)内，决定 DIB 特性的 DIT 结构、DIT 内容、客体类以及属性类型、句法和匹配规则的一系列规则和限制的集合。

13.1.4

DIT 内容规则 DIT content rule

控制某个特定的结构客体类的条目内容的规则。它规定了在指定的结构客体类的条目中，允许出现的或不允许出现的辅助客体类和附加的属性类型。

13.1.5

DIT 上下文用法 DIT context use

控制可能与特定属性类型的属性值相关联的上下文类型的规则。它规定了该属性类型所允许的和必选的上下文类型。

13.1.6

DIT 结构规则 DIT structure rule

控制 DIT 结构的规则，它规定了所允许的上级条目与下级条目之间的关系。一个结构规则将一个名(称)格式以及相应的结构客体类，与其上级结构规则联系起来。这就使得由名(称)格式所标识的结构客体类的条目可以处于 DIT 内，并且是作为所指定的上级结构规则所控制的条目的下级。

13.1.7

(条目的)控制结构规则 governing structure rule (of an entry)

是应用于某个特定条目的一个单独的 DIT 结构规则。该规则通过操作属性governingStructureRule 来指示。

13.1.8

名(称)格式 name form

名(称)格式规定了某个特定结构客体类的条目所允许的 RDN。名(称)格式标识了一个已命名的客体类以及一个或多个用于命名(即：用于 RDN)的属性类型。名(称)格式是在 DIT 结构规则的定义中需要主要规范的部分。

注：名(称)格式被登记，且具有全球范围。DIT 结构规则不被登记，且其范围是与之相关联的管理区。

13.1.9

上级结构规则 superior structure rule

是关于一个特定条目的，控制该条目上级的 DIT 结构规则。

13.2 概述

目录模式是关于下面内容的一系列的定义和约束，包括：DIT 的结构，条目命名的可能方法，条目可能拥有的信息，用来表示这些信息的属性，为了便于搜索和查询这些信息所组织成的层次结构以及属性值在属性值和匹配规则断言中是如何匹配的等。

注 1：模式使得目录系统能够，例如：

——防止创建具有错误客体类的下级条目(如国家作为人的下级);

——防止向一个条目增加不适合该客体类的属性类型(如对人的条目增加一个序列号);

——防止增加一个句法与所定义的属性类型不匹配的属性值(如将可打印字符串赋值给比特串)。

从形式上,目录模式由下述部分组成:

a) *名(称)格式定义*:为结构客体类定义了基本的命名关系;

b) *DIT 结构规则*定义:定义了条目可能拥有的名(称)以及 DIT 中条目之间可能相互关联的方法;

c) *DIT 内容规则*定义:扩展了条目所允许具备的属性的规范,超出了条目的结构客体类所指示的那些属性;

d) *客体类*定义:分别定义了一个给定类型的条目中,必须出现的和可能出现的必选及可选属性的基本集合,并且指示了被定义的客体类的类型(见 7.3);

e) *属性类型*定义:标识了用以确定属性的客体标识符、句法、相关的匹配规则,是否为一个操作属性,如果是的话,其类型是什么,是否为一个集合属性,是否允许具有多个值以及该属性是否是从另外的属性类型派生而来等;

f) *匹配规则*定义:定义了匹配的规则;

g) *DIT 上下文用法*定义:控制可能与任意一个特定属性类型的属性值相关联的上下文类型。

图 11 中,一方面举例说明了模式和子模式定义之间的关系,另一方面举例说明了 DIT、目录条目、属性和属性值之间的关系。

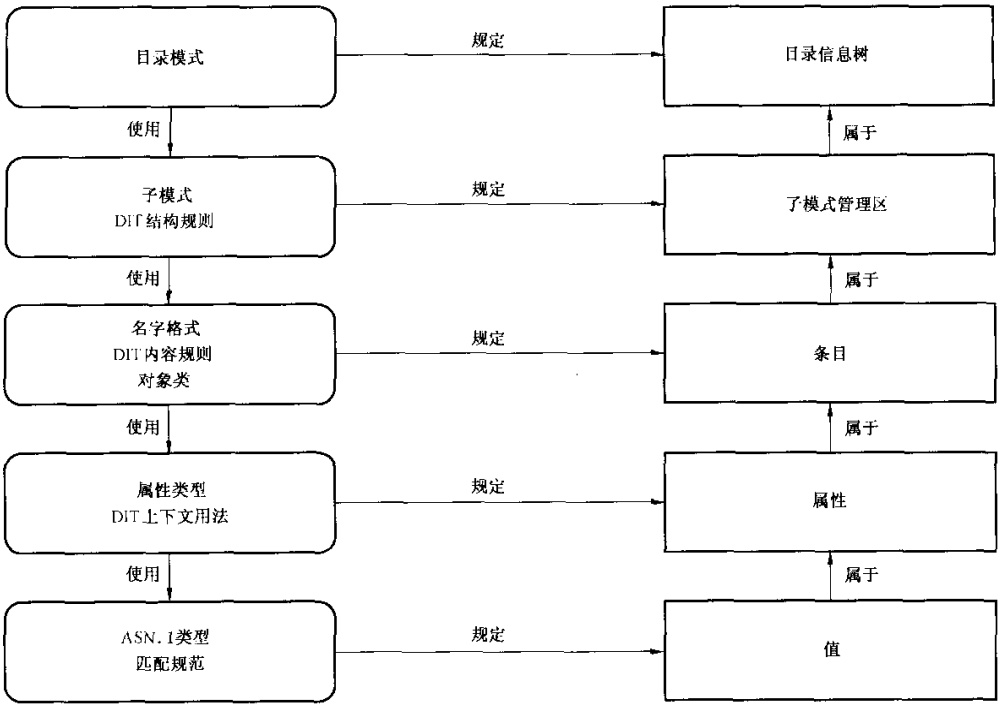


图 11 目录模式概述

对图 11 解释如下:

——左侧垂直列出的项表示模式的元素;

——右侧垂直列出的项表示相应的模式项的实例,是根据这些模式项所定义的规则进行实例化的;

——模式项之间的关系为示例中的"使用"关系;

一模式不同方面的实例之间的关系为示例中的"属于"关系。

如同DIB本身一样,目录模式也是分布式的。它体现为一系列互不重叠的子模式,每个子模式控制着一个自治管理区(或其中的一个子模式特定部分)内的条目。子模式管理机构建立组成子模式的规则和限制。

子模式的管理机构可能会选择使用本系列目录规范中定义的具有全球范围的目录模式中的单独元素:如名(称)格式、客体类和属性(类型和匹配规则)等。也可能会选择定义这些元素的替代方式来更好地符合自身的环境;再或者选择一种中间方式,既使用标准的模式元素又定义私有的模式元素。

子模式的管理机构定义模式元素的范围仅限于子模式,如:DIT结构规则、DIT内容规则和DIT上下文用法。另外,子模式的管理机构还可能会规定哪个属性类型应用哪个匹配规则。

目录模式仅与目录用户信息相关。尽管在本条所定义的表示法中提供了对操作信息规范的部分支持,但目录管理和操作信息的规范是目录系统模式所关心的。

注2:目录系统模式在第14章描述。

13.3 客体类定义

客体类的定义包括如下内容:

a) 指示该类是哪个客体类的子类;

b) 指示被定义的客体类是哪种类型;

c) 列出除了客体类所有上级类的*必选*属性类型外,该客体类的条目中还应当包含的必选属性类型;

d) 列出除了客体类所有上级类的*可选*属性类型外,该客体类的条目中还可能包含的可选属性类型;

e) 为客体类分配一个客体标识符。

注:集合属性不会出现在一个客体类定义的属性类型中。

13.3.1 子类划分

在子类划分中有一些限制,即:

只有抽象客体类才能作为其他抽象客体类的上级类。

有一种特殊的客体类,每个结构客体类都是它的子类。该客体类被称为top。top是一个抽象客体类。

13.3.2 客体通用性

每个条目应当包含一个类型为objectClass的属性,该属性标识了条目所属于的客体类及其上级类。该属性的定义在13.4.8中给出。该属性是多值的。

在objectClass属性中,须有一个值表示该条目的结构客体类,另外,对于其每个上级类都有一个相应的值。top可能被忽略。

一个条目的结构客体类不得改变。objectClass属性的初始值在条目创建时由用户提供。

当使用辅助客体类的时候,条目的objectClass属性值中可能会包含DIT内容规则所允许的辅助客体类及其上级类。如果存在一个所允许的辅助客体类的值,则该辅助客体类的上级类的值也应当存在。

当objectClass属性中包含一个辅助客体类的客体标识符值时,则条目应当包含该客体类所指示的必选属性。

注1:在每个条目中都应出现objectClass属性的需求在top客体类的定义中体现。

注2:由于一个客体类被认为是属于其所有的上级类的,因此,其直到top客体类的上级类链中的每个成员都在属性objectClass的值中表示出来(且上级类链中的任意值都可能被过滤器所匹配)。

注3:在修改objectClass属性时,可能会使用访问控制限制。

与所应用的DIT内容规则一起,目录为DIB中的每个条目都实施了所定义的客体类。任何试图要修改条目,但违反了条目客体类的定义,而条目的DIT内容规则又没有明确允许的,则这样的修改将

失败。

注4：特别地，目录一般来说还将：

a) 如果属性类型在条目的结构客体类的定义中没有定义，且条目的DIT内容规则不允许，则这样的属性类型不能被加入到该客体类的条目中；

b) 如果条目不具备相应的客体类所定义的一个或多个必选属性类型，则该条目不能被创建；

c) 条目的客体类所定义的必选属性类型不能被删除。

13.3.3 客体类规范

客体类被定义为OBJECT-CLASS信息客体类的值：

```
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) }
```

对于每个使用此信息客体类定义的客体类：

a) &Superclasses是一个表示其直接上级类的客体类集合；

b) &kind表示其类别；

c) &MandatoryAttributes是属性的集合，该客体类的条目必须包含这些属性；

d) &OptionalAttributes是属性的集合，该客体类的条目可能包含这些属性，如果一个属性既出现在必选属性集中，又出现在可选属性集中，则该属性必须被认为是必选的；

e) &id为客体类分配的客体标识符。

之前提及的客体类(top和alias)定义如下：

```
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 }
```

注1：客体类alias没有为别名条目的RDN规定适当的属性类型。管理机构可能会定义alias类的子类来为别名条

目的 RDN 规定有用的属性类型。

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

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

parent 客体类和child 客体类都不能与alias 客体类结合起来形成一个别名条目。

parent 客体类是根据直接下级家庭成员的出现而推派生的，标志为出现了child 客体类的值。它可能不能被直接管理。child 客体类的值被增加或删除，只有当增加或删除的结果与复合条目的结构保持一致时才可以(例如，家族成员的下级应当总是具有一个child 客体类)。

注 2：客体类parent 和child 没有为家族成员的 RDN 规定任何合适的属性类型。这种规定可以通过常规方法完成，即通过这些条目的适当的结构客体类和名(称)格式来完成。

13.4 属性类型定义

属性类型的定义包括如下内容：

a) 可选地指示该属性类型是某个之前定义的属性类型它的直接上级类型的子类型；

b) 规定该属性类型的属性句法；

c) 可选地指示该属性类型的相等、排序和/或子串匹配规则；

d) 指示该类型的属性是否应当仅具有一个值，或是可能具有多个值；

e) 指示该属性类型是操作属性还是用户属性；

f) 可选地指示一个用户属性类型是集合属性；

g) 可选地指示一个操作属性是不能被用户修改的；

h) 对于操作属性，指示其应用；

i) 为该属性类型分配一个客体标识符。

任何一个用户属性都可以被管理机构标识为一个具有友人属性的锚(anchor)属性。因此，属性类型的定义不需要指定一个锚属性的友人。这个可能会根据子模式的不同而不同。

13.4.1 操作属性

一些操作属性是由用户直接控制的。而在另外的情况下，操作属性的值是由目录控制的。在后一种情况下，操作属性的定义中应当指示不允许用户对属性值进行修改。

一个操作属性类型的规范须指示它的应用，须是如下所列的一种：

——目录操作属性(如访问控制属性)；

——DSA 共享的操作属性(如一个主访问点属性)；

——DSA 特定的操作属性(如一个拷贝状态属性)。

13.4.2 属性的层次结构

一个属性的层次结构须或者包括用户属性，或者包括操作属性，但不得两种都包括。因为，一个用户属性不得从操作属性派生而来，而一个操作属性也不得从用户属性派生而来。

作为另一个操作属性的子类型的操作属性应当具有与其上级类型相同的应用。

如果一个属性类型不是另一个属性类型的子类型，则在属性类型的定义中须规定属性句法和匹配规则(如果应用的话)。对属性句法的规定可以通过直接定义 ASN.1 数据类型而完成。

如果一个属性类型是某个指定类型的子类型，且在属性类型定义中没有规定属性句法时，在这种情况下，它的属性句法与其直接上级类型的属性句法相同。如果规定了属性句法，且该属性有一个直接上级类型，则所规定的句法须与上级类型的句法兼容，即每个符合该属性句法的可能取值也须符合上级类

型的句法。

如果一个属性类型是另一个属性类型的子类型，则应用于上级类型的匹配规则也可应用于子类型，除非在子类型的定义中对其进行了扩展或修改。在定义一个子类型时，其上级类型定义的匹配规则可能不会被删除。

13.4.3 友人属性

一个锚属性的友人列表须只能包含用户属性。这种关系对友人属性的语义、句法或其他特性都不会加以限制。

注：一个锚属性可能被定义为一个哑属性。

13.4.4 集合属性

一个操作属性不得被定义为是集合属性。

一个用户属性可能被定义为是集合属性。这就表示同一个属性值可能会出现在一个条目集合中的多个条目中，并符合属性collectiveExclusions 的规定。

集合属性须是多值的。

13.4.5 派生属性

派生属性所包含的信息符合属性信息的句法，但是其值是返回后计算出的，而不是 DIB 中直接存储的。

引入派生属性family-information 用于目录服务中以便容纳家族信息。它的特性在 GB/T 16264.3—2008 的 7.7.1 定义。

DSA 也可能会使用派生属性技术来提供其他属性。例如，所有包含某个特定 DSA 的AccessPoint 值的操作属性，可能(或应当)会从一个单独的信息源中派生一个适合管理的值。

13.4.6 属性句法

如果为某个属性类型规定了一个相等匹配规则，则目录须确保为该属性类型的每个取值都使用正确的属性句法。

13.4.7 匹配规则

在属性类型的定义中，可能会指定相等、排序和子串匹配规则。同一个匹配规则可以用于一个或多个这种类型的匹配，如果规则的语义允许这些多种不同类型的匹配的话。

注 1：在相应的匹配规则的定义中，应当反映出这个事实。

如果没有指定相等匹配规则，则目录将：

a) 认为该属性的值具有类型ANY，即目录不可能检查这些值是否与为属性所指示的数据类型或其他规则相符合；

b) 不允许该属性用于命名；

c) 不允许增加或删除多值属性的单个值；

d) 不能对属性值进行比较；

e) 不能使用该属性类型的值来评估AVA 。

如果指定了一个相等匹配规则，则目录将：

a) 认为该属性的值具有属性定义中(或该属性所派生的属性的定义中)的&Type 字段所定义的类型；

b) 在与属性相关的属性值断言的评估中，将使用指定的相等匹配规则；

c) 仅与一个当前存在的，具有适当数据类型(在属性类型定义中指定)的值进行匹配。

注 2：对于这样的属性，其相等匹配规则所使用的声明句法不同于属性类型的句法，本条也可以相同地应用于这样的属性。

如果没有指定排序匹配规则，则目录将认为任何一个使用目录抽象服务所提供的句法构造的排序匹配声明是未定义的。

如果没有指定子串匹配规则，则目录将认为任何一个使用目录抽象服务所提供的句法构造的子串匹配声明是未定义的。

一个属性类型应当仅规定应用于该属性的属性句法的匹配规则。

13.4.8 属性定义

属性可被定义为ATTRIBUTE 信息客体类的值：

```
ATTRIBUTE  ::=CLASS  {
    &derivation                     ATTRIBUTE OPTIONAL,
    &Type                           OPTIONAL,--或者是 &Type，或者是所需的 &derivation--
    &equality-match                 MATCHING-RULE OPTIONAL,
    &ordering-match                 MATCHING-RULE OPTIONAL,
    &substrings-match               MATCHING-RULE  OPTIONAL,
    &single-valued                  BOOLEAN DEFAULT FALSE,
    &collective                     BOOLEAN DEFAULT FALSE,
    &dummy                          BOOLEAN DEFAULT FALSE,
    --操作扩展--
    &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 ]
    [ DUMMY                         &dummy ]
    [ NO USER MODIFICATION          &no-user-modification  ]
    [ USAGE                         &usage ]
    ID                              &id }

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

对于每个使用此信息客体类定义的属性：

a) &derivation 是一个属性，如果存在的话，被定义的属性是其子类型；

b) &Type 是属性句法。应当是一个 ASN.1 数据类型，但不得包含 EmbeddedPDV 的类型；

c) &equality-match 是它的相等匹配规则(如果存在的话)；

d) &ordering-match 是它的排序匹配规则(如果存在的话)；

e) &substrings-match 是它的子串匹配规则(如果存在的话)；

f) &single-valued 将取值为TRUE，如果该属性是单值的，否则取值为FALSE；

g) &collective 将取值为TRUE，如果该属性是一个集合属性的话，否则取值为FALSE；

h) &dummy 将取值为TRUE，如果该属性是一个哑属性的话，否则取值为FALSE；

i) &no-user-modification 将取值为TRUE，如果该属性是一个不能被用户修改的操作属性的话；

j) &usage 指示了属性的操作用法。userApplications 表示该属性是一个用户属性，directoryOperation、distributedOperation 和dSAOperation 分别表示该属性是一个目录操作属性、分布式操作属性和 DSA 操作属性；

k) &id 是分配给该属性的客体标识符。

由目录已知并出于其自身目的而使用的在本目录规范的第 1 版中所定义的属性类型，如下所述：

```
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 }
```

注：在这些定义中涉及的匹配规则本身在 13.5.2 定义。

objectClass 属性和aliasedEntryName 属性被定义为用户属性，即使它们用于目录操作，而且在语义上也应当被定义为操作属性。这是因为在操作属性概念提出之前，这些客体已经被定义成为用户属性，因此为了方便实现了本目录规范不同版本的系统之间的互通，必须维持它们为用户属性。

13.5 匹配规则定义

13.5.1 概述

匹配规则的定义包括：

a) 可选地定义派生当前匹配规则的双亲匹配规则；

b) 定义匹配规则断言的句法；

c) 规定规则所支持的匹配的不同类型；

d) 定义适当的规则，以便为所提出的关于 DIB 内的目标属性值的声明进行评估赋值；

e) 为匹配规则分配一个客体标识符。

一个匹配规则应当用于为属性的属性值断言进行赋值，该属性指定此规则为它的相等匹配规则。在属性值断言(即属性值断言中的assertion 组件)中使用的句法是匹配规则的声明句法。

一个匹配规则可能会应用到具有不同属性句法的不同类型的属性中。

匹配规则的定义应当包括对匹配规则断言的句法规范以及如何用该句法的值来执行一个匹配的方法。这不要求对匹配规则可能应用的属性句法给出完整的规范。在定义为具有不同 ASN.1 句法的属性使用的匹配规则时，应当规定匹配是如何被执行的。

在一个子模式规范中所定义的匹配规则对于属性的适用性(超越了在这些属性类型的定义中所使用的匹配规则)是通过子模式规范操作属性matchingRuleUse 来指示的，该属性在 15.7.7 中定义。

13.5.2 匹配规则定义

匹配规则可被定义为 MATCHING-RULE 信息客体类的值：

```
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 }
```

对于每个使用此信息客体类定义的匹配规则：

a) &ParentMatchingRules 字段，当被定义的匹配规则与两个或多个其他匹配规则的特性相结合时使用该字段。该字段中给出两个或多个为被定义的匹配规则提供基本特性（如匹配算法）的匹配规则的客体标识符；对于一个基本的匹配规则而言，该字段被忽略。

b) &AssertionType 字段，定义了使用该匹配规则的声明的句法；如果该字段被忽略，则声明句法与应用规则的属性的句法相同，除非匹配规则有相反的规定。如果该字段存在，则可能规定对可能存在的双亲匹配规则的限制，但是在这种情况下，它必须与双亲匹配规则的句法兼容（即符合&AssertionType 的一个值应当也符合其双亲匹配规则的&AssertionType）。

c) &uniqueMatchIndicator 字段，是一个通知属性类型。若该字段存在，则表示要求唯一的匹配。对于一个基于映射的匹配规则（见 13.6），则意味着根据映射表进行的映射应当产生一个无二义性的结果。如果有多个匹配都符合映射表，则搜索请求将被拒绝，并在参数serviceError 中给出原因值为ambiguousKeyAttributes 。另外，本字段所规定类型的一个通知属性将在返回的错误中置于CommonResults 字段中。

注 1：在进行地理匹配时可能发生上述情况，例如，一个声明中指定"*Newton* "是英国的一个地名；但有很多不同的城镇具有该名（称），因此必须通过一个限定符来进行区分（如"Newton,Cambs "）。

d) &id 是为该匹配规则分配的客体标识符。

如果两个或多个匹配规则都用于ParentMatchingRules ，则结果是一个组合的匹配规则，并作为一个结果返回，其值符合AssertionType ，并使用如下规定的规则：

a) 如何任何一个双亲匹配规则的结果为TRUE ，则组合匹配规则须返回TRUE ；

b) 否则，如果任何一个双亲匹配规则的结果为FALSE ，则组合匹配规则须返回FALSE ；或者

c) 否则，组合匹配规则须返回UNDIFINED 。

下表显示了两个匹配规则 A 和 B 的组合规则；原则上，该表可以扩展到多维以覆盖 3 个或更多的双亲匹配规则，结果将具有类似的形式：

		规则A		
		TRUE	FALSE	UNDEFINED
规则B	TRUE	TRUE	TRUE	TRUE
	FALSE	TRUE	FALSE	FALSE
	UNDEFINED	TRUE	FALSE	UNDEFINED

通过上述规定的方法进行规则的组合，则可能获得合法的匹配，否则匹配将失败。

注 2：使用双亲匹配规则的一个特定用例是：将任意一个匹配规则与一个特定匹配规则。

ignoreIfAbsentMatch 组合起来。后一个匹配规则规定如果属性不存在的话，则过滤项将返回 TRUE；如果属性出现，则应用正常的匹配规则。这就规定了若搜索过滤器中规定的属性如果不存在的话，一个搜索过滤器如何来检查条目。见 GB/T 16264.6—2008 的 7.7.1。

匹配规则objectIdentifierMatch 定义如下：

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

某个类型为客体标识符的当前值与一个类型为客体标识符的目标值进行比较时，当且仅当两个客体标识符具有相同数量的整数组件，且第一个标识符的每个整数组件与第二个标识符的相应组件都相等的情况下，这两个客体标识符才被认为是匹配的。这个匹配规则是客体标识符的 ASN.1 类型定义所固有的。

objectIdentifierMatch 是一个相等匹配规则。

匹配规则distinguishedNameMatch 定义如下：

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

某个当前可辨别名值与一个目标可辨别名值进行比较时，当且仅当下述所有情况都为真时，这两个可辨别名值才被认为是匹配的：

a) 每个可辨别名值的 RDN 个数相同；

b) 每对相应的 RDN 都具有相同数量的 AttributeTypeAndValue；

c) 每对相应的 AttributeTypeAndValue(即它们都在每个相应的 RDN 内，且具有相同的属性类型)都具有与 9.4 的描述相匹配的属性值。

distinguishedNameMatch 是一个相等匹配规则。

13.6 放宽和收紧

*放宽*和*收紧*是两种功能，通过使用系统的方法来修改对一个或多个过滤项的匹配。如果执行了放宽功能，则以提高相似度的方式对匹配进行修改，使其得到更多的匹配条目。当匹配的条目低于某个最低值时，则执行放宽功能。当匹配的条目高于某个最高值时，则以类似的方法执行收紧功能。有两种方式的放宽和收紧：

a) 为某个特定属性类型所应用的匹配规则可逐步被替代匹配规则所替换，直到获得所要求的效果，或者直到用完所有的可能性，如 13.6.1 的描述；和

b) 放松和收紧功能可作为 13.6.2 所描述的*基于映射的匹配*中的一部分来应用。

13.6.1 匹配规则的替换

匹配规则的替换可被一个特定服务管理区(见 16.10.7)内的某个控制搜索规则所控制。它也可以被搜索请求中的用户所控制(见 GB/T 16264.3—2008 的 10.2.1)。在这两种情况下，都使用在 16.10 中定义的结构RelaxationPolicy 来控制这种替换。

通过匹配规则的替换而实现的放宽和收紧，是通过系统地替换所选属性之前所应用的匹配规则来修改过滤器动作，以便提供更张弛的(或更紧的)匹配。通过匹配规则的替换进行了放宽或收紧，则整个的搜索过程将针对搜索范围内的同一个条目集进行重新评估。重新评估可一直继续进行，直到不再需要放宽，或者直到获得了满意的结果(通过引用控制的RelaxationPolicy 元素，满意的结果指的是低于或等于maximum 值，或者高于minimum 值)。

结果是每个被重新评估的过滤器是保持不变的，但是用于评估过滤器的各个匹配规则在必要时将被替换(如图 12 所示)。放宽的评估可能会在逐个 DSA 的基础上进行，每个 DSA 之间使用不同的放宽；或者可能使用ChainingArguments 中的组件chainedRelaxation 来定义将要使用的放宽。

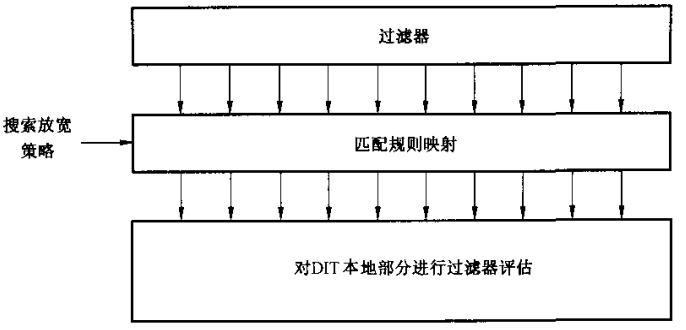


图 12 匹配规则的替换

当一个放宽策略将要被使用时，之前启动了一个本地搜索的 DSA 将为每个属性类型执行一个在放宽策略中指定的基本替换，基本替换是为这些属性定义的。

注 1：基本替换的一个具体的实用的应用是：例如，对于属性类型localityName，当匹配规则更适合，且用户希望正确地表示substrings 过滤项的时候，可以使用generalWordMatch 匹配规则来替换caseIgnoreSubstringMatch 匹配规则。

如果应用于某个特定 DSA 的搜索结果返回了太少的条目，则使用第一个放宽策略；如果返回的搜索结果还是太少，则使用下一个放宽策略；以此类推。

类似的，如果搜索结果返回了太多的条目，则以相似的方式使用第一个收紧策略。不能从收紧再逆转到放宽，或从放宽逆转到收紧。

使用一个MRSubstitution 的集合应用于某个特定属性的放宽，将一直应用直到被另一个MRMapping 所撤销。可以通过指定一个匹配规则来显式地撤销，或者通过忽略oldMatchingRule 标识符来隐含地撤销。

如果根据之前的评估返回了太少的结果而执行了一个放宽评估，但根据该放宽评估又返回了太多的结果，则这些根据放宽评估所得出的部分或全部结果须被返回。如果根据之前的评估返回了太多的结果而执行了一个收紧评估，但根据该收紧评估又返回了太少的结果，则须返回根据之前的评估所得出的部分或全部结果。在任何一种情况下，放宽和收紧过程都应当停止。

一个可应用的放宽策略在适当时候可应用于filter 或extendedFilter 。

注 2：因为放宽允许对普通过滤器的过滤项的评估进行放宽或收紧，因此，为了获得更复杂的过滤而使用扩展过滤器的必要性减少了。

一个 DSA 可能会在搜索结果中提供通知属性proposedRelaxation（见 GB/T 16264.6—2008 的 5.12.15），在PartialOutcomeQualifier 的notification 子组件中提供。因此，这里的信息在后续的搜索请求中可用做用户所提供的放宽策略。

作为一个放宽策略的极端用例，该策略能够根据nullMatch 匹配规则，使某个特定的过滤器被评估为TRUE（或FALSE，如果过滤项为否定的话）。

在一个特定服务管理区内，对搜索规则的合法性判断是在可能的基本替换完成后才执行的，正如同正在被评估的搜索请求所应用的搜索规则所指示的那样。一个控制搜索规则的选择，应在任何其他后续的匹配规则替换的选择之前，包括在搜索请求中指定的可能的基本替换。

13.6.2 基于映射的匹配

基于映射的匹配是与这样的搜索操作相关的，即在某些方式下，用户对现实世界的概念与目录常使用的理想化模型是不同的。例如，用户对位置名（称）以及这些位置之间关联关系的表示，可能与目录中对位置的表示是完全不同的。为了弥补这些差距以提高搜索的成功率，在用户对某些现实世界客体及其相互关系的概念和目录对同样客体的模型之间有必要进行映射。同样的映射也应当允许使用“模糊”匹配，即允许一些属性值反映比其精确的定义更多的内容。

注 1：例如，一个用户可能会在过滤器中指定一个位置名（称），但是所查找的客体可能包括附近边界的位置。

基于映射的匹配可应用于白页搜索中的地理位置方面、黄页搜索中的商务分类方面等。

基于映射的匹配为了控制映射，会使用一些被称为“映射表”的中间表。基于映射的匹配的确切行为以及映射表的具体结构属于本地事物。然而，该技术的基本原则是共同的，如图 13 所示。

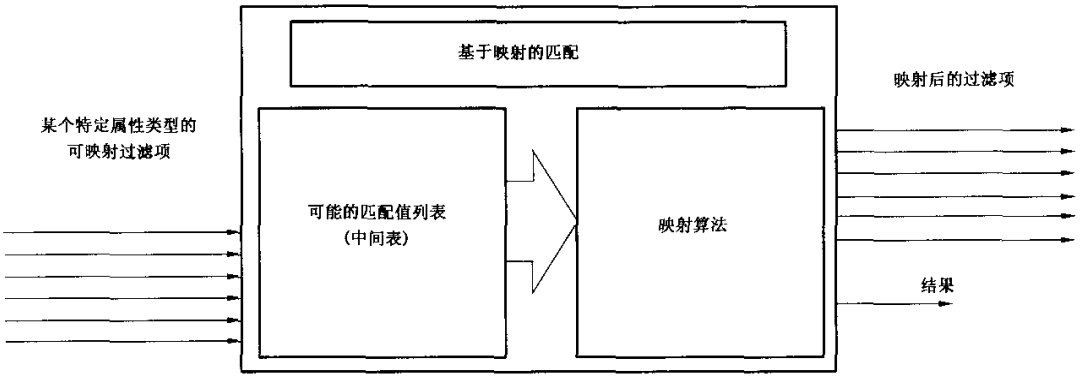


图 13 基于映射的匹配

使用该技术，某个指定属性类型的过滤项(*可映射的过滤项*)使用映射表和某种类型的映射算法完成一个映射过程。这样映射后，会形成一些被称为“*映射后的过滤项*”的新的过滤项，替代了可映射的过滤项。在异常情况下，将不执行映射，且异常的确切特性信息被返回。

映射后的过滤项的数量与可映射的过滤项的数量不一定相等，而且一般来说是不同的。

一个类型为extensibleMatch 的过滤项，但type 规范缺失，则该过滤项不能成为一个可映射的过滤项。

基于映射的映射可能属于 DSA 的本地事物。如果搜索评估是分布式的，而且在搜索的评估阶段有其他 DSA 的参与，则其他 DSA 可能会应用它们自己的基于映射的映射。然而，所使用的映射可以通过 ChainedArguments 中的chainedRelaxation 组件被传送到其他 DSA。

注 2：为了向用户提供一致的服务，在一个分布式搜索评估中可能参与的 DSA 的管理者应当考虑其映射表和映射功能是否是协调一致的。

图 14 举例说明了在现实世界和表示该世界的目录模型之间建立映射功能的原则。用户对现实世界有感知，这些感知可能不会考虑到现实世界的所有方面。而现实世界的一些方面对用户如何构造搜索请求的表达式是重要的，因此这些方面组成了表示现实世界的模型。这些模型构成了映射如何执行的基础。现实世界的精确模型一定是基于经验的，并且还要求基于所观察到的用户的搜索行为进行定期的修改。

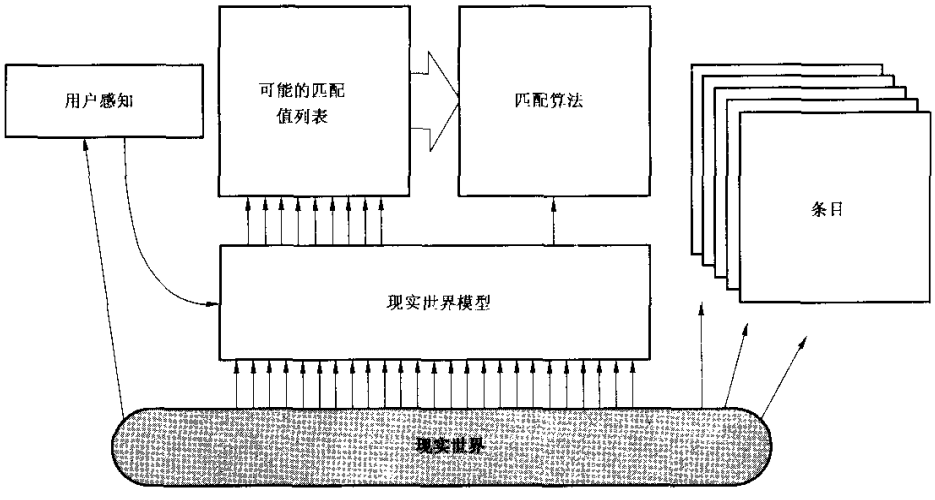


图 14 信息的推导

现实世界的模型可能仅包含用户在搜索请求中使用的属性类型的一个子集，并且可能仅有一个属

性类型是相关的。例如，当考虑一个与现实世界的位置相关的模型时，仅需相应考虑与位置相关的属性类型即可。不涉及这些属性类型的过滤项将不被映射，但是在进行条目匹配时，这些过滤项将保留且与映射后的过滤项共同使用。

现实世界的模型用来建立*可匹配值*的映射表，即可匹配值指的是潜在地将与可映射的过滤项进行匹配的值的集合。如何建立该可匹配值的映射表属于本地事物。与映射表的匹配可能会得到零个或多个匹配，每个匹配又得到一个或多个映射后的过滤项。映射算法决定了映射后的过滤项如何应用到条目中。但如何决定属于本地事物。可以基于条目中的传统属性值，或可以基于安置在条目中的但在目录外则无意义的属性值，如数字标识符。

进行映射的方法以及得到映射后的过滤项后应当如何处理等，可参见 16.5 定义的子过滤器以及附录 Q 中更详细的定义。子过滤器的概念用在这里，仅仅是当做一个描述工具。在实现时可以使用任何其他可获得同样结果的算法。

每个子过滤器都与映射表进行比较评估，所得到的映射后的过滤项将根据详细的映射算法所指定的方法与非映射的过滤项组合起来。所得到的匹配后的条目是与每个子过滤器都匹配的联合体。

注 3：在许多情况下，可映射的过滤项将被映射后的过滤项的逻辑或所替代。

原则上，有两种不同的映射方式。每个可映射过滤项在同一时间能够被映射为一个过滤项；或者多个组合的可映射过滤项可被用于满足一个单独的与映射表的匹配。多个过滤项可应用于一个单独的基于映射的匹配，当且仅当它们是组合的过滤项时才可以，也就是说，作为一个单元包含在一个单独的子过滤器内。

注 4：例如，在一个子过滤器中两个不同的地理名(称)的逻辑与(AND)，能够用来指定一个单独的具有适当范围的地理位置，而如果单独使用每个地理名(称)，可能会确定出一个具有二义性的或大范围的地理位置。

过滤项与映射表的匹配，是通过使用该过滤项所隐含定义的或明确指定的匹配规则来执行的，可能是在控制搜索规则(如果有的话)或者搜索请求中指定的基本匹配规则的替换之后才执行的。

注 5：这可以包含一个复杂的匹配规则，如在 GB/T 16264.6—2008 中定义的通用字匹配generalWordMatch，允许字的旋转、字的截断以及近似字匹配等。

注 6：本系列目录规范不指定在实现时如何将相关的匹配规则组合成为组合的匹配规则。所希望的是在实现时，对所支持的过滤项和匹配规则能够进行何种组合给出限制。

如果一个过滤项或组合过滤项与映射表的匹配没有得到任何子过滤器的匹配结果时，即匹配获得的结果为 FALSE 或未定义，则将得到零个映射后的过滤项。如果在每个子过滤器中有可映射的过滤项，则搜索将不会有结果。然后必须有一个错误信息返回给用户。

在某些情况下，如在地理区域匹配时，要求与映射表的匹配得到一个单独的无二义性的结果。如果一个子过滤器在映射表中与多个条目都匹配，或者不同的子过滤器在映射表中匹配了不同的条目，则搜索可能会返回太多的无用条目。一种替代方法是给用户返回信息，允许发起一个新的、更好定位的搜索请求。

注 7：有一种更简单的情况，即仅核查可映射的过滤项与映射表。如果这种匹配成功，则可映射的过滤项将不被改变。

映射可以是动态的，即若搜索找到的匹配条目为零个或太少时，则映射能够被调整(放宽)。有关放宽执行的细节不在本系列目录规范的定义范围之内，而是由本地需求所决定的。放宽的执行可以是逐步渐进的，潜在地可以发现越来越多的条目。放宽应当以这样的方式来完成，即每当新执行了一个放宽步骤时，之前步骤所返回的所有条目将与可能的新匹配的条目一起返回。

放宽的执行是逐步渐进的，这可以通过指定不同级别的放宽来实现的。级别为零表示不放宽。级别为 1 表示放宽的第一级别，等等，依此类推。图 15 对逐步放宽机制给出了一个抽象的示例。放宽的每个不同级别都确切意味着什么不在本系列目录规范的定义范围之内。放宽级别可以通过RelaxationPolicy 结构来控制，该结构可能在搜索规则中提供，或搜索请求中提供，或两者中都提供。这就允许两种放宽，一种是基于映射的映射放宽，另一种是通过匹配规则替代的放宽，两者可以彼此同步起来，因为

两种都能够通过RelaxationPolicy 中指定的每一放宽步骤所决定。

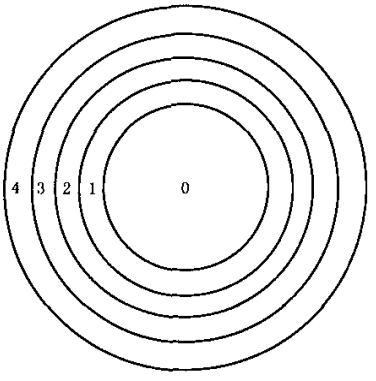


图 15 搜索放宽

extendedArea 搜索控制是一个整数，为基于映射的匹配算法的放宽级别的控制提供了一种替代方式。它是基于映射的映射算法的客户化的一部分，无论它是否能够被此搜索控制所控制。

如果extendedArea 搜索控制在一个搜索请求中出现，且它被允许用于基于映射的算法中，则在RelaxationPolicy 中的任何级别的规范，无论是否包含在搜索中或控制搜索规则中，都将被忽略。

当放宽方式被extendedArea 搜索控制所控制时，includeAllAreas 搜索控制选项规定了放宽的方式。如果设置了该选项，则放宽的执行如前所述，即如果放宽级别越高，则可能会有更多的条目返回(*包含放宽*)。如果没有设置该选项，则用户仅对相应于增量放宽的结果感兴趣(*排斥放宽*)。如果用户在逐步地执行放宽，且对之前返回结果中的条目不感兴趣，而仅对最后一步放宽的结果感兴趣时，后一种情况将比较适合。

注 8：并不能保证(特别是具有一个复杂的过滤器)用户不会得到之前已经检索到的某些条目，也不能保证那些感兴趣的所有条目都会返回。例如，在 Winkfield 寻找一家法国餐厅会失败；将其放宽到寻找 Winkfield 区内但不包括 Winkfield 的所有餐厅，则在搜索结果中会找到一家位于 Winkfield 的具有混合风味的 White Hart Inn 餐厅。

某些基于映射的匹配算法可能不支持排斥放宽，或者可能被客户化为不允许进行排斥放宽。在这种情况下，搜索控制选项includeAllAreas 对该映射功能来说，应当被忽略，且可能的放宽应当作为包含放宽被执行。

在某些环境中，可能相对地能够指定放宽级别为负数，相当于一个收紧的匹配。在这种情况下，搜索控制选项includeAllAreas 没有意义，如果存在则被忽略。对于所有类型的基于映射的匹配而言，收紧可能不是相对的。

一个 DSA 可能会同时支持多个映射功能，即拥有多个映射表以及其相应的映射算法。具有多个映射功能的理由如下：

a) 要执行的映射功能取决于应用类型。地理区域匹配(见 GB/T 16264.6—2008 的 7.8)是基于映射的匹配的一种特定的重要应用。其他的示例还有黄页搜索、书籍目录搜索中的基于映射的匹配等；

b) 在一个特定的应用中，如何执行映射的详细定义可能会根据具体条件而变化。例如，地理区域匹配的映射可能取决于地理地区(如体现在搜索的baseObject 中)，或者取决于用户所尝试的搜索类型，即基于搜索过滤器中的信息。另外一个例子是，映射可能取决于请求所使用的语言。

如果同步应用了多个映射功能，且其中一个功能的执行引起了一个异常条件，则需要向用户报告，这时不要求实现去检查是否存在多个异常(但有可能是这样的)。

一个基于映射的映射规范(见以下)决定了extendedArea 搜索控制是否可应用于一个正在讨论的

映射功能。如果有多个映射功能对于同一个搜索操作来说都是激活的，且其中的一些能够被extendedArea搜索控制所控制，则它们都将根据extendedArea搜索控制，并且如果可用的话，还将根据includeAllAreas搜索控制选项，来同步地执行放宽或收紧。

注9：之前给出的示例显示了多个基于映射的映射使用includeAllAreas时，将增加复杂度。

如果extendedArea搜索控制指定了一个放宽或收紧级别，但对于某些受此搜索控制影响的映射功能而言，DSA不支持这些级别，则DSA应当尽最大努力执行映射。如果extendedArea搜索控制指定了一个放宽或收紧级别，但对于所有受此搜索控制影响的映射功能而言，DSA都不支持这些级别，则应当在CommonResults的notification参数中，返回searchServiceProblem通知属性，取值为id-pr-unavailableRelaxationLevel。

注10：如果搜索操作的评估是分布在多个DSA中进行的，则这些DSA可能部署不同的映射功能给出不一致的结果，除非在DSA之间建立了某种协调。

尽管基于映射的匹配的细节定义属于本地事物，但还是可能定义基于映射的匹配的全局特性，即定义一个特定类型的匹配规则，它被称为"*基于映射的匹配规则*"。该匹配规则作为MATCHING-RULE信息客体类的一个实例而定义。然而，它与传统的匹配规则不同之处在于它不以常规方式指定匹配，因此不指定匹配的句法。然而，作为它定义的一部分，它给出了关于它的目的、它如何被应用以及如何处理异常条件等的规范，一个基于映射的匹配规则的特定特性可以通过一个从具有较低通用性(即已经参数化)的MAPPING-BASED-MATCHING信息客体类派生而来的ASN.1信息客体类的实例部分地描述出来。该信息客体类仅用来指定可能会客户化的方面。本目录规范不指定该信息客体类的实例如何存储以及在何处存储等，仅是通过某种方式，使其对DSA可用。

```
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.&id  (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 }
```

MAPPING-BASED-MATCHING 信息客体类的字段规定如下：

a) &selectBy 字段是对规范的一个虚拟引用，该规范定义了如何选择信息客体类的一个具体实例来进行基于映射的映射。如果可应用的话，信息客体类的具体实例将通过一个 ASN.1 数据类型以及相应的文字描述来规定如何执行此选择。如果用户在搜索请求的 RelaxationPolicy 结构中提供了一个非空的 mapping 组件，则该字段应当被忽略；

注 11：原则上，同一个搜索请求能够选择不同派生信息客体类的多个可能的实例。

b) &ApplicableTo 字段规定了哪些过滤项应当被认为是可映射的过滤项，通过指定该过滤项的属性类型来完成。该字段所列出的属性类型对应的任何过滤项都应当可以进行基于映射的匹配。该字段必须存在。但该字段所列出的属性类型可能没有必要都出现在过滤器中。该值由该信息客体类的特定化信息客体实例决定。

c) &subtypesIncluded 字段值为布尔型，规定了一个派生的信息客体类的实例除了所指定的属性外，是否还能够接受&ApplicableTo 属性的子类型。如果该字段缺失，则表示允许接受子类型，如果他们没有被其他机制关闭的话。该值由派生的信息客体类的信息客体实例决定。

d) &combinable 字段值为布尔型，如果取值为 TRUE，则表示允许基于映射的匹配使用多个组合的过滤项来满足与映射表的匹配。combinable 是该字段值的一个虚拟引用，由该信息客体类的某个具体实例所决定。

e) &mappingResults 字段是对规范的一个虚拟引用，该规范规定了异常条件如何被上报。派生的信息客体类应定义一个 ASN.1 数据类型来上报相应的异常条件。

f) &userControl 字段值为布尔型，规定了派生的信息客体类的一个实例及其相关的基于映射的匹配规则是否能够被extendedArea 搜索控制所控制。

注 12：如果有多个基于映射的匹配都同时应用的话，可能这样是适当的，即仅让其中的一个允许使用extendedArea 搜索控制。

g) &exclusive 字段值为布尔型，规定了派生的信息客体类的一个实例及其相关的基于映射的匹配规则是否允许执行"排斥放宽"。该值如果存在的话，由派生的信息客体类的信息客体实例所决定。如果该值为 FALSE 或者如果 DSA 不对此基于映射的匹配支持排斥匹配，则该特定映射的行为将如同设置了includeAllAreas 搜索控制选项一样。

注 13：如果有多个基于映射的匹配都同时应用的话，可能这样是适当的，即仅让其中的一个允许排斥放宽。

h) &matching-rule 字段值的类型为客体标识符，标识了一个基于映射的匹配规则，该实例对此匹配规则提供了附加的规范，并且应当应用于基于映射的匹配。该字段值的虚拟引用matchingRule 由该信息客体类的某个具体实例所决定。所指定的匹配规则应当用于特定的基于映射的匹配。

i) &id 字段是为特定的基于映射的映射分配的客体标识符。

13.7 DIT 结构定义

13.7.1 概述

目录模式的一个基本方面是规范某个特定类的条目可能置于 DIT 的何处以及它是如何被命名的。考虑如下：

- DIT 内条目的层次关系(DIT 结构规则)；
- 用来构造条目 RDN 的一个或多个属性(名(称)格式)。

13.7.2 名(称)格式定义

名(称)格式的定义包括：

a) 指定要命名的客体类；

b) 指示此名(称)格式所应用的客体类条目的 RDN 中所用的必选属性；

c) 指示此名(称)格式所应用的客体类条目的 RDN 中可能使用的可选属性，如果有的话；

d) 为名(称)格式分配一个客体标识符。

如果对于某个给定的结构客体类的条目,需要有不同的命名属性集,则应当为每个用于命名的不同的属性集定义一个名(称)格式。

仅有结构化客体类可用于名(称)格式中。

对于存在于 DIB 某部分中的某个特定结构客体类的条目,至少应当有该客体类的一个名(称)格式包含在模式的可应用部分中。模式包含了所需要的附加名(称)格式。

RDN 属性(或 RDN 属性集)没有必要从结构客体类的结构定义或别名客体类定义中指定的允许属性列表中选择。

注:命名属性由 DIT 内容规则来管理,且 DIT 上下文以与其他属性一样的方式使用该属性。

名(称)格式仅仅是构造 DIT 格式所要求的全部规范中的一个基本元素,DIT 格式的需求由管理和命名机构提出,该机构可以决定某个给定的 DIT 领域的命名策略。DIT 结构的其余方面的规范在 13.7.5 讨论。

13.7.3 名(称)格式规范

名(称)格式可能被定义为NAME-FORM 信息客体类的值:

```
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 }
```

对于每个使用此信息客体类定义的名(称)格式:

a) &namedObjectClass 字段标识了它所命名的结构客体类;

b) &MandatoryAttributes 字段指示此名(称)格式所控制的 RDN 中必须出现的属性集;

c) &OptionalAttributes 字段指示此名(称)格式所控制的 RDN 中可能出现的属性集;

d) &id 是为名(称)格式所分配的客体标识符。

处于必选和可选属性列表中的所有属性类型都必须是不同的。

13.7.4 条目的结构客体类

某些子模式规范中将包括这样一些名(称)格式,这些名(称)格式是为在子模式中表示的结构客体类的每个上级类链的最多一个结构客体类所定义的。

某些子模式规范中将包括这样一些名(称)格式,这些名(称)格式是为在子模式中表示的结构客体类的每个上级类链的多个结构客体类所定义的。

在任何一种情况下,关于某个特定的条目,只有出现在条目的属性objectClass 中的结构上级类链中的最下级结构客体类,才能决定应用于该条目的 DIT 内容规则和 DIT 结构规则。该类是条目的结构客体类,且由操作属性*structuralObjectClass* 指示。

13.7.5 DIT 结构规则定义

DIT 结构规则是一个规范,由子模式管理机构提供,目录使用该规范在子模式范围内控制条目的放置位置和命名。每个客体和别名条目都被一个单独的 DIT 结构规则所控制。控制某个 DIT 子树的子模式将典型地包含多个 DIT 结构规则,以便允许在子树内有多种类型的条目存在。

一个 DIT 结构规则的定义包括：

a) 一个整数标识符，在子模式范围内唯一；

b) 指示由 DIT 结构规则所控制的条目的名(称)格式；

c) 所允许的上级结构规则的集合，如果需要的话。

子模式的 DIT 结构规则的集合规定了被子模式控制的条目的可辨别名格式。

DIT 结构规则允许一个给定子模式内的条目定购某个特定的名(称)格式。条目属性DistinguishedName 中的"最后的 RDN(lastRDN)"组件的格式由控制条目的 DIT 结构规则中的名(称)格式所决定。

名(称)格式中的namedObjectClass 组件(名(称)格式的客体类)与条目的结构客体类应一致。

DIT 结构规则应当仅允许属于结构客体类的条目才能够被相关的名(称)格式所标识，而不允许属于结构客体类的任何子类的条目被该名(称)格式所标识。

对于某个特定的条目，控制条目的 DIT 结构规则被称为条目的"控制结构规则"。该规则可以通过检查条目的governingStructureRule 属性来进行确定。

对于某个特定的条目，控制条目上级的 DIT 结构规则被称为条目的"*上级结构规则*"。

一个条目可能仅仅作为另一个条目(上级)的下级而存在于 DIT 中，如果一个 DIT 结构规则存在于控制子模式中，且：

——为条目的结构客体类指示了名(称)格式；且

——或者将包含条目的上级结构规则作为一个可能的上级结构规则，或者不指定上级结构规则，在这种情况下，条目应当是一个子模式的管理点。

如果条目本身是一个子模式的管理点，但出于子模式管理的目的它没有被包含在它的子模式子条目中，则将使用其直接上级子模式管理区中的子模式来控制该条目。

作为管理点但没有子模式子条目的条目(如新创建的管理点条目)将没有控制结构规则。目录不得允许在这些条目的下面创建下级，直到增加了子模式子条目为止。

如果一个条目被转换为一个新的子模式管理点，则在新的子模式管理区内所有条目的控制结构规则将被自动转变为新的子模式隐含的控制结构规则。

13.7.6 DIT 结构规则规范

DIT 结构规则的抽象语法由下述 ASN.1 类型表示：

```
DITStructureRule  ::=SEQUENCE {
    ruleIdentifier              RuleIdentifier,
                                    --在子模式范围内应当唯一
    nameForm                    NAME-FORM.&id,
    superiorStructureRules      SET SIZE (1..MAX) OF RuleIdentifier OPTIONAL }

RuleIdentifier  ::=INTEGER
```

13.7.5 所列的定义的不同部分，与如上所定义的 ASN.1 类型的各组件之间的对应关系，如下所述：

a) ruleIdentifier 组件在子模式范围内唯一地标识了该 DIT 结构规则；

b) DIT 结构规则的*nameForm* 组件规定了该 DIT 结构规则所控制的条目的名(称)格式；

c) superiorStructureRules 组件标识了该规则所控制的条目所允许的上级结构规则。如果该组件被省略，则表示该 DIT 结构规则应用于一个子模式管理点。

提供了STRUCTURE-RULE 信息客体类来简化 DIT 结构规则的文档。

```
STRUCTURE-RULE  ::=CLASS {
    &nameForm                   NAME-FORM,
```

```
    &superiorStructureRules     STRUCTURE-RULE   OPTIONAL,
    &id                         RuleIdentifier }

WITH SYNTAX {
    NAME FORM                   &nameForm
    [ SUPERIOR RULES            &SuperiorStructureRules  ]
    ID                          &id }
```

13.8 DIT 内容规则定义

13.8.1 概述

DIT 内容规则规定了某个特定的结构客体类的条目所允许具有的内容，这是通过指定一个辅助客体类、必选属性、可选属性以及排斥属性等的集合来完成的。如果条目中允许有集合属性，则它们须被包含在 DIT 内容规则中。

DIT 内容规则的定义包括：

a) 指示本 DIT 内容规则所应用的结构客体类；

b) 可选地，指示本 DIT 内容规则所控制的条目允许的辅助客体类；

c) 可选地，指示本 DIT 内容规则所控制的条目需要的必选属性，但结构客体类和辅助客体类已经要求的除外；

d) 可选地，指示本 DIT 内容规则所控制的条目允许的可选属性，但结构客体类和辅助客体类已经要求的除外；

e) 可选地，从条目的结构客体类和辅助客体类的可选属性中，指示出不能出现在本 DIT 内容规则所控制的条目中的属性。

对于任何合法的子模式规范，对每个结构客体类最多定义一个 DIT 内容规则。

DIT 中的每个条目最多被一个 DIT 内容规则所控制。该规则可通过条目的structuralObjectClass 属性值来标识。

如果一个结构客体类没有 DIT 内容规则，则该类的条目必须仅包含结构客体类定义中所允许的属性。

条目的结构客体类上级类的 DIT 内容规则，不能应用于本条目。

由于 DIT 内容规则与某个结构客体类相关联，因此同一结构客体类的所有条目都将具有同一个 DIT 内容规则，而不论控制它们在 DIT 内所处位置的 DIT 结构规则。

被 DIT 内容规则所控制的条目，除了与 DIT 结构规则的结构客体类相关外，还可能与 DIT 内容规则所标识的辅助客体类的某个子集相关。这种相关关系体现在条目的objectClass 属性中。

条目的内容应当与它的objectClass 属性所指示的客体类保持一致，在如下方面：

——objectClass 属性所指示的客体类的必选属性必须总是出现在条目中；

——由 DIT 内容规则所指示的辅助客体类的可选属性（不是 DIT 内容规则附加的可选或必选属性）只有在objectClass 属性指示了这些辅助客体类的情况下，才可能存在于条目中。

与结构客体类或所指示的辅助客体类相关的必选属性不得被 DIT 内容规则所排除。

13.8.2 DIT 内容规则规范

DIT 内容规则的抽象语法由下述 ASN.1 类型表示：

```
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 }
```

13.8.1 中所列的定义部分，与如上所定义的 ASN.1 类型的各组件之间的对应关系，如下所述：

a) structuralObjectClass 组件标识了本 DIT 内容规则所应用的结构客体类；

b) auxiliaries 组件标识了本 DIT 内容规则所应用的条目允许的辅助客体类；

c) mandatory 组件指定了本 DIT 内容规则所应用的条目除了根据其结构客体类和辅助客体类必须包含的属性类型外，还必须包含的用户属性类型；

d) optional 组件指定了本 DIT 内容规则所应用的条目除了根据其结构客体类和辅助客体类可能包含的属性类型外，还可能包含的用户属性类型；

e) precluded 组件指定了结构客体类和辅助客体类中的可选用户属性类型的一个子集，该子集中的属性类型被本 DIT 内容规则所应用的条目所排除。

注：直接标识属性(如在必选属性、可选属性和排除属性列表中所列的属性)的内容规则仅应用于所指定的属性，而不能应用于子类型和友人属性。

提供了CONTENT-RULE 信息客体类来简化 DIT 内容规则的文档：

```
CONTENT-RULE  ::=CLASS {
    &structuralClass        OBJECT-CLASS.&id  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 上下文类型定义

上下文类型的定义包括：

a) 规定上下文的句法；

b) 规定上下文断言的句法；

c) 可选地，规定上下文的缺省值；

d) 定义上下文的语义；

e) 规定如何执行匹配；

f) 规定上下文值如果缺失时的行为；以及

g) 为上下文类型分配一个客体标识符。

13.9.1 上下文值匹配

当前的上下文断言与已存储的同一上下文类型的某个上下文值相匹配，是根据上下文定义中匹配部分的描述而判断的。

13.9.2 上下文定义

上下文的定义使用CONTEXT 信息客体类：

```
CONTEXT  ::=CLASS  {
    &Type,
    &DefaultValue     OPTIONAL,
```

```
    &Assertion          OPTIONAL,
    &absentMatch        BOOLEAN  DEFAULT  TRUE,
    &id                 OBJECT IDENTIFIER UNIQUE }

WITH SYNTAX {
     WITH  SYNTAX       &Type
    [ DEFAULT-VALUE     &DefaultValue ]
    [ ASSERTED  AS      &Assertion ]
    [ ABSENT-MATCH      &absentMatch ]
    ID                  &id     }
```

DEFAULT-VALUE 将抵消ABSENT-MATCH 的效果(1),而对定义了DEFAULT-VALUE 的任意上下文都可以假定ABSENT-MATCH 的效果(2),在这种情况下,ABSENT-MATCH 字段能够被省略。

如果&defaultValue 被指定,则要求增加值及其上下文的条目修改请求,应当符合下述预处理和后处理规范所定义的行为。

注:DSA 没有义务按照下述的步骤精确地实现,只要最终的结果表现出同样的外部观察行为即可。

预处理

对于每个要求增加值及其上下文,或者删除值及其上下文,或者删除所有值及其上下文的Entry-Modification 请求。对于每个应用于该属性类型的上下文类型,如果上下文类型定义时带有&defaultValue ,则:

1) 如果上下文类型没有显式地列在请求中,则在请求中增加&defaultValue 中定义的上下文类型;

2) 对于属性类型的每个已存储的属性值,如果属性值没有包含上下文类型,则在属性值中增加&defaultValue 中定义的上下文类型。

正常处理

后处理

对于每个要求增加值及其上下文,或者删除值及其上下文,或者删除所有值及其上下文的Entry-Modification 请求。对于每个应用于该属性类型的上下文类型,如果上下文类型定义时带有&defaultValue ,则对于该属性类型的每个已存储的属性值:

3) 如果属性值没有上下文类型,则删除该属性值;

4) 如果属性值具有上下文类型,且该上下文类型仅有的上下文值为&defaultValue ,则删除该上下文(但不删除属性值)。

如果&Assertion 被省略,则上下文断言的句法与&Type 相同。

在上下文定义中,如果指定&absentMatch 值为 FALSE,则有如下两种效果:

a) 一个属性值,如果没有具备所指定的上下文类型的上下文,则它被认为不具备该上下文类型的值。也就是说,如果一个属性值没有包含所声明的contextType 的上下文,则ContextAssertion 将被评估为 FALSE;

b) 该上下文类型的上下文值中的fallback 组件被认为设置为 FALSE,而不管它的实际设定值。

在定义一个上下文时,在规范中应当包括对上下文语义的描述以及如何对匹配进行评估赋值的描述。

GB/T 16264.6—2008 规定了所选择的上下文定义。

13.10 DIT 上下文用法定义

13.10.1 概述

DIT 上下文用法是一个规范，由子模式管理机构提供，该规范规定了可能与属性一起存储的允许的上下文类型以及应当与属性一起存储的必选的上下文类型。

DIT 上下文用法的定义包括：

a) 指示该用法所应用的属性类型；

b) 可选地，指示任何时候当属性被存储时，应当与该属性类型的值相关的必选上下文类型；

c) 可选地，指示任何时候当属性被存储时，可能与该属性类型的值相关的可选上下文类型。

如果对于某个给定的属性类型，没有定义 DIT 上下文用法，则该属性类型的值不得包含上下文列表。

对于给定的子模式管理区，对于某个给定的属性类型，可以仅有一个 DIT 上下文用法。DIT 上下文用法可能会定义为应用到所有的属性类型，在这种情况下，它应当是子模式中的唯一的 DIT 上下文用法。

13.10.2 DIT 上下文用法规范

DIT 上下文用法的抽象语法由下述 ASN.1 类型表示：

```
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 }
```

13.10.1 所列定义的各部分与如上所定义的 ASN.1 类型的各组件之间的对应关系，如下所述：

a) attributeType 组件标识了本 DIT 上下文用法所应用的属性类型，或者任意属性类型(取值为 id-oa-allAttributeTypes)；

b) mandatoryContexts 组件规定了任何时候当属性被存储时都必须与给定属性类型的属性值相关的上下文类型。如果该组件省略，则属性值可能没有上下文列表；

c) optionalContexts 组件规定了任何时候当属性被存储时，都可能与给定属性类型的属性值相关的上下文类型。如果该组件省略，但mandatoryContexts 组件存在，则所有的属性值出现时都应当具有必选的上下文类型，此外没有其他。如果该组件省略，且mandatoryContexts 组件也省略，则情况与该属性类型没有 DIT 上下文用法时相同，也就是说，给定属性类型的属性值不得具有相关的上下文列表。

提供了DIT-CONTEXT-USE-RULE 信息客体类来简化 DIT 上下文用法规则的文档。

```
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 ] }
```

13.11 友人定义

友人集的定义包括：

a) 规定具有该友人集的锚属性；

b) 规定该锚属性的友人属性集。

提供了FRIENDS信息客体类来简化友人集的文档：

```
FRIENDS  ::=CLASS {
        &anchor                 ATTRIBUTE. &id UNIQUE,
        &Friends                ATTRIBUTE }

WITH SYNTAX {
        ANCHOR                  &anchor
        FRIENDS                 &Friends }
```

任何给定的属性在任何一个子模式中，都只能有一个友人集。

示例：

```
postal FRIENDS ::={
        ANCHOR                  {postalAddress}
        FRIENDS     { physicalDeliveryOfficeName |
                postalCode |
                postOfficeBox |
                streetAddress }
```

14 目录系统模式

14.1 概述

目录系统模式是关于信息的一系列定义和限制，这些信息是目录本身为了正确操作而需要知道的。这些信息通过子条目和操作属性来规定。

注：系统模式使得目录系统能够，如：

——防止错误类型的子条目与管理条目相关联起来（例如，为一个仅被定义为安全管理条目的管理条目创建一个子模式子条目作为其下级）；

——防止为条目或子条目增加不正确的操作属性（例如，为一个用户条目增加一个子模式操作属性）。

从形式上，目录信息模式包含如下内容的集合：

a) 客体类定义：在这些客体类定义中定义了应当或可能出现在一个给定类的子条目中的属性；

b) 操作属性类型定义：规定了由目录所知并使用的操作属性的特性。

一个操作属性的完整定义包括对目录在操作中，使用和提供（如果合适的话）或管理属性的方法的规范。

目录系统模式是分布式的，如同DIB本身一样。每个管理机构建立了系统模式的一部分，将应用于该机构所管理的DIB的那些部分。

本目录规范中定义的目录系统模式是目录系统本身的一个完整组成部分。每个加入到目录系统中的DSA都要求对它的管理机构所建立的系统模式有一个完整的认识。一个管理区的系统模式可能被其管理机构使用本条所定义的表示法来进行定义。

目录系统模式不被DIT结构规则或内容规则所控制。当一个系统模式的元素被定义时，它应该提供规范规定它如何被使用以及它出现在DIT的什么位置等。

目录系统模式的特定方面在后续部分中规定。

为支持目录分布所需的目录系统模式在第25章到第28章规定。

14.2 支持管理和操作信息模型的系统模式

尽管使用第13章的表示法定义了subentry和subentryNameForm，但子条目不被DIT结构规则或内容规则所控制。

14.2.1 子条目客体类

客体类subentry是一个结构客体类，它的定义如下：

```
subentry   OBJECT-CLASS ::={
           SUBCLASS OF          { top }
           KIND                 structural
           MUST CONTAIN         { commonName | subtreeSpecification }
           ID                   id-sc-subentry }
```

14.2.2 子条目名(称)格式

名(称)格式subentryNameForm 允许使用commonName 属性对类subentry 的条目命名：

```
subentryNameForm   NAME-FORM   ::={
           NAMES                    subentry
           WITH ATTRIBUTES          { commonName }
           ID                       id-nf-subentryNameForm }
```

子条目不得使用其他名(称)格式。

14.2.3 子树规范操作属性

操作属性subtreeSpecification 的语义在第 12 章规定，它的定义如下：

```
subtreeSpecification   ATTRIBUTE   ::={
    WITH SYNTAX          SubtreeSpecification
    USAGE                directoryOperation
    ID                   id-oa-subtreeSpecification }
```

该属性出现在所有的子条目中；每个值都定义了一个条目集(可能是由客体类过滤器所选择提炼后的管理区的一部分)，这些条目集可能是子条目所定义的策略实施的客体。

注：允许在一个单独子条目中定义的某个单独的复杂策略(如一个搜索规则)，在一个管理区的不相交区域，用于多个客体类组合中。

14.3 支持管理模型的系统模式

在第 11 章定义的管理模型要求管理条目包含一个administrativeRole 属性，此属性指示了相关的管理区担任的一种或多种管理角色。

操作属性administrativeRole 的定义如下：

```
administrativeRole   ATTRIBUTE   ::={
    WITH SYNTAX                      OBJECT-CLASS. &id
    EQUALITY MATCHING RULE           objectIdentifierMatch
    USAGE                            directoryOperation
    ID                               id-oa-administrativeRole }
```

本目录规范定义的该属性的值可能为：

id-ar-autonomousArea

id-ar-accessControlSpecificArea

id-ar-accessControlInnerArea

id-ar-subschemaAdminSpecificArea

id-ar-collectiveAttributeSpecificArea

id-ar-collectiveAttributeInnerArea

id-ar-contextDefaultSpecificArea

id-ar-serviceSpecificArea

这些值的语义在第 12 章定义。

操作属性administrativeRole 还可用于规范允许作为一个管理条目下级的子条目。一个administrativeRole 属性所不允许的类的子条目不能作为管理条目的下级。

14.4 支持通用管理和操作需求的系统模式

下面的各条描述了子模式操作属性，这些属性与常规属性不同(即不是由条目所拥有的)，但可能被认为是"虚拟"属性，表示了可派生的信息(例如，从当前存在的操作属性、属性值以及其他信息等)。这些哑属性对于一个管理区内的所有条目都是合法的。因此，结果是这些子模式操作属性出现在每个条目中。

14.4.1 时戳

createTimestamp 指示条目被创建的时间：

```
createTimestamp   ATTRIBUTE  ::={
    WITH SYNTAX                        GeneralizedTime
                  ——GB/T 16262.1—2006 中 42.3 b)或 c)的定义
    EQUALITY MATCHING RULE             generalizedTimeMatch
    ORDERING MATCHING RULE             generalizedTimeOrderingMatch
    SINGLE VALUE                       TRUE
    NO USER MODIFICATION               TRUE
    USAGE                              directoryOperation
    ID                                 id-oa-createTimestamp }
```

modifyTimestamp 指示条目被最后修改的时间：

```
modifyTimestamp   ATTRIBUTE  ::={
    WITH SYNTAX                        GeneralizedTime
                  ——GB/T 16262.1—2006 中 42.3b)或 c)的定义
    EQUALITY MATCHING RULE             generalizedTimeMatch
    ORDERING MATCHING RULE             generalizedTimeOrderingMatch
    SINGLE VALUE                       TRUE
    NO USER MODIFICATION               TRUE
    USAGE                              directoryOperation
    ID                                 id-oa-modifyTimestamp }
```

subschemaTimestamp 指示条目的子模式子条目(见 15.3)创建的时间或最后修改的时间。在每个条目中都可用：

```
subschemaTimestamp   ATTRIBUTE  ::={
    WITH SYNTAX                        GeneralizedTime
          ——GB/T 16262.1—2006 中 42.3b)或 c)的定义
    EQUALITY MATCHING RULE             generalizedTimeMatch

    ORDERING MATCHING RULE             generalizedTimeOrderingMatch
    SINGLE VALUE                       TRUE
    NO USER MODIFICATION               TRUE
    USAGE                              directoryOperation
    ID                                 id-oa-subschemaTimestamp }
```

generalizedTimeMatch 和 generalizedTimeOrderingMatch 匹配规则在 GB/T 16264.6—2008 中定义。

14.4.2 条目修改者操作属性

操作属性creatorsName 指示创建一个条目的目录用户的可辨别名：

```
creatorsName   ATTRIBUTE   ::={
    WITH SYNTAX                     DistinguishedName
    EQUALITY MATCHING RULE          distinguishedNameMatch
    SINGLE VALUE                    TRUE
    NO USER MODIFICATION            TRUE
    USAGE                           directoryOperation
    ID                              id-oa-creatorsName }
```

操作属性modifiersName 指示最后修改条目的目录用户的可辨别名：

```
modifiersName   ATTRIBUTE   ::={
    WITH SYNTAX                     DistinguishedName
    EQUALITY MATCHING RULE          distinguishedNameMatch
    SINGLE VALUE                    TRUE
    NO USER MODIFICATION            TRUE
    USAGE                           directoryOperation
    ID                              id-oa-modifiersName }
```

这些操作属性应当使用主辨别名。

14.4.3 子条目标识操作属性

操作属性subschemaSubentryList 指示控制条目的子模式子条目。它在每个条目中都可用：

```
subschemaSubentryList   ATTRIBUTE   ::={
    WITH SYNTAX                     DistinguishedName
    EQUALITY MATCHING RULE          distinguishedNameMatch
    SINGLE VALUE                    TRUE
    NO USER MODIFICATION            TRUE
    USAGE                           directoryOperation
    ID                              id-oa-subschemaSubentryList }
```

操作属性accessControlSubentryList 标识影响条目的所有访问控制子条目。它在每个条目中都可用。

```
accessControlSubentryList   ATTRIBUTE   ::={
    WITH SYNTAX                     DistinguishedName
    EQUALITY MATCHING RULE          distinguishedNameMatch
    NO USER MODIFICATION            TRUE
    USAGE                           directoryOperation
    ID                              id-oa-accessControlSubentryList }
```

操作属性collectiveAttributeSubentryList 标识影响条目的所有集合属性子条目。它在每个条目中都可用：

```
collectiveAttributeSubentryList   ATTRIBUTE   ::={
    WITH SYNTAX                     DistinguishedName
    EQUALITY MATCHING RULE          distinguishedNameMatch
    NO USER MODIFICATION            TRUE
    USAGE                           directoryOperation
    ID                              id-oa-collectiveAttributeSubentryList }
```

操作属性contextDefaultSubentryList 标识了影响条目的所有上下文缺省子条目。它在每个条目中都可用：

```
contextDefaultSubentryList   ATTRIBUTE   ::={
        WITH SYNTAX                      DistinguishedName
        EQUALITY MATCHING RULE           distinguishedNameMatch
        NO USER MODIFICATION             TRUE
        USAGE                            directoryOperation
        ID                               id-oa-contextDefaultSubentryList }
```

操作属性serviceAdminSubentryList标识了影响条目的所有服务管理子条目，如果存在的话。它在受这些子条目影响的每个条目中都可用。

```
serviceAdminSubentryList   ATTRIBUTE   ::={
        WITH SYNTAX                      DistinguishedName
        EQUALITY MATCHING RULE           distinguishedNameMatch
        NO USER MODIFICATION             TRUE
        USAGE                            directoryOperation
        ID                               id-oa-serviceAdminSubentryList }
```

14.4.4 具有下级操作属性

操作属性hasSubordinates指示拥有该属性的条目下是否有任何一个下级条目存在。值为TRUE表示有下级存在。值为FALSE表示没有下级存在。如果该属性缺失，则关于该条目是否存在下级没有相应的信息可以提供。一般来说，该属性会告知是否存在下级，即使直接下级是被访问控制所隐蔽的——为了防止下级存在信息的泄漏，该操作属性本身也必须被访问控制所保护。

注：如果所有可能的下级都只有通过一个非特定的下级引用才可用时(见GB/T 16264.4—2008)，或者如果唯一的下级为子条目或孩子家族成员时，则没有下级存在也可能会返回值为TRUE。

```
HasSubordinates ATTRIBUTE ::={
        WITH SYNTAX          BOOLEAN
        EQUALITY MATCHING RULE       booleanMatch
        SINGLE VALUE         TRUE
        NO USER MODIFICATION     TRUE
        USAGE            directoryOperation
        ID           id-oa-hasSubordinates }
```

14.5 支持访问控制的系统模式

如果一个子条目包含了预定的访问控制信息，则它的属性objectClass须包含值accessControlSubentry：

```
accessControlSubentry   OBJECT-CLASS   ::={
        KIND                 auxiliary
        ID                     id-sc-accessControlSubentry }
```

这种客体类的子条目须精确地包含一个prescriptiveACI属性，其类型与相应的访问控制特定点的属性accessControlScheme的值相一致。

14.6 支持集合属性模型的系统模式

支持集合属性特定管理区或内部管理区的子条目定义如下：

```
collectiveAttributeSubentry   OBJECT-CLASS   ::={
        KIND                 auxiliary
        ID                     id-sc-collectiveAttributeSubentry }
```

这种客体类的子条目须包含至少一个集合属性。

包含在这种客体类子条目中的集合属性，从概念上来说，在对子条目的subtreeSpecification属性值

指定范围内的每个条目进行查询和过滤时都可应用，但它们是通过子条目进行管理的。

操作属性collectiveExclusions 允许条目排斥某些特定的集合属性：

```
collectiveExclusions   ATTRIBUTE   ::={
        WITH SYNTAX          OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        USAGE                directoryOperation
        ID              id-oa-collectiveExclusions }
```

该属性对每个条目都是可选的。

OBJECT IDENTIFIER 的值id-oa-excludeAllCollectiveAttributes 可能会作为属性collectiveExclusions 的值出现，这样将表示所有的集合属性都被某个条目排斥在外。

14.7 支持上下文断言缺省值的系统模式

提供上下文断言的缺省值的子条目定义如下：

```
contextAssertionSubentry   OBJECT-CLASS   ::={
        KIND          auxiliary
        MUST CONTAIN      {contextAssertionDefaults}
        ID        id-sc-contextAssertionSubentry }
```

这种客体类的子条目须包含一个contextAssertionDefaults 属性：

```
contextAssertionDefaults   ATTRIBUTE   ::={
        WITH SYNTAX          TypeAndContextAssertion
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                directoryOperation
        ID              id-oa-contextAssertionDefault   }
```

无论何时，当一个上下文被评估且用户没有提供上下文断言时，目录都将在控制被访问条目的上下文断言子条目中提供与该属性值相等的上下文断言的缺省值，如 8.9.2.2 的描述。

注：TypeAndContextAssertion 在 GB/T 16264.3—2008 的 7.6 定义(对该属性值的评估在 7.6.3 定义)。

14.8 支持服务管理模型的系统模式

```
serviceAdminSubentry   OBJECT-CLASS   ::={
        KIND          auxiliary
        MUST CONTAIN      { searchRules }
        ID        id-sc-serviceAdminSubentry }
```

这种客体类的子条目须包含一个searchRules 操作属性：

```
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 }
```

操作属性searchRules 的值或者是一个包含实际搜索限制的搜索规则，或者是一个根本没有指定任何搜索限制的虚拟搜索规则。该虚拟搜索规则是通过id 值为零，且不包含serviceType 组件(或者不包

含SearchRule 中的除id 和dmdId 外的其他任何组件)来标识的。dmdId 是为控制 DMD 赋予的一个标识符(见 6.4)。

14.9 支持层次结构组的系统模式

```
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 }

hierarchyTop ATTRIBUTE  ::={
        WITH SYNTAX                 DistinguishedName
        EQUALITY MATCHING RULE      distinguishedNameMatch
        SINGLE VALUE                TRUE
        USAGE                       directoryOperation
        ID                          id-oa-hierarchyTop }
```

操作属性hierarchyLevel 须出现在作为某个层次结构组成员的任何条目中。目录应当负责创建和维护该属性。当条目不再是此层次结构组中的成员时,目录应当删除该属性。对于层次结构中的顶端(top),该属性值为零。在孩子家族成员中,不得出现该属性。

操作属性hierarchyBelow 指示了条目是否具有层次结构孩子。值为TRUE 表示存在层次结构孩子。值为FALSE 或者该属性类型不存在,则表示不存在层次结构孩子。目录必须负责创建和维护该属性。当条目不再是此层次结构组中的成员时,目录必须删除该属性。

当一个新的条目或一个现有的条目变成层次结构的孩子时，必须在增加条目或修改条目操作中出现操作属性hierarchyParent。该属性的值必须为其直接层次结构双亲的可辨别名。如果其直接层次结构双亲为一个组合条目，则该值必须为其祖先的可辨别名。否则，目录必须返回一个修改错误，错误原因为parentNotAncestor。在一个孩子家族成员中，或者在非此层次结构组内的条目中，或者在层次结构的顶端条目中，都不得出现该属性。

操作属性hierarchyTop指向该层次结构组的顶端条目。由目录负责提供和维护该属性。该属性的值必须为顶端条目的可辨别名。如果顶端条目是一个复合条目，则该值必须是其祖先的可辨别名。在一个孩子家族成员中，或者在非此层次结构组内的条目中，或者在层次结构的顶端条目中，都不得出现该属性。

注：该属性提供了一个条目所属于的层次结构组的唯一标识符。

当一个层次结构组中的某个条目通过删除条目操作被删除后，则其所有的层次结构孩子都将从层次结构组中删除。

14.10 系统模式的维护

DSA应当负责维护子条目和操作属性与系统模式之间的一致性。系统模式不同方面的不一致性以及系统模式与子条目和操作属性间的不一致性，都不得发生。

当一个新的子条目加入到DIT中，或者对一个当前的子条目进行修改时，由目录来执行条目的增加和修改过程。目录应当判断所执行的操作是否会违反系统模式，如果违反的话，修改操作将失败。

特别地，目录应当确保增加到DIT中的子条目与administrativeRole属性的值是一致的，且子条目内的属性与子条目的objectClass属性的值是一致的。

属性administrativeRole的值可能会被修改，以便允许还未出现的子条目类可以成为管理条目的下级。但如果会引起当前的子条目变得不一致时，则属性administrativeRole的值不得被修改。

当操作属性的值由目录提供时，目录还应当确保这些值是正确的。

14.11 第一级下级的系统模式

对于作为DIT根的直接下级而创建的条目，目录施加了如下的规则和限制：

—— 所有的这些条目都必须被创建为管理点条目；

—— 这些条目的客体类和命名属性必须与GB/T 16264.1—2008中规定的一致。

15 目录模式管理

15.1 概述

对于全球DIT的目录模式的全面管理，是通过对组成全球DIT的各个DIT域自治管理区子模式进行独立管理来完成的。

在DIT域之间的边界上，对目录模式管理的一致性保证是DMO之间双边商定要解决的课题，不在本目录规范的定义范围之内。

出于管理某个DIT域的目的，而在本条定义的子模式管理能力包括：

a) 创建、删除和修改子模式子条目；

b) 为了允许DSA在操作绑定协议交互中包含模式信息，并允许DUA通过DAP来获取子模式信息，则需要支持发布机制；

c) 为了确保任何修改操作的执行都与所应用的子模式规范一致，可以对子模式进行调整。

15.2 策略客体

子模式策略客体可能是如下中的一种：

——子模式管理区；

——子模式管理区内的客体或别名条目；

——该客体或别名条目的用户属性。

为了管理子模式，自治管理区可能会被设计为一个子模式特定管理区。这应当通过在相关联的管理条目的administrativeRole 属性中出现值id-oa-subschemaAdminSpecificArea 来标识(除了出现值id-oa-autonomousArea 以及其他可能的值以外)。

为了为某些特定部分部署和管理子模式，这样的一个自治管理区可能会被分割为不同的部分。在这种情况下，每个子模式特定管理区的管理条目都是通过在这些条目的属性administrativeRole 中出现值id-oa-subschemaAdminSpecificArea 来标识的。

15.3 策略参数

子模式策略参数用于表示子模式管理机构的策略。这些参数以及用以表示参数的操作属性，包括：

——*DIT 结构*参数：用于定义子模式管理区的结构，并且用于存储已废弃的 DIT 结构规则的信息，这些结构规则可能还被某些条目标识为其控制 DIT 结构规则。该参数由操作属性 dITStructureRules 和 nameForms 来表示；

——*DIT 内容*参数：用于定义在子模式管理区内包含的客体和别名条目的内容类型，并且用于存储已废弃的 DIT 内容规则的信息，这些内容规则可能还被目录使用来判断某些条目的内容。该参数由操作属性dITContentRules 、objectClasses 、attributeTypes 、contextTypes 、friends 和dITContextUse 来表示；

——*匹配能力*参数：用于定义在子模式管理区内定义的属性类型所应用的匹配规则所支持的匹配能力。该参数由操作属性matchingRules 和matchingRuleUse 来表示。

子模式管理机构将使用一个单独的子模式子条目来管理子模式管理区的子模式。出于这种目的，子模式子条目中包含了表示策略参数的操作属性，这些策略参数用于表示子模式的策略。子模式子条目中的属性subtreeSpecification 须指定整个的子模式管理区，即它须是一个空序列。

子模式子条目的定义如下：

```
subschema   OBJECT-CLASS  ::={
            KIND                auxiliary
            MAY CONTAIN   {
                                dITStructureRules   |
                                nameForms   |
                                dITContentRules   |
                                objectClasses   |
                                attributeTypes   |
                                friends   |
                                contextTypes   |
                                dITContextUse   |
                                matchingRules   |
                                matchingRuleUse   }
            ID                  id-soc-subschema   }
```

子模式子条目的操作属性在 15.7 中定义。

15.4 策略规程

有两个与子模式管理相关的策略规程：

——子模式修改过程；和

——条目修改过程。

15.5 子模式修改过程

子模式管理机构可能以一种动态方式来管理子模式，包括执行受限的子模式修改。这种动态方式可能通过使用目录修改操作来修改子模式操作属性的值来实现，这种方式可以有效地修改子模式管理

区中已生效的子模式。一个子模式管理机构还可能分别通过创建或删除子模式子条目来创建一个新的子模式区，或者删除已存在的子模式区。

子模式管理机构可以通过增加一个新规则、或者增加一个新的辅助客体、或者为某个已存在的规则增加一个必选或可选属性等方式等来扩展 DIT 结构规则或 DIT 内容规则，在扩展之前，所涉及的模式信息必须在子模式子条目的适当属性中描述。由 dITStructureRule 、dITContentRule 或者 matchingRuleUse 等属性所(直接或间接)提及的名(称)格式、客体类、属性类型以及匹配规则等，不得从子模式子条目中删除。

已经登记的信息客体的定义，如客体类、属性类型、匹配规则以及名(称)格式等(即已经被分配了一个类型为客体标识符的名(称))都是静态的，不能被修改。对这些信息客体语义的修改要求分配一个新的客体标识符。

DIT 结构规则和 DIT 内容规则可能是活动的或是废弃的。只有活动的规则可用于规范 DIT。对废弃规则的标识和保留是为了提供一种管理上的便利，便于对根据旧的规则而增加的条目进行定位(或修改)，尽管这些旧的规则已经被修改了。

在对 DIT 结构规则或 DIT 内容规则执行受限的修改，并引起 DIB 的不一致时，必须使用此废弃机制。否则，适当的活动规则将被直接修改。目录允许在任何时间删除废弃的规则。

注：在子模式操作属性中提供的废弃机制可以确保所有具有废弃模式的条目，能够在废弃的子模式操作属性被删除之前，还能被标识和修改。

子模式管理机构负责维护条目与活动子模式之间的一致性，可以通过采用目录抽象服务方式或通过其他本地方式来实现。可根据子模式管理机构的便利性来选择。不会定义何时对这种不一致条目执行调整。然而，如果在不一致条目被定位并修复之前就删除此废弃规则，将使得该任务更加困难。

15.6 条目增加和修改过程

任何时候，当一个新的条目被加入到 DIT 中，或一个已存在的条目被修改时，目录都会执行条目增加和修改过程。目录必须判断正在执行的操作是否会违反子模式策略。

特别地，目录必须确保增加到 DIT 中的条目与相应的活动 DIT 结构规则和 DIT 内容规则是相一致的。

目录必须允许对与其活动规则不一致的条目进行询问。

当目录被请求修改 DIB 时，它应当运用活动的规则。如果一个条目与其活动规则不一致，而如果修改条目的请求是为了修复目前的不一致，或者不会引起新的不一致时，则该请求须被允许。如果会引起新的不一致时，请求将失败。

在一个合法的子模式管理区内，对于任何一个合法的条目，在其结构客体类上级类链中，能够仅有一个最下级结构客体类。当一个条目被增加到 DIT 中时，目录将根据所提供的 objectClass 属性值来判断此最下级结构客体类，并且将该结构客体类通过条目的 structuralObjectClass 属性与条目关联起来。

当一个条目被创建时，必须提供属性 objectClass 的值，这样条目的内容就与控制该条目的 DIT 内容规则相一致起来。特别地，当属性 objectClass 的值标识了一个拥有上级类的特定客体类，且其上级类并非 top 时，则须提供所有这些上级类的值。否则，创建条目的目录操作将失败。

随后，目录用户可能会在属性 objectClass 中为条目的辅助客体类增加或删除值。在属性 objectClass 的值被修改后，条目的内容必须与控制此条目的 DIT 内容规则保持一致。特别地，当属性 objectClass 的值标识了一个拥有上级类的特定客体类，且其上级类并非 top，当该特定客体类被增加或删除时，则所有这些上级类的值也都必须被增加或删除，除非这些上级类还分别存在于与未被增加或删除的值相关的上级类链中。

15.7 子模式策略属性

后续的子条规定了子模式策略操作属性，这些属性将：

——出现在子模式子条目中。这些属性的值通过目录修改操作来进行管理，并使用了子模式子条

目的可辨别名；

——可用于子模式所控制的所有条目的查询。

下面定义中使用的 ASN.1 参数数据类型DirectoryString { ub-schema }，在 GB/T 16264.6—2008 中定义。

相等匹配规则 integerFirstComponentMatch 和 objectIdentifierFirstComponentMatch 也在 GB/T 16264.6—2008 中定义。

出于管理的目的，许多人类可读的name 组件和一个description 组件可以可选地作为后续子条定义的多个子模式策略操作属性的组件。

后续子条定义的多个子模式策略操作属性包含了一个obsolete 组件。该组件用于指示该定义在子模式管理区内是活动的，还是废弃的。

15.7.1 DIT 结构规则操作属性

操作属性dITStructureRules 定义了在子模式内使用的 DIT 结构规则：

```
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 }
```

操作属性dITStructureRules 是多值的；每个值都定义了一个 DIT 结构规则。

dITStructureRule 中的组件与 13.7.6 中相应的 ASN.1 定义具有相同的语义。

15.7.2 DIT 内容规则操作属性

操作属性dITContentRules 定义了子模式内使用的 DIT 内容规则。该操作属性的每个值都被打上了它所属于的结构客体类的客体标识符的标签。

```
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 }
```

操作属性dITContentRules 是多值的，每个值都定义了一个 DIT 内容规则。

dITContentRule 中的组件与 13.8.2 中相应的 ASN.1 定义具有相同的语义。

15.7.3 匹配规则操作属性

操作属性matchingRules 规定了一个子模式内使用的匹配规则：

```
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 }
                                                    ——描述了 ASN.1 句法
```

属性*matchingRules* 的值中的组件*identifier* 是标识了该匹配规则的客体标识符。

组件description 包含了与规则相关联的算法的自然语言描述。

组件information 包含了规则声明句法的 ASN.1 定义。

这一个 ASN.1 定义须作为一个可选的 ASN.1 输入产品，后跟一个可选的 ASN.1 分配产品，再后跟一个 ASN.1 类型产品。所有在目录模块中定义的类型名(称)都被隐含地输入进来，并且不需要显式地输入。所有的类型名(称)，无论是输入的还是通过分配定义的，都在本句法定义的本地使用。如果 ASN.1 类型包括一个用户定义的约束，且不是目录模块中定义的 ASN.1 类型之一，则约束中的最后一个用户定义的约束参数(UserDefinedConstraintParameter)应当是一个实际的参数，控制它的类型为 SyntaxConstraint，且取值为分配给此约束的客体标识符。

```
SyntaxConstraint ::=OBJECT IDENTIFIER
```

注 1：ASN.1 输入产品、分配产品以及类型产品在 GB/T 16262.1—2006 中定义。参数 UserDefinedConstraintParameter 在 GB/T 16262.3 2006 中定义。

注 2：典型的 ASN.1 定义只是一个类型名(称)。

操作属性matchingRules 是多值的；每个值都描述了一个匹配规则。

15.7.4 属性类型操作属性

操作属性attributeTypes 规定了子模式内使用的属性类型：

```
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 }
```

属性attributeTypes 的值中的组件identifier 是标识了该属性类型的客体标识符。

操作属性attributeTypes 是多值的；每个值都描述了一个属性类型：

```
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 }
```

组件derivation、equalityMatch、attributeSyntax、multi-valued、collective 以及application 等与相应的信息客体类引入的表示法的对等部分具有相同的语义。

组件attributeSyntax 包括一个文本字符串，给出了属性句法的 ASN.1 定义。这个 ASN.1 定义应当与为匹配规则操作属性的information 组件的规定相同。

15.7.5 客体类操作属性

操作属性objectClasses 规定了子模式内使用的客体类。

```
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 }
```

属性objectClasses 的值中的组件identifier 是标识了该客体类的客体标识符。

操作属性objectClasses 是多值的；每个值都描述了一个客体类：

```
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 }
```

组件subclassOf、kind、mandatories 以及optionals 等与相应的信息客体类引入的表示法的对等部分具有相同的语义。

15.7.6 名(称)格式操作属性

操作属性nameForms 规定了子模式内使用的名(称)格式。

```
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 }
```

属性nameForms 的值中的组件identifier 是标识了该客体类的客体标识符。

操作属性nameForms 是多值的;每个值都描述了一个名(称)格式:

```
NameFormInformation  ::=SEQUENCE  {
        Subordinate     OBJECT-CLASS.&id,
        namingMandatories SET OF ATTRIBUTE.&id,
        namingOptionals     SET SIZE (1..MAX) OF ATTRIBUTE.&id OPTIONAL }
```

组件subordinate 、mandatoryNamingAttributes 以及optionalNamingAttributes 等与相应的信息客体类引入的表示法的对等部分具有相同的语义。

15.7.7 匹配规则用法操作属性

操作属性matchingRuleUse 用于指示子模式内一个匹配规则所应用的属性类型:

```
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 }
```

属性matchingRulesUse 的值中的组件identifier 是标识了该匹配规则的客体标识符。

值中的information 组件标识了匹配规则所应用的属性类型的集合。

15.7.8 结构客体类操作属性

DIT 中的每个条目都拥有一个操作属性structuralObjectClass ,该操作属性指示了条目的结构客体类:

```
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 控制结构规则操作属性

DIT 中的每个条目，除了没有子模式子条目的管理点条目外，都拥有一个操作属性governingStructureRule，该操作属性指示了条目的控制结构规则：

```
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 上下文类型操作属性

操作属性contextTypes 规定了在子模式内使用的上下文类型。

```
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 }
```

属性contextTypes 的值中的组件identifier 是标识了该上下文类型的客体标识符。

操作属性contextTypes 是多值的；每个值都描述了一个上下文类型：

```
ContextInformation  ::=SEQUENCE  {
        syntax                     DirectoryString { ub-schema } ,
        assertionSyntax            DirectoryString { ub-schema } OPTIONAL }
```

组件syntax 和assertionSyntax 与相应的信息客体类引入的表示法的对等部分具有相同的语义。

每个组件syntax 和assertionSyntax 都包含一个文本字符串，分别给出了上下文句法和上下文断言句法的 ASN.1 定义。这一个 ASN.1 定义须作为一个可选的 ASN.1 输入产品，后跟一个可选的 ASN.1 分配产品，再后跟一个 ASN.1 类型产品。所有在目录模块中定义的类型名(称)都被隐含地输入进来，并且不需要显式地输入。所有的类型名(称)，无论是输入的还是通过分配定义的，都在本句法定义的本地使用。如果 ASN.1 类型包括一个用户定义的约束，且不是目录模块中定义的 ASN.1 类型之一，则约束中的最后一个用户定义的约束参数(UserDefinedConstraintParameter)须是一个实际的参数，控制它的类型为SyntaxConstraint，且取值为分配给此约束的客体标识符。

注 1：ASN.1 输入产品、分配产品以及类型产品在 GB/T 16262.1—2006 中定义。参数UserDefinedConstraintParameter 在 GB/T 16262.3—2006 中定义。SyntaxConstraint 在 15.7.3 中定义。

注 2：一个典型的 ASN.1 定义只是一个类型名(称)。

15.7.11 DIT 上下文用法操作属性

操作属性dITContextUse 用于指示必须或可能与某个属性一起使用的上下文：

```
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 }
```

操作属性dITContextUse的值中的组件identifier标识该操作属性所应用的属性类型的客体标识符。值id-oa-allAttributeTypes指示其应用于所有的属性类型。

值中的information组件标识与identifier所标识的属性类型相关联的必选和可选的上下文类型：

```
DITContextUseInformation  ::=SEQUENCE  {
        mandatoryContexts [1]    SET SIZE (1..MAX) OF CONTEXT. &id  OPTIONAL,
        optionalContexts [2]     SET SIZE (1..MAX) OF CONTEXT. &id  OPTIONAL }
```

15.7.12 友人操作属性

操作属性friends用于指示在子模式内为友人的属性类型集合：

```
friends ATTRIBUTE  ::={
        WITH SYNTAX       FriendsDescription
        EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
        USAGE          directoryOperation
        ID          id-soa-friends }

FriendsDescription  ::=SEQUENCE  {
        anchor       ATTRIBUTE. &id,
        name         SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description      DirectoryString { ub-schema }          OPTIONAL,
        obsolete      BOOLEAN DEFAULT FALSE,
        friends          [0]        SET OF ATTRIBUTE. &id }
```

属性friends的值中的组件anchor是标识该友人集所在的锚属性类型的客体标识符。值中的组件friends是标识作为锚属性友人的属性类型的客体标识符的集合。

第七篇：目录服务管理

16 服务管理模型

本章提供了一个表示管理机构如何控制、限制并调整服务的模型，内容包括用户能够在一个搜索、阅读或修改条目的请求中指定什么以及什么样的信息可以返回等。

16.1 定义

本目录规范使用下列术语和定义：

16.1.1

有效出现的属性类型 effectively present attribute type

一个属性类型，出现在搜索过滤器的每个子过滤器内的至少一个非否定过滤项中，且满足在相关的搜索规则中为该属性类型所规定的需求。关于否定过滤器和非否定过滤器项的定义，见 GB/T 16264.3—2008 的 7.8.1。

16.1.2

控制搜索规则 governing-search-rule

某个特定的操作所遵循的一个搜索规则，且该搜索规则被选择用以控制该操作。

16.1.3

命名的服务 named-service

服务类型的集合，共同提供一个完整的服务，如一种白页服务。

16.1.4

请求属性表 request-attribute-profile

对于一个过滤项，对将要有效出现的相应属性类型有什么要求的一个规范。

16.1.5

请求属性类型 request-attribute-type

根据某个搜索规则规范可能出现在一个搜索操作的过滤器中的一个属性类型。

16.1.6

搜索规则 search-rule

为某个给定服务类型提供的关于服务限制/增强方面的详细规范，主要是为某个给定的用户类使用，后调整为某个特定的用户组使用。

16.1.7

服务类型 service-type

在一个定义良好的范围内，完成某种特定目的的服务能力的一个全球唯一标识符，如在 DIT 的某个地区内搜索某个特定类型条目的能力。并不是一个服务类型的所有方面都要对所有用户可用。

16.1.8

子过滤器 subfilter

过滤器中的一个布尔型组件，仅由非否定过滤项和否定过滤项的逻辑与(AND)组成，否定过滤项能够被表达为"NOT(过滤项)"。任何一个过滤器都由子过滤器的逻辑或组成，并能够以一种规范的格式表示，如同在附录 Q 中讨论的那样。

16.1.9

用户类 user-class

一个根据其功能和在组织中的位置等而标识的用户集，能够在一个命名的服务内调用服务类型的某些方面。一个用户类内以名(称)标识的不同用户组可能会看到所提供服务的不同方面。一个用户组的范围可以是一个用户类。

16.2 服务类型/用户类模型

GB/T 16264.3—2008 中所定义的目录抽象服务是本系列目录规范所提供的所有服务能力的表示。*服务类型*是该服务中执行某个特定功能的一个子集，例如在某个定义的范围内搜索某个特定类型的客体。

*已命名的服务*是为某种特定目的而服务的服务类型的集合，例如提供一个白页服务、一个特定类型的黄页服务等。

服务类型主要是通过搜索操作实现的，但也可以通过其他能够指定条目信息选择的操作来实现，即阅读和修改条目操作。出于服务管理的目的，阅读(read)或修改条目(modifyEntry)请求可以被认为

在某种程度上与subset 字段等于baseObject 而filter 字段等于and ：{}的搜索(search)请求是相等的。服务管理不影响一个修改条目操作能够修改什么样的信息。这个仅由访问控制来控制。

客体标识符标识了服务类型，因此给予了服务类型一个全球唯一的标识符。不同的用户类，取决于他们的作用或是在组织中的地位等，可能会对一个服务类型有些不同的感知。一个用户类由一个整型值所标识，该整型值仅要求在一个 DMD 内唯一。不同的 DMD 能够对认为是相同的用户类分配不同的标识符。然而，希望相互协作在跨多个 DMD 内提供了一个共同的命名的服务的管理机构，能够在用户组的标识符方面也协调合作。即使对于一个特定的用户类，可应用于该类用户的服务也可能具有多样性。这样的多样性基于用户的可辨别名。例如，一个国家内属于某个特定用户类的用户，与另一个国家的同一用户类的用户相比，可能不会对同一个服务类型有完全一致的视图，如对本地隐私法的反映。为某个用户组的服务类型的定义通过一个搜索规则earch-rule 来表达，该搜索规则规定了诸如操作如何执行等的细节。

服务类型以及它主要将应用的用户类在搜索规则中规定。

一个用户组可能会跨多个用户类。某个用户类中的用户可能也会实现将主要应用于其他用户类的搜索规则，例如，某个拥有强大能力的用户类中的用户也可以被准予拥有其他通常被赋予较低服务能力的用户类的许可。

一个用户组不是由搜索规则直接标识的，但是可以通过拥有该搜索规则的调用许可而被间接标识。一个用户组能够调用任何一个其拥有调用许可的搜索规则。如果某个特定的用户拥有为同一个服务类型但为不同用户组而定义的多个搜索规则的调用许可，则本系列目录规范中定义的过程将选择具有最高用户组标识符的搜索规则，如果其他任何值都相等的话。这就允许管理机构可以通过正确地分配用户类标识符来控制这种选择。

16.3 特定服务管理区

为了部署和管理搜索规则，一个自治管理区可能会被设计为一个特定服务管理区。这应当通过在相关管理条目的administrativeRole 属性值中出现id-ar-serviceSpecificArea 值来标识(除了出现id-ar-autonomousArea 值以及其他可能出现的值以外)。

为了在特定的部分部署和管理搜索规则，这样的一个自治管理区可能会被分割为不同的部分。在这种情况下，每个特定服务管理区的管理条目，都应当通过在这些条目的属性administrativeRole 中出现值id-ar-serviceSpecificArea 来标识。上级特定服务管理区的服务策略不是该管理条目的相对下级。

如果这样的一个自治管理区不进行分割，则对于搜索规则有一个单独的特定服务管理区，该管理区包含了整个自治管理区。

在目录信息模型中，一个或多个搜索规则通过一个子条目来表示，该子条目被称为服务子条目，它的objectClass 属性中包含值id-sc-serviceAdminSubentry，如 14.8 中的定义。该类子条目应当是管理条目的直接下级，该管理条目的administrativeRole 属性包含值id-ar-serviceSpecificArea 。

在一个特定服务管理区内，操作的评估表达式取决于操作使用了哪个基客体，可能是在别名解除引用之后。因此将搜索规则与条目相关联起来。当一个操作的基客体被决定后，与该条目关联的搜索规则便可作为控制搜索的候选规则。一个子条目内的搜索规则与一个特定服务管理区内的条目之间的关联联系，是通过子条目的操作属性subtreeSpecification 来建立的。通过这种方式，由操作属性subtreeSpecification 的值所标识的条目便与置于同一子条目内的搜索规则关联起来了。

一个特定的条目能够与多个子条目中的搜索规则相联系起来；这些子条目可能拥有相同的或不同的子树规范。相反的，管理区的不同部分能够被一个子条目所指向，这可以通过使用子树规范中的多值来实现。

一个操作的变量是否与搜索规则相违背，可以通过使用一个被称为搜索合法性验证功能的算法来进行判断。

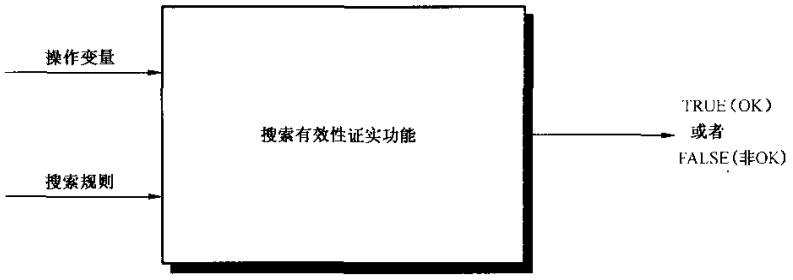


图 16 搜索有效性功能

一个操作是合法的并被允许执行，在且只有在与该操作基客体关联的至少一个可用搜索规则，可以使得“搜索合法性验证功能”取值为 TRUE 的情况下才可以。对于某个操作可用的搜索规则，请求者必须对包含搜索规则的属性值拥有*调用*许可。如果操作只有一个可用的搜索规则可以遵循，则该搜索规则被称为该操作的“*控制搜索规则*”，即操作后续执行时使用的搜索规则。如果有多个这样的搜索规则，则其中的一个将通过本地方式被选择为“控制搜索规则”。选择控制搜索规则的过程在 ISO/IEC 16264.4—2008 的 19.3.2.2.1 描述。因此，控制搜索规则与操作永久性地关联起来，用于其在特定服务管理区内的评估。同样的，当操作的一部分是由拥有特定服务管理区中一部分的其他 DSA 所执行时，也同上所述。

管理机构可以选择，是否：

—— 集合多个需要不同调用许可的搜索规则在一个单独的子条目内（如果这些调用许可是根据值的不同而不同时，则要求属性值层次的访问控制）；或者

—— 集合具有相同访问控制许可的搜索规则在不同的子条目内，因此访问控制许可的准予可基于对完整属性的许可；而不同的子条目能够拥有不同的访问控制许可。

对于一个在某特定服务管理区内指定了基客体条目的操作，如果没有可应用的搜索规则，或者对于所有可用的搜索规则，搜索合法性验证功能都返回 FALSE，则操作被拒绝，并有错误返回。

如果一个特定服务管理区没有子条目，则对该管理区没有相应的服务限制。

可能有用户不得被服务限制所约束，例如管理者；可能有条目，当作为基客体条目时，不得要求有服务限制，例如 DIT 下的条目。因此，管理机构能够包含一个特殊的搜索规则——*空搜索规则*。

一个特定服务管理区内的层次结构组必须被完整地包含在该管理区内。

搜索操作的范围不能跨越特定服务管理区的边界。GB/T 16264.4—2008 规定了一个过程，该过程不允许在某个特定服务管理区内发起的一个搜索操作超出该管理区，即使在搜索评估过程中，别名被解除引用。类似的，在某个特定服务管理区外发起的搜索操作不能扩展至该管理区。

16.4 搜索规则介绍

搜索规则是策略的表达式，一方面可以限制和调整在一个 DIT 领域内执行的操作；另一方面通过对操作过程的指导可以辅助操作的执行。一个搜索规则具有如下主要的特性：

—— 它给出了一个操作应当符合的需求，如果该操作是基于此搜索规则而执行的；

—— 它规定了对操作请求的调整；

—— 它提供了详细的操作评估规范，例如通过规定当搜索操作查找到太多或太少的条目时的放宽策略；以及

—— 它提供了条目信息选择规范。

当一个操作过程开始时，操作的基条目将会符合一个或多个服务子条目，这些服务子条目的子树规范值中包含该基条目。因此潜在的一系列候选搜索规则被标识。操作的详细信息根据这些候选搜索规则进行评估。只有能够找到一个符合的搜索规则时，该操作才能够被执行。

16.5 子过滤器

如果一个搜索规则被设计为去控制搜索操作，则它可能会指定一个会出现在Search 请求过滤器中的属性类型集。这些属性类型被称为搜索规则的“*请求属性类型*”。其他的属性类型不得以任何形式出现在过滤器中，无论是否定形式还是非否定形式。本条将对出现在一个搜索过滤器中的属性类型所具有的含义进行限定。一个搜索规则还指定了请求属性类型的合法组合要求。它可能是某些属性应当出现的要求；或者是两个属性类型中至少一个应当出现的要求；或者是如果不伴随一个已经出现的属性，则某个属性类型不允许出现的要求等。为了更详细地描述如何表达这些组合，有必要引入一个概念：子过滤器。

根据命题演算，任何过滤器都可以写成一个子过滤器的序列，这些子过滤器之间以 OR 操作符分隔。这可以写做：

$$f=f_1+f_2+\cdots+f_r$$

其中，每个子过滤器，f_i，是一个以 AND 操作符分隔的过滤项或是否定过滤项的序列，这可以写做：

$$f_i=f_{i1}f_{i2}\cdots f_{ij}$$

其中，f_{ij} 或者是一个过滤项，或者是其否定。

该子过滤器概念在附录 Q 中有更详细描述。

对于一个过滤器，如果它与某个搜索规则相符合，则它的每个子过滤器都应当与搜索规则相符合。

对于一个在子过滤器中有效表示了一个属性类型的过滤项，要求符合该属性类型的*请求属性表*中的要求。请求属性表是搜索规则规范中的一部分。如果在每个子过滤器中，至少有一个该属性类型的过滤项符合该属性类型的请求属性表，则该属性类型被称为*有效出现的属性类型*。

16.6 过滤器需求

对于一个将有效出现在过滤器中的属性类型，则该属性类型，或者如果请求属性表中的includeSubtypes 选项被设置的话，该属性类型的一个子类型，应当出现在每个子过滤器的至少一个非否定过滤项中。这样的一个非否定过滤项须符合如下的所有要求：

——它须是一个不属于如下类型之一的非否定过滤项：

- greaterOrEqual；
- lessOrEqual；
- present 或contextPresence，除非被请求属性表明确地允许。

——它须与该属性类型的请求属性表规范相一致；

——如果它是一个extensibleMatch 过滤项，则该属性类型须在MatchingRuleAssertion 的type 组件中被指定。

注：如果上述最后一个限制没有被引入的话，则该过滤项能够隐含地在搜索过滤器中包含不定数量的属性类型，因此可能会破坏搜索合法性验证过程。

如果一个属性类型在一个过滤器中有表示，则它须是有效出现的。

可以允许在过滤器中拥有没有type 组件的extensibleMatch 过滤项。它们的存在不会影响到搜索规则的搜索合法性判断过程。然而，这样一个过滤项仅应当被应用到类型为请求属性类型的属性，即由一个请求属性表在控制搜索规则中表示（见 16.10.2）。

16.7 基于搜索规则的属性信息选择

在一个特定服务管理区外，返回的属性信息由操作请求的selection 组件选择，可能会被CommonArguments 中的operationContext 组件修改，或者被任何上下文缺省值修改，这些上下文缺省值或者在一个上下文缺省的特定管理区内缺省建立，或者被本地上下文缺省值缺省建立。对于一个搜索操作，信息的选择也可能会被SearchArgument 中的组件matchedValuesOnly 修改。然而，当一个操作被控制搜索规则所控制时，该搜索规则可能会指定什么样的信息可以返回。在这种情况下，返回的用户属性信息须是控制搜索规则指定的信息和如果没有控制搜索规则时可能返回的信息的交集。如果在selection 组

件中的条目信息选择规定了对操作属性的选择，则相同的规则须应用于操作属性。如果条目信息选择没有指定操作属性信息的返回，则返回的操作属性信息须仅被控制搜索规则所决定。

一个控制搜索规则可能会规定什么样的属性信息将要返回，这与在Search过滤器中能够指定什么样的属性类型之间是完全无关的。

当基于层次结构组的信息要返回时，则从这些条目中进行属性信息的选择应基于上述原则，除matchedValuesOnly规范无效外。

注：家族成员的选择不受上述原则支配(见16.10.6)。

16.8 搜索规则的访问控制方面

除了第18章描述的能力外，搜索规则还提供一些附加的访问控制能力。在一个关注服务的方式下，有必要对如何构建操作表达式以及什么样的信息可以返回等应用限制。限制不仅应当基于用户的身份，还应当基于服务类型和用户类等，因此允许管理机构可以基于信息质量、计费考虑等来对服务进行裁剪。

第18章定义的访问控制能力用于确保仅有适当的用户组能够调用搜索规则。这些能力还可以保护信息永远不会被某个特定的用户组所访问。

一个高速缓存了源于某个特定服务管理区内信息的DSA，可能没有对这些信息限制进行控制的搜索规则。如对于访问控制(见18.8.2)，一个安全管理者应当意识到那样一个具有高速缓存能力的DSA可能会对其他DSA施加严重的安全风险。

16.9 搜索规则的上下文方面

由于一个上下文断言可以是搜索操作过滤项的一部分，因此搜索规则规范必须考虑到上下文。将上下文包含在搜索规则中，将给上下文特性带来新的能力，可能可以简化对DUA和DSA实现的要求。

基本的上下文特性允许用户为搜索过滤器和条目信息选择指定上下文；还允许管理机构在一个上下文缺省特定管理区内建立上下文缺省值。这些缺省值无区别地应用于所有的用户和所有的服务类型。然而，搜索规则所提供的上下文特性允许用户指定一个最小的上下文信息，并且允许管理机构为每个搜索规则给出不同的上下文规范。另外，如16.8所示，通过对搜索规则上下文规范的合理设计，还可能提供类似访问控制的功能。使用搜索规则中的上下文规范能够建立冗余的上下文缺省特定管理区。

16.10 搜索规则规范

ASN.1数据类型SearchRule给出了搜索规则的句法。

```
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}) }
```

—— *下述信息客体集的定义被推迟，可能推迟到标准化概要或协议实现一致性声明时。*

*该集合被要求用来规定一个对*SupportedMatchingRestrictions*组件的限制表。*

```
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 {
    defaulit            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               Attribute Type,
    oldMatchingRule     [0] MATCHING-RULE. &id OPTIONAL,
    newMatchingRule     [1] MATCHING-RULE. &id OPTIONAL }
```

16.10.1 **搜索规则标识符组件**

组件id 允许对一个 DMD 范围内的搜索规则进行唯一地标识。值为零表示为空搜索规则保留。空

搜索规则的目的在 16.3 描述。

组件dmdId 给出了建立此搜索规则的 DMD 的唯一标识符。本组件与id 组件一起对搜索规则给出了一个唯一的全球标识符。

注：这种唯一性如何被管理监督不在本规范的定义范围之内。

组件id（值为零）与组件dmdId 是与空搜索规则相关的仅有的两个组件。

组件serviceType 是一个客体标识符，标识了此搜索规则支持的服务类型。除空搜索规则外，本组件应当始终存在。

组件userClass 指示了搜索规则主要打算应用的用户类。对于一个给定的服务类型，可以有多个搜索规则都指定了同一个用户类。除空搜索规则外，本组件必须始终存在。

16.10.2 请求属性表

组件inputAttributeTypes 必须为所有的必须或可能出现在搜索过滤器中的属性类型指定请求属性表。如果一个搜索过滤器包含了一个过滤项，且该过滤项中的属性类型没有出现在请求属性表中时，则根据此搜索规则的搜索合法性判断将失败。数据类型RequestAttribute 对在过滤项中指定的将有效出现在过滤项中的属性类型做了规定。如果本组件缺失，则搜索规则不会对属性类型的出现设置任何限制，即任何操作都符合本组件。如果本组件存在，但值为空，则仅有缺省过滤器为（and：{ }）的阅读（read）请求、修改条目（modifyEntry）请求或搜索（search）请求才符合本组件。

下述子组件与搜索规则控制的所有操作类型都相关：

a) 子组件attributeType 规定了本规范所应用的属性类型。它是唯一的必选组件。在一个搜索规则内，为某个给定的属性类型只能有一个RequestAttribute 规范。如果除了可能的includeSubtypes 子组件外，仅有此一个子组件，则对此属性类型没有搜索过滤项的限制，除非如果此过滤项在过滤器中，且至少有一个须是非否定的；

b) 子组件includeSubtypes 规定了该请求属性表能够被一个含有此属性类型子类型的过滤项所满足。

下述子组件仅与搜索操作相关：

c) 子组件selectedValues 提供了一个在attributeType 中给定类型的属性值的集合。如果此属性类型出现在过滤器中，则须至少有该属性类型的一个非否定过滤项与该子组件的至少一个值相匹配。否则，该属性类型不会有效出现在过滤器中。

如果这个子组件缺失，则上述的匹配评估值为TRUE 。

如果给定了一个值为空集的属性值，则该属性类型仅能有效出现在：

——一个 present 过滤项中，如果子组件 Contexts 不出现的话；或者

——一个 contextPresent 过滤项中，如果 Contexts 子组件出现的话。

d) 子组件defaultValues 不会影响搜索请求针对搜索规则的评估，但是当一个搜索规则被选择作为控制搜索规则时，该子组件会控制搜索操作。该组件提供了一个在attributeType 中给定类型的属性值的集合。如果在过滤器中定义了一个使用该属性类型的过滤项，但是在条目（或一个家庭组）中没有出现该类型的属性，则当该过滤项与本子组件中的一个值相匹配时，该过滤项被判断为 TRUE（或如果为否定时，被判断为 FALSE）。如果本子组件缺失，则没有缺省值。

如果本子组件存在，但取值为空，它表示本组件可取所有可能的值，即如果条目中没有出现该属性类型时，该属性类型的过滤项总是被判断为 TRUE（或如果为否定时，被判断为 FALSE）。

注：这就反映了一个情况，即如果一个所涉及类型的属性不存在时，则过滤项必须被忽略。

如果一个条目拥有该类型的一个属性，则将对该属性执行正常的匹配。

e) 子组件contexts 规定了允许出现在该属性类型的一个过滤项中的上下文类型。某个特定的上

下文类型不得多于一次地出现在本子组件内。

- 如果该子组件缺失，则任何上下文信息都可能出现在该属性类型的一个过滤项中；
- 如果该子组件存在，则仅有该子组件指定的上下文类型可能出现在该属性类型的一个过滤项中。如果该子组件的取值为一个空序列，则没有任何上下文信息可能出现在该属性类型的一个过滤项中；
- 如果仅有一个上下文类型被指定，则该类型的任何上下文值都可能出现在上下文断言中；
- 如果一个给定上下文类型的上下文值出现在本子组件中，则仅有这些值才可能出现在过滤项的相应上下文断言中。

如果过滤项中的上下文规范不符合上述要求，则该过滤项也不符合该属性类型的请求属性表。

f) 子组件contextCombination 规定了在该请求属性表的contexts 子组件中所列的上下文类型的合法组合。如果该子组件缺失，则对这些上下文类型的组合没有任何限制。如果出现了一个上下文类型的不合法的组合，则过滤项不符合该属性类型的请求属性表。该子组件可能会指定某些应当无条件出现的上下文类型。

g) 子组件matchingUse 用于规定对可应用的匹配规则的用法的可能限制，例如对子串匹配的最小长度的限制等。可应用的匹配规则指的是在放宽之前，但在可能的基本替换之后，确实将要使用的匹配规则。限制的详细信息以及它们是如何被评估的将作为限制规范的一部分被描述。如果该子组件为将要使用的匹配规则指定了一个匹配限制，则将检查是否违反了该匹配限制，或者检查匹配规则的某些不被支持的方面是否要应用。如果是这种情况，则：

——如果 performExactly 搜索控制选项未被设置，则实现时将使用一个本地规则，该本地规则规定了如何以不同的方式应用该匹配规则；

注 2：这样一个本地规则要求一个客户化的能力应用于所讨论的匹配规则。

——如果 performExactly 搜索控制选项被设置，或者它可能不应用一个本地规则，则搜索请求不符合该搜索规则。

16.10.3 属性组合

组件attributeCombination 规定了在inputAttributeTypes 组件中所列的请求属性类型的合法组合。如果该组件缺失，或者具有缺省值(and：{ })，则对于请求属性类型的组合没有任何限制，且所有相关的操作类型都符合此组件。如果出现了请求属性类型的一个不合法组合，则针对此搜索规则的搜索合法性检查将失败。该组件可能会指定某些属性类型必须无条件有效出现在过滤器中。如果inputAttributeTypes 缺失或取值为空，则该组件不得出现。如果该组件出现，且具有一个非缺省的值，则仅仅是具有一个非缺省过滤器的搜索操作才可能潜在地符合此组件。

16.10.4 结果中的属性

组件outputAttributeTypes 规定了什么样的属性类型(或者当没有设置服务控制选项noSubtypeSelection 时，还包括它们的子类型)可能潜在地出现在结果中，并作为访问控制的目标(见 16.7)。如果一个匹配的条目或复合条目中没有包含该组件定义的任何属性，则该条目或复合条目将不被包含在结果中。对于被标识为匹配结果的单个家族成员，或者通过additionalControl 组件中的控制属性所规定的操作得到的单个家族成员，都将应用类似规则。如果这样的家族成员没有拥有本组件所定义的任何一个属性类型，则家族成员以及它们所有下级的这种不符合性都会被显式地标记出来。数据类型ResultAttribute 规定了关于这些属性类型应当如何在结果中表示的详细信息。本组件不会影响搜索的合法性判断。如果组件缺失，则搜索规则不会影响条目信息的选择，除非可能在组件familyReturn 和additionalControl 中有规定。本组件具有如下子组件：

a) 子组件attributeType 规定了本规范应用的属性类型。它是唯一的必选子组件。在一个搜索规则中，对于某个给定的属性类型有且仅有一个ResultAttribute 规范。

b) 子组件outputValues 规定了该属性类型的哪些属性值可作为返回结果的候选。该属性值的

集合还将被上下文子组件、请求者提供的条目信息选择以及访问控制等作更严格的限制。如果本子组件缺失，则所有的属性值都是候选。选项selectedValues 提供了在attributeType 中给定类型的属性值的一个集合。仅有这些列出的值可作为返回结果中属性值的候选。选项matchedValuesOnly 规定了仅有那些属性值才可作为被返回值的候选，那些属性值指的是通过过滤项对过滤器返回 TRUE 做出贡献的，而不仅仅是出现的属性值(关于术语"贡献"的定义，见 GB/T 16264.3—2008 的 10.2.2)。

c) 子组件context 拥有一个上下文表的集合，规定了该属性类型有什么样的属性值信息可以返回。

——如果本子组件缺失，则搜索规则不会对返回的属性值有任何基于上下文方面的限制；

——如果本子组件中没有包含某个上下文类型，则该类型的上下文信息不会伴随此属性类型的任何返回属性值返回；

——如果在一个上下文表中没有包含数据类型contextValue，则该上下文类型的所有上下文值都与伴随每个属性值一起返回；

——如果一个或多个上下文表中都包含数据类型contextValue，则每个这样的上下文表都被认为是一个ContextAssertion，将被应用于 8.9.2.4 中规定的属性值。如果属性值的所有上下文类型都使得声明的评估结果为 TRUE，则那样的属性值才被返回。如果选择结果是没有该类型的属性值返回，则在结果中将不包含该属性。类似的，如果选择结果是没有留下一个条目的任何信息，则该条目不会被返回；

——如果该属性类型的所有返回属性值都具有同样的{上下文类型，上下文值}对要返回，则这样的一个上下文值将从所有的属性值中删除。如果留下一个没有任何上下文值的上下文，则它将被完全删除；

注：将允许一个服务以这样的一种方式被裁剪，即在大部分情况下，用户使用简单的设备就能够获取到没有上下文的信息。

16.10.5 服务和搜索控制

组件defaultControls 如果存在，将用于规定比特的设置，这些比特没有在操作变量的服务控制内，在ServiceControlOptions 中为操作显式地设置，如果操作是一个搜索操作，则是在SearchControlOptions 和HierarchySelections 中设置。如果任一特定选项缺失，则将使用defaultControls 元素(如果它存在的话)。

如果所有的子组件hierarchyOptions 在defaultControls 中都缺失，或者defaultControls 缺失，则不得使用层次结构选择。如果组件hierarchySelection 存在于一个search 变量中，并且规定了除self 外的其他内容，则针对此搜索规则的搜索合法性判断将失败。相应的mandatoryControls 和searchRuleControls 中的元素将被忽略。

如果组件defaultControls 完全缺失，则应当被认为是具有标准的缺省值{serviceControls { }，searchOptions {searchAliases} }。

组件mandatoryControls 通过设置特定的比特，规定了在defaultControls 中指定的应当出现的比特串选项；如果mandatoryControls 指定的任何比特与用户从defaultControls 中提供的选项不同，则针对此搜索规则的搜索合法性判断将失败。组件mandatoryControls 没有指定的比特将取值为零。如果该操作是阅读或修改条目操作，则仅考虑子组件serviceControls 。

组件searchRuleControls 通过设置特定的比特，规定了应从defaultControls 中而不是从用户提供的选项中获取的比特串选项，组件searchRuleControls 没有指定的比特将取值为零。如果该操作是一个阅读或修改条目操作，则仅考虑子组件serviceControls 。

注：如果用户在搜索操作中提供了 U0 to p，且缺省的比特为 D0 to N，则应用 defaultControls 组件的结果是一个比特串 C0 to N，其中，比特 0 到 p 从 U 中取值，其余的从 D 中取值。如果比特串 C&M 不等于 D&M，其中 C 表

示 C0 to N."&"代表一个'比特与'操作，M0 to N 是由 mandatoryControls 指定的比特串，则针对此搜索规则的搜索合法性判断将失败。否则，所使用的选项值为(C&~S | D&S)，其中，S 是由 searchRuleControls 指定的字符串，~S 是比特位的非，"|"表示一个"比特或"操作。后一个运算的结果是：删除 searchRuleControls 中指定的比特，并且使用缺省的比特值来替换它们。组件 familyGrouping 规定了一个家族分组规范，如果存在的话，该规范的优先级高于(即将替换)搜索变量 CommonArguments 的 familyGrouping。

16.10.6 家族规范

组件familyGrouping 规定了一个家族分组选择，如果存在的话，该选择的优先级高于(即替换)搜索变量CommonArguments 中的familyGrouping 。

组件familyReturn 规定了家族成员返回选择。它调整了搜索变量EntryInformationSelection (或其缺省值)中的familyReturn 中给定的规范。搜索规则规范的优先级是根据组件memberSelect 中规范的优先级，而搜索变量规范的优先级是根据组件familySelect 的优先级，即如果在搜索变量中存在familySelect 组件的话，则一个可能出现在搜索规则中的familySelect 组件须被忽略。

16.10.7 放宽的控制

组件relaxation 定义了使用RelaxationPolicy 结构的放宽策略。同样的组件可能包含在搜索请求的relaxation 组件中。与该结构相关的过程描述于此，涵盖了两种情况，一种情况是它包含在一个搜索规则中，另一种情况是它包含在一个搜索请求中。如果RelaxationPolicy 既包含在搜索规则中，又包含在搜索请求中，则附加的规范在 GB/T 16264.3—2008 的 10.2.2 中给出。

RelaxationPolicy 具有如下子组件：

a) 子组件basic ，如果存在的话，定义了MRMapping ，即一个匹配规则替换集和/或应用于搜索过滤器的基于映射的匹配功能集，这些匹配功能是供第一次评估使用的(即无任何收紧或放宽)。这就允许选择一个比初始的匹配更有效的匹配。如果忽略该项，或者该项的值为空集，则结果是所有正常的没有应用任何基于映射的匹配的匹配规则将用于第一次评估；

b) 子组件tightenings ，如果存在的话，组成了一个替换和映射的序列，每个都由 MRMapping 定义，如果匹配条目的数目太多的话(大于maximum 指定的值)，则这些替换和映射将按照给定的序列使用，一次使用一个；

c) 子组件relaxations ，如果存在的话，组成了一个替换和映射的序列，每个都由MRMapping 定义，如果匹配条目太少的话(小于minimum 指定的值)，则这些替换和映射将按照给定的序列使用，一次使用一个；

d) 如果tightenings 存在的话，子组件maximum 须总是存在，且该子组件规定了条目的数量，如果发现的条目在此数量之上，则将应用收紧策略；

e) 子组件minimum 规定了条目的数量，如果发现的条目为此数(或低于此数)时，则将应用放宽策略；如果该子组件缺失，则其缺省值为零。

注 1：放宽/收紧不受performExactly 搜索控制选项的影响。

匹配规则替换和映射由元素MRMapping 定义，每个都包含一个Mapping 元素的序列和一个MRSubstitution 元素的序列。这些元素的序列顺序不重要。

一个Mapping 元素包含如下组件：

a) 组件mappingFunction 标识了一个基于映射的映射功能，与将要应用的映射表相关联；

b) 组件level 标识了基于映射的匹配将应用的放宽级别(或是相反的收紧级别)。当&userControl 为了基于映射的匹配而被设置，且extendedArea 搜索控制包含在搜索请求中时，该组件必须被忽略，在这种情况下，将应用extendedArea 中指定的值。

注 2：对于 basic 替换和映射，level 应当在很多情况下都被设置为零。

一个MRSubstitution 元素包含如下组件：

a) 组件attribute 描述了将要被应用替换的属性；

b) 组件oldMatchingRule是将要被替换的匹配规则。如果该组件缺失，则它应用于该指定的属性类型之前所应用的匹配规则，如果有的话。对于基本的替换，或者如果基本替换没有执行，对于第一次的放宽/收紧替换，则所应用的匹配指的是如果没有此次替换将要使用的那个匹配。对于后续的替换，所应用的匹配规则指的是之前的替换所引入的匹配规则。如果该子组件指定的匹配规则不是之前所应用的匹配规则，则没有替换可以执行。

注3：作为一个示例：如果类型为equality的一个过滤项，并因此选择了一个相等匹配规则，但该子组件指定了一个子串匹配规则，则将没有替换可以执行。

c) 组件newMatchingRule是一个将要用于替换旧的匹配规则的替换匹配规则的客体标识符。如果该组件缺失，则任何相应的过滤项对于非否定项都判断为TRUE，而对否定项则判断为FALSE(即与id-mr-nullMatch的含义一致)。

下述仅应用于search请求中指定的匹配规则替换。如果指定了一个匹配规则，而对于该匹配规则，有一个关于属性类型的匹配限制(见16.10.2的g)项)将使得搜索请求不符合控制搜索规则；或者指定了一个不支持的匹配规则或不一致的匹配规则，则该替换被放弃，且对于该属性类型，后续将不再应用任何替换。

注4：假设一个DSA将不允许在一个搜索规则中出现不合法的替换。

假若属性和oldMatchingRule(如果存在的话)的组合是唯一的，则该属性能够在一个MRMapping中拥有多个MRSubstitution元素。当oldMatchingRule组件在一个MRSubstitution中缺失，而在另一个MRSubstitution中存在时，则后面的一个在映射oldMatchingRule中定义的匹配规则时，其优先级将高于前一个。

16.10.8 附加的控制组件

组件additionalControl允许控制搜索规则的效果能够适应于一个特定的环境，在这个环境中需要一个附加的搜索操作控制。它规定了一个或多个控制属性类型。本组件所提及的此类控制属性类型的语义、句法和放置位置等应当作为控制属性定义的一部分来定义。这样的一个规范可能不在本系列目录规范的定义范围之内。一个指定的控制属性的定义中，包括基于控制属性所提供的信息而将执行的过程。

本组件不会影响搜索有效性证实功能。

一个控制属性可以以这样的方式来放置，即可以影响多个条目，例如：放置在一个特定服务管理点，或放置在一个服务管理子条目内。它还可以被放置在单个的条目中。当控制属性被放置在单个的条目中时，它仅能够影响那些条目的条目信息选择。一个控制属性可能会引起某些条目或家族成员被*显式地不标注*，这样将防止它们出现在搜索结果中。

注1：通过将控制属性放置在特定服务管理点中，可以使得控制属性影响匹配执行的方式。例如，在某个过滤项中指定的一个属性类型能够被映射为一个属性类型集，或者被一个属性类型集(如"友人"属性类型)所补充，则对此属性类型集，匹配可以以某种已定义的方式来执行，例如达到与属性子类型化所提供的同样效果。类似的，一个控制属性能够调整返回的条目信息。

注2：通过将控制属性放置在一个给定的条目中，可以使得单个条目的请求可能被重点考虑，例如覆盖用户的数据保护要求。

如果由于一个或多个控制属性的执行结果而使得复合条目被标注或未被标注，则必须在应用"家族返回规范"(正如在EntryInformationSelection中的familyReturn所指定的那样，或者被familyReturn搜索规则组件所覆盖)之前就执行。如果显式地不标注所引起的结果是没有该复合条目的任何一个成员会返回，即该复合条目将被完全地从结果中删除。

16.10.9 其他组件

组件allowedSubset规定了搜索请求subset规范的合法选择。如果imposedSubset搜索规则组件存在，且在某个搜索请求中没有设置useSubset搜索控制的话，则此搜索规则组件被忽略，缺省的情况是，任何subset选择都是可能的。如果一个搜索请求的subset参数没有指定符合该搜索规则组件的值，则

针对此搜索规则的搜索合法性判断将失败。对于一个符合此组件的阅读或修改条目操作，必须包括值baseObject 。

组件imposedSubset 指定了一个subset ，该子集将替代搜索请求中的subset 规定。如果该组件没有出现，或者在搜索请求中设置了useSubset 搜索控制，则不执行任何替换，并且allowedSubset 所表示的限制将被执行。当针对某个搜索规则，对read 或modifyEntry 请求进行评估时，该组件必须被忽略。

组件entryLimit 有两个子组件。子组件default 指示了当sizeLimit 服务控制没有被设置时，目录所施加的尺寸限制。子组件max 指示了sizeLimit 服务控制所允许的最大值。如果超出此最大值，则生效的sizeLimit 被限制为此max 值。当针对一个搜索规则，对read 或modifyEntry 请求进行评估时，该组件必须被忽略。

16.10.10 ASN.1 信息客体类

提供了信息客体类SEARCH-RULE 、REQUEST-ATTRIBUTE 和RESULT-ATTRIBUTE 来简化搜索规则的文档：

```
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,
    &addifionalControl      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-C LASS. &id      OPTIONAL,
                              valuesSEQUENCE 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.
    &outputVlues            CHOICE {
        selectedValues      SEQUENCE OF ATTRIBUTE. &Type,
        matchedValuesOnly   NULL }                              OPTIONAL,
    &contexts               ContextProfile                      OPTIONAL }

WITH SYNTAX {
    ATTRIBUTE TYPE          &attributeType
    [OUTPUTVALUES           &outputValues ]
    [CONTEXTS               &contexts ] }
```

16.11 匹配限制定义

管理机构可能会希望对如何应用匹配规则设置限制。例如，关于一个子串匹配规则的限制可能会规定一个搜索过滤项中应当提供的子串最小长度。这样的限制是永久性的，不具备如搜索放宽那样的动态调整特性。

在一个特定服务管理区内，可以通过构造适当的搜索规则来应用这些限制，并且这也是引入匹配限制的唯一方法。

匹配限制可以被定义为MATCHING-RESTRICTION 信息客体类的值：

```
MATCHING-RESTRICTION ::=CLASS {
        &Restriction,
        &Rules          MATCHING-RULE. &id,
        &id             OBJECT IDENTIFIER UNIQUE }

WITH SYNTAX  {
        RESTRICTION     &Restriction
        RULES           &Rules
        ID              &id }
```

对于每一个使用此信息客体类定义的匹配规则限制：

a) &Restriction 是要应用的匹配限制的句法；

b) &Rules 是该匹配限制可以应用的匹配规则的集合。匹配限制仅能为某个基本的匹配规则而定义，即一个在其定义中没有双亲匹配规则的匹配规则；

c) &id 是分配给该匹配限制的客体标识符。

可以为任意一个匹配规则定义多个匹配限制，但是在某个给定的情况下，只能应用其中的一个。

16.12 搜索有效性证实功能

搜索有效性证实功能是一个抽象功能，用于判断一个搜索请求是否符合一个特定的搜索规则。如果搜索请求符合搜索规则，则搜索有效性证实功能结果为 TRUE。否则，结果为 FALSE。对于一个符合搜索规则的搜索请求应当：

——除了inputAttributeTypes 中指定的属性类型外，其他属性类型不得以任何形式出现在搜索过滤器中，无论是否定的，还是非否定的；

——如果一个属性类型出现在一个过滤器中，则它也必须是有效出现的；

注：这意味着一个属性类型不得仅仅通过否定过滤项来表示。

——在搜索规则的组件attributeCombination 中规定的请求属性有效出现的条件必须被满足；

——如果有请求属性表包含selectedValues 子组件，则相应的属性必须仅通过非否定过滤项来表示；

——在搜索变量中的subset 规范必须符合搜索规则的subset 规范；

——组件mandatoryControls 中规定的必选控制应当与为搜索规则所定义的defaultControls 相同。

对于通过在一个子过滤器的一个或多个过滤项中表示的，且在此子过滤器中有效出现的一个属性类型，至少有一个过滤项必须符合该属性类型的RequestAttribute 规范，即：

——过滤项的类型必须如 16.6 中的规定；

——如果子组件 selectedValues 在请求属性表中出现，且非空，则过滤项必须与此子组件相匹配；

——过滤项中的上下文规范必须符合请求属性表中的上下文规范；

——过滤项中的匹配规则规范必须符合请求属性表中的匹配规则规范；以及

——任何匹配限制都须被满足。

详细的搜索合法性判断过程在 GB/T 16264.3—2008 的第 13 章规定。

第八篇:安　　全

17　安全模型

17.1　定义

本目录规范使用 GB/T 9387.2 中定义的下述术语:

——访问控制　access control

——鉴别　authentication

——安全策略　security policy

——保密性　confidentiality

——完整性　integrity

本目录规范使用下列术语和定义:

17.1.1

访问控制方案　access control scheme

对目录信息的访问进行控制的方式以及潜在的对其访问权限进行控制的方式。

17.1.2

被保护项　protected item

目录信息的一个元素,对其的访问可以被分别控制。目录的被保护项可以是条目、属性、属性值和名(称)等。

17.2　安全策略

在目录所处的环境中,有各种管理机构对他们的 DIB 部分进行访问控制。一般来说,这样的访问与某些主管部门所控制的安全策略是相适应的(见 ISO/IEC 9594-8)。

安全策略中,影响目录访问的两个方面,或者说两个组件是:鉴别过程和访问控制方案。

注:第 18 章定义了两个访问控制方案:基本访问控制和简化的访问控制,第 19 章定义了基于规则的访问控制。这些方案可能会与本地管理控制联合使用;然而,由于本地管理策略没有标准的表示,因此不能以影像信息进行通信。

17.2.1　鉴别过程和机制

在目录上下文中的鉴别过程和机制包括对下述信息进行验证的方法以及将其传播到所需地方的方法:

——DSA 和目录用户的身份;

——在某个访问点接收到的消息的消息源身份。

注 1:相比较对于非管理员用户的鉴别配置,管理机构可能会为管理员用户的鉴别规定不同的配置。

通用用法(General-use)鉴别过程在 ISO/IEC 9594-8 中定义,并且能够与本目录规范中定义的访问控制方案联合使用,以便增强安全策略。

注 2:本系列目录规范将来的版本可能会定义其他访问控制方案。

注 3:本地管理策略可规定在其他某些 DSA(如在其他 DMD 中的 DSA)中发生的鉴别将被忽视。

通常来说,从被鉴别的身份(如被某个鉴别交互进行鉴别的人类用户身份)到访问控制身份(如用于表示用户的条目的可辨别名以及一个可选的唯一可辨别名)之间有一个映射功能。某个特定的安全策略可能会声明被鉴别的身份和访问控制身份是相同的。

用于访问控制身份中的名(称),必须使用主辨别名。类似的,当访问控制在它的准予和拒绝规范中使用名(称)时,也必须使用主辨别名。

17.2.2　访问控制方案

在目录上下文中定义的访问控制方案中包括的方法可以:

——规定访问控制信息(ACI);

——施加上述访问控制信息所定义的访问权限;

——维护访问控制信息。

施加访问权限,是用于控制对如下信息的访问:

——与名(称)相关的目录信息;

——目录用户信息;

——目录操作信息,包括访问控制信息。

管理机构在实现其安全策略时,可能会使用所有的或部分的标准访问控制方案,或者也可能会根据其判断力,自由地定义他们自己的方案。

然而,管理机构可能会为保护某些或全部目录操作信息而规定不同的配置。不要求管理机构为普通用户提供方法,以便检测到为保护操作信息而进行的配置。

注 1:管理策略可能会准予或拒绝对特定属性(如操作属性)的任何形式的访问,而不考虑可能会应用的访问控制。

目录提供一种方法,通过使用accessControlScheme 操作属性,使得在 DIB 的某个特定部分生效的访问控制方案可以被标识出来。该方案的范围由一个"访问控制特定区(ACSA)"来定义,这是一个特定的管理区,由相应的安全机构负责。该属性置于相应管理点的管理条目内。仅有访问控制特定点的管理条目才被允许包含一个accessControlScheme 属性。

注 2:如果在访问某个给定条目时,该操作属性不存在,则 DSA 的行为必须同第 1 版的 DSA(即决定使用哪种访问控制机制及其对操作、结果和错误的影响等,都属于本地事物)。

```
accessControlScheme   ATTRIBUTE   ::={
        WITH SYNTAX          OBJECT IDENTIFIER
        EQUALITY MATCHING RULE        objectIdentifierMatch
        SINGLE VALUE          TRUE
        USAGE          directoryOperation
        ID          id-aca-accessControlScheme }
```

任何一个 ACSA 中的子条目或条目都被允许包含条目 ACI,当且仅当该 ACI 是允许的,且与相应 ACSA 的accessControlScheme 属性值一致。

17.3 目录操作的保护

有两种保护形式可用于目录操作:保密性和完整性。

保密性仅在一个点对点基础上通过使用 TLS 才可用,可能是为了 IDM 目录协议和 LDAP 而调用的。

TLS 不用于 OSI 目录协议。应当注意的是在一个分布式环境中,点对点保护可能是不够的;然而,端到端的保密性仅靠通过对属性自身的保护才能够提供。

完整性的提供有两种方式。点到点完整性可能通过使用 TLS 为 IDM 目录协议和 LDAP 提供。端到端完整性的提供是通过签名和可选的链接签名的目录操作完成的,而不是通过使用如下面所述的 OPTIONALLY-PROTECTED 的 LDAP 完成的。包含目录操作的 PDU 是不被保护的;相反的,参数,结果和错误等是被保护的。并没有一个机制用于提供一个安全持久事件(如 DAP 操作)的记录。LDAP 操作不通过本目录规范定义的方法进行保护。

注:实验性的 IETF RFC 2649"拥有操作签名的一个 LDAP 控制和模式",提出了这样一个机制:对包含 LDAP 操作的 PDU 进行签名,并为这些操作提供一个安全持久的记录。

OPTIONALLY-PROTECTED 是一个参数化的数据类型,这样的参数是一个数据类型,其值根据发起者的选择可能会伴随一个数字签名。这个能力通过下述类型来指定:

```
OPTIONALLY-PROTECTED { Type }   ::=CHOICE {
        unsigned        Type,
```

```
        signed      SIGNED {Type} }
```

当被保护的数据类型是一个未加标记的序列型数据类型时，将使用OPTIONALLY-PROTECTED-SEQ 替换OPTIONALLY-PROTECTED 。

```
OPTIONALLY-PROTECTED-SEQ { Type } ::=CHOICE {
        unsigned      Type,
        signed  [0]   SIGNED { Type } }
```

SIGNED 是一个参数化数据类型，描述了信息被签名的方式，在 ISO/IEC 9594-8 中规定。

18 基本访问控制

18.1 范围和应用

本章为目录定义了一个特定的访问控制方案(可能会有多个)。这里定义的访问控制方案通过将操作属性accessControlScheme 赋值为basic-access-control 来进行标识。17.2.2 描述了哪个条目包含操作属性accessControlScheme 。

注：一个名为“简化的访问控制”的访问控制方案在 18.9 定义。它被定义为“基本访问控制方案”的一个子集。当使用简化的访问控制时，操作属性accessControlScheme 必须取值为simplified-access-control 。另外的名为“基于规则的访问控制”的访问控制方案在第 19 章定义。

这里定义的方案仅仅是提供了对 DIB 内的目录信息(潜在地包括树型结构和访问控制信息)进行访问控制的方法。它没有指出为了与某个 DSA 应用实体进行通信的目的而进行的访问控制。对信息的访问进行控制指的是防止对信息进行非授权的检测、泄漏或修改等。

18.2 基本访问控制模型

目录的基本访问控制模型为每个目录操作都定义了一个或多个点，在这些点上将执行访问控制决策。每个访问控制决策包括：

——被访问的目录信息元素，被称为“被保护项”；

——发起请求的用户，被称为“请求者”；

——一个为完成操作的一部分而必需的特定权限，被称为“许可”；

——一个或多个操作属性，这些操作属性共同包含了对该被保护项进行访问控制的安全策略，被称为“ACI 项”。

因此，基本的访问控制模型定义了：

——被保护项；

——用户类；

——执行每个目录操作所需的许可类别；

——应用范围和 ACI 项的句法；

——一个基本的算法，被称为访问控制决策功能(ACDF)，用于决定某个特定的请求者根据所应用的 ACI 项是否拥有某种特定的许可。

18.2.1 被保护项

一个被保护项是目录信息中的一个元素，对于这些元素的访问可以分别进行控制。目录的被保护项有：条目、属性、属性值和名(称)。为了便于规定访问控制策略，基本的访问控制提供了方法来标识相关项的集合，如一个条目中的属性或者一个给定属性的所有属性值等，并且为它们规定了共同的保护。

18.2.2 访问控制许可和它们的范围

访问控制是通过对许可的准予和拒绝来实现的。许可的类别在 18.2.3 和 18.2.4 中描述。

访问控制的范围可以是一个单独的条目或条目的一个集合，这些集合中的条目处于某个特定管理点的子条目范围内，因此在逻辑上相关。

许可类别一般来说是独立的。由于所有的目录条目都在 DIT 内有一个相对位置，对用户和操作信

息的访问总是包含对 DIT 相关信息的某种形式的访问。因此，与目录操作相关的访问控制决策有两种主要的形式：对称为客体的条目的访问（被称为条目访问）；对包含用户信息和操作信息的属性的访问（被称为属性访问）。对许多目录操作而言，这两种许可形式都是需要的。另外，当可应用时，不同的许可分别控制返回的名（称）或错误类型。许可类别的某些重要方面，访问的形式和访问控制决策的产生等描述如下：

a) 为了对所有条目执行目录操作（如阅读一个条目，或增加一个条目），通常有必要对该条目内所包含的属性及其值的许可进行准予。例外情况是控制条目重命名和删除的许可：在任何一种情况下，属性或属性值许可都不会被考虑。

b) 为了执行需要对属性或属性值进行访问的目录操作，有必要对包含那些属性或属性值的条目或条目集拥有条目访问许可。

注 1：删除一个条目或一个属性不要求对条目或属性的内容进行访问。

c) 是否允许对条目进行访问，此决策严格地来说是由 DIT 内的条目位置所决定的，即根据其可辨别名来决定，并且与目录如何定位该条目无关。

d) 基本访问控制的一个设计原则是：只有当在做访问控制决策的目录所使用的访问控制信息中显式提供了许可准予的情况下，才有可能允许访问。准予一种形式的访问（如条目访问）从来不会自动或隐含地准予其他形式的访问（如属性访问）。为了管理有效的目录访问控制策略，因此经常有必要显式地为两种访问形式设置访问策略。

注 2：准予或拒绝的某些组合是不合逻辑的，但这是用户的责任，而不是目录的责任，来确保不出现这样的组合。

注 3：与上述设计原则相一致，对某个属性值许可的准予或拒绝不会自动地控制对相应属性的访问。此外，为了在一个目录查询操作中访问到一个属性值，则用户必须被准予访问该属性类型及其值。

e) 模型所提供的唯一一个缺省的访问决策是：在没有显式的访问控制信息准予访问时，则拒绝访问。

f) 在访问控制信息中规定的拒绝，其优先级总是高于准予，此外其他因素都相等。

g) 一个特定的 DSA 可能不会拥有控制它所高速缓存的目录数据的访问控制信息。安全管理员应当意识到，此具有高速缓存能力的 DSA 可能会对其他 DSA 形成重大的风险，因为它可能会将信息泄露给非授权的用户。

h) 出于查询的目的，与某个条目相关的集合属性被保护起来，就好像它们是组成该条目的属性一样。

注 4：出于修改的目的，集合属性应与拥有它们的子条目相关，而不是与子条目范围内的条目相关。因此，与修改相关的访问控制与集合属性是无关的，除非它们应用于子条目内的集合属性及其值。

18.2.3 条目访问的许可类别

用于控制条目访问的许可类别是：*Read*，*Browse*，*Add*，*Remove*，*Modify*，*Rename*，*DiscloseOnError*，*Export* 以及*Import* 和*ReturnDN* 等。它们的用法在 GB/T 16264.3—2008 中给出详细描述。

附录 L 对在一般情况下它们的含义给出了一个概述。本条通过简要地指出与每个类别的准予相关的目的，从而引入这些类别。然而，一个特定的准予许可对访问控制决策的实际影响，是由 ACDF 的完整的上下文和每个目录操作的访问控制决策点所决定的。

a) *Read*，如果被准予，则允许目录操作的阅读访问，该操作明确地指定了条目的名（称）（即与列表和搜索操作相反），并对它所应用的条目中包含的信息提供了可视性。

b) *Browse*，如果被准予，则允许使用目录操作对条目进行访问，该操作没有显式地提供条目名。

c) *Add*，如果被准予，则允许在 DIT 中创建一个条目，且在创建时置于新条目中的所有属性和属性值都可被控制。

注 1：为了增加一个条目，还应当至少准予增加必选属性及其值。

注 2：没有一个特定的“增加下级许可”。增加一个条目的许可通过使用 18.3 描述的操作属性prescriptiveACI

来控制。

d) *Remove*，如果被准予，则允许从 DIT 中删除条目，而不考虑对条目内属性或属性值的控制。

e) *Modify*，如果被准予，则允许修改条目中包含的信息。

注 3：为了修改包含在条目中的除可辨别名属性值外的信息，其他相应的属性及其值的许可也必须被准予。

f) *Rename* 的准予对于一个要使用新的 RDN 进行重新命名的条目来说是必要的，如果有的话，还需要考虑由此而产生的下级条目可辨别名的改变；如果上级的名(称)没有改变，则许可是足够的。

注 4：为了重命名一个条目，对于所包含的属性或属性值不需要有先决条件的许可，包括 RDN 属性；即使由于 RDN 的改变，使得该操作引起新的属性值被增加或被删除，上述说明也是正确的。

g) *DiscloseOnError*，如果被准予，则允许在一个错误(或空)结果中显示条目的名(称)。

h) *Export*，如果被准予，则允许一个条目及其下级(如果有的话)被输出；也就是说，将其从当前的位置删除，然后放置到一个新的位置，并符合目标位置处适当的许可准予。如果最后的 RDN 有改变，则在当前位置还需要执行*Rename*。

注 5：为了输出一个条目及其下级，对于所包含的属性或属性值不需要有先决条件的许可，包括 RDN 属性；即使由于 RDN 的改变，使得该操作引起新的属性值被增加或被删除，上述说明也是正确的。

i) *Import*，如果被准予，则允许一个条目及其下级(如果有的话)被输入；也就是说，将其从某个其他的位置删除，并放置到许可应用的位置上(符合在源位置处适当的许可准予)。

注 6：为了输入一个条目及其下级，对于所包含的属性或属性值不需要有先决条件的许可，包括 RDN 属性；即使由于 RDN 的改变，使得该操作引起新的属性值被增加或被删除，上述说明也是正确的。

j) *ReturnDN*，如果被准予，则允许在操作返回的结果中显示条目的可辨别名。

18.2.4 属性和属性值访问的许可类别

用于控制属性和属性值访问的许可类别有：*Compare*，*Read*，*FilterMatch*，*Add*，*Remove* 和*DiscloseOnError*。它们的用法在 GB/T 16264.3—2008 中给出详细的描述。附录 L 对在一般情况下它们的含义给出了一个概述。本条通过简要地指出与每个类别的准予相关的目的，从而引入了这些类别。然而，一个特定的准予许可对访问控制决策的实际影响，是由 ACDF 的完整的上下文和每个目录操作的访问控制决策点所决定的。

a) *Compare*，如果被准予，则允许属性及其值用于一个比较操作中。

b) *Read*，如果被准予，则允许属性及其值可以作为阅读或搜索访问操作返回的条目信息。

c) *FilterMatch*，如果被准予，则允许对一个搜索条件内的过滤器进行评估。

d) *Add*，如果增加一个属性被准予，则允许增加一个属性，并能够增加所有指定的属性值。如果增加一个属性值被准予，则允许为某个现有的属性增加一个值。

e) *Remove*，如果删除一个属性被准予，则允许完全删除一个属性及其所有的值。如果删除一个属性值被准予，则允许从某个现有的属性中删除一个值。

f) *DiscloseOnError*，如果是针对一个属性被准予，则允许该属性出现在属性或安全错误中。如果是针对一个属性值被准予，则允许该属性值出现在一个属性或安全错误中。

g) *Invoke*，如果被准予，则许可所应用的客体(总是一个操作属性或一个操作属性的值)，能够被 DSA 当做授权的用户来调用。该调用所完成的功能是取决于属性的。对于用户来说，不需要拥有操作属性，或拥有此操作属性的条目/子条目的其他许可。

18.3 访问控制管理区

DIT 被分割为不同的子树，这些子树被称为“自治管理区”，每个自治管理区都由一个单独的域管理组织内的管理机构所控制。出于特定方面的管理目的，它还可再被分割为被称为“特定管理区”的子树；或者整个自治管理区可能由一个单独的特定管理区构成。每个这样的特定管理区都由相应的特定管理机构负责。某个特殊的管理区可能由多个特定的管理机构共同负责，见第 11 章。

18.3.1 访问控制区和目录访问控制域

在访问控制情况下,特定的管理机构是一个安全机构,且特定的管理区被称为一个"访问控制特定区(ACSA)"。ACSA 的根被称为"访问控制特定点"。每个访问控制特定点在 DIT 中都由一个管理条目来表示,且条目的操作属性administrativeRole 中都包含值access-control-specific-area ;它(潜在地)拥有一个或多个包含访问控制信息的子条目。类似的,每个访问控制内部点在 DIT 中都由一个管理条目来表示,且条目的操作属性administrativeRole 中都包含值access-control-inner-area ;它也(潜在地)拥有一个或多个包含访问控制信息的子条目。每个这样的拥有子条目的管理条目,其中子条目中包含了指定的 ACI 信息,其操作属性accessControlScheme 都取值为basic-access-control 、simplified-access-control 或其他相关值。每个属于一个访问控制特定点,且包含了访问控制信息的子条目,都将accessControlSubentry 作为其客体通用性的值。一个管理条目及其子条目可能会拥有一些操作属性(如访问控制信息),这些操作属性分别与管理点(以及可能的子条目)和由子条目的subtreeSpecification 定义的(在管理区内的)条目集合相关。

当且仅当拥有它的管理条目是一个访问控制特定条目时,属性accessControlScheme 才必须存在。

一个管理条目永远都不可能既是一个访问控制特定条目,又是一个访问控制内部条目;因此,相应的值不可能同时出现在administrativeRole 属性中。

一个包含访问控制信息的子条目的范围,在它的subtreeSpecification 中定义(可能包含子树精选),被称为目录访问控制域(DACD)。

注:一个 DACD 能够包含零个条目,也能够包含尚未加入到 DIT 中的条目。

安全机构可能会允许一个访问控制特定区被分割为不同的子树,这样的子树被称为内部(管理)区。每个这样的内部区,被称为一个"访问控制内部区(ACIA)",其操作属性administrativeRole 取值为access-control-inner-area 。相应管理点的包含了指定 ACI 信息的每个子条目,如前所述,都在其客体类属性中取值accessControlSubentry 。

一个 ACIA 中的子条目所规定的范围(subtreeSpecification)也是一个 DACD,并且包含了相关的访问控制内部区中的条目。

ACIA 允许在 ACSA 内对访问控制权力进行一定程度的授权。ACSA 的权力在 ACIA 内仍然保留,因为 ACSA 的管理点子条目内的 ACI 的应用与相关 ACIA 子条目的 ACI 应用相同(18.6 解释了 ACSA 是如何控制权力的)。

总地来说,在评估访问控制时,访问控制方案的类型(如基本访问控制)通过相关的访问控制特定条目中的accessControlScheme 属性的值来指示;ACSA 内的每个相关管理条目的作用由administrativeRole 属性的值来指示;指定的访问控制在一个特定子条目内的出现由其客体类属性中的值accessControlSubentry 来指示。

子条目,同其他条目一样,为了保护它自己的内容,可拥有一个entryACI 属性。

18.3.2 将控制与管理区相关联

对一个给定条目的访问(潜在地)是由全体上级访问控制管理点(包括内部的和特定的)所控制的,所有上级访问控制管理点指的是在 DIT 内从本条目一直向上朝着根的方向,期间所遇到的所有内部访问控制管理点或自治管理点,直到遇到第一个非内部访问控制管理点或自治管理点为止。作为此访问控制管理点上级的访问控制特定点,对给定条目的访问控制没有影响。

注 1:出于描述的目的,一个自治管理点被隐含地认为是一个访问控制特定点,即使它没有与任何预定的控制相关。

关于访问控制与管理区之间的关联关系的重要方面如下所列:

a) 目录信息的访问控制可能仅应用于所选择的条目,或者范围扩展至 DIB 的一部分,这一部分通过一个共同的安全策略和一个共同的访问控制管理在逻辑上相关联起来。

b) 访问控制可能施加在一个 ACSA 内或 ACIA 内的条目上,通过将属性prescriptiveACI (见 18.5)放置在相应的访问控制管理条目的一个或多个子条目内,且其范围由相应的 sub-

treeSpecification 来定义。

注 2：属性prescriptiveACI 不是集合属性。在prescriptiveACI 属性和集合属性之间有显著的不同：

——尽管一个prescriptiveACI 属性可能会影响该属性所在的子条目范围内每个条目的访问控制决策，但prescriptiveACI 属性不考虑为任何这样的条目提供可访问的信息，或者作为这些条目的有效组成部分；

——prescriptiveACI 属性与管理的访问控制方面相关，并且与访问控制特定点和内部点相关，但与条目集合的管理点不相关；

——prescriptiveACI 属性的目的是表示一个策略，该策略影响所定义的一个条目集；而一个集合属性的目的是提供信息，将用户可访问的属性集合与所定义的条目集合关联起来；

——prescriptiveACI 属性表示策略信息，一般来说，这些策略信息不能由普通用户普遍地访问。需要访问prescriptiveACI 信息的管理员用户能够访问此信息，如同访问子条目内的操作属性一样。

c) 操作属性prescriptiveACI 包含ACIItems（见 18.4.1），这些ACIItems 对于出现操作属性prescriptiveACI 的子条目范围内（即 DACD）的所有条目都是共同的。正常情况下，一个 DACD 内包含了相关的访问控制特定区内的所有条目（但也可以根本不包含任何一个条目）。

d) 尽管具体的ACIItems 可能会指定属性或属性值作为被保护项，但ACIItems 与条目在逻辑上是相关联的。与一个条目以及此条目的内容相关联的特定ACIItems 的集合是由如下组合而成的：

——应用于该具体条目的ACIItems，如果存在的话，作为操作属性entryACI 的值来指定（见 18.5.2）；

——操作属性prescriptiveACI 中的ACIItems，可应用于该条目，通过将其放置在管理条目的子条目内，而管理条目的范围包含了该具体的条目（见 18.5.1）。

e) 每个条目（由entryACI 和/或prescriptiveACI 所控制）有必要属于一个且仅属于一个 ACSA。每个这样的条目还可能属于一个或多个 ACSA 内嵌套的包含该条目的 ACIA。潜在地影响某个给定条目的访问控制决策结果的prescriptiveACI 被置于为 ACSA 和每个包含该条目的 ACIA 的（管理条目的）子条目内。其他子条目不能影响与此条目相关的访问控制决策。

f) 如果一个条目处于多个 DACD 的范围内，则那些可能潜在地影响该条目相关的访问控制决策的ACIItems 的全集中，除了包含条目自身的所有entryACI 属性外，还包含所在的 DACD 中的所有prescriptiveACI 属性，如图 17 所示。在条目 E1 中生效的访问控制是 DACD1、DACD2、DACD3 中的prescriptiveACI 以及条目 E1 中的entryACI（如果存在的话）的组合。在条目 E2 中生效的访问控制是 DACD1 和 DACD3 中的prescriptiveACI 以及条目 E2 中的entryACI（如果存在的话）的组合。

注 3：访问控制信息的保护在 18.6 描述。

g) 每个子条目中的属性subtreeSpecification 在某个管理区内定义了一个条目集合。由于一个subtreeSpecification 可能会定义一个子树精选，因此 DACD 可能会在它们各自管理区的交集中随意地重叠。为了简化，图 17 没有显示管理点、子条目或管理区；然而，它可被认为是在同一个 ACSA 中的三个 DACD，其中每个 DACD 都与该 ACSA 管理点的一个单独的子条目相对应（图中没有 ACIA）。换言之，图 17 还可被认为是在一个单独的 ACSA 的上下文中，该 ACSA 包含了一个单独的 ACIA，其中，DACD1 等同于 ACSA，DACD3 等同于 ACIA（DACD1 和 DACD2 应符合 ACSA 管理点的子条目，而 DACD3 应符合 ACIA 管理点的子条目）。当 DACD 内的条目集合与管理区隐含定义的子树中的条目集合相同时，则认为一个管理区与一个 DACD 是重叠的。见附录 M 中的示例，通过图形描述了管理条目、管理区、子条目以及 DACD 之间的关系。

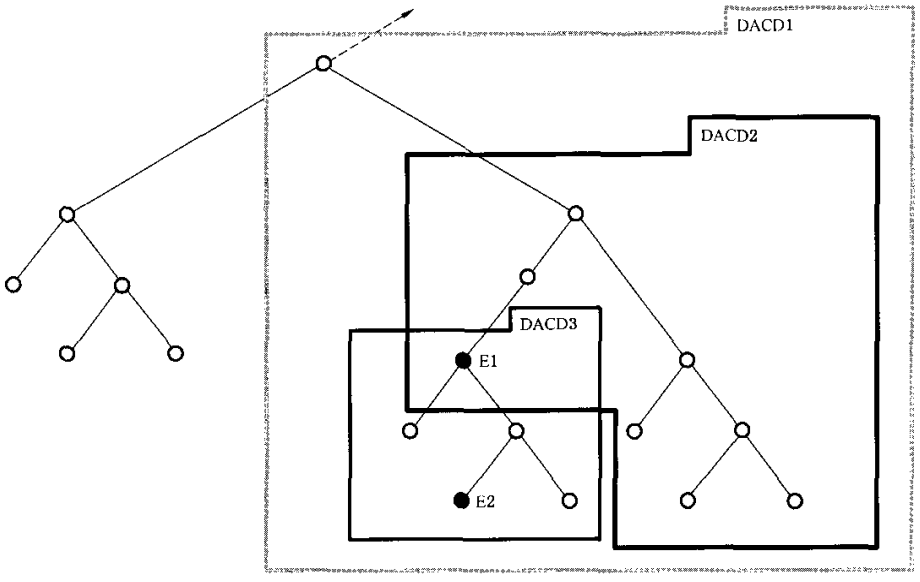


图 17 使用 DACD 的有效的访问控制

18.4 访问控制信息的表示

18.4.1 访问控制信息的 ASN.1

访问控制信息被表示为一个 ACIItem 的集合，其中，每个 ACIItem 准予或拒绝关于某些指定用户和被保护项的许可。

信息以 ASN.1 表示如下：

```
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            AttrributeType,
    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,
                --缺省值为 ACIItem 中的优先级
            userClasses             UserClasses,
            grantsAndDenials        GrantsAndDenials }
UserPermission  ::=SEQUENCE  {
            precedence              Precedence OPTIONAL,
                --缺省值为 ACIItem 中的优先级
            protectedItems          ProtectedItems,
            grantsAndDenials        GrantsAndDenials }

AuthenticationLevel  ::=CHOICE  {
            basicLevels  SEQUENCE  {
              level                 ENUMERATED { none (0),simple (1),strong (2) },
              localQualifier        INTEGER OPTIONAL,
              signed                BOOLEAN DEFAULT FALSE },
            other        EXTERNAL }
```

```
GrantsAndDenials  ::=BIT STRING  {
        --可能与被保护项的任何组件一起使用的许可
        grantAdd                (0),
        denyAdd                 (1),
        grantDiscloseOnError    (2),
        denyDiscloseOnError     (3),
        grantRead               (4),
        denyRead                (5),
        grantRemove             (6),
        denyRemove              (7),
        --只能与条目组件一起使用的许可
        grantBrowse             (8),
        denyBrowse              (9),
        grantExport             (10),
        denyExport              (11),
        grantImport             (12),
        denyImport              (13),
        grantModify             (14),
        denyModify              (15),
        grantRename             (16),
        denyRename              (17),
        grantReturnDN           (18),
        denyReturnDN            (19),
        --可能与被保护项的任何组件,除条目外,一起使用的许可
        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 ACIItem 参数的描述

18.4.2.1 标识标签

identificationTag 用于标识一个特定的ACIItem 。出于保护、管理和经营的目的,该参数用于区分不同的ACIItem 。

18.4.2.2 优先级

Precedence 用于控制ACIItem 的相对顺序,在做出访问控制决策的过程中,将根据此顺序考虑ACIItem ,如 18.8 所述。在其他方面都相同的情况下,具有高优先级值的ACIItem 可能会胜过其他具有低优先级值的ACIItem 。优先级值为整数,并可进行比较。

优先级可被安全管理机构内的上级机构使用，以便允许 ACSA 内设置的访问控制策略可以被部分授权。

这可以通过上级机构设置一个具有高优先级的通用策略，并授权代表下级机构的用户（如与某个 ACIA 相关）来创建和修改 ACI，使其具有较低优先级，以此为了某个具体的目的而将通用策略进行裁减。因此，部分授权要求上级机构限制最高优先级，而下级机构可以在上级机构的控制下将其分配给 ACI。

基本访问控制不会指定或描述如何限制下级机构可使用的最高优先级，这可以通过本地方式完成。

18.4.2.3 鉴别级别

AuthenticationLevel 定义了该ACIItem 所要求的最小的请求者安全级别。它有两种形式：

——basicLevels 指示了鉴别级别，可选地，通过正整数或负整数的localQualifier 值以及请求是否需要签名等来给出限定资格；

——other：一个外部定义的方法。

当使用basicLevels 时，一个包含了level 和可选的localQualifier 的AuthenticationLevel，必须根据本地策略由 DSA 分配给请求者。对于一个请求者而言，其鉴别级别达到或超出最小需求，指的是请求者的level 必须达到或超出在ACIItem 中指定的级别；另外，请求者的localQualifier 必须在算术上大于或等于在ACIItem 中指定的值。请求者的强鉴别是指超出简单鉴别或无鉴别要求的鉴别，而简单鉴别是指超出无鉴别要求的鉴别。出于访问控制的目的，“简单”鉴别级别要求提供一个口令；而只需要标识符，不需要提供口令的情况被认为是“无”鉴别级别。如果没有在ACIItem 中指定localQualifier，则请求者不必具有相应的值（如果有值存在，其值将被忽略）。除了要达到或超出上述要求外，如果ACIItem 中规定的signed 等于 TRUE，则请求还必须被签名。

当使用other 时，一个适当的AuthenticationLevel 必须根据本地策略由 DSA 分配给请求者。该AuthenticationLevel 的形式以及与 ACI 中的 AuthenticationLevel 进行比较的方法，属于本地事物。

注 1：与一个显式拒绝相关联的鉴别级别，指示了请求者为了不被拒绝访问应当被鉴别的最小级别。例如，对于一个拒绝访问某个特定用户类，并且要求强鉴别的 ACIItem 来说，如果请求者不能通过强鉴别的身份证明他们不属于此用户类，则 ACIItem 将拒绝此类所有请求者的访问。

注 2：DSA 可能以鉴别级别作为基础，而不采用其他从协议交互中获得的值。

18.4.2.4 itemFirst 和 userFirst 参数

每个ACIItem 都包含一个选择：itemFirst 或userFirst。该选择允许将许可分组，取决于许可根据用户类分组或被保护项分组哪个是最方便的。在获取访问控制信息方面，itemFirst 和userFirst 是一致的，它们都获取了相同的访问控制信息；但它们组织那些信息的方式是不同的。它们之间选择的根据是是否便于管理。用于itemFirst 或userFirst 中的参数描述如下：

a) ProtectedItems 定义了指定的访问控制所应用的项。它被定义为一个集合，从下述内容中选取：

——entry 表示作为一个整体的条目内容。若是一个家族成员，则它还意味着同一个复合属性内每个下级家族成员的条目内容。它不是必需要包含这些条目的信息。如果 classes 元素存在的话，该元素须被忽略，因为 classes 元素会基于它们的客体类来选择被保护的条目（及其下级家族成员）。

——allUserAttributeTypes 表示所有的与条目相关的用户属性类型信息，但不包含与这些属性相关的值。

——allUserAttributeTypesAndValues 表示所有的与条目相关的用户属性类型信息，包含所有用户属性的所有值。

——attributeType 表示与特定的属性相关的属性类型信息，但不包含与此属性类型相关的值。

——allAttributeValues 表示与特定属性相关的所有属性值信息。

——attributeValue 表示一个特定属性的一个特定值。

——selfValue 表示与当前请求者相关的属性值断言。被保护项selfValue 仅在当访问控制准备应用到一个特定的被鉴别用户的时候才应用。它能够仅应用于一个特定的情况下，即指定的属性具有DistinguishedName 或uniqueMember 句法，且指定属性的属性值与操作发起者的可辨别名相匹配。

注 1：*allUserAttributeTypes* 和*allUserAttributeTypesAndValues* 参数不包括操作属性，操作属性应当以每个属性为基础来规定，使用属性类型：allAttributeValues 或attributeValue 。

- **rangeOfValues 表示任何与指定的过滤器匹配的属性值，即指定的过滤器针对此属性值进行评估时，将返回 TRUE。**

注 2：过滤器不会对 DIB 中的任何条目进行评估；它在评估时使用 GB/T 16264.3—2008 的 7.8 定义的语义，在一个虚构的条目上进行操作，该虚构条目仅包含了一个作为被保护项的属性值。

下述各项提供了限制，这些限制可能会使得对同一序列中的被保护项的某个许可不被准予：

——maxValueCount 限制了一个特定的属性类型所允许的属性值的最大数量。如果被保护项是某个指定类型的属性值，且准许增加，则此时应当检查该限制值。条目中该属性类型的值被计数，而不考虑上下文或访问控制，并且假设增加属性值的操作是成功的。如果属性值的数量超出了*maxCount*，则该 ACI 项被视为*不准予进行增加访问*。

——maxImmSub 限制了被增加条目或被输入条目的上级条目的直接下级的最大数量。如果被保护项是一个条目，准许增加或输入，且其直接上级条目与被增加或输入的条目在同一个 DSA 内，则此时应当检查该限制值。上级条目的直接下级被计数，而不考虑上下文或访问控制，并且假设条目增加或输入操作是成功的。如果下级的数量超过了maxImmSub，则该 ACI 项被视为不准予进行增加或输入访问。

— restrictedBy 限制了加到该属性类型上的值应当是已经出现在同一个条目中的属性valuesIn 中的值。如果被保护项是某个指定类型的一个属性值，且准许增加，则此时应当检查该限制值。属性valuesIn 的值被检查，而不考虑上下文或访问控制，并且假设增加属性值的操作是成功的。如果要增加的属性值没有出现在valuesIn 中，则该 ACI 项被视为不准予进行增加访问。

——contexts 限制了加到条目中的值应当具有上下文列表，且此上下文列表满足contexts 中的所有上下文断言。如果被保护项是一个属性值，且准许增加，则此时应当检查该限制值。如果要加入的属性值不满足上下文断言，则该 ACI 项被视为不准予进行增加访问；如果属性值满足了所有的上下文断言，则该 ACI 项被视为不拒绝进行增加访问。

注 3：仅当许可被准予增加，且所有的上下文断言都被满足时，才相关。它不提供关于上下文的通用用法来将被保护项从其他许可中区分出来。

— classes 表示条目（可能是某个家族成员）的内容被限制为具有满足Refinement（见 12.3.5）所定义的谓词的客体类值，该条目内容与（祖先或其他家族成员）条目的内容一起作为每个下级家族成员条目的一个整体；它不是必需要包含这些条目中的信息。

注 4：根据为*entry* 和*classes* 所定义的规则，所有的家族成员继承了同一家族内的祖先或上级家族成员的访问控制。这并不排除家族成员符合entryACI 或prescriptiveACI 未来的增强或降低保护的策略。

b) UserClasses 定义了许可所应用的零个或多个用户的集合。用户集从下述内容中选取：

——allUsers 表示每个目录用户（具有可能的authenticationLevel 要求）；

— thisEntry 表示具有与正在被访问的条目相同的可辨别名的用户，或者如果条目是一个家族中的成员，则表示具有与祖先相同的可辨别名的用户；

——name 表示具有指定可辨别名的用户（可能具有可选的唯一标识符）；

——userGroup 表示用户的集合，该集合中的成员是groupOfUniqueNames 条目中的成员，由指定的可辨别名所标识（可能具有可选的唯一标识符）。具有唯一名（称）的组中的成员

被视为具有单独的客体名(称),而不是其他组的名(称)。如何判断组内成员在 18.4.2.5 中描述;

——subtree 是用户的一个集合,其可辨别名符合(未精选)子树的定义。

用于指定一个用户、分组或子树的名(称)必须是主辨别名。不得包括上下文和可替代可辨别值。不需要访问控制决策功能来判断主辨别名,而在可替代名(称)时需要提供该功能。

注 5:这就意味着如果一个请求者已经提供了一个可替换名,而此可替换名在后续未被目录解析为主辨别名,则基于主辨别名的访问控制在判断请求者是否属于准予或拒绝访问的用户类时,可能会失败。

c) SubtreeSpecification 用于指定相对于名为base 的根条目的一棵子树。base 代表子树根的可辨别名。子树将一直扩展到 DIT 的叶结点,除非在chop 中有其他规定。不允许使用组件 specificationFilter;如果该组件存在的话,它必须被忽略。

注 6: SubtreeSpecification 不允许子树精选,因为一棵子树精选可能会为了判断一个给定用户是否属于一个特定的用户类,而要求 DSA 使用分布式操作。在做出一个访问控制决策的过程中,基本访问控制被设计为避免进行远程操作。在定义中仅包括base 和chop 的子树中的成员可以进行本地评估,而在定义中使用了specificationFilter 的子树中的成员,仅可以通过获取用户条目的信息来进行评估,而这些用户条目潜在地处于其他 DSA 中。

d) ItemPermission 包括了一个与itemFirst 规范内的ProtectedItems 相关的用户及其许可的集合。这些许可在grantsAndDenials 中规定,如同本条 f)项所描述。出于评估 18.8 所讨论的访问控制信息的目的,在grantsAndDenials 中规定的每个许可被认为拥有 precedence 中指定的优先级。如果precedence 在ItemPermission 内被忽略,则优先级从为ACIItem 规定的precedence 中获取(见 18.4.2.2)。

e) UserPermission 包括了一个与userFirst 规范内的userClasses 相关的被保护项及其许可的集合。被保护项在protectedItems 中规定,如同 18.4.2 的讨论。相关的许可在grantsAndDenials 中规定,如同本条 f)项所描述。出于评估 18.8 所讨论的访问控制信息的目的,在grantsAndDenials 中规定的每个许可被认为拥有precedence 中指定的优先级。如果precedence 在UserPermission 内被忽略,则优先级从为ACIItem 规定的precedence 中获取(见 18.4.2.2)。

f) GrantsAndDenials 规定了在ACIItem 规范中被准予或拒绝的访问权限。这些与每个被保护项相关的许可的精确语义在 GB/T 16264.3—2008 中讨论。

g) *UniqueIdentifier* 可能被鉴别机制使用,用于辨别重用的可辨别名实例。唯一标识符的值由鉴别机构根据其策略来分配,并通过鉴别的 DSA 来提供。如果该字段存在,则对于一个正在访问的用户,要与一个准予访问的ACIItem 中的name 用户类相匹配,则除了要求用户的可辨别名应当与指定的可辨别名匹配外,还要求用户的鉴别产生一个相关的唯一标识符,该标识符应当与指定的值相等。

注 7:当鉴别是基于所提供的SecurityParameters 时,与用户相关的唯一标识符可能通过可选的 Certification-Path,从发送者的Certificate 的subjectUniqueIdentifier 字段中获取。

18.4.2.5 判断分组成员资格

判断一个给定的请求者是否是一个分组成员,要求检查两个标准。另外,如果分组的定义在本地未知,则判断可能会受到限制。成员资格的标准以及对非本地分组的处理讨论如下:

a) 出于基本访问控制的目的,不要求 DSA 执行一个远程操作来判断请求者是否属于一个特定的分组。如果分组内的成员资格无法判断,则当 ACI 项准予许可时,DSA 必须假设请求者不属于此分组,而当 ACI 项拒绝许可时,DSA 必须假设请求者属于此分组。

注 1:访问控制管理者应当注意下述情形:对非本地可用的分组内的成员资格进行的访问控制,或者对那些仅通过复制才可用的分组(因此,可能会过期)内的成员资格进行的访问控制。

注 2:从性能原因考虑,从远端 DSA 处获取分组成员资格作为访问控制判断的一部分,是不切实际的。然而,在某些情况下,它可能是可行的,且 DSA 也是允许的,例如,执行远端操作获取或恢复一个分组条目的

本地拷贝，或者在应用本条之前，使用比较操作来检查成员资格。

b) 为了判断请求者是否是一个userGroup 用户类中的成员，应用下述标准：

——由userGroup 规范命名的条目应当是客体类groupOfNames 或groupOfUniqueNames 的一个实例。

——请求者的名(称)必须是该条目的属性member 或uniqueMember 的一个值。

注 3：如果属性 member 或 uniqueMember 的值与请求者的名(称)不匹配，则这些值被忽略，即使它们表示分组的名(称)，而且能够发现发起者是该分组中的成员。因此，当对访问控制进行评估时，是不支持嵌套分组的。

注 4：用在 member 或 uniqueMember 中的名(称)必须是主辨别名。不得包括上下文以及带上下文的可替代值。

18.5 ACI 操作属性

访问控制信息，作为条目和子条目的一个操作属性存储于目录中。此操作属性是多值的，允许 ACI 被表示为一个ACIItems 的集合(在 18.4 中定义)。

18.5.1 预定的访问控制信息

一个预定的 ACI 属性被定义为某个子条目的一个操作属性。它包含了可应用于该子条目范围内的所有条目的访问控制信息：

```
prescriptiveACI  ATTRIBUTE  ::={
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-prescriptiveACI }
```

18.5.2 条目访问控制信息

一个条目 ACI 属性被定义为某个条目的一个操作属性。它包含了可应用于该条目以及该条目内容的访问控制信息：

```
entryACI  ATTRIBUTE  ::={
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-entryACI }
```

18.5.3 子条目访问控制信息

子条目 ACI 属性被定义为管理条目的一个操作属性，它提供了可应用于相应管理点下的每个子条目的访问控制信息。某个特定管理点的子条目内的PrescriptiveACI 永远不会应用到该管理点的同一个或任意一个其他的子条目，但可应用到其下级管理点的子条目。子条目 ACI 属性仅包含于管理点内，且除直接下级子条目外，将不会影响 DIT 内的其他任何元素。

在为某个特定的子条目进行访问控制评估时，必须考虑的 ACI 包括：

——子条目自身内的entryACI(如果有的话)；

——相关管理条目内的subentryACI(如果有的话)；

——在同一访问控制特定区内，与其他相关管理点关联的prescriptiveACI(如果有的话)。

```
subentryACI  ATTRIBUTE  ::={
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-subentryACI }
```

18.6 保护 ACI

ACI 操作属性可能与其他普通属性一样被保护机制所保护。一些相关要点为：

a) identificationTag 为每个ACIItem 提供了一个标识符。该标签可被用于删除一个特定的ACIItem 值，或者通过prescriptiveACI 或条目 ACI 来保护它。

注 1：目录规则确保每个访问控制属性仅有一个ACIItem 才拥有任何特定的identificationTag 值。

b) 为某个管理条目创建一个子条目可能会通过管理条目内的操作属性subentryACI 来进行访问控制。

注 2：创建prescriptive 访问控制的权限可能也是由安全策略直接控制的；在一个新的自治管理区内创建访问控制时需要该指配。

18.7 访问控制和目录操作

每个目录操作都包括对该操作所访问的各种被保护项做出一系列的*访问控制决策*。

对于某些操作(如修改操作)，为了保证操作能够成功，每个这样的访问控制决策都必须是准予访问；如果对任何一个被保护项的访问被拒绝，则整个操作都将失败。而对于另外一些操作，对某个被保护项的访问被拒绝仅仅是在操作结果中忽略，而操作还会继续。

如果被请求的访问被拒绝，可能还需要其他的访问控制决策来判断用户对被保护项是否具有DiscloseOnError 许可。仅当DiscloseOnError 许可被准予时，目录才可能在错误响应中显示此被保护项的存在；在其他所有情况下，目录都将隐藏此被保护项的存在。

对每个操作的访问控制要求，即被保护项以及访问每个被保护项时所需的访问许可，在 GB/T 16264.3—2008 中规定。

如何做出访问控制决策的算法在 18.8 规定。

18.8 访问控制决策功能

本条规定了如何做出对某个特定的被保护项的访问控制决策。它为basic-access-control 的访问控制决策功能(ACDF)提供了一个概念上的描述。描述了为了决定如何准予或拒绝某个请求者对一个给定被保护项的指定许可，ACI 项是如何被处理的。

18.8.1 输入和输出

对于 ACDF 的每次调用，输入参数包括：

a) 请求者的可辨别名(在 GB/T 16264.3—2008 的 7.3 定义)，唯一标识符，鉴别级别，或者其他可用的参数；

b) 在 ACDF 被调用的当前决策点正在考虑的被保护项(一个条目、一个属性或一个属性值)；

c) 在当前决策点指定的被请求的许可类别；

d) 与包含被保护项的条目相关联的 ACI 项(或本身就是被保护项)。被保护项在 18.4.2.4 中描述。在prescriptiveACI 属性内的 ACI 项的影响范围在 18.3.2 和 18.5.1 中描述。在entryACI 属性内的 ACI 项的影响范围在 18.3.2 和 18.5.2 中描述。在subentryACI 属性内的 ACI 项的影响范围在 18.5.3 中描述。

当条目是一个家族成员时，它还从同一家族内的祖先或上级家族成员那里继承了访问控制。这并不排除家族成员符合entryACI 或prescriptiveACI 未来的增强或降低保护的策略。

另外，如果 ACI 项包含任何在 18.4.2.4 中描述的被保护项限制时，则要求整个条目和其上级条目的直接下级的数量也应当作为输入参数。

输出是一个决策，表示是否*准予*或*拒绝*对被保护项的访问。

对于做出访问控制决策的任何特定实例，如果 18.8.2 到 18.8.4 的步骤被执行，则输出必须是相同的。

18.8.2 元组

对于 18.8.1 中 d)中描述的 ACI 项内的每个 ACI 值，将值扩展至一个元组(tuples)的集合，item-

Permissions 和userPermissions 集合中每个元素都对应一个元组。从所有的 ACI 值中收集所有的元组形成一个单独的集合。每个元组包含如下项：

(userClasses,authenticationLevel,protectedItems,grantsAndDenials,precedence)

对于任何一个在grantsAndDenials 中都既规定了准予又规定了拒绝的一个元组，可将其替换为两个元组，一个仅规定准予，而另一个仅规定拒绝。

18.8.3 放弃非相关的元组

执行下述步骤来放弃所有非相关的元组：

a) 如果元组中不包含在其中userClass (18.4.2.4 中 b)项中描述)中指定的请求者，则放弃所有这样的元组，如下所述：

——对于准予访问的元组，如果在元组的userClasses 元素(18.4.2.4 中 b)项中描述)中没有包括请求者身份，则放弃所有这样的元组，并且如果相关的话，还要考虑uniqueIdentifier 元素。当元组指定了一个uniqueIdentifier，且如果该元组没有被放弃，则匹配值应当出现在请求者的身份中。如果元组指定的鉴别级别高于与请求者相关的鉴别级别(根据 18.4.2.3)，则放弃此元组。

——对于拒绝访问的元组，如果在元组的userClasses 元素中包含了请求者，则保留所有这样的元组，并且如果相关的话，还需要考虑uniqueIdentifier 元素。如果元组被拒绝访问，且指定的鉴别级别高于与请求者相关的鉴别级别(根据 18.4.2.3)，则这样的元组也应当保留。所有其他拒绝访问的元组都被放弃。

注 1：上述第 2 个子项中的第二个需求(即如果元组被拒绝访问，且指定的鉴别级别高于与请求者相关的鉴别级别，则这样的元组也应当保留)反映了一个事实，即请求者不能充分地证明拒绝所指定的元组在用户类中不具有成员资格。

b) 如果元组没有在protectedItems (18.4.2.4 中 a)项)中包含被保护项，则放弃所有这样的元组。

c) 检查所有包含maxValueCount 、maxImmSub 、restrictedBy 或contexts 的元组。如果元组是准予访问，但不满足这些限制中的任何一项(18.4.2.4 中 a)项)，则放弃所有这样的元组。

d) 如果元组没有在grantsAndDenials (18.4.1 和 18.4.2.4 中 f)项)中对被请求的许可相应的比特进行了设置，则放弃所有这样的元组。

注 2：放弃非相关元组的执行顺序不会改变 ACDF 的输出。

18.8.4 选择最高优先级，最特殊的元组

执行下述步骤来选择具有最高优先级和特殊性的那些元组：

a) 如果元组拥有的优先级(precedence)低于剩余的最高优先级，则放弃所有这些元组。

b) 如果有多个元组被保留，则选择其中的具有最特殊用户类的元组。如果有任意一个元组与某个拥有UserClasses 中的元素name 或thisEntry 的请求者相匹配，则放弃其他所有的元组。否则，如果有任意一个元组与UserGroup 匹配，则放弃其他所有的元组。再否则，如果有任意一个元组与subtree 匹配，则放弃其他所有的元组。

c) 如果有多个元组被保留，则选择其中的具有最特殊被保护项的元组。如果被保护项是一个属性，并且有元组明确指定了该属性类型，则放弃其他所有的元组。如果被保护项是一个属性值，并且有元组明确指定了该属性值，则放弃其他所有的元组。如果被保护项是一个rangeOfValues，则被认为是明确指定了一个属性值。

当且仅当有一个或多个元组被保留且都准予访问时，才被准予访问。否则拒绝访问。

18.9 简化的访问控制

18.9.1 引言

本条描述了一个访问控制方案的功能，它被称为简化的访问控制，设计该访问控制是用来提供基本

访问控制功能中的一个子集。

18.9.2 简化的访问控制功能的定义

简化的访问控制功能定义如下：

a) 访问控制决策仅应当基于操作属性prescriptiveACI 和subentryACI 的ACIItem 值而做出。

注 1：如果存在entryACI 的话，不得用于做出访问控制决策。

b) 应当支持访问控制特定管理区。不得使用访问控制内部管理区。特定访问决策的做出应当基于从一个单独的管理点获取的 ACIItem 值，或者从该管理点的子条目中获取的ACIItem 值。

注 2：出现在管理点子条目中的属性prescriptiveACI 的值，如果不包含管理作用属性值id-ar-accessControl-SpecificArea，则不得用于做出访问控制决策。

c) 所有其他的指配都应当与为基本访问控制定义的一样。

19 基于规则的访问控制

19.1 范围和应用

本章为目录定义了一种特殊的访问控制方案(可能有多种)。这里定义的访问控制方案通过将操作属性accessControlScheme 赋值为rule-based-access-control 来进行标识，或者如果与第 18 章定义的基本访问控制方案或简化的访问控制方案联合使用时，则还要赋值为rule-and-basic-access-control 或rule-and-simple-access-control 。17.2.2 描述了哪些条目包含操作属性accessControlScheme 。

这里所定义的方案仅关系到对 DIB 内目录信息(潜在地包括树型结构和访问控制信息)的访问进行控制。它没有指出为了与某个 DSA 应用实体进行通信的目的而进行的访问控制。对信息的访问进行控制指的是防止对信息进行非授权的检测、泄漏或修改等。

19.2 基于规则的访问控制模型

可能有这样的环境，该环境中，在判断对某个属性值的访问是否被拒绝时，使用了与请求者的许可证(替代了身份)相关的信息。这被定义为基于规则的访问控制，并且在判断对目录的某些内容的访问是否被拒绝时，使用了所施加的访问控制策略规则。如果根据基于规则的访问控制，访问被拒绝，则在其他访问控制方案下也不被允许。基于规则的访问控制模型标识了用于判断访问是否被拒绝的信息。这些信息将被应用于每个操作。每个访问控制决策包含：

a) 与正在访问的属性值相关联的访问控制信息。这些访问控制信息被称为一个安全标签；

b) 与发出操作请求的用户相关联的访问控制信息。这些访问控制信息被称为许可证。发出操作请求的用户被称为请求者；

c) 一个规则，该规则定义了某个访问是否被授权给予一个安全标签和一个许可证。这样的规则被称为安全策略。

见图 18。

通过使用数字签名或其他完整性机制，可将标签与信息绑定起来，这样安全标签便与属性值安全地关联起来了。一个安全标签由属性值所有，并作为一个上下文与属性值关联起来。

在与安全标签进行比较时，需要用到许可证。通过一个证书扩展字段(属于目录的一个属性)或者通过一个属性证书，可以将许可证与请求者的可辨别名绑定起来。选择何种方法来提供许可证是属于当前生效的安全策略的内容。

注：其他许可证信息(如与任意一个可能会将操作链接的中间 DSA 相关的信息)的用法，不在本目录规范的定义范围之内。

在做访问控制决策时所应用的安全规则被定义为安全策略的一部分。安全策略或者在安全标签中标识，或者为包含已标注客体的环境而定义。

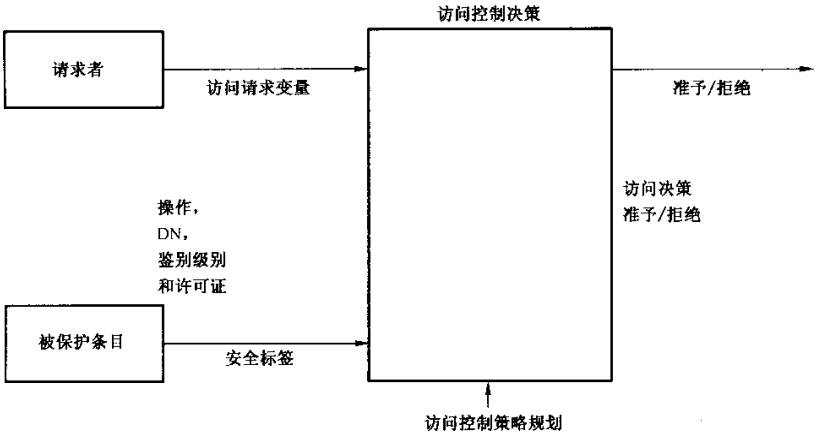


图 18 基于规则的访问控制决策模型

19.3 访问控制管理区

对于基本访问控制(18.3),DIT 被划分为管理区,包括访问控制特定区(ACSA)。一个 ACSA 的管理条目标识了如何为应用于该管理区的安全策略(访问规则)进行标注以及如何为可应用的访问控制方案(rule-based-access-control 或 rule-and-basic-access-control 或 rule-and-simple-access-control 或一些其他的访问控制方案)进行标注。

19.4 安全标签

19.4.1 引言

安全标签可能会用于将安全相关的信息与目录内的属性关联起来。

安全标签可能会分配给某个属性值,并且符合该属性所应用的安全策略。安全策略还可能会定义如何使用安全标签来执行安全策略。

一个安全标签由一个元素集组成,可选地包括:一个安全策略标识符,一个安全分类,一个私有标记以及一个安全类别集等。通过使用数字签名或其他完整性机制,将安全标签与属性值绑定起来。

19.4.2 安全标签的管理

在将一个属性值放置在目录内之前,通过一个管理功能将安全标签分配给该属性值。

该管理功能负责将安全标签分配给属性值,并且符合 ACSA 所应用的安全策略。

安全标签的绑定通过使用数字签名或其他完整性机制来保护。这种保护由管理功能或者属性值的创建者来提供。

19.4.3 加标签的属性值

一个安全标签上下文将安全标签与一个属性值关联起来。一个属性值仅能够与一个单独的标签相关联。

也就是说,安全标签上下文是单值的。另外,不支持安全标签上下文的匹配规则。

注:上下文的概念在 8.8 中介绍。

```
attributeValueSecurityLabelContext CONTEXT ::={
        WITH SYNTAX        SignedSecurityLabel    -- 一个属性值最多可以分配一个安全标签
                                                  上下文
        ID                 id-avc-attributeValueSecurityLabelContext }

SignedSecurityLabel  ::=SIGNED {SEQUENCE  {
        attHash            HASH {AttributeTypeAndValue},
```

```
        issuer              Name       OPTIONAL,        ——加标签的机构的名(称)
        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 ( {——无,至少须有一个组件存在——} ) )

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
```

该上下文同其他上下文不一样,不是用于过滤或选择特定的属性,而且不使用与上下文相关的机制(回退、缺省上下文值等)来提供基于规则的访问控制。

组件attHash 包含了对 DER-编码八比特组应用密码散列过程后的结果值,如 ISO/IEC 9594-8 中的定义。

组件issuer 承载了加标签的机构的名(称)。

组件keyIdentifier 可能是一个已经鉴别的公钥的标识符,与存储在 ITU-T X.509 建议书 | ISO/IEC 9594-8 中定义的主体公开密钥标识符扩展域中的相同,或者是一个对称密钥标识符以及相关的安全控制信息。

组件securityLabel 由一个元素集组成,可选地包括:一个安全策略标识符,一个安全分类,一个私有标记以及一个安全类别集等,在 ISO/IEC 10021-4 的 8.5.9 中定义。

19.5 许可证

许可证属性将一个许可证与一个已命名的实体(包括 DUA)关联起来。

```
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 ::={ ... }
```

组件policyId 承载了一个标识符，该标识符可能用于标识与许可证的classList 和securityCategories 相关联的有效的安全策略。

组件classList 包含了一个与已命名的实体相关联的分类列表。

组件securityCategories（见 ISO/IEC 10021-4:2003 的 8.5.9），如果存在的话，在classList 的上下文内提供了额外的限制。

注：将许可证与一个已命名实体安全地绑定起来，可以通过使用一个属性证书（见 ISO/IEC 9594-8），一个公钥证书扩展域（例如，在SubjectDirectoryAttribute 扩展内）（见 ISO/IEC 9594-8），或者通过使用本目录规范定义之外的方法。

19.6 访问控制和目录操作

每个目录操作都包括对该操作所访问的属性值做出一系列的访问控制决策。

对于某些操作（如删除条目操作），如果有一个或多个属性值的访问被拒绝，则即使访问操作可能会成功，但隐藏的属性还会保留在目录中。而对于另外一些操作，如果某个被保护项的访问被拒绝，则仅仅是在操作结果中被忽略，而操作还会继续。

每个操作的访问控制需求在 GB/T 16264.3—2008 中规定。

可以做出任何具体访问控制决策的算法规定如下：

——如果根据rule-based-access-control，对某个条目的所有属性值的访问都被拒绝，则对该条目所有操作的访问都将被拒绝；

——如果根据rule-based-access-control，对某个属性的所有属性值的访问都被拒绝，则对该属性所有操作的访问都将被拒绝；

——基于规则的访问控制会影响关于属性值阅读的操作（如阅读、搜索操作），如果对该属性值的访问被拒绝，则这些属性值将不可见（即操作被执行，就好像属性值不存在一样）；

——基于规则的访问控制会影响关于删除条目的操作（如删除条目操作），如果对属性值的访问被拒绝，则这些属性值不能被删除；

——基于规则的访问控制会影响关于删除属性类型的操作（如修改条目中的删除属性），如果对属性值的访问被拒绝，则这些属性值不能被删除；

——基于规则的访问控制会影响关于删除属性值的操作（如修改条目中的删除属性值），如果对属

性值的访问被拒绝，则这些操作将失败。

19.7 访问控制决策功能

本条规定了针对某个特定属性值的访问控制决策是如何做出的。它为rule-based-access-control 的访问控制决策功能(ACDF)提供了一个概念上的描述。描述了为决定是否准予或拒绝某个请求者对一个给定属性值的指定许可，许可证和安全标签是如何被处理的。决策功能应用了安全策略规则，该规则根据安全标签和请求者证书确定了针对某个属性值的一个访问是否被授权。安全规则的定义不在本目录规范的定义范围之内。一个安全策略规则的简化示例在 M.10 中给出，该安全策略规则是为rule-based-access-control 而定义的。

对于 ACDF 的每次调用，输入参数有：

a) 请求者的许可证(如 19.5 的定义)；

b) 在 ACDF 被调用的当前决策点正在考虑的属性值；

c) 访问控制特定区内生效的安全策略；

d) 与属性值绑定的安全标签。

输出是一个决策，表示是否拒绝对属性值的访问。

对于做出访问控制决策的任何特定实例，如果 19.6 的步骤被执行，则输出须是相同的。

19.8 基于规则的访问控制和基本访问控制的用法

如果基于规则的访问控制和基本的访问控制两种都生效，则它们的应用顺序属于本地事物，除非如果根据任意一个机制，对条目、属性类型或属性值的访问被拒绝的话，则根据另一个机制，该访问也不得被准予。在这种情况下，basic-access-control 中的*DiscloseOnError*（见 18.2.3 和 18.2.4)的许可不得优先于rule-based-access-control 做出的拒绝决定。

20 存储时的数据完整性

20.1 引言

在某些情况下，目录可能不会给出足够的保障来确保数据在存储时是不变的，而不考虑访问控制。

存储在目录中的数据的完整性可能会通过作为目录信息的一部分的数字签名来验证。有两种情况，一种情况是：条目或条目内所选择属性的数字签名，可能会作为一个属性而存储(见 20.1.2)；另一种情况是：单个属性值的数字签名可能会作为一个上下文而存储(见 20.1.3)。

注 1：DSA 必须维护目录内属性的编码，以确保在返回结果中计算的签名是正确的。

注 2：属性值的保密性不在本目录规范的定义范围之内。

20.2 条目或所选属性类型的保护

所存储属性的数据完整性通过使用伴随着被保护属性的数字签名来提供。整个条目的完整性或条目内所选属性的所有属性值的完整性，是通过一个属性来保护的，该属性拥有被保护的所有属性值的数字签名。

该数字签名由管理机构创建，或者由负责将信息放置到目录条目中的目录用户创建。数字签名可由阅读该条目属性值的任何用户来验证。在对属性所拥有的数字签名进行创建和验证的过程中，目录服务本身不包括在内。

该完整性机制既保护了存储中的目录属性的完整性，又保护了在目录不同组件(DSA 和 DUA)间传输过程中的目录属性的完整性。该完整性机制不依赖于目录服务本身的安全。

应用于整个条目的数字签名不包括操作属性、集合属性或attributeIntegrityInfo 本身。包括任何属性值上下文。

下面定义了一个拥有一个数字签名的属性类型以及相关的控制信息，这些信息提供了整个条目或条目内所选属性的所有属性值的完整性。

```
attributeIntegrityInfo ATTRIBUTE  ::={
```

```
        WITH SYNTAX     AttributeIntegrityInfo
        ID              id-at-attributeIntegrityInfo}

AttributeIntegrityInfo  ::=SIGNED { SEQUENCE  {
        scope           Scope,                  ——标识了被保护的属性
        signer          SignerOPTIONAL,         ——管理机构或数据发起方的名(称)
        attribsHash     AttribsHash } }         ——被保护属性的散列值

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,——签名保护该条目内的所有属性值
        selectedTypes   [1]     SelectedTypes
                                    ——签名保护所选属性类型的所有属性值
        }

SelectedTypes  ::=SEQUENCE SIZE (1..MAX) OF AttributeType

AttribsHash  ::=HASH { SEQUENCE SIZE (1..MAX) OF Attribute }
        ——在所选择范围内的属性类型和属性值以及相关的上下文值
```

AttributeIntegrityInfo 的值有三种不同的创建方法：

a) 管理机构能够创建并对其赋值，且用于验证签名的公钥是通过脱机方式获知的；

b) 条目的拥有者，即条目所代表的客体，能够创建并对其赋值。如果拥有者有多个证书，或者将来可能拥有多个证书，则证书必须通过发布证书的 CA 以及证书序列号来共同标识；

c) 第三方可能创建并对其赋值。此时，需要签名者的名(称)、发布证书的 CA 名(称)以及证书序列号。

如果范围是wholeEntry，则所有可应用的属性必须按照 ISO/IEC 9594-8 中 6.1 的集合类型的规定来排序。如果范围是selectedTypes，则必须同SelectedTypes 中给定的顺序一样来排序。

注：如果用户不获取Scope 数据类型所定义的所有完整属性，则对于用户来说，验证属性的完整性是不可能的。

20.3 单个属性值保护的上下文

下面定义了一个拥有一个数字签名的上下文以及相关的控制信息，这些信息提供了一个单个属性值的完整性。在完整性检查中，应包括除了用于存储签名的上下文外的任意属性值上下文。

```
attributeValueIntegrityInfoContext   CONTEXT   ::={
        WITH SYNTAX          AttributeValueIntegrityInfo
        ID                   id-avc-attributeValueIntegrityInfoContext }

AttributeValueIntegrityInfo   ::=SIGNED { SEQUENCE {
        signer       Signer    OPTIONAL,        ——管理机构或数据发起者的名(称)
        aVIHash      AVIHash } }                 ——被保护属性的散列值

AVIHash   ::=HASH { AttributeTypeValueContexts }
          ——属性类型和值以及相关的上下文值

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

contextList 的排列顺序必须按照 ISO/IEC 9594-8 中 6.1 的集合类型的规定来排序。

第九篇：DSA 模型

21 DSA 模型

本章关注的是一个通用模型，描述了组成目录的组件，即目录系统代理(DSA)的不同方面。后续章条论述了附加的 DSA 模型。

21.1 定义

本目录规范使用下列术语和定义：

21.1.1

DIB 片段 DIB fragment

DIB 的一部分，由一个主 DSA 拥有，包含一个或多个命名上下文。

21.1.2

上下文前缀 context prefix

RDN 的一个序列，该序列从 DIT 的根开始一直到一个命名上下文的初始顶点结束；相当于该顶点的可辨别名。

21.1.3

命名上下文 naming context

条目的一棵子树，由一个单独的主 DSA 拥有。

21.2 目录功能模型

目录体现为一个或多个应用进程的集合，该应用进程被称为*目录系统代理(DSA)*和/或*LDAP 服务器*。每个 DSA 都提供零个、一个或多个访问点。每个 LDAP 服务器都提供一个或多个访问点。如图

19 所示。图中，目录由多个 DSA 或 LDAP 服务器组成，该目录被称为是*分布式*的。分布式目录的操作流程在 GB/T 16264.4—2008 中规定。

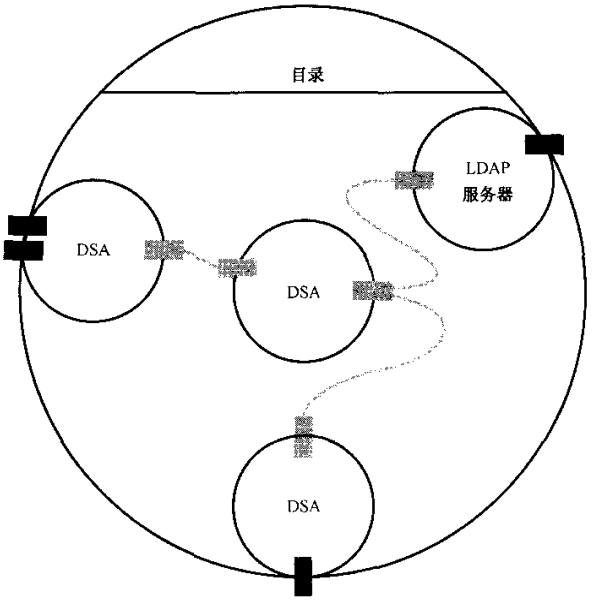


图 19　由多个 DSA 提供的目录

注 1：一个 DSA 可能会具有本地的行为特性以及结构，这些不在目前的目录规范的定义范围之内。例如，一个负责拥有 DIB 中某些或所有信息的 DSA，可能会通过数据库拥有这些本地特性，但与数据库的接口属于本地事物。

在目录服务的指配中需要进行交互的特定的应用进程对，可能会处于不同的开放系统中。这样的交互可通过目录协议来承载，目录协议可以是 GB/T 16264.5—2008 中规定的协议，也可以是 IETF RFC 3377 中规定的轻量级目录访问协议(LDAP)。

注 2：LDAP 服务器的行为在 IETF RFC 3377 中规定，可能与本条所规定的 DSA 行为不同。

第 23 章规定了模型，这些模型可作为规定目录分布特性的基础。关于目录组件(即 DSA)操作的各种特定方面的操作模型规范框架分别在第 25 章到第 28 章提供。

21.3　目录分布模型

本条定义了原则，根据这些原则，DIB 可以跨多个 DSA 分布。

注 1：DIB 也可能是跨多个 LDAP 服务器来分布，LDAP 服务器可能与一个或多个 DSA 共存，也可能不与 DSA 共存。LDAP 服务器及其特性和行为在 IETF RFC 3377 中规定，可能与本条所规定的 DSA 的特性和行为不同。

DIB 内的每个条目都应当由一个且仅由一个 DSA 的管理者来管理，该 DSA 的管理者被称为拥有该条目的主管权力。条目的维护和管理必须在一个由此条目的管理机构来管理的 DSA 内执行。该 DSA 被称为此条目的主 DSA。

目录内的每个主 DSA 都拥有 DIB 的一个*片段*。主 DSA 所拥有的 DIB 片段由术语 DIT 来描述，并包含一个或多个*命名上下文*。一个命名上下文是 DIT 的一棵子树，其上的所有条目都有一个共同的管理机构，并由同一个主 DSA 所拥有。一个命名上下文起始于 DIT 的一个顶点(非根顶点)，并向下扩展至叶顶点或非叶顶点。这些顶点组成了命名上下文的边界。一个命名上下文的起始顶点的上级不被该主 DSA 所拥有，而属于命名上下文边界的非叶顶点的下级则表示了另一个命名上下文的开始。

注 2：因此，DIT 被分割为多个不相交的命名上下文，每个命名上下文都在一个单独的主 DSA 的管理机构的管理之下。

注 3：一个命名上下文本身并不是一个拥有管理点或者明确的子树规范的管理区，但它可能与一个管理区是一

致的。

条目的家族必须处于一个单独的命名上下文内。

对于主 DSA 的管理者而言，它可能会拥有多个不相交的命名上下文的主管权力。对于由主 DSA 所主管的每一个命名上下文，它必须在逻辑上拥有一个 RDN 的序列，该序列从 DIT 的根开始，一直向下直到组成该命名上下文子树的初始顶点。这个 RDN 的序列被称为该命名上下文的*上下文前缀*。

注 4：命名上下文的主辨别名必须被用做上下文前缀。上下文以及伴随上下文的替代值可能会包含在 RDN 中。

一个主 DSA 的管理者可能会将本地所拥有的任何条目的任何直接下级的主管权力授权给另外的主 DSA。一个授权的主 DSA 被称为一个*上级 DSA*，而主管权力被授权的那个上级条目的上下文被称为*上级命名上下文*。主管权力的授权可以始于根顶点，并且在 DIT 内一直向下，也就是说，它仅能够出现在从一个条目到其下级。

图 20 举例说明了一个假定的 DIT，该 DIT 在逻辑上被分割为五个命名上下文（它们被称为：A、B、C、D 和 E），每个上下文都在物理上分布于三个 DSA 上（DSA1、DSA2 和 DSA3）。

从例中可知，由某个特定主 DSA 所拥有的命名上下文可能会被配置为符合某个大范围的操作需求。这些主 DSA 可能会被配置为拥有这样一些条目，这些条目在 DIB 的某些逻辑部分（如一个大公司的组织结构）内表示高层命名域，但对所有的下级条目而言不是必需的。替代的，主 DSA 可能会被配置为仅拥有表示主叶条目的命名上下文。

从上述的定义中，一个命名上下文的限制情况可以或者是一个单个条目，或者是整个 DIT。

同时，从逻辑的 DIT 到物理的主 DSA 的映射，潜在来说是任意的，如果主 DSA 被配置为拥有少量的命名上下文，则信息定位和管理的任务可被简化。

DSA 在拥有条目的同时，还可能会拥有条目复制。影像条目是在本系列目录规范中被考虑的唯一一种条目复制种类，对其的维护是通过在 ISO/IEC 9594-9 中描述的影像服务来实现的。

除了这种标准的复制信息种类外，在目录中还可能会遇到另外两种非标准的条目拷贝种类。

——条目的拷贝可能会通过双边协议存储于另外的 DSA 中。

——条目的拷贝可能会通过（本地或动态）存储一个条目的高速缓存拷贝而获取，该高速缓存由一个请求而产生。

注 5：这些拷贝的维护和管理方式不在本系列目录规范的定义范围之内。由于一些更精确的处理特性，如访问控制，建议使用影像服务替代高速缓存拷贝。

拥有一个条目拷贝的 DSA 被称为该条目的一个*影像 DSA*。一个影像 DSA 可能会拥有一个命名上下文的拷贝或其中一部分的拷贝。被影像的命名上下文的一部分，术语称之为一个“*复制单元*”。

正如在 ISO/IEC 9594-9 中 9.2 所描述的，一个复制单元被定义在目录信息模型内，并且提供了一个规范机制。目录中的影像机制是以将被影像的 DIT 子集的定义为基础的。该子集被称为*复制单元*。复制单元由一个包含三部分的规范组成，分别定义了被复制的 DIT 部分的范围、此范围内将被复制的属性以及对下级知识的需求等。复制单元还隐含地使影像信息包含策略信息，该策略信息以操作属性的形式存在于条目或子条目中（如访问控制信息等），将被用于正确地执行目录操作。被包含的前缀信息始于一个自治管理点，并且扩展至复制基条目。

目录请求的发起者将被通知到（通过fromEntry）：为响应请求而返回的信息是否是来自条目拷贝。

定义了一个服务控制dontUseCopy，可以允许用户阻止使用条目拷贝来满足请求（尽管拷贝信息可能会被用于进行名（称）解析）。

注 6：在某些实例情况下，针对合法替换名的名（称）解析将失败，一种是拥有拷贝的 DSA 是按照第 3 版之前的规范实现的 DSA；另一种是拥有拷贝的 DSA 是按照第 3 版的后续版本实现的，但是该 DSA 拥有的拷贝具有不完整的名（称）信息，即 RDN 包含一个属性类型，而该属性类型有多个由上下文所区分的可辨别值。

对于 DUA 而言，为了能够开始处理一个请求，它必须拥有一些关于它能够初始接触的至少一个 DSA 的信息，特别是表示地址的信息。这些信息是任何获取的，并且是如何保存的，属于内部事物。

在修改条目的处理过程中，有可能目录会变得不一致。尤其是当修改包含了可能处于不同 DSA 内的别名或别名客体时。这种不一致必须通过特定的管理动作进行纠正，例如：如果相应的被起别名的客体被删除，则别名也必须被删除。在这种不一致的过程中，目录还应当继续操作。

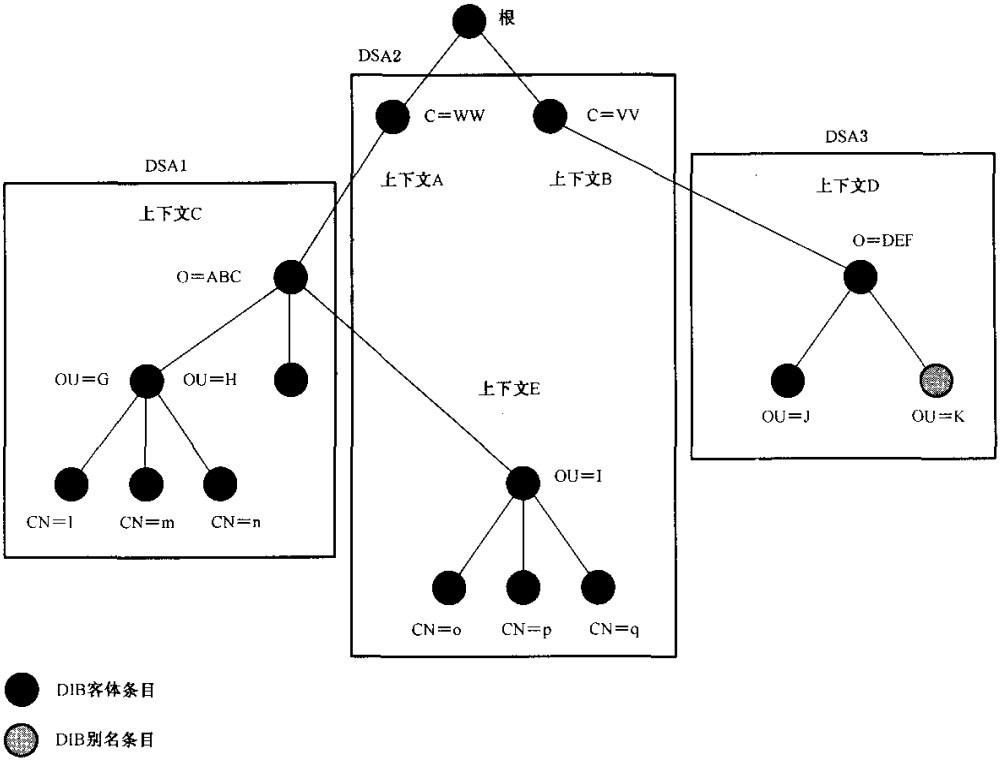


图 20 假设的 DIT

第十篇：DSA 信息模型

22 知识

22.1 定义

本目录规范使用下列术语和定义：

22.1.1

种类 category

知识引用的一种特性，被限定为用于标识一个 DSA 是主 DSA 还是影像 DSA。

22.1.2

公共可用的 commonly usable

复制区的一种特性，允许拥有它的 DSA 访问点可以进行一般性的分布；一个公共可用的复制区一般来说是一个命名上下文的完整的影像拷贝。

22.1.3

交叉引用 cross reference

一种知识引用，包含了关于一个拥有某个条目或条目拷贝的 DSA 的信息。它用于优化。该条目与

拥有交叉引用的 DSA 内的任何条目之间都应当没有上级或下级关系。

22.1.4

DIT 桥接知识引用　DIT bridge knowledge reference

一种知识引用，包含了关于一个拥有不同 DIT 内条目的 DSA 的信息。该条目与拥有其他 DIT 的 DSA 内的任何条目之间都应当没有上级或下级关系。

22.1.5

直接上级引用　immediate superior reference

一种知识引用，包含了关于一个拥有命名上下文（或一个从它派生而来的公共可用的复制区）的 DSA 的信息，且此命名上下文是知识引用相关的 DSA 所拥有的命名上下文的直接上级。

22.1.6

知识（信息）　knowledge（information）

某个 DSA 所拥有的 DSA 操作信息，DSA 使用此信息来定位远端条目或条目拷贝信息。

22.1.7

知识引用　knowledge reference

将一个 DIT 条目或条目拷贝与其所在的 DSA 直接或间接相关联起来的知识。

22.1.8

主知识　master knowledge

某个命名上下文的主 DSA 的知识。

22.1.9

非特定下级引用　non-specific subordinate reference

一种知识引用，包含了关于一个拥有一个或多个非特定下级条目或条目拷贝的 DSA 的信息。

22.1.10

引用路径　reference path

知识引用的一个连续的序列。

22.1.11

影像知识　shadow knowledge

某个命名上下文（如果知识是特定的）或某个上下文（如果知识是非特定的）的一个或多个影像 DSA 的知识。

22.1.12

下级引用　subordinate　reference

一种知识引用，包含了关于一个拥有一个特定下级条目或条目拷贝的 DSA 的信息。

22.1.13

上级引用　superior reference

一种知识引用，包含了关于一个被认为能够解析整个 DIT（即发现 DIT 内的任何条目）的 DSA 的信息。

22.2　引言

DIB 可分布在大量的主 DSA 内，每个主 DSA 都拥有一个 DIB 片段，且对其具有主管权力。控制分布的原则在 21.3 规定。

另外，这些 DSA 以及其他 DSA 还可能会拥有 DIB 部分的拷贝。

对目录有一个需求，即对于用户交互的某些特定方式，目录的分布应当是透明的，因此给出的效果是在每一个 DSA 内，DIB 都是完整出现的。

为了支持该操作需求，有必要让每个 DSA 都能够获得对 DIB 内所拥有的与任意名（称）相关的信息的访问（即任何客体的可辨别名或别名）。如果 DSA 本身没有拥有与名（称）相关的客体条目或客体

条目拷贝，则它必须能够与另一个拥有这些信息的 DSA 进行交互，与其他 DSA 的交互可以通过直接的或间接的方式进行。

当目录用户指出不得使用条目拷贝信息来满足其需求时，则服务于该请求的 DSA 必须能够获得对相应主 DSA 的访问，这些主 DSA 拥有与用户请求中所提供的名(称)相关的条目信息，访问可以是直接的或间接的。

本条定义了知识，DSA 操作信息需要这些知识来达到它们的技术目标。后续的部分规定了知识在一个通用 DSA 信息模型上下文中的表示。

注：前述的陈述表达了目录的技术目标。这些技术目标的实现除了依赖 DSA 内知识的一致性配置外，还依赖于其他问题(如策略问题)。第 25 章至第 28 章为处理这些问题，建立了一个框架。

附录 O 包含了一个对知识进行建模的示例。该示例基于图 20 中给出的一个假设 DIT。

22.3 知识引用

*知识*是某个 DSA 所拥有的操作信息，表示其他 DSA 内所拥有的条目信息或条目拷贝信息的局部描述。知识由 DSA 使用，当 DSA 本地存储的信息不能满足从 DUA 或其他 DSA 发来的请求时，DSA 使用知识来判断与哪个适当的 DSA 进行交互。

知识由*知识引用*组成。一个知识引用将目录条目的名(称)与一个拥有该条目或条目拷贝的 DSA 相关联起来，或者是直接的，或者是间接的。

在知识引用中所使用的名(称)，无论是上下文前缀，DSA 名(称)或条目名，都须是主可辨别值。上下文以及伴随了上下文的替代值等，也可能包含在 RDN 中。

注：在两种情况下，针对合法替换名的名(称)解析将失败，一种是拥有知识引用的 DSA 是按照第 3 版之前的规范实现的 DSA，它不能够辨别由不同上下文所区分的多个可辨别值；另一种是拥有知识引用的 DSA 没有在知识引用或条目拷贝中包含所有的替代辨别名。

22.3.1 知识种类

有两种知识引用的种类：主知识引用和影像知识引用。

*主知识*是某个命名上下文的主 DSA 访问点的知识。

*影像知识*是拥有复制的目录信息的 DSA 的知识；它可能会通过复制过程由影像提供者分布到影像使用者处，复制过程在 ISO/IEC 9594-9 中描述。影像知识是一个复制区(一个命名上下文或其中的一部分)内一个或多个影像 DSA 集合的访问点的知识。

称一个 DSA 是影像知识的客体，则它须拥有一个公共可用的复制区。公共可用复制区的形式之一是一个命名上下文的完整影像拷贝。某个 DSA 所拥有的一个命名上下文的非完整影像拷贝也可能是公共可用的，如果它能足够完整地满足用户一般情况下向 DSA 发出的查询请求。如果管理机构将拥有一个命名上下文非完整拷贝的 DSA 的影像知识分布起来，则确保复制区是公共可用的应当是管理机构的责任。

一个给定的 DSA 可能既拥有主知识，又拥有影像知识，拥有影像知识的 DSA 包含多个与特定命名上下文相关的影像 DSA。在对从 DUA 或另一个 DSA 发来的请求进行处理的过程中(如在名(称)解析过程中)所使用的特定知识，由一个 DSA 特定选择过程来决定的，根据此过程，并基于管理机构所认为适当的任何非标准的条件，由 DSA 计算得出在某个 DSA 内的某个有能力来处理该请求的访问点。

注：本系列目录规范不限制 DSA 如何使用主知识和影像知识(不同于间接地对 DSA 行为的限制，例如在 GB/T 16264.3—2008 中所规定的dontUseCopy 和copyShallDo 服务控制)。

22.3.2 知识引用类型

由 DSA 所处理的知识使用一个或多个知识引用的集合来进行定义，每个知识引用将 DIB 的条目(或条目拷贝)与拥有这些条目(或条目拷贝)的 DSA 相关联起来，通过直接的或间接的方式。

一个 DSA 可能会拥有下述类型的知识引用：

——上级引用；

——直接上级引用；

——下级引用；

——非特定下级引用；以及

——交叉引用。

一个特定类型的知识引用应当或者是一个主知识引用，或者是一个影像知识引用。

另外，参与影像的 DSA 可能是一个影像提供者和/或一个影像使用者，都拥有一个或多个下述类型的知识引用：

——提供者引用；和

——使用者引用。

这些知识引用类型在后续部分描述。

22.3.2.1 上级引用

一个上级引用包含：

——一个 DSA 的访问点。

每一个非第一级别的 DSA(见 22.5)都必须至少维护一个上级引用。上级引用必须构成到根的引用路径的一部分。除非在 DMD 内实施了一些非标准的方法来确保上述要求，否则上述要求的实现都是通过引用一个 DSA 来实现的，该 DSA 拥有一个命名上下文或复制区，其复制区的上下文前缀所具有的 RDN 比拥有该引用的 DSA 所拥有的具有最少 RDN 的上下文前缀的 RDN 还要少。

22.3.2.2 直接上级引用

一个直接上级引用包括：

——一个命名上下文的上下文前缀，是拥有该引用的 DSA 所拥有的条目或条目拷贝的直接上级；

——拥有该命名上下文(如条目或条目拷贝)的 DSA 的访问点。

直接上级引用是一种可选的引用类型，仅当一个层次结构的操作绑定到被引用的 DSA 时才出现该类型(见 GB/T 16264.4—2008 的第 24 章)。如果没有显式地出现这样的操作绑定，则直接上级命名上下文的引用可能会通过一个交叉引用来实现。

22.3.2.3 下级引用

一个*下级引用*包括：

——一个上下文前缀，该上下文前缀是与拥有该引用的 DSA 所拥有的条目或条目拷贝的直接下级的命名上下文相应的；

——拥有该命名上下文(如条目或条目拷贝)的 DSA 的访问点。

主 DSA 所拥有的命名上下文的所有直接下级命名上下文，都须由下级引用来表示(或是由非特定下级引用来表示，在 22.3.2.4 中描述)。

在 DSA 拥有条目拷贝的情况下，下级命名上下文可能表示也可能不表示，这取决于所生效的影像商定。

22.3.2.4 非特定下级引用

一个*非特定下级引用*包含：

——拥有一个或多个直接下级命名上下文内的条目(或条目拷贝)的 DSA 的访问点。

该类型的引用是可选的，允许用于下面这种情况，即已知 DSA 包含某些下级条目(或条目拷贝)，但这些条目(或条目拷贝)的特定 RDN 未知。该类型的引用不能被用于引用中 LDAP 服务器。

对于一个主 DSA 所拥有的每个命名上下文，可能会拥有零个或多个非特定下级引用。通过一个非特定引用被访问的 DSA 须能够直接实现请求(成功或者失败)。在失败的情况下，将通过serviceError 上报问题unableToProceed，并将其返回给请求者。

在 DSA 拥有条目拷贝的情况下，非特定下级引用可能会表示也可能不表示，依赖于所生效的影像商定。

22.3.2.5 交叉引用

一个交叉引用包含：

——一个上下文前缀；

——一个拥有此命名上下文内的条目或条目拷贝的 DSA 的访问点。

该类型的引用是可选的，用于优化名(称)解析。一个 DSA 可能会拥有任意数量(包括零)的交叉引用。

22.3.2.6 提供者引用

影像使用者 DSA 所拥有的一个提供者引用包括：

——一个命名上下文的上下文前缀，从影像提供者处接收到的复制区从该上下文前缀派生而来；

——影像使用者与影像提供者建立起来的影像商定的标识符；

——影像提供者 DSA 的访问点；

——指示复制区的影像提供者是否是主 DSA；以及

——可选的，如果提供者不是主 DSA 时，主 DSA 的访问点。

22.3.2.7 使用者引用

影像提供者 DSA 所拥有的一个使用者引用包括：

——一个命名上下文的上下文前缀，影像提供者所提供的复制区从该上下文前缀派生而来；

——影像提供者与一个影像使用者建立起来的影像商定的标识符；以及

——影像使用者 DSA 的访问点。

22.4 最小知识

目录有一个特性，即每个条目都能够被访问，而不依赖于请求是从哪里发出的。

目录还有一个特性，即为了达到足够级别的性能和可用性，某些请求可以通过使用一个条目拷贝来满足，而另外一些请求仅能够通过使用条目本身来满足(即条目的主 DSA 所拥有的信息)。

为了实现这些与位置无关的目录特性，每个 DSA 须维护一个最小数量的知识，这些知识取决于 DSA 的特定配置。

这些最小需求的目标是：允许分布式名(称)解析过程能够为目录内的所有命名上下文建立一个引用路径，该路径由主知识引用的连续序列构成。

在这些最小需求之外，可能会部署另外的知识为命名上下文的拷贝建立其他的引用路径。可能会部署交叉引用知识(主知识或影像知识)为命名上下文和命名上下文的拷贝建立优化的引用路径。

DSA 的最小知识需求在 22.4.1 至 22.4.4 中规定。

22.4.1 上级知识

每个不是第一级 DSA 的 DSA 都必须至少维护一个上级引用。为了操作方面的原因，可能会拥有附加的上级引用作为到 DIT 根的可替代路径。

22.4.2 下级知识

作为一个命名上下文主 DSA 的 DSA，必须为每个拥有一个直接下级命名上下文的主 DSA 维护主知识类别的下级引用或非特定下级引用。

22.4.3 提供者知识

对于在复制区内向某个影像使用者 DSA 提供服务的每个影像提供者 DSA，该影像使用者 DSA 都必须维护一个相应的提供者引用。如果影像使用者所拥有的命名上下文拷贝的下级知识是不完整的，则它必须使用它的提供者引用来建立一个到下级信息的引用路径。该过程在 GB/T 16264.4—2008 的第 20 章描述。

22.4.4 使用者知识

对于在复制区内由某个影像提供者 DSA 所提供服务的每个影像使用者 DSA，该影像提供者 DSA 都必须维护一个相应的使用者引用。

22.5 第一级 DSA

由某个上级引用所引用的 DSA 假定了向 DIT 内所有条目建立的引用路径的边界，而此边界对于引用的 DSA 来说是未知的。由另一个 DSA 所引用的 DSA 自身也可能会维护一个或多个上级引用。这种递归的上级引用过程到第一级 DSA 后将停止，在该第一级 *DSA* 处，建立引用路径的最终责任将落于此。

第一级 DSA 的特性如下所述：

a) 它不拥有一个上级引用；

b) 它可能会拥有作为 DIT 根的直接下级的一个或多个命名上下文(作为该命名上下文的主 DSA 或影像 DSA)；以及

c) 它拥有的下级引用(主 DSA 和/或影像 DSA)和非特定下级引用(主 DSA 和/或影像 DSA)构成了 DIT 根的直接下级的所有的命名上下文，但不是 DIT 根本身所拥有的。

第一级 DSA 的管理机构共同负责对 DIT 根的直接下级进行管理。控制此共同管理的过程由多边商定来决定，多边商定不在本系列目录规范的定义范围之内。

注：在一个相关的条目环境中，可能某些第一级的条目拥有相同的名(称)，因此构成了多个 DIT。相应第一级 DSA 的管理机构应当共同负责这些 DIT 的管理。

为了限制可能向第一级主 DSA(即作为 DIT 根的直接下级的命名上下文的主 DSA)直接发出的查询请求的数量，可能为该第一级主 DSA 建立第一级的影像 DSA。这些影像 DSA 拥有第一级主 DSA(或提供者)所拥有的作为根的直接下级的条目以及下级引用的拷贝，因此它们可能作为非第一级 DSA 的上级引用。

23 DSA 信息模型的基本元素

23.1 定义

本目录规范使用下列术语和定义：

23.1.1

DSA 信息树 DSA information tree

从名(称)角度所看到的由一个 DSA 所拥有的所有 DSE 的集合。

23.1.2

DSA 共享属性 DSA-shared attribute

DSA 信息模型中与某个特定名(称)相关的一个操作属性，如果该操作属性被多个 DSA 所拥有的话，则它们的值或值集是相等的(除了短暂的不一致以外)。

23.1.3

DSA 特定属性 DSA-specific attribute

DSA 信息模型中与某个特定名(称)相关的一个操作属性，如果该操作属性被多个 DSA 所拥有的话，则它们的值或值集可以不必相等。

23.1.4

DSA 特定条目 DSA-specific entry；DSE

DSA 所拥有的与某个特定名(称)相关的信息；DSE 可能会(但不是必需)包含与相应的目录条目相关的信息。

23.1.5

DSE 类型 DSE type

指示一个 DSE 的某种特定目标；一个 DSE 可能服务于多个目标，因此具有多个类型。

23.2 概述

目录信息模型描述了作为一个整体的目录是如何表示关于客体的信息的，这些客体具有一个可辨

别名和可选的多个别名。在 DIT、条目和属性的描述中，目录的组成部分，即潜在地可能进行协作的 DSA 的集合，从模型中抽象出来。

从另一方面来说，DSA 信息模型尤其是与 DSA 以及 DSA 所必须拥有的信息相关，因为组成目录的 DSA 的集合可能联合起来实现目录信息模型。它所关注的包括：

——目录信息（客体条目和别名条目以及子条目等）是如何映射到 DSA 的；

——目录信息的拷贝如何被 DSA 所拥有；

——DSA 为执行名（称）解析和操作评估所需的操作信息；以及

——DSA 为实现影像和使用影像信息所需的操作信息。

对 DSA 操作信息（如知识）的表示进行建模，其目的是为了管理对 DSA 操作信息的访问而建立一个通用的框架。

23.3 DSA 特定条目及其名（称）

在 DSA 信息模型中，信息库拥有与某个特定名（称）相关的信息，此信息库被称为*DSA 特定条目（DSE）*。处于 DSA 信息模型中的目录条目仅仅是作为可能构成 DSE 的信息元素。特定于 DSA 信息模型的操作属性由其他各种可能构成 DSE 的信息元素组成。

如果一个 DSA 拥有与某个名（称）直接相关的任意信息（即存储于信息库中的由名（称）辨别的信息），则可认为 DSA 已知该名（称），或具有该名（称）相关的知识。

对于每个 DSA 已知的名（称），该 DSA 所拥有的与该名（称）直接相关的信息，除了名（称）本身外，都由一个 DSE 来表示。名（称）本身的信息（即 RDN 以及与 DIT 之间的关系）没有显式地表示为 DSA 信息模型内的一个属性；

一个 DSA 已知的所有名（称）的集合组成了一个隐含的结构，该结构可被认为是附着在相关的 DSE 上。

注 1：这种 DSA 信息模型处理名（称）的方法，带来的一个结果是，对于类型不是条目、别名或子条目的 DSE 来说，表示 DSE 的 RDN 的 AVA 没有建模在属性中。

当命名属性拥有多个由上下文所区分的可辨别值时，则存在替代名（称），在这种情况下，一个单独的 DSE 可以表示 DSA 拥有的关于所有替代名（称）的所有信息。在 DSA 信息模型中，这种情况被建模为一个单独的名（称），该名（称）具有上下文特定的变量，而不是建模为不同的名（称）。

注 2：为了进行一致的名（称）解析，并与第 3 版之前的 DSA 进行交互，每个 DSA 必须至少拥有组成名（称）的所有属性的主可辨别值信息，并且尽可能地拥有尽量多的替代可辨别值。

DSA 所已知的所有名（称）的集合以及与每个名（称）相关的信息，如果从这些名（称）的视角来看，它们被称为该 DSA 的 DSA 信息树。一棵 DSA *信息树*如图 21 所述。

DSA 与名（称）关联的最少信息（因此才能够知晓该名（称））包括表达名（称）被知晓的目的（即在 DSA 知晓该名（称）的操作中，名（称）所发挥的作用），该目的在 DSA 信息模型中由一个 DSA 特定属性 dseType 来表示。

另外，一个 DSE 可能还会拥有其他与名（称）相关的信息，如条目或条目拷贝、DSA 共享属性和 DSA 特定属性等。

一个 DSE 可能会直接表示一个目录条目，一个条目的一部分或不表示目录信息。在 DSE 中拥有的信息是变化的，取决于它的类型或目的。一般来说，下述类型的 DSE 可能会出现在 DSA 中。

——一个直接表示一个目录条目的 DSE，包括与该目录条目相应的用户属性和操作属性（在图 21 的 DSE2 中描述）。该 DSE 还可能包含 DSA 共享属性和 DSA 特定属性；

——一个表示一个条目中部分信息的 DSE（影像的结果），包括与该目录条目相应的某些用户属性和操作属性以及 DSA 特定属性，另外还可能包括 DSA 共享属性；

——一个表示惯例 ACI 或集合属性的子条目 DSE，包括与该目录子条目相应的有关用户属性和操作属性（在图 21 中的 DSE 3 中描述）。该 DSE 还可能包括 DSA 共享属性和 DSA 特定属性；

——一个没有表示任何目录条目信息的 DSE，仅包括 DSA 共享属性和/或 DSA 特定属性（在图 21 的 DSE 1 和 DSE 4 中描述）。例如，一个表示下级引用的 DSE 可能会拥有一个 DSA 共享属性来指示主访问点，拥有一个 DSA 特定属性来指示该 DSE 是一个下级引用。

注 3：DSE 是一个概念上的实体，以一种一致方便的方式简化了信息组件的规范和建模。尽管 DSE 被称为是“拥有”或“存储”信息，但它实际上对实现没有施加任何特定的限制或数据结构要求。

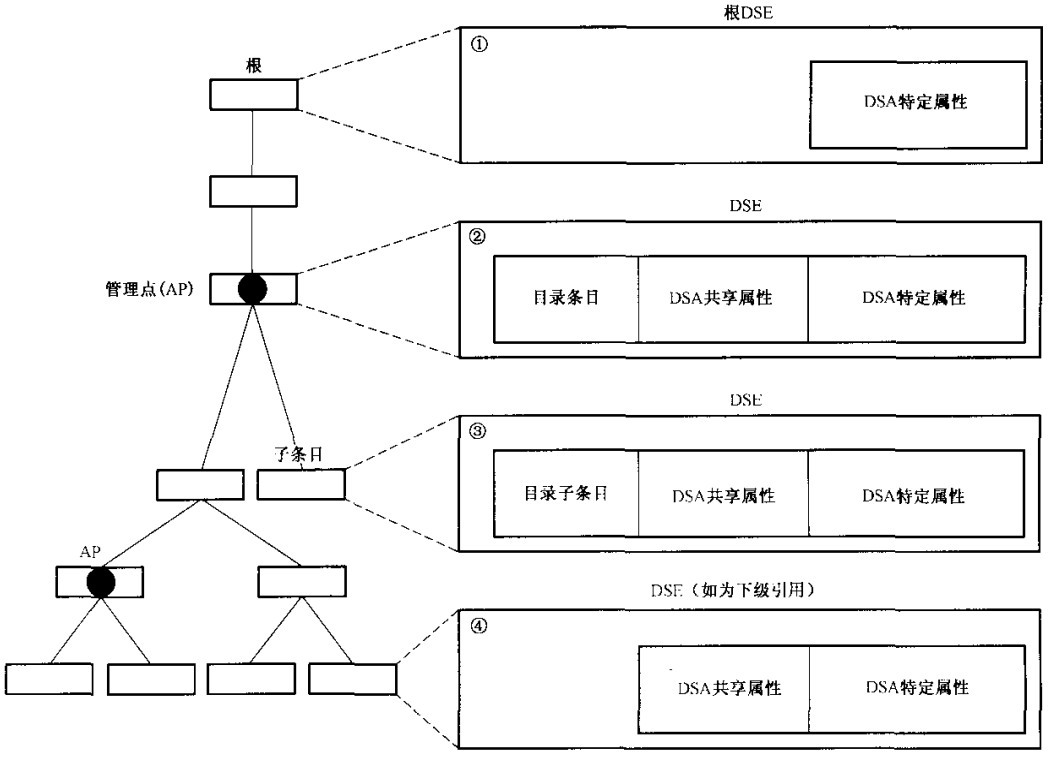


图 21　DSA 信息树

23.4　基本元素

一个 DSE 由 3 个基本元素组成：DSE 类型、一些数量的 DSA 操作属性（DSE 类型是其中之一）以及可选的一个条目或条目拷贝。

23.4.1　DSA 操作属性

有两种操作属性出现在 DSA 信息模型中，它们不符合目录条目中的信息。这些操作属性为 DSA 共享属性和 DSA 特定属性。

DSA 共享属性 是 DSA 信息模型中与某个特定名（称）相关的一种操作属性，如果该操作属性由多个 DSA 所拥有，则它们的值或值集是相等的（除了短暂的不一致以外）。一个 DSA 可能会拥有 DSA 共享属性的一个影像拷贝。

DSA 特定属性 是 DSA 信息模型中与某个特定名（称）相关的一种操作属性，如果该操作属性由多个 DSA 所拥有，则它们的值或值集不必是相同的。一个 DSA 特定属性所表示的操作信息是针对拥有该属性的 DSA 的特定功能。一个 DSA 不能拥有 DSA 特定属性的一个影像拷贝。

注：当一个影像提供者 DSA 可能向影像使用者 DSA 提供一个 DSA 特定属性时，在概念上该特定属性不是提供者所拥有信息的一个影像拷贝，更确切的应当是提供者为使用者所产生的信息，使用者可能会使用并修改这些信息。

23.4.2　DSE 类型

DSE 的类型，在 DSA 信息模型中由 DSA 特定操作属性 dseType 来表示，指示了某个 DSE 的特定

目的(或作用)。该目的由属性dseType的单值中的命名比特来指示。由于一个DSE可能会服务于多个目的,因此可能会设置属性dseType的多个命名比特来表示这些目的。附录N中规定了可能会出现的命名比特的多种组合。

在目录规范中使用短语"类型为x的DSE"来表示DSE的属性dseType中的命名比特x被设置。

对于一个类型为x的DSE,其他命名比特可能会根据需求设置或不设置。另外还可能会使用替代短语"类型包括x的DSE"。

操作属性dseType的句法规范可能使用下述的属性表示法来表示:

```
dseType   ATTRIBUTE   ::={
          WITH SYNTAX                    DSEType
          EQUALITY MATCHING RULE         bitStringMatch
          SINGLE VALUE                   TRUE
          NO USER MODIFICATION           TRUE
          USAGE                          dSAOperation
          ID                             id-doa-dseType }
```

该DSA特定操作属性由DSA自身来管理。

表示属性dseType的可能值的句法的ASN.1类型为DSEType。它的定义如下:

```
DSEType   ::=BIT STRING {
          root           (0),    ——根 DSE——
          glue           (1),    ——仅表示一个名(称)的知识——
          cp             (2),    ——上下文前缀——
          entry          (3),    ——客体条目——
          alias          (4),    ——别名条目——
          subr           (5),    ——下级引用——
          nssr           (6),    ——非特定下级引用——
          supr           (7),    ——上级引用——
          xr             (8),    ——交叉引用——
          admPoint       (9),    ——管理点——
          subentry       (10),   ——子条目——
          shadow         (11),   ——影像拷贝——
          immSupr        (13),   ——直接上级引用——
          rhob           (14),   ——rhob 信息——
          sa             (15),   ——别名条目的下级引用——
          dsSubentry     (16),   ——DSA 特定子条目——
          familyMember   (17),   ——家族成员——
          ditBridge      (18),   ——DIT 桥接引用——
          writeableCopy  (19) }  ——可写拷贝——
```

DSEType的值为:

a) root:根DSE包含DSA特定属性,由DSA使用,表示DSA作为一个整体的特性。与根DSE相应的名(称)是包含零个RDN序列的退化名(称)。

注1:通过目录抽象服务而使其可用的一个DSA的特性信息包含在DSA的条目中。一个DSA可能但不必拥有其自身条目或自身条目的拷贝。

b) glue:一个粘接DSE(glueDSE),该DSE仅表示一个名(称)的知识。一个拥有上下文前缀DSE或交叉引用DSE的DSA可能会拥有glueDSE,来表示上下文前缀或交叉引用DSE的上

级的名(称),如果没有其他的操作属性(如知识)与这些名(称)相关的话。如图 22 中的示例。一个类型为 glue 的 DSE 不得设置其他任何的DSEType 比特。

c) cp :表示一个命名上下文的上下文前缀的 DSE。

d) entry :一个 DSE,该 DSE 拥有一个客体条目。

e) alias :一个 DSE,该 DSE 拥有一个别名条目。

f) subr :一个 DSE,该 DSE 拥有一个表示下级引用的特定知识属性。

g) nssr :一个 DSE,该 DSE 拥有一个表示非特定下级引用的非特定知识属性。

h) supr :一个 DSE,该 DSE 拥有一个表示 DSA 上级引用的特定知识属性。

i) xr :一个 DSE,该 DSE 拥有一个表示交叉引用的特定知识属性。

j) admPoint :一个 DSE,该 DSE 与一个管理点相应。

k) subentry :一个 DSE,该 DSE 拥有一个子条目。

l) shadow :一个 DSE,该 DSE 拥有一个条目(或部分条目)的影像拷贝或其他从影像提供者处接收到的信息;该命名比特由影像使用者来设置。

m) immSupr :一个 DSE,该 DSE 拥有一个表示直接上级引用的特定知识属性。

n) rhob :一个 DSE,该 DSE 拥有从上级 DSA 处接收到的管理点和子条目信息,此上级 DSA 在一个相关的层次关系操作绑定(RHOB)中(即或者在一个层次化的操作绑定中,或者在一个非特定层次绑定中,在 GB/T 16264.4—2008 的第 24 章到第 25 章中描述。

o) sa :一个类型为subr 的 DSE 的限定符,指示了该下级命名上下文条目是一个别名。

p) dsSubentry :一个 DSE,该 DSE 拥有一个 DSA 特定的子条目。

q) familyMember :一个 DSE,该 DSE 拥有一个家族成员。

r) ditBridge :一个 DSE,该 DSE 拥有一个 DIT 桥接引用。

s) writeableCopy :一个 DSE,该 DSE 拥有在多个主实现中复制的条目或其他信息(如知识)的一个可写拷贝。

注 2:某些目录操作要求有能力为任意给定的条目标识一个单独的主。因此,在一个多主的实现中,每个条目的所有主拷贝都须具有该 DSE 类型,只有一个除外,即必要时,其中一个主拷贝要作为第一主拷贝,则该主拷贝不得具有该 DSE 类型。

使用该操作属性可表示 DSA 信息模型的各个方面,其用法在第 23 章描述。

24 DSA 信息的表示

本章论述了 DSA 信息的表示。它描述了 DSA 操作信息(知识)、目录用户信息以及目录操作信息的表示等。

24.1 目录用户信息和操作信息的表示

本条规定了在 DSA 信息模型中目录用户信息和目录操作信息的表示。

24.1.1 客体条目

客体条目由一个类型为entry 的 DSE 来表示,该 DSE 包含与该目录条目相关的用户属性和目录操作属性。DSE 的名(称)即为客体条目的名(称)(即客体的识别名)。

如果 DSE 中拥有该条目的一个拷贝,则 DSE 的类型还包括shadow 。

如果客体条目的名(称)中包含任何由上下文所区分的替代识别名,则 DSE 的名(称)可能也包含这些由上下文所区分的替代识别名。当 DSE 拥有该条目的一个影像时,DSE 的名(称)可能包含这些替代识别名中的一个子集。当 DSE 不是拷贝的情况下,DSE 的名(称)必须包含所有的识别名。

注:为了进行一致的名(称)解析,并与第 3 版之前的 DSA 进行交互,一个拥有拷贝的 DSA 的名(称)须至少包含所有命名属性的主识别名。因此,拷贝至少应拥有该客体条目的主识别名。如果每个识别值都存在(因此每个替代识别名都存在),则名(称)解析将被增强。

24.1.2 别名条目

别名条目由一个类型为*alias* 的 DSE 来表示，该 DSE 包含与别名条目相关的属性(即 RDN 属性和被起了别名的客体名(称)属性)。DSE 的名(称)即为别名条目的名(称)。

如果 DSE 拥有该别名条目的一个拷贝，则 DSE 的类型还包括shadow 。

如果别名条目的名(称)包含任何由上下文所区分的替代识别名，则 DSE 的名(称)可能也包含这些由上下文所区分的替代识别名。当 DSE 拥有该别名条目的一个影像时，DSE 的名(称)可能包含这些替代识别名中的一个子集。当 DSE 不是拷贝的情况下，DSE 的名(称)必须包含所有的识别名。

注：为了进行一致的名(称)解析，并与第 3 版之前的 DSA 进行交互，一个拥有拷贝的 DSA 的名(称)必须至少包含所有命名属性的主识别名。因此，拷贝至少应拥有该别名条目的主识别名。如果每个识别值都存在(因此每个替代识别名都存在)，则名(称)解析将被增强。

24.1.3 管理点

管理点由一个类型为admPoint 的 DSE 来表示，该 DSE 包含与管理点相关的属性。DSE 的名(称)即为管理点的名(称)。

如果 DSE 表示一个条目，则 DSE 的类型中包括entry 。如果 DSE 拥有该管理点的一个拷贝，则 DSE 的类型中还包括shadow 。

如果管理点的名(称)包含任何由上下文所区分的替代识别名，则 DSE 的名(称)可能也包含这些由上下文所区分的替代识别名。当 DSE 拥有该管理点的一个影像时，DSE 的名(称)可能包含这些替代识别名中的一个子集。当 DSE 中不是拷贝的情况下，DSE 的名(称)应当包含所有的识别名。

注：为了进行一致的名(称)解析，并与第 3 版之前的 DSA 中进行交互，一个拥有拷贝的 DSA 的名(称)必须至少包含所有命名属性的主识别名。因此，拷贝至少应拥有该管理点的主识别名。如果每个识别值都存在(因此每个替代识别名都存在)，则名(称)解析将被增强。

24.1.4 子条目

子条目由一个类型为subentry 的 DSE 来表示，该 DSE 包含与子条目相关的操作信息和用户信息。DSE 的名(称)是子条目的名(称)。

如果该 DSE 中拥有一个子条目的拷贝，则 DSE 的类型为subentry 和shadow 。

24.1.5 家族成员

家族成员(包含祖先)由一个类型为familyMember 的 DSE 来表示。该祖先也是一个类型为entry 的 DSE；它是家族成员中唯一一个被允许具有该 DSE 类型的成员。

24.2 知识引用的表示

一个知识引用包括一个具有适当类型的 DSE，该 DSE 拥有相应正确的 DSA 操作属性，且由一个名(称)来标识，该名(称)承载了与被引用的 DSA 所拥有的命名上下文之间的已定义关系。

该 DSE 的名(称)须是主识别名，如果替代名(称)和上下文信息出现在被引用的 DSA 的命名上下文的上下文前缀中，则 DSE 的名(称)也可能包括替代名(称)和上下文信息。当 DSE 拥有一个影像时，DSE 的名(称)中可能包含这些替代识别名中的一个子集。当 DSE 不是拷贝的情况下，DSE 的名(称)必须包含所有的识别名。

注：如果每个识别值都存在(因此每个替代识别名都存在)，则名(称)解析将被增强。

24.2.1 知识属性类型

DSA 操作属性在 DSA 信息模型中定义，它可以表示一个 DSA 的下述信息：

—— DSA 自身访问点的知识；

—— 上级知识；

——特定知识(它的下级引用)；

—— 非特定知识(它的非特定下级引用)；

—— 如果 DSA 是一个影像使用者，则可表示它的提供者知识，可选地包括主 DSA；

——如果 DSA 是一个影像提供者，则可表示它的使用者知识；

——如果 DSA 是一个影像提供者，则可表示它的第二影像的知识；以及

——另一个 DIT 的知识。

这些操作属性的客体标识符在附录 F 中分配。

24.2.1.1 我的访问点

操作属性类型myAccessPoint 由 DSA 使用，用以表示它自身的访问点。它是一个 DSA 特定属性。所有的 DSA 都必须在其根 DSE 中拥有该属性。它是单值的，并且由 DSA 自身来管理。

```
myAccessPoint   ATTRIBUTE   ::={
        WITH SYNTAX                         AccessPoint
        EQUALITY MATCHING RULE              accessPointMatch
        SINGLE VALUE                        TRUE
        NO USER MODIFICATION                TRUE
        USAGE                               dSAOperation
        ID                                  id-doa-myAccessPoint }
```

类型 AccessPoint 的 ASN.1 定义在 GB/T 16264.4—2008 中定义。为了便于读者阅读，将其 ASN.1 规范复制在此。

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

注：在ae-title 中的Name 可能是主辨别名或一个可替代辨别名；然而，如果使用主辨别名，可以增强名(称)解析的一致性以及与第 3 版之前的 DSA 的交互性。

一个 DSA 如何获得myAccessPoint 中存储的信息，不在本系列目录规范中描述。

属性类型myAccessPoint 存储于类型为root 的 DSE 中。

当建立或修改一个操作绑定时，可能会在 DOP 中部署使用存储于myAccessPoint 中的信息。

24.2.1.2 上级知识

操作属性类型superiorKnowledge 由一个非第一级 DSA 使用，用以表示它的上级引用。它是一个 DSA 特定属性。所有的非第一级 DSA 都须在其根 DSE 中拥有该属性。它是多值的，并且由 DSA 自身来管理。

```
superiorKnowledge   ATTRIBUTE   ::={
        WITH SYNTAX                         AccessPoint
        EQUALITY MATCHING RULE              accessPointMatch
        NO USER MODIFICATION                TRUE
        USAGE                               dSAOperation
        ID                                  id-doa-superiorKnowledge }
```

一个 DSA 可能会获取存储于superiorKnowledge 中的信息，获取方法不在本系列目录规范中描述。它也可能会根据其直接上级引用来构造这些信息，例如，根据上下文前缀的名(称)中具有最小数量 RDN 的直接上级引用来构造此信息。

属性类型superiorKnowledge 存储于类型为root 的 DSE 中。

当构建一个从 DAP 或 DSP 参照指示中返回的连续引用或者当执行一个链接时，可能会由 DSA 部署使用存储于superiorKnowledge 中的信息。

24.2.1.3 特定知识

特定知识包括一个命名上下文的主 DSA 的访问点和/或该命名上下文的影像 DSA。说它是特定

的，是因为该命名上下文的上下文前缀已知，并与访问点信息相关联。该特定知识由操作属性类型specificKnowledge表示。它是一个DSA共享属性，是单值的，并且由DSA自身来管理。

```
specificKnowledge ATTRIBUTE ::={
    WITH SYNTAX                     MasterAndShadowAccessPoints
    EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
    SINGLE VALUE                    TRUE
    NO USER MODIFICATION            TRUE
    USAGE                           distributedOperation
    ID                              id-doa-specificKnowledge }
```

ASN.1类型MasterAndShadowAccessPoints在GB/T 16264.4—2008中定义。为了便于读者阅读，将其ASN.1规范复制在此。

```
MasterAndShadowAccessPoints ::=SET OF MasterOrShadowAccessPoint

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

DSA可能会获取存储于specificKnowledge中的信息，获取方法不在本系列目录规范中描述。在交叉引用的情况下(DSE的类型为xr)，它也可能会根据从DSP回复的ChainingResults的组件crossReference中接收的信息来构造这些信息。在下级引用的情况下(DSE的类型为subr)，它也可能会根据在创建或修改HOB时从DOP中接收的信息来构造这些信息。

属性类型specificKnowledge存储于类型为subr、immSupr或xr的DSE中。它被DSA使用，分别用以表示下级引用、直接上级引用和交叉引用。

当构建一个从DAP或DSP参照指示中返回的连续引用(或者当执行一个链接)时，或者当构建DISP提供的类型为subr、immSupr或xr的影像DSA的特定条目(SDSE)时，可能会由DSA部署使用存储于specificKnowledge中的信息。

24.2.1.4 非特定知识

非特定知识包括一个或多个命名上下文的主DSA的访问点和/或该一个或多个命名上下文的影像DSA。说它是非特定的，是因为该命名上下文的上下文前缀是未知的。然而，命名上下文的直接上级已知，且访问点信息与此名(称)相关联。非特定知识由操作属性类型nonSpecificKnowledge表示。它是一个DSA共享属性，是多值的，且由DSA自身来管理。

```
nonSpecificKnowledge ATTRIBUTE ::={
    WITH SYNTAX                     MasterAndShadowAccessPoints
    EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
    NO USER MODIFICATION            TRUE
    USAGE                           distributedOperation
    ID                              id-doa-nonSpecificKnowledge }
```

MasterAndShadowAccessPoints的值包括拥有一个或多个下级命名上下文的主DSA的一个访问点以及拥有这些命名上下文的部分或全部影像的DSA的零个或多个访问点。

DSA可能会获取存储于nonSpecificKnowledge中的信息，获取方法不在本系列目录规范中描述。在非特定下级引用的情况下(DSE的类型为nssr)，它也可能会根据在创建或修改一个NHOB时，从

DOP 中接收的信息来构造这些信息。

属性类型nonSpecificKnowledge 存储于类型为nssr 的 DSE 中。它用以表示非特定下级引用。

当构建一个从 DAP 或 DSP 参照指示中返回的连续引用(或执行一个链接)时,或者当构建一个在 DISP 中提供的类型为nssr 的 SDSE 时,可能会由 DSA 部署使用存储于nonSpecificKnowledge 中的信息。

24.2.1.5 提供者知识

影像消费者 DSA 所拥有的提供者知识包括:访问点和一个复制区内的拷贝(或拷贝集)提供者的影像商定标识符。可选的,如果该提供者不是派生此复制区的命名上下文的主 DSA,则在提供者知识中还可能会包括主 DSA 访问点的信息。提供者知识由操作属性类型supplierKnowledge 表示。它是一个 DSA 特定属性,是多值的,且由 DSA 自身来管理。

属性supplierKnowledge 的值的 ASN.1 句法定义为SupplierInformation 。组成该属性值的信息包括:一个影像提供者 DSA 的访问点,在提供者 DSA 和拥有该 DSA 特定属性的消费者 DSA 之间的影像商定的商定 ID(表示为类型SupplierOrConsumer 的值),指示复制区的提供者是否为派生该复制区的命名上下文的主 DSA,如果不是主 DSA,可选的还包括主 DSA 的访问点等。

```
SupplierOrConsumer ::= SET {
    COMPONENTS OF              AccessPoint,      --提供者或消费者--
    agreementID          [3]   OperationalBindingID }

SupplierInformation ::= SET {
    COMPONENTS OF              SupplierOrConsumer,      --提供者--
    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 }
```

DSA 可能会获取存储于supplierKnowledge 中的信息,获取方法不在本系列目录规范中描述。一个影像消费者 DSA 可能会根据在创建或修改一个影像商定时从 DOP 中接收的信息来构造这些信息。

属性类型supplierKnowledge 存储于类型为cp 的 DSE 中。它用于表示一个或多个提供者引用。所有的影像消费者 DSA 必须为它们做为消费者的每个影像商定拥有该属性的一个值。

当构建一个从 DAP 或 DSP 参照指示中返回的连续引用时,可能会由 DSA 部署使用存储于supplierKnowledge 中的信息。在管理一个影像商定的 DOP 操作中以及所有的 DISP 操作中都需要supplierKnowledge 的组件agreementID(其类型为OperationalBindingID ,在 28.2 中定义)。

24.2.1.6 消费者知识

影像提供者 DSA 所拥有的消费者知识包括:访问点和提供者向消费者提供命名上下文拷贝(或拷贝集)的影像商定的标识符。消费者知识由操作属性类型consumerKnowledge 表示。它是一个 DSA 特定属性,是多值的,且由 DSA 自身来管理。

属性consumerKnowledge 的值的 ASN.1 句法定义为ConsumerInformation (与SupplierOrConsumer 具有相同的句法,但指向一个消费者访问点)。

```
ConsumerInformation ::= SupplierOrConsumer      --消费者--
```

```
consumerKnowledge ATTRIBUTE ::={
    WITH SYNTAX                 ConsumerInformation
    EQUALITY MATCHING RULE      supplierOrConsumerInformationMatch
    NO USER MODIFICATION        TRUE
    USAGE                       dSAOperation
    ID                          id-doa-consumerKnowledge }
```

DSA 可能会获取存储于consumerKnowledge 中的信息，获取方法不在本系列目录规范中描述。影像提供者 DSA 可能会根据在创建或修改一个影像商定时从 DOP 处接收的信息来构造这些信息。

属性类型consumerKnowledge 存储于类型为cp 的 DSE 中。它用于表示一个或多个消费者引用。所有的影像提供者 DSA 必须为它们做为提供者的每个影像商定拥有该属性的一个值。

在管理一个影像商定的 DOP 操作中以及所有的 DISP 操作中都需要consumerKnowledge 的组件agreementID 。

24.2.1.7 二次影像知识

二次影像知识包括一个提供者 DSA(例如一个主 DSA)可能会选择维护的信息，这些信息是从提供者 DSA 的视角来看，参与二次影像的消费者 DSA 的信息。二次影像知识由操作属性类型secondary-Shadows 来表示。它是一个 DSA 特定属性，是多值的，且由 DSA 自身来管理。属性secondaryShadows 的值的 ASN.1 句法定义为SupplierAndConsumers 。它包括一个影像提供者的访问点及其直接消费者的列表。

```
SupplierAndConsumers ::=SET {
COMPONENTS OF       AccessPoint,     --提供者--
consumers           [3] SET OF AccessPoint }

secondaryShadows ATTRIBUTE ::={
    WITH SYNTAX                 SupplierAndConsumers
    EQUALITY MATCHING RULE      supplierAndConsumersMatch
    NO USER MODIFICATION        TRUE
    USAGE                       dSAOperation
    ID                          id-doa-secondaryShadows }
```

SuppliersAndConsumers 的组件consumers 中仅包括拥有一个复制区内公共可用拷贝的 DSA 的访问点。

一个提供者 DSA 可能会从一个消费者 DSA 中获取到构造该属性的值所需的信息，获取的过程在 GB/T 16264.4—2008 的 23.1.1 中描述。

属性类型secondaryShadows 由类型为cp 的 DSE 所拥有。

对二次影像知识的支持为可选。

24.2.1.8 DIT 桥接知识

处于另一个 DIT 中的某个命名上下文的主 DSA 由ditBridgeKnowledge 表示，它包括一个域标识符及其访问点。操作属性dITBridgeKnowledge 包括所有已知的这些 DSA 的DITBridgeKnowledge 。它是一个多值的 DSA 共享属性，且由 DSA 管理者来管理。该属性由类型为root 的 DSE 所拥有，另外该 DSE 还因为 DIT 桥接引用而具有 DSE 类型ditBridge。

```
ditBridgeKnowledge ATTRIBUTE ::={
    WITH SYNTAX                 DitBridgeKnowledge
    EQUALITY MATCHING RULE      directoryStringFirstComponentMatch
```

NO USER MODIFICATION TRUE
USAGE dSAOperation
ID id-doa-ditBridgeKnowledge}

ASN.1 类型DitBridgeKnowledge 在 GB/T 16264.4—2008 中定义。为了便于读者阅读，将其ASN.1 规范复制在此。

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

当执行一个包含相关条目的搜索操作时，DSA 可能会部署使用ditBridgeKnowledge 中的信息。

24.2.1.9 匹配规则

之前所述的知识属性的四个相等匹配规则在下面规定。它们所应用的属性的句法类型分别为：AccessPoint，MasterAndShadowAccessPoints，SupplierInformation，ConsumerInformation 和SuppliersAndConsumers。

24.2.1.9.1 访问点匹配

访问点匹配规则规定如下：

```
accessPointMatch MATCHING-RULE ::={
SYNTAX          Name
ID              id-kmr-accessPointMatch }
```

匹配规则accessPointMatch 应用于类型为AccessPoint 的属性值。该声明句法的一个值是从该属性句法的一个值派生而来的，使用了[0]上下文特定标签(Name)组件的值。如果每个值的Name 组件使用DistinguishedName 值的匹配过程后是匹配的，则这两个值被认为是相等匹配的。

24.2.1.9.2 主访问点和影像访问点匹配

主访问点和影像访问点的相等匹配规则规定如下：

```
masterAndShadowAccessPointsMatch MATCHING-RULE ::={
SYNTAX          SET OF Name
ID              id-kmr-masterShadowMatch }
```

匹配规则masterAndShadowAccessPointsMatch 应用于类型为MasterAndShadowAccessPoints 的属性。该声明句法的一个值是从该属性句法的一个值派生而来的，是通过删除SET OF MasterOrShadowAccessPoints 中的每个SET 的category 和address 组件来实现的。如果两个值都具有相同数量的SET OF 元素，且以任意一种便利的方式对这每个值中的SET OF 组件进行排序后，每对SET OF 元素中的ae-title 组件使用DistinguishedNameMatch 的匹配过程后是都是匹配的，则这两个值被认为是相等匹配的。

24.2.1.9.3 提供者或消费者信息匹配

提供者或消费者信息匹配规则规定如下：

```
supplierOrConsumerInformationMatch MATCHING-RULE ::={
SYNTAX          SET {
    ae-title                        [0]     Name,
    agreement-identifier            [2]     INTEGER }
ID              id-kmr-supplierConsumerMatch }
```

匹配规则supplierOrConsumerInformationMatch 应用于类型为SupplierInformation 或ConsumerInformation 的属性值(以及其他的符合SupplierInformation 或ConsumerInformation 的属性值)。该声明句法的一个值是从该属性句法的一个值派生而来的，是通过选择SET 组件来实现的，该SET 组件所

具有的标签与声明句法的SET 组件相匹配。如果两个值中的ae-title 组件(在删除了显式的[0]标签信息后)在使用了DistinguishedName 值的匹配过程后是匹配的,且包含在每对值中的agreement 组件中的identifier 组件(在删除了显式的[2]以及SEQUENCE 标签信息后)在使用了INTEGER 值的匹配过程后也是匹配的,则这两个值被认为是相等匹配的。

24.2.1.9.4 提供者和消费者匹配

提供者和消费者匹配规则规定如下:

```
supplierAndConsumersMatch MATCHING-RULE ::={
    SYNTAX          Name
    ID              id-kmr-supplierConsumersMatch }
```

提供者和消费者匹配规则应用于类型为SupplierAndConsumers 的属性值(以及其他的符合SupplierAndConsumers 的属性值)。如果两个值中的ae-title 组件(在删除了显式的[0]标签信息后)在使用了DistinguishedName 值的匹配过程后是匹配的,则这两个值被认为是相等匹配的。

24.2.2 知识引用类型

本条规定了知识在 DSA 信息模型中的表示法。

24.2.2.1 自我引用

一个自我引用表示一个 DSA 自身访问点的知识。它由 DSA 内的根 DSE(类型为root 的 DSE)所拥有的属性myAccessPoint 的值表示。

24.2.2.2 上级引用

一个上级引用由一个类型为*supr* 和*root* 的 DSE 表示,该 DSE 包含一个*superiorKnowledge* 属性。由于一个superiorKnowledge 属性的值可能包含多个 DSA 的访问点,因此它可能表示多个上级引用。

24.2.2.3 直接上级引用

一个直接上级引用由一个类型为immSupr 的 DSE 表示,该 DSE 包含一个specificKnowledge 属性。拥有该属性的 DSE 的名(称)与被引用的上级 DSA 所拥有的命名上下文的上下文前缀相一致。

由于specificKnowledge 的一个属性值可能包含多个 DSA 的访问点,因此他可能表示多个直接上级引用,其中,最多有一个种类为master ,有零个或多个种类为shadow 。

如果拥有该直接上级引用的 DSE 中是从一个影像提供者处获取的,则该 DSE 的类型还包括shadow 。

24.2.2.4 下级引用

一个下级引用由一个类型为subr 的 DSE 表示,该 DSE 包含一个specificKnowledge 属性。拥有该属性的 DSE 的名(称)与被引用的下级 DSA 所拥有的相关命名上下文的上下文前缀相一致。

由于specificKnowledge 的一个属性值可能包含多个 DSA 的访问点,因此它可能表示多个下级引用,其中,最多有一个种类为master ,有零个或多个种类为shadow 。

如果拥有该下级引用的 DSE 是影像的信息,且是从一个影像提供者处获取的,则该 DSE 的类型还包括shadow 。

DSE 可能还会在拥有两个命名上下文的 DSA 中包含immSupr ,这两个命名上下文中,其中一个是另一个的上级,并且由另一个 DSA 所拥有的一个第三方单独条目命名上下文所分离。这种情况的一个示例在附录 O 中描述。

24.2.2.5 非特定下级引用

一个非特定下级引用由一个类型为nssr (一般来说还包括entry)的 DSE 表示,该 DSE 包含一个nonSpecificKnowledge 属性。拥有该属性的 DSE 的名(称)与被引用的下级 DSA 所拥有的相关命名上下文的上下文前缀中将最后的 RDN 排除后所构造的名(称)相一致。

注:SSR 不能够引用 LDAP 服务器。

由于nonSpecificKnowledge 的一个属性值可能会包括多个 DSA 的访问点,因此它可能表示多个非

特定下级引用，其中，最多有一个种类为master，有零个或多个种类为shadow。每个nonSpecificKnowledge的属性值表示一个相关的非特定下级引用的集合 — 种类为shadow的DSA拥有一个或多个复制区，这些复制区是从种类为master的DSA所拥有的命名上下文派生而来的。

如果拥有非特定下级引用的DSE是从一个影像提供者处接收的影像信息，则DSE的类型还包括shadow。

DSE包括种类shadow是在影像DSA的如下情况下：当DSE与某个条目相符合，而对于该条目，主DSA拥有该条目的非特定下级知识，且仅有非特定下级引用的属性nonSpecificKnowledge被影像。

DSE包括种类cp和shadow是在影像DSA的如下情况下：该影像DSA的复制区不包括上下文前缀条目，而命名上下文的主DSA拥有该上下文前缀的非特定下级知识。

DSE包括种类admPoint和shadow是在影像DSA的如下情况下：当DSE与一个管理点相符合，该管理点的条目信息没有被影像，且命名上下文的主DSA拥有该管理点的非特定下级知识。

当管理点与前面所述的两种情况中的上下文前缀相符合时，则DSE可能包括admPoint、cp和shadow。

24.2.2.6 交叉引用

一个交叉引用由一个类型为xr的DSE表示，该DSE包含一个specificKnowledge属性。拥有该属性的DSE的名(称)与被引用的DSA所拥有命名上下文的上下文前缀相一致。

由于一个specificKnowledge属性的值可能会包含多个DSA的访问点，因此它可能会表示多个交叉引用，其中，最多有一个种类为master，有零个或多个种类为shadow。

24.2.2.7 提供者引用

一个提供者引用由一个类型为cp的DSE表示，该DSE包含一个supplierKnowledge属性。拥有该属性的DSE的名(称)与影像的命名上下文的上下文前缀相一致。

由于一个supplierKnowledge属性可能会有多个值，因此它可能会表示多个提供者引用。每个属性值表示一个提供者引用。

24.2.2.8 消费者引用

一个消费者引用由一个类型为cp的DSE表示，该DSE包含一个consumerKnowledge属性。拥有该属性的DSE的名(称)与影像的命名上下文的上下文前缀相一致。

由于一个consumerKnowledge属性可能会有多个值，因此它可能会表示多个消费者引用。每个属性值表示一个消费者引用。

24.3 名(称)和命名上下文的表示

24.3.1 名(称)和粘接DSE

正如在23.3中描述的那样，一个DSA可能与某个名(称)相关联的最小信息是它拥有该名(称)的目的，由一个拥有属性dseType的一个值的DSE来表示。当一个DSE仅包含这样的最小信息，则该DSE的类型须为glue。在这种情况下，该DSE不得拥有一个条目或子条目(或条目或子条目的一个影像拷贝)或一个DSA共享属性。

粘接DSE出现在DSA信息模型中，是用来表示由于拥有与其他名(称)相关的信息而被某个DSA已知的名(称)。例如，考虑在图22中描述的交叉引用。拥有该交叉引用的DSA也"已知"(在23.3描述的情况)一个名(称)，该名(称)是与交叉引用相关的上下文前缀名(称)的上级。如果没有其他信息与这样的上级名(称)相关联，则它们在DSA信息模型中由粘接DSE来表示。

24.3.2 命名上下文

一个命名上下文包括一个上下文前缀；该上下文前缀的零个或多个下级条目的一棵子树(上下文前缀是该子树的根)；如果有作为其下级的命名上下文，则还包括能够充分地组成完整的下级知识的下级引用和/或非特定下级引用。

一个上下文前缀由类型为cp的DSE来表示。如果上下文前缀与一个条目相符合，则DSE的类型

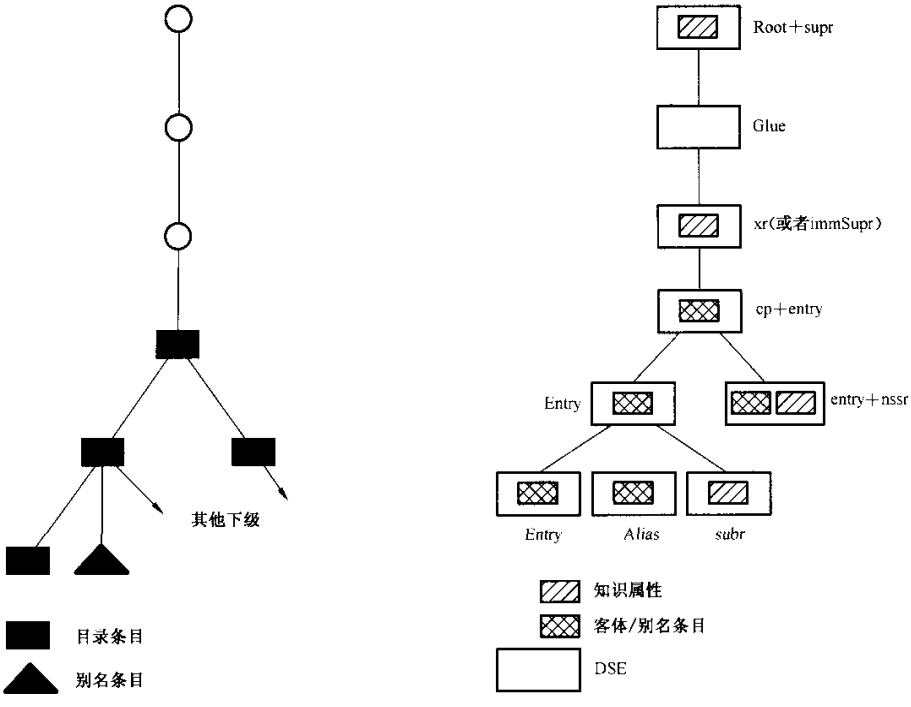
中包括entry 。如果上下文前缀与一个别名相符合，则 DSE 的类型中包括alias 。如果上下文前缀与一个管理点相符合，则 DSE 的类型中包括admPoint 。

作为上下文前缀下级的条目和子条目所组成的一棵子树，由 24.1.1 到 24.1.5 所描述的 DSE 来表示。命名上下文的下级知识，由 24.2.2 描述的 DSE 来表示。

一个复制区(一个命名上下文的所有或部分的影像拷贝)的表示如上所述，除了一种情况，即如果 DSE 的用户属性或操作属性是从某个影像提供者处接收的，则这样的每个 DSE 的类型中还包括shadow 。在不完整的复制区中，可能会存在类型为glue 的 DSE 来表示影像信息的各分散部分之间的桥接。没有任何用户属性或操作属性与这些粘接 DSE 相关联。

24.3.3 示例

图 22 举例说明了 DIT 的一部分(相应于一个命名上下文)如何映射到一个 DSA 信息树。除了命名上下文信息本身外，图中还描述了：包含其上级引用的 DSA 的根 DSE(本例中不是第一级 DSA 的 DSA 信息树)，一个粘接 DSE 和一个表示直接上级命名上下文引用的 DSE(或者是一个交叉引用，或者是一个直接上级引用)。



与一个命名上下文相符合的 DIT 子树　　命名上下文的 DSA 信息树

图 22　命名上下文的 DSE

第十一篇：DSA 操作框架

25　概述

25.1　定义

本目录规范使用下列术语和定义：

25.1.1

合作状态　cooperative state

与另一个 DSA 相关，表示 DSA 在与这个 DSA 的一个操作绑定实例已经建立但尚未终止期间的状态。

25.1.2

目录操作框架 directory operational framework

提供了一个框架，与目录组件(DSA)操作的某些特定方面(如影像或创建一个命名上下文)相关的特定操作模型可能通过该框架的应用派生出来。它提出了在目录组件的所有交互中都会出现的公共要素。

25.1.3

非合作状态 non-cooperative state

与另一个 DSA 相关，表示 DSA 在与这个 DSA 建立一个操作绑定实例之前或该操作绑定实例已经终止后的状态。

25.1.4

操作绑定 operational binding

指两个 DSA 之间的共同理解，一旦被建立，则表示它们"同意"参与后续的某些种类的交互。

25.1.5

操作绑定建立 operational binding establishment

建立一个操作绑定实例的过程。

25.1.6

操作绑定实例 operational binding instance

两个 DSA 之间某种特定类型的操作绑定。

25.1.7

操作绑定管理 operational binding management

建立、终止或修改一个操作绑定实例的过程。这种管理可以通过本系列目录规范所定义的信息交换实现，或者通过其他规范中定义的交互实现，或者通过其他方式实现等。

25.1.8

操作绑定修改 operational binding modification

修改一个操作绑定实例的过程。

25.1.9

操作绑定终止 operational binding termination

终止一个操作绑定实例的过程。

25.1.10

operational binding type 操作绑定类型

为了某种明确的目的而规定的操作绑定的一种特定类型，表达了两个 DSA 间参与特定类型的交互(如影像)的"商定"。

25.2 简介

本系列目录规范定义了应用协议信息交换和相关的定义目录分布式操作的 DSA 过程。第 25 章至第 28 章定义了一个 DSA 操作框架，对在这些信息交互和过程中的某些公共元素进行了建模。

两个 DSA 是以一种合作的方式进行交互的，因为，除了它们交互信息和执行与这些交互相关的过程时的技术能力外，每个 DSA 都被配置为接受与另一方 DSA 的某些类型的交互。

这几章所表述的都与一个公共框架相关，用以规范两个 DSA 间进行合作的元素结构。

该框架的一个目标是能够足够通用，可以解决在本系列目录规范的当前版本和后续版本可能定义的所有类型的 DSA 合作。该框架被用于本系列目录规范内，用以定义影像和层次化的操作绑定类型。

26 操作绑定

26.1 概述

第26章定义了一个通用框架，即DSA操作框架，在该框架内可以构建DSA组件间合作交互特性的规范，以达到取得共识的目标。

该通用框架提出了公共的特性，这些特性表现了DSA间交互的特色。通过将DSA操作框架应用于DSA间特定的合作交互方面，则得出的规范将是简明一致的，因此一个DSA须支持的机制的数量将得到缩减。

两个DSA间的共同理解，一旦被建立，则表示它们"同意"参与后续的某些种类的交互，这种共同理解被称为一个"操作绑定"。两个DSA可能会根据需要共享所需数量的某个特定类型的操作绑定实例。

DSA操作框架提供了一个公共的方法来定义一个*操作绑定类型*。一个操作绑定类型是为了某种明确的目的而规定的一种特定类型的操作绑定，表达了两个DSA间参与特定类型的交互(如影像)的"商定"。这种交互允许商定的一方或另一方能够调用一个定义良好的集合中的操作。

两个取得这样一个"商定"的特定的DSA，将共享一个特定操作绑定类型的一个操作绑定实例。它们被称为处于这样一个操作绑定类型实例的"*合作状态*"中。

在建立一个操作绑定实例之前或在终止一个操作绑定实例之后，这两个DSA被称为处于"*非合作状态*"中。

*操作绑定管理*是建立、终止或修改一个操作绑定实例的过程。这种管理可以通过本系列目录规范所定义的信息交换实现，或者通过其他规范中定义的交互实现，或者通过其他方式实现等。

这些通用概念如图23所描述。

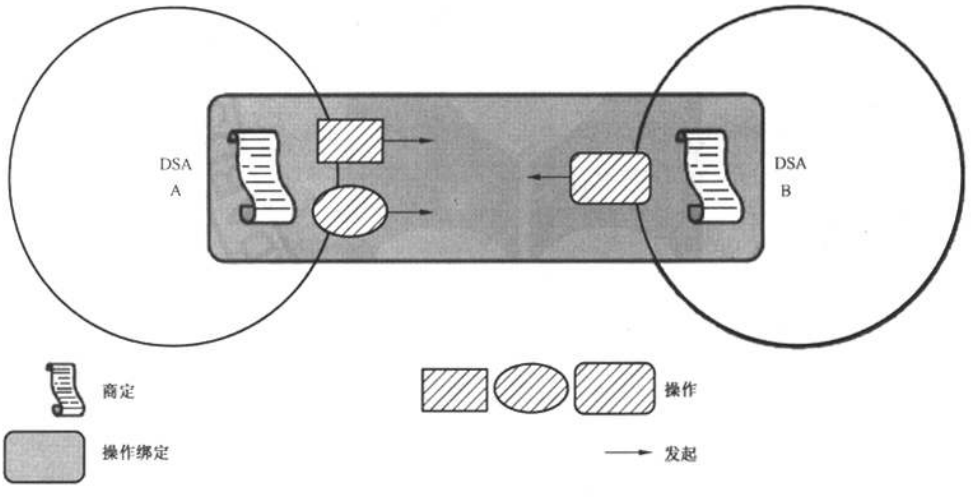


图23 操作绑定

26.2 操作框架的应用

应用DSA操作框架来定义一个操作绑定类型，有下述基本的元素参与：

a) 两个DSA；

b) 一个DSA将提供给另一个DSA的服务"商定"；

c) 一个或多个操作的集合以及相伴随的过程，DSA须遵循此过程才能够实现服务；

d) 在管理商定时所需要的DSA交互的规范。

这些基本元素之间的关系通过一个操作绑定来表示。一个操作绑定由这些基本元素集组成，用以通过技术术语来表示抽象的商定。它表示了一个由某个“商定”所控制的环境，在该环境内，一个 DSA 向另一个 DSA 提供某个已定义的服务(或者相反)。

26.2.1 两个 DSA

DSA 操作框架提供了一个结构，在该结构内，一个 DSA 与另一个 DSA 的交互以及它们因此而执行的过程可能会被规定。

两个 DSA 在操作绑定中可能会发挥相同的作用，在这种情况下，这两个 DSA 都可能管理操作绑定，两个 DSA 都可能互相之间调用相同的操作，且两个 DSA 都被限制为遵循相同的过程。这种情况被称为一个“对称的操作绑定”。

另一种情况是，两个 DSA 在操作绑定中可能发挥不同的作用，因此每个 DSA 都可能应用不同的操作和过程集。任意一个 DSA 或者两个 DSA 可能被用于管理操作绑定。这种情况被称为一个“非对称操作绑定”。

26.2.2 商定

“商定”是两个 DSA 的管理机构之间所取得的关于一个 DSA 必须向另一个 DSA 提供(或者相反)的服务的一种共同的理解。“商定”的最初形成是通过 DSA 的管理机构之间进行协商的，其协商方法不在本系列目录规范的定义范围之内。

该“商定”的参数可以形式化，方法是通过在 DSA 内记录在操作绑定管理的协议交互中需要用到的一个 ASN.1 数据类型。在这种方式下，两个 DSA 都可以取得对一方提供给另一方的服务的共同理解。

26.2.3 操作

操作是 DSA 间用来交互的基本媒介。为了提供已经达成共识的服务，一对 DSA 之间可能会传递一个或多个操作。

尽管一个 DSA 可能在技术上有能力支持大数据量的操作，但它可能仅愿意与另一个 DSA 在少量的操作中合作，或者与另一个 DSA 在某些参数的特定的值集内合作。

一个操作绑定类型的定义要求列举出可以进行交互的操作。它还允许对在操作中定义的参数值进行限制。

26.2.4 商定的管理

该框架为管理一个操作绑定的实例提供了通用的操作。这些操作可以对一个操作绑定进行建立、修改和终止。

可将该框架应用于对某个特定操作绑定类型进行规范，这就要求规定三个管理操作的每个发起者，并定义建立、修改和终止操作的过程。当一个管理操作应用于一个特定类型的操作绑定时，DSA 都必须遵循相应的过程。

26.3 合作的状态

通用的操作模型定义了两个 DSA 间合作的两种状态(一个 DSA 所看到的另一个 DSA 相关的状态)，由一个特定的操作绑定类型的实例所控制以及在两个状态之间的 3 种迁移。由两个 DSA 共享的一个操作绑定类型的每个实例都有各自的合作状态。合作的状态包括：

a) *非合作状态*：两个 DSA 之间的一个操作绑定类型的特定实例尚未被建立，或者已经被终止了。这两个 DSA 之间的(关于该操作绑定类型的实例)交互尚未定义。一个 DSA 被另一个与其处于非合作状态的 DSA 所联系，则它可能会拒绝与该 DSA 的任何合作，也可能会准备服务该请求。

b) *合作状态*：在两个 DSA 之间有一个操作绑定类型的实例正存在。它们的合作行为由操作绑定

类型及其参数的定义以及相关的过程所控制。

这两个合作状态之间的迁移可能会有两种调用方式:通过标准的协议交互或通过其他方式。

两个 DSA 间为管理一个操作绑定实例的交互(例如:建立和终止一个影像商定)与它们潜在地由绑定所控制的交互(例如:修改一个复制单元的交互)是不同的。

状态的迁移包括如下:

a) *建立*(establishment)迁移在两个 DSA 间创建一个特定类型的操作绑定的实例,结果是从非合作状态转移到合作状态;

b) *终止*(termination)迁移删除了两个 DSA 间的一个特定类型的操作绑定的实例,结果是从合作状态转移到非合作状态;

c) *修改*(modification)迁移修改了两个 DSA 间的一个特定类型的操作绑定实例的参数,结果是从合作状态转移到合作状态。

这些通用的状态及其迁移在图 24 中举例说明。

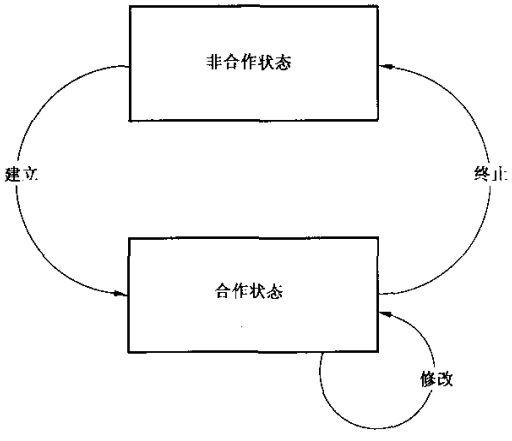


图 24 合作的状态

27 操作绑定规范和管理

27.1 操作绑定类型规范

当将该框架应用于定义一个特定类型的操作绑定时,该类型的下述特性应当被规定:

a) *对称(Symmetry)*

规定作为操作绑定组成部分的两个 DSA 的各自的作用。

操作绑定可能是对称的,在这种情况下,一个 DSA 的作用与另一个 DSA 的作用是可互换的,且两个 DSA 都表现了相同的外部交互行为。操作绑定还可能是非对称的,在这种情况下,每个 DSA 都发挥了不同的作用,且两个 DSA 表现了不同的外部交互行为。在后一种情况下,目录操作框架将这两种作用区分为抽象的"ROLE-A"和"ROLE-B"。

每个抽象作用"ROLE-A"和"ROLE-B"都必须与一个已定义语义的具体作用相关联(例如,"ROLE-A"为影像提供者,"ROLE-B"为影像使用者)。

b) *商定(Agreement)*

对"商定"组件的语义及其表示的定义。该信息确定了两个 DSA 之间一个操作绑定的特定实例的参数。

c) *发起者(Initiator)*

定义了两个抽象作用"ROLE-A"和"ROLE-B"中的哪一个被允许发起对该类型的一个操作绑定实例进行建立、修改和终止的操作。

d) *管理过程(Management procedures)*

在该类型的一个操作绑定被建立、修改或终止时,一个 DSA 应当遵循的一系列过程。

e) *类型标识符(Type identification)*

标识了由操作绑定所决定的 DSA 交互的类型。这些标识符的取值为客体标识符。

f) *应用上下文、操作和过程(Application-contexts, operations and procedures)*

标识了一系列的应用上下文,其操作(或操作的子集)可能在操作绑定的合作阶段被部署。

对于由该操作绑定类型所引用的每一个操作,如果该操作被调用,则要求对 DSA 所遵循的过程进行描述(这可能会通过引用本系列目录规范的另一部分来完成)。

对于那些使用本条所提供的通用操作绑定管理操作进行管理的操作绑定,应当使用本条所定义的 3 个信息客体类来指定绑带类型,它们是 OPERATIONAL-BINDING 、OP-BINDING-COOP 和 OP-BIND-ROLE 。

27.2 操作绑定管理

一般来说,对一个操作绑定的管理要求首先建立一个操作绑定实例。后续可选地会对初始商定的某些或所有参数进行一次或多次修改,最后可能包括将该操作绑定实例终止。关于一个实例如何被管理的精确细节在操作绑定类型的定义过程中进行定义。该类型定义要求规范如下内容:

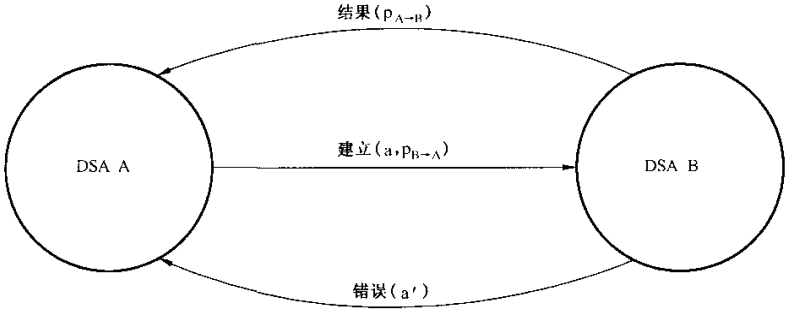
a) 每个管理操作的发起者(可能是两个 DSA 中的其中一个,或两个都是,或两个都不是);

b) 每个管理操作的参数;以及

c) 每个管理操作中,每个 DSA 应当遵循的过程。

在建立一个操作绑定实例的过程中,一个操作绑定实例标识符(绑定 id)被创建。该标识符,与包含在操作绑定内的两个 DSA 的可辨别名相结合,将构成该绑定实例的唯一标识符。后续所有的关于该操作绑定实例的管理操作都必须使用绑定 id 来标识正在被修改或终止的是哪个操作绑定实例。

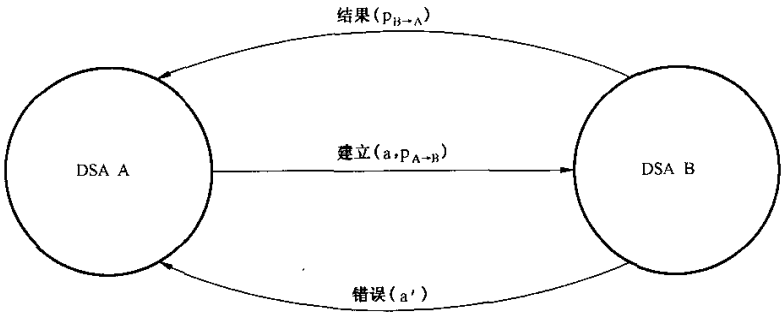
建立操作的发起者总是将"商定"的参数传递到第二个 DSA。另外,该发起者还可能会传递一些建立时的参数,这些参数在操作绑定中是与其作用相应的。如果响应 DSA 愿意进入该操作绑定中,则它可能会在结果中返回建立参数,这些参数是与响应 DSA 的作用相应的。如果响应 DSA 不愿意进入该操作绑定,则它必须返回一个错误,在错误中可能会可选地包含一个参数经过校正的商定。图 25 中描述了 Role A 是建立操作发起者的情况,图 26 中描述了 Role B 是建立操作发起者的情况。



a 商定

b 建立参数

图 25 具有 Role A 的 DSA 发起建立



a 商定

b 建立参数

图 26 具有 Role B 的 DSA 发起建立

27.3 操作绑定规范模板

对于一个特定类型的操作绑定的定义,可以使用下面 3 个 ASN.1 的信息客体类作为定义模板。它们允许使用 ASN.1 来定义那些可以形式化的操作绑定类型部分。而操作绑定类型的其他方面,如在建立或终止一个操作绑定时 DSA 必须遵循的过程,必须通过其他方式进行定义(可以以一种方式实现,这种方式类似于在 GB/T 16264.4—2008 中描述的名(称)解析过程中对 DSA 过程的非正式描述)。

27.3.1 操作绑定信息客体类

```
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 }
```

信息客体类 OPERATIONAL-BINDING 用做操作绑定类型定义的一个规范模板。为该类定义了一种变量记法来简化其作为一个模板的使用。一个操作绑定类型的定义与该变量记法的各字段之间的对应关系如下所述:

a) 该类型的操作绑定所使用的商定参数的 ASN.1 类型由字段 AGREEMENT 表示;

b) 应用上下文以及所定义类型的操作绑定实例在操作绑定的合作阶段被部署的这些应用上下文的操作等,在字段 APPLICATION-CONTEXTS 中被列举。被列举的应用上下文的所有操作都将被选择,除非可选字段 APPLIES TO 存在,且给出了一个操作引用的列表,该列表中的操作是从应用上下文中选择出的。该列表是一个客体类的集合,由信息客体类OPERATION 的实例组成;

c) 操作绑定的类别由字段 SYMMETRIC 或 ASYMMETRIC 来定义。在对称操作绑定的情况下，字段 SYMMETRIC 后跟随的是一个单独的信息客体类 OP-BIND-ROLE，该客体类对两种操作绑定的角色都有效。在非对称操作绑定的情况下，字段 ASYMMETRIC 后跟随的是两个信息客体类 OP-BIND-ROLE，其中一个由子字段 ROLE-A 所指向，另一个由子字段 ROLE-B 所指向；

d) 客体标识符值用于标识该类型的操作绑定，由字段ID来定义。

27.3.2 操作绑定合作信息客体类

```
OP-BINDING-COOP  ::=CLASS {
        &applContext    APPLICATION-CONTEXT,
        &Operations     OPERATION OPTIONAL }

WITH SYNTAX {
        &applContext
        [ APPLIES TO   &Operations ] }
```

信息客体类 OP-BINDING-COOP 作为标识一个已命名的应用上下文操作的规范模板，该应用上下文的某些方面由操作绑定所决定。该类的一个实例仅在一个特定操作绑定类型的上下文内有意义。为该类定义了一个可变的表示法来简化其作为一个模板的使用。一个操作绑定类型的定义与该可变表示法的各字段之间的对应关系如下所述：

a) 字段applContext 标识了一个应用上下文，该应用上下文的所有或部分操作都通过某种方式由一个操作绑定所决定；

b) 字段 APPLIES TO，如果存在，则标识了操作绑定所应用的特定操作。如果该字段不存在，则操作绑应用于该应用上下文的所有操作。

27.3.3 操作绑定角色信息客体类

```
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 ] }
```

信息客体类 OP-BIND-ROLE 作为定义一个操作绑定类型所起作用的一个规范模板。该类的一个实例仅在一个特定操作绑定类型的上下文内有意义。为该类定义了一个可变的表示法来简化其作为一个模板的使用。一个操作绑定类型的定义与该可变表示法的各字段之间的对应关系如下所述：

a) 字段 ESTABLISHMENT INITIATOR 指示：承担了所定义作用的 DSA 是否可能发起建立一个特定类型的操作绑定；

b) 字段 ESTABLISHMENT PARAMETER 定义：当操作绑定类型的一个实例被建立时，承担了所定义作用的 DSA 所交互的 ASN.1 类型；

c) 字段 MODIFICATION INITIATOR 指示：承担了所定义作用的 DSA 是否可能发起修改一个特定类型的操作绑定；

d) 字段 MODIFICATION PARAMETER 定义：当操作绑定类型的一个实例被修改时，承担了所定义作用的 DSA 所交互的 ASN.1 类型；

e) 字段TERMINATION INITIATOR 指示：承担了所定义作用的 DSA 是否可能终止一个特定类型的操作绑定；

f) 字段 TERMINATION PARAMETER 定义：当操作绑定类型的一个实例被终止时，承担了所定义作用的 DSA 所交互的 ASN.1 类型。

28 操作绑定管理的操作

本章定义了操作的一个集合，这些操作可用于建立、修改和终止各种类型的操作绑定。这些操作是通用的，因为它们能够被用于管理各种类型的操作绑定。这些操作规范使用了为某种类型的操作绑定所提供的定义模板，即应用了 OPERATIONAL-BINDING 信息客体类模板。

注：通过使用这种工具，任意类型的操作绑定都可能被管理。这些操作（以及相应的应用上下文）提供了关于 DSA 交互的扩展方式。将来可以定义新出现的操作绑定类型，以扩展 DSA 之间所提供的功能。

28.1 应用上下文定义

用于管理操作绑定实例的操作集合可被用来定义一个应用上下文，通过下面两种方式：

a) 一个应用上下文可被构造为仅包含用于操作绑定管理的操作。一个用于通用操作绑定管理的应用上下文在 GB/T 16264.5—2008 中定义。

在操作绑定的合作阶段可能交互的操作构成了一个或多个分离的应用上下文。

b) 一个操作集合能够被输入到模块中，用以定义一个特定的应用上下文。因此，用于操作绑定管理的操作能够与一个单独应用上下文内合作阶段的操作一起使用。

注：当 DSA 的一个专门组件希望使用一个仅为管理该 DSA 的操作绑定集合的关联，且它不准备接收任意一个为合作阶段所定义的操作（如 updateShadow）时，在这种情况下第一种方式有用。

28.2 建立操作绑定操作

“建立操作绑定”操作允许在两个 DSA 间建立一个预定义类型的操作绑定实例。这是通过传递建立参数以及在操作绑定类型的定义中所定义的商定条款来实现的。操作的变量可能由请求者签名（见 17.3）。如果请求者签名的话，则响应者可能会对结果进行签名。

在对称操作绑定的情况下，两个 DSA 中的任意一个都可能会主动建立一个预定义类型的操作绑定实例。

在非对称操作绑定的情况下，承担了“ROLE-A”或“ROLE-B”作用的某个 DSA 会建立一个操作绑定，这取决于该操作绑定类型的特定定义。

```
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,
                    ——对称的,Role A 发起,或 Role B 发起——
        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]   ValidityDEFAULT { },
        securityParameters  [8]   SecurityParameters OPTIONAL } }

OpBindingSet  OPERATIONAL-BINDING  ::={
        shadowOperationalBinding  |
        hierarchicalOperationalBinding  |
        nonSpecificHierarchicalOperationalBinding }

OperationalBindingID  ::=SEQUENCE {
        identifier  INTEGER,
        version  INTEGER }
```

组件bindingType 规定了哪种类型的操作绑定将被建立。操作绑定类型的定义是通过使用信息客体类模板 OPERATIONAL-BINDING 来定义的,该模板为操作绑定类型分配了一个客体标识符值。组件bindingType 的值来自 OpBindingSet 所指向的某个操作绑定类型的一个实例内的 ID 字段。该集合为EstablishOperationalBindingArgument 中的一个参数,是一个参数化类型。

发起请求的 DSA 可能会通过组件bindingID 为操作绑定实例分配一个标识符。如果组件bindingID 没有出现在操作的参数中,则响应的 DSA 必须为操作绑定实例分配一个ID ,并且在establishOperationalBindingResult 的组件bindingID 中将此ID 返回。在这两种情况下,当建立一个操作绑定时,OperationalBindingID 值的两个组件identifier 和version 都必须被分配,并且由执行分配的 DSA 将值发布。

组件accessPoint 规定了请求发起者进行后续交互的访问点。

发起"建立操作绑定"操作的 DSA 所承担的角色由 CHOICE 类型所指示,其可选项为 symmetric 、roleA-initiates 和roleB-initiates 。CHOICE 选项控制了由请求发起方 DSA 和响应方 DSA 所部署的特定的建立参数。角色的语义定义是操作绑定类型定义中的组成部分。CHOICE 的 ASN.1 类型通过请求发起者的OP-BIND-ROLE 信息客体类模板中的ESTABLISHMENT PARAMETER 来决定。如果操作绑定类型的建立不需要从请求者处获取建立参数,则该CHOICE 类型被忽略。

组件agreement 包含了控制操作绑定实例的商定的条款。它的实际内容取决于要建立的操作绑定的类型。该参数的 ASN.1 类型由该操作绑定类型的信息客体类模板OPERATIONAL-BINDING 中的字段AGREEMENT 来定义。

操作绑定实例必须存在的持续时间在valid 内定义。该操作绑定实例存在的起始时间由validFrom 规定,而操作绑定实例的终止时间由validUntil 给出。

```
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 }
```

在任何一个对比操作中使用Time值之前，且如果 Time 的句法选择了UTCTime类型，则两位数的年字段必须合理转化为 4 位数的年值，规则如下：

——如果两位数的值为 00 到 49，则最后的值应当是加上 2000 后的值；

——如果两位数的值为 50 到 99，则最后的值应当是加上 1900 后的值。

使用GeneralizedTime 可能会阻止与某些实现的互操作，这些实现不关注选择UTCTime 或GeneralizedTime 的可能性。谁规定了本目录规范将应用的域，例如profiling groups，则应当由谁来负责什么时候可能会使用GeneralizedTime 。在任何情况下，UTCTime 都不能被用于表示超出 2049 年的日期。

如果“建立操作绑定”的操作成功，则将返回如下结果，并且可能被响应者签名(见 17.3)。

```
EstablishOperationalBindingResult  ::=OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType  [0]    OPERATIONAL-BINDING. &id ({OpBindingSet}),
        bindingID    [1]    OperationalBindingID OPTIONAL,
        accessPoint  [2]    AccessPoint,
        ——对称的，Role A 响应，或 Role B 响应
        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 } }
```

结果中包含的组件 bindingType 指示了CHOICE 元素内使用的操作绑定的类型。它的值与建立请求的发起者所提供的值相同，且来自于 OpBindingSet 所指向的某个操作绑定类型的一个实例内的ID字段。

该集合为EstablishOperationalBindingResult 中的一个参数，是一个参数化类型。

所建立的操作绑定实例的标识符可能在 bindingID 中返回。在任何一个后续的修改或终止操作绑定的操作中，应当使用此标识符来标识该操作绑定实例，该标识符还可用于在所建立的操作绑定实例的合作阶段所执行的任何其他的操作中。

组件accessPoint 规定了请求响应者进行后续交互的访问点。

请求发起方的 DSA 可能会通过组件bindingID 为操作绑定实例分配一个标识符。如果在操作参数中没有出现bindingID，则响应者 DSA 须为操作绑定实例分配一个ID，并且在 establishOperationalBindingResult 的组件bindingID 中将此ID 返回。

对“建立操作绑定”操作进行响应的 DSA 所承担的角色由CHOICE 类型所指示，其可选项为sym-

metric、roleA-initiates 和 roleB-initiates。角色的语义定义是操作绑定类型定义中的组成部分。CHOICE 的 ASN.1 类型通过请求响应者的OP-BIND-ROLE 信息客体类模板中的ESTABLISHMENT PARAMETER 来决定。如果操作绑定类型的建立不需要从响应者处获取建立参数，则该CHOICE 类型被忽略。

28.3 修改操作绑定操作

"修改操作绑定"操作被用于修改一个已经建立的操作绑定。修改权限通过操作绑定类型定义中的字段MODIFICATION INITIATOR 来指定，操作绑定类型的定义使用了信息客体类模板OP-BIND-ROLE 和OPERATIONAL-BINDING。

一个操作绑定中可以被修改的组件是操作绑定及其有效期商定中所规定的内容。另外，修改参数能够由请求发起者来指定。操作的变量可能被请求者签名(见 17.3)。如果请求者签名的话，则响应者可能会对结果进行签名。

```
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,
        ——对称的，Role A 发起，或 Role B 发起——
        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 } }
```

组件 bindingType 规定哪种类型的操作绑定将被修改。组件bindingType 来自于OpBindingSet 所指向的某个操作绑定类型的一个实例内的 ID 字段。该集合为ModifyOperationalBindingArgument 中的一个参数，是一个参数化类型。

要修改的操作绑定实例的标识符由bindingID 给出。修正后的操作绑定实例的标识符由newBindingID 给出。newBindingID 的version 组件值须大于bindingID 的version 组件值。

如果请求发起者后续交互的访问点要被修改，则可选组件accessPoint 须存在。

发起"修改操作绑定"操作的 DSA 所承担的角色由 CHOICE 类型所指示，其可选项为 symmetric、roleA-initiates 和roleB-initiates。角色的语义定义是操作绑定类型定义中的组成部分。CHOICE 的 ASN.1 类型通过请求发起者的 OP-BIND-ROLE 信息客体类模板中的MODIFICATION PARAME-

TER 来决定。如果操作绑定类型的修改不需要从请求者处获取修改参数，则该 CHOICE 类型被忽略。

组件newAgreement 如果存在的话，包括了对控制操作绑定实例的商定中要修改的条款。该参数的 ASN.1 类型由该操作绑定类型的信息客体类模板OPERATIONAL-BINDING 中的字段 AGREEMENT 来定义。如果newAgreement 组件没有出现，则操作不对商定中的参数进行修改。

可选组件 valid 用于指示改变后的商定的有效时长。如果valid 组件没有出现，则假设validFrom 组件的值为now，而validUntil 组件的值假设为不变。如果 validFrom 组件存在，并指向未来一个具体的时间，则当前的商定将一直生效直到该时间点为止。

如果“修改操作绑定”的操作成功，则将返回下述结果，并且可能被响应者签名(见 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 } }
```

对于响应者 DSA 而言，它不可能向修改发起者返回根据它的角色所定义的修改参数。

28.4 终止操作绑定操作

“终止操作绑定”操作用于请求终止一个已经建立的操作绑定实例。终止权限通过操作绑定类型定义中的字段TERMINATION INITIATOR 来指定，操作绑定类型的定义使用了信息客体类模板 OP-BIND-ROLE 和 OPERATIONAL-BINDING 。操作的变量可能被请求者签名(见 17.3)。如果请求者签名的话，则响应者可能会对结果进行签名。

```
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,
      对称的，Role A 发起，或Role B 发起——

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 } }
```

组件bindingType 规定哪种类型的操作绑定将被终止。组件bindingType 来自于OpBindingSet 所指向的某个操作绑定类型的一个实例内的 ID 字段。该集合为 TerminateOperationalBindingArgument 中的一个参数，是一个参数化类型。

要终止的操作绑定实例的标识符由bindingID 给出。组件bindingID 中出现的version 组件被忽略。

发起"终止操作绑定"操作的 DSA 所承担的角色由CHOICE 类型所指示，其可选项为symmetric 、roleA-initiates 和roleB-initiates 。角色的语义定义是操作绑定类型定义中的组成部分。CHOICE 的 ASN.1 类型由请求发起者的OP-BIND-ROLE 信息客体类模板中的TERMINATION PARAMETER 来决定。如果操作绑定类型的终止不需要从请求者处获取终止参数，则该CHOICE 类型被忽略。

如果该操作绑定不是被立即终止，则可在terminateAt 中定义一个延时的终止时间。

如果"终止操作绑定"的操作成功，则将返回下述结果，并且可能被响应者签名(见 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 } } }
```

对于响应者 DSA 而言，它不可能向终止请求发起者返回根据它的角色所定义的终止参数。

28.5 操作绑定错误

一个操作绑定错误报告了与操作绑定的管理操作的使用相关的问题。错误的参数可能被响应者签名(见 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 }

problem 的取值有如下含义：

a) invalidID：请求中给出的操作绑定 ID 对接收方 DSA 来说是未知的，或者对于被请求的操作而言是处于错误的状态。

b) duplicateID：在建立请求中给出的操作绑定 ID 在响应者处已经存在了。这种情况的出现可能是由于之前有一个建立操作绑定实例的请求，但结果丢失，因此请求发起者重复了此建立请求。

c) unsupportedBindingType：被请求的操作绑定的类型不被 DSA 所支持。

d) notAllowedForRole：对操作绑定实例所发起的管理操作请求，对该发起者所承担的角色而言是不允许的(例如，一个不允许发起终止操作绑定实例的 DSA 发起了一个终止操作绑定的操作)。

e) parametersMissing：为该类型的操作绑定所定义的任意一个所需的建立参数或终止参数缺失。

f) roleAssignment：一个非对称操作绑定实例所请求的角色分配没有被接受。

g) invalidStartTime：为一个操作绑定实例所指定的起始时间没有被接受。

h) invalidEndTime：为一个操作绑定实例所指定的终止时间没有被接受。

i) invalidAgreement：被请求的操作绑定实例的商定条款没有被接受。响应方 DSA 可以接受的商定条款可在 agreementProposal 内返回。

j) currentlyNotDecidable：DSA 不能够在线决定对于被请求的操作绑定实例的修改或终止。什么时间能够再次重复发出该请求，在 retryAt 中给出。

k) modificationNotAllowed："修改操作绑定"操作被拒绝，由于该绑定实例不允许进行修改。

组件bindingType 的值须与失败的操作绑定管理操作的发起者所传递的值相同。

组件agreementProposal 仅须用于对一个EstablishOperationalBinding 请求进行响应，以提出一个修正后的商定参数集，如 28.2 所描述。

组件retryAt 仅须与 problem 的值currentlyNotDecidable 共同使用，以指示什么时间EstablishOperationalBinding 或ModifyOperationalBinding 请求能够再次尝试。

组件CommonResultsSeq (见 GB/T 16264.3—2008 的 7.4)中包括SecurityParameters 。而如果错误参数将被响应者签名时，则组件 SecurityParameters (见 GB/T 16264.3—2008 的 7.10)应当包括在 CommonResultsSeq 中。

28.6 操作绑定管理的绑定和解绑定

DSA 操作绑定管理绑定操作和 DSA 操作绑定管理解绑定操作，在 28.6.1 和 28.6.2 中定义，由 DSA 在操作绑定管理活动的一个特定周期的起始和结束时所使用。

对于dSAOperationalBindingManagementBind 和dSAOperationalBindingManagementUnbind 的保护须与应用于DSABind 和DSAUnbind 操作的保护相同。

注：鉴别所要求的证书可能会由安全交互服务元素所承载(见 GB/T 16264.5—2008)，在这种情况下，它们不会出现在绑定参数或结果中。

28.6.1 DSA 操作绑定管理绑定

一个 DSA 操作绑定管理绑定操作用于开始一个操作绑定管理周期。

dSAOperationalBindingManagementBind OPERATION ::=directoryBind

dSAOperationalManagementBind 中的组件与在directoryBind (见 GB/T 16264.3—2008)中定义的相应组件相同，除了下面的一些区别外。

注：鉴别所要求的证书可能会由安全交互服务元素所承载(见 GB/T 16264.5—2008)，在这种情况下，它们不会出现在绑定参数或结果中。

28.6.1.1 发起者证书

DirectoryBindArgument 中的Credentials 允许标识发起方 DSA 的 AE-Title 的信息被发送到响应方 DSA。AE-title 的格式应当采用目录可辨别名格式。

28.6.1.2 响应者证书

DirectoryBindResult 中的Credentials 允许标识响应方 DSA 的AE-Title 的信息被发送到发起方 DSA。AE-title 的格式须采用目录可辨别名格式。

28.6.2 DSA 操作绑定管理解绑定

在一个提供操作绑定管理的周期结束时将执行解绑定，为 OSI 环境使用的解绑定在 GB/T 16264.5—2008 的 7.6.4 和 7.6.5 中定义，为 TCP/IP 环境使用的解绑定在 GB/T 16264.5—2008 的 9.3.2 中定义。

附 录 A
（规范性附录）
客体标识符的用法

本附录记录了客体标识符子树的上层分支，在这棵子树上有本系列目录规范中所分配的所有客体标识符。这是通过提供一个名为 UsefulDefinitions 的 ASN.1 模块来实现的，在该模块中，子树中的所有非叶结点都被分配了名(称)。

```
UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
```

——本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供其使用，

——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。

——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行的扩展和修改。

```
ID                  ::=   OBJECT IDENTIFIER
ds          ID      ::=   {joint-iso-itu-t ds(5)}
——信息客体的种类——
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}--本规范当前没有定义
controlAttributeType               ID  ::=  {ds 37}
keyPurposes                        ID  ::=  {ds 38}
——模块——
usefulDefinitions                  ID  ::=  {module usefulDefinitions(0) 5}
informationFramework               ID  ::=  {module informationFramework(1) 5}
directoryAbstractService           ID  ::=  {module directoryAbstractService(2) 5}
distributedOperations              ID  ::=  {module distributedOperations(3) 5}
--protocolObjectIdentifiers        ID  ::=  {module protocolObjectIdentifiers(4) 5}
selectedAttributeTypes             ID  ::=  {module selectedAttributeTypes(5) 5}
selectedObjectClasses              ID  ::=  {module selectedObjectClasses(6) 5}
authenticationFramework            ID  ::=  {module authenticationFramework(7) 5}
algorithmObjectIdentifiers         ID  ::=  {module algorithmObjectIdentifiers(8) 5}
directoryObjectIdentifiers         ID  ::=  {module directoryObjectIdentifiers(9) 5}
upperBounds                        ID  ::=  {module upperBounds(10) 5}
--dap                             ID  ::=  {module dap(11) 5}
--dsp                             ID  ::=  {module dsp(12) 5}
distributedDirectoryOIDs           ID  ::=  {module distributedDirectoryOIDs(13) 5}
directoryShadowOIDs                ID  ::=  {module directoryShadowOIDs(14) 5}
directoryShadowAbstractService     ID  ::=  {module directoryShadowAbstractService(15) 5}
--disp                            ID  ::=  {module disp(16) 5}
--dop                             ID  ::=  {module dop(17) 5}
opBindingManagement                ID  ::=  {module opBindingManagement(18) 5}
opBindingOIDs                      ID  ::=  {module opBindingOIDs(19) 5}
hierarchicalOperationalBindings    ID  ::=  {module hierarchicalOperationalBindings(20) 5}
dsaOperationalAttributeTypes       ID  ::=  {module dsaOperationalAttributeTypes(22) 5}
schemaAdministration               ID  ::=  {module schemaAdministration(23) 5}
basicAccessControl                 ID  ::=  {module basicAccessControl(24) 5}
directoryOperationalBindingTypes   ID  ::=  {module directoryOperationalBindingTypes(25) 5}
certificateExtensions              ID  ::=  {module certificateExtensions(26) 5}
```

```
directoryManagement                ID   ::=  {module directoryManagement(27) 5}
enhancedSecurity                   ID   ::=  {module enhancedSecurity (28) 5}
--directorySecurityExchanges       ID   ::=  {module directorySecurityExchanges (29) 5}
iDMProtocolSpecification           ID   ::=  {module iDMProtocolSpecification(30) 5}
directoryIDMProtocols              ID   ::=  {module directoryIDMProtocols(31) 5}
attributeCertificateDefinitions    ID   ::=  {module attributeCertificateDefinitions(32) 5}
serviceAdministration              ID   ::=  {module serviceAdministration(33) 5}
——下述定义是为这样的模块而定义,该模块拥有外部定义的模式元素,这些模式元素不是使用
——使用正式的ASN.1 记法(见实现者指南的最新版本)
externalDefinitions                ID   ::=  {module externalDefinitions(34) }
commonProtocolSpecification        ID   ::=  {module commonProtocolSpecification (35) 5}
oSIProtocolSpecification           ID   ::=  {module oSIProtocolSpecification (36) 5}
directoryOSIProtocols              ID   ::=  {module directoryOSIProtocols (37) 5}
——同义词——
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
id-kp                              ID   ::=  keyPurposes
```

```
——废弃的模块标识符——
--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}
——未使用的模块标识符——
--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
```

附 录 B
（规范性附录）
用 ASN.1 描述的信息框架

本附录提供了本目录规范中包含的所有 ASN.1 类型、值和宏定义的摘要。这些定义构成了 ASN.1 模块InformationFramework 。

```
InformationFramework {joint-iso-itu-t ds(5) module(1) informationFramework(1) 5}
DEFINITIONS ::=
BEGIN

--EXPORTS All--
--本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供
  其使用，
——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。
——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行
  的扩展和修改。
IMPORTS
——来自GB/T 16264.2—2008
    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) 5}

    SearchRule
        FROM ServiceAdministration serviceAdministration
——来自GB/T 16264.3—2008
    TypeAndContextAssertion
        FROM DirectoryAbstractService directoryAbstractService
——来自GB/T 16264.6—2008
    booleanMatch, commonName, DirectoryString {},
    generalizedTimeMatch, generalizedTimeOrderingMatch,
    integerFirstComponentMatch, integerMatch,
    integerOrderingMatch, objectIdentifierFirstComponentMatch
        FROM SelectedAttributeTypes selectedAttributeTypes

    ub-search
        FROM UpperBounds upperBounds;
--属性数据类型 --
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 }
```

——下述信息客体集的定义被推迟，可能推迟到标准化概要或协议实现一致性声明时。

——要求该集合为属性的值组件、AttributeTypeAndValue 的值组件以及属性值断言的。

——声明组件等规定一个表限制。

```
SupportedAttributes ATTRIBUTE ::= { objectClass | aliasedEntryName, ... }
```

——下述信息客体集的定义被推迟，可能推迟到标准化概要或协议实现一致性声明时。

——要求该集合为上下文规范规定一个表限制。

```
SupportedContexts CONTEXT ::= { ··· }
```

--命名数据类型--

```
Name ::= CHOICE { --目前仅有一种可能--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 }
--子树数据类型--
SubtreeSpecification ::= SEQUENCE {
    base                [0]  LocalName DEFAULT { },
                             COMPONENTS OF ChopSpecification,
    specificationFilter [4]  Refinement OPTIONAL }
      ——空序列表示整个管理区
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 信息客体类规范--
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) }
--客体类--
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 信息客体类规范--
ATTRIBUTE ::= CLASS {
        &derivation                     ATTRIBUTE OPTIONAL,
        &Type                           OPTIONAL,--或者是&Type ,或者是所需的&derivation --
        &equality-match                 MATCHING-RULE OPTIONAL,
        &ordering-match                 MATCHING-RULE OPTIONAL,
        &substrings-match               MATCHING-RULE OPTIONAL,
        &single-valued                  BOOLEAN DEFAULT FALSE,
        &collective                     BOOLEAN DEFAULT FALSE,
        &dummy                          BOOLEAN DEFAULT FALSE,
        ——操作扩展——
        &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 ]
        [ DUMMY                             &dummy ]
        [ NO USER MODIFICATION              &no-user-modification ]
        [ USAGE                             &usage ]
        ID                                  &id }

AttributeUsage ::= ENUMERATED
        { userApplications      (0),
        directoryOperation      (1),
        distributedOperation    (2),
        dSAOperation            (3) }
--属性--
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 信息客体类规范--
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 }
```

```
--匹配规则--
        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.&id           (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 信息客体类规范--
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 类和 DIT 结构规则数据类型--
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 ,
                                    ——须在子模式范围内是唯一的
        nameForm                    NAME-FORM.&id,
        superiorStructureRules      SET SIZE (1..MAX) OF RuleIdentifier OPTIONAL }
RuleIdentifier ::= INTEGER
--CONTENT-RULE 类和 DIT 内容规则数据类型--
CONTENT-RULE ::= CLASS {
        &structuralClass            OBJECT-CLASS.&id            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,
        &DefaultValue           OPTIONAL,
```

```
        &Assertion              OPTIONAL,
        &absentMatch            BOOLEAN DEFAULT TRUE,
        &id                     OBJECT IDENTIFIER UNIQUE }

    WITH SYNTAX {
        WITH SYNTAX             &Type
        [ DEFAULT-VALUE         &DefaultValue ]
        [ ASSERTED AS           &Assertion ]
        [ ABSENT-MATCH          &absentMatch ]
        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 ] }

FRIENDS ::= CLASS {
        &anchor                 ATTRIBUTE.&id UNIQUE,
        &Friends                ATTRIBUTE }

WITH SYNTAX {
        ANCHOR                  &anchor
        FRIENDS                 &Friends }
--系统模式信息客体--
--客体类--
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 }
--属性--
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
                                -- 按照 GB/T 16262.1—2006 中 42.3b) 或 c)的定义
        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
                                -- 按照 GB/T 16262.1—2006 中 42.3b) 或 c)的定义
        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
                        -- 按照 GB/T 16262.1-2006  中 42.3b) 或 c)的定义
        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          distinguished NameMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-creatorsName }

modifiersName ATTRIBUTE ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedNameM
                                        atch
        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          Boolean
        Match SINGLE VALUE          TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-oa-hierarchyBelow }

HierarchyBelow ::= BOOLEAN
hierarchyParent ATTRIBUTE ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedName
```

```
        Match SINGLE VALUE              TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyParent }

hierarchyTop ATTRIBUTE ::= {
        WITH SYNTAX                     DistinguishedName
        EQUALITY MATCHING RULE          distinguishedName
        Match SINGLE VALUE              TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyTop }
--客体标识符的分配--
--客体类--
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}
--属性--
id-at-objectClass                       OBJECT IDENTIFIER   ::=   {id-at 0}
id-at-aliasedEntryName                  OBJECT IDENTIFIER   ::=   {id-at 1}
--匹配规则--
id-mr-objectIdentifierMatch             OBJECT IDENTIFIER   ::=   {id-mr 0}
id-mr-distinguishedNameMatch            OBJECT IDENTIFIER   ::=   {id-mr 1}
--操作属性--
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}
```

```
id-oa-hierarchyTop                      OBJECT IDENTIFIER  ::=  {id-oa 20}
--子条目类--
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}
--名(称)格式--
id-nf-subentryNameForm                  OBJECT IDENTIFIER  ::=  {id-nf 16}
--管理者角色--
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  --信息框架
```

附 录 C
（规范性附录）
用 ASN.1 描述的子模式管理模式

本附录包含了第 15 章中定义的子模式管理的 ASN.1 类型、值和信息客体的定义，其形式为 ASN.1 模块 SchemaAdministration 。

```
SchemaAdministration {joint-iso-itu-t ds(5) module(1) schemaAdministration(23) 5}
DEFINITIONS ::=
BEGIN

--EXPORTS All--
——本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供
  其使用，
——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。
——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行
  的扩展和修改。
IMPORTS
——来自 GB/T 16264.2—2008
    id-soa, id-soc, informationFramework, selectedAttributeTypes, upperBounds
        FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

    ATTRIBUTE, AttributeUsage, CONTEXT, DITContentRule, DITStructureRule, MATCH-
ING-RULE, NAME-FORM, OBJECT-CLASS, ObjectClassKind, objectIdentifierMatch
        FROM InformationFramework informationFramework
——来自 GB/T 16264.6—2008
    DirectoryString {}, integerFirstComponentMatch, integerMatch,
    objectIdentifierFirstComponentMatch
        FROM SelectedAttributeTypes selectedAttributeTypes
    ub-schema
        FROM UpperBounds upperBounds;
--类型--
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 }
                        -描述了 ASN.1 句法

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 }
--客体类--
subschema  OBJECT-CLASS  ::=  {
        KIND                auxiliary
        MAY CONTAIN  {
                            dITStructureRules  |
```

```
                        nameForms     |
                        dITContentRules    |
                        objectClasses    |
                        attributeTypes    |
                        friends    |
                        contextTypes    |
                        dITContextUse    |
                        matchingRules    |
                        matchingRuleUse    }
        ID              id-soc-subschema    }
--属性--
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 }

friends ATTRIBUTE ::= {
        WITH SYNTAX                 FriendsDescription
        EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
        USAGE                       directoryOperation
        ID                          id-soa-friends }
```

```
FriendsDescription  ::=  SEQUENCE  {
        anchor          ATTRIBUTE. &id,
        name            SET SIZE (1..MAX) OF DirectoryString { ub-schema }  OPTIONAL,
        description     DirectoryString { ub-schema }          OPTIONAL,
        obsolete        BOOLEAN DEFAULT FALSE,
        friends         [0]  SET OF ATTRIBUTE. &id }
--客体标识符分配--
--模式客体类--
id-soc-subschema    OBJECT  IDENTIFIER  ::=  {id-soc  1}
--模式操作属性--
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}
id-soa-friends                   OBJECT IDENTIFIER  ::=  {id-soa 13}

END  --子模式管理
```

附 录 D
（规范性附录）
用 ASN.1 描述的服务管理

本附录包含了第 16 章中定义的服务管理的 ASN.1 类型、值和信息客体的定义，其形式为 ASN.1 模块 ServiceAdministration 。

```
ServiceAdministration {joint-iso-itu-t ds(5) module(1) serviceAdministration(33) 5}
DEFINITIONS ::=
BEGIN

--EXPORTS All--
——本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供
  其使用，
——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。
——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行
  的扩展和修改。
IMPORTS
——来自 GB/T 16264.2—2008
        directoryAbstractService, informationFramework
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        ATTRIBUTE, AttributeType, CONTEXT, MATCHING-RULE, OBJECT-CLASS,
        SupportedAttributes, SupportedContexts
                FROM InformationFramework informationFramework
——来自 GB/T 16264.3—2008
        FamilyGrouping, FamilyReturn, HierarchySelections, SearchControlOptions,
        ServiceControlOptions
                FROM DirectoryAbstractService directoryAbstractService;
--类型--
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]  BOOLEN       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}) }
--下述信息客体集的定义被推迟，可能推迟到标准化概要或协议实现一致性声明时。
```

```
--需要该集合对组件 SupportedMatchingRestrictions 指定一个表限制。
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, --如果 tightenings 存在,则为必选
        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 { --必须是一个
                                    --基于映射的匹配算法的客体标识符 --} ),
        level       INTEGER  DEFAULT 0 }

MRSubstitution  ::=  SEQUENCE {
```

```
        attribute       AttributeType,
        oldMatchingRule     [0]  MATCHING-RULE.&id  OPTIONAL,
        newMatchingRule     [1]  MATCHING-RULE.&id  OPTIONAL }
--ASN.1 信息客体类--
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  ——服务管理
```

附 录 E
（规范性附录）
用 ASN.1 描述的基本访问控制

本附录提供了基本访问控制的所有 ASN.1 类型和值定义的摘要，该定义构成了 ASN.1 模块BasicAccessControl 。

```
BasicAccessControl {joint-iso-itu-t ds(5) module(1) basicAccessControl(24) 5}
DEFINITIONS ::=
BEGIN

--EXPORTS All--
——本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供
    其使用，
——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。
——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行
    的扩展和修改。
IMPORTS
——来自 GB/T 16264.2—2008
        directoryAbstractService, id-aca, id-acScheme, informationFramework,
        selectedAttributeTypes, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}
        ATTRIBUTE, AttributeType, ContextAssertion, DistinguishedName, MATCHING-RULE,
        objectIdentifierMatch, Refinement, SubtreeSpecification, SupportedAttributes
                FROM InformationFramework informationFramework
——来自 GB/T 16264.3—2008
        Filter
                FROM DirectoryAbstractService directoryAbstractService
——来自 GB/T 16264.6—2008
        DirectoryString{}, directoryStringFirstComponentMatch, NameAndOptionalUID,
        UniqueIdentifier
                FROM SelectedAttributeTypes selectedAttributeTypes
        ub-tag  FROM UpperBounds upperBounds;
--类型--
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 组件须是一个名为 GroupOfUniqueNames 的条目的名(称)
        subtree         [4] SET SIZE (1..MAX) OF SubtreeSpecification    OPTIONAL }

ItemPermission  ::=  SEQUENCE  {
        precedence      Precedence OPTIONAL,
            --缺省值为在 ACIItem 中定义的优先级
        userClasses     UserClasses,
        grantsAndDenials  GrantsAndDenials }
```

```
UserPermission ::= SEQUENCE {
        precedence          Precedence OPTIONAL,
                    --缺省值为在 ACIItem 中定义的优先级
        protectedItems      ProtectedItems,
        grantsAndDenials    GrantsAndDenials }

AuthenticationLevel ::= CHOICE {
        basicLevels SEQUENCE {
                level               ENUMERATED { none (0), simple (1), strong (2) },
                localQualifier          INTEGER OPTIONAL,
                signed                      BOOLEAN DEFAULT FALSE },
        other               EXTERNAL }

GrantsAndDenials ::= BIT STRING {
        --可与被保护项的任意组件联合使用的许可
        grantAdd        (0),
        denyAdd         (1),
        grantDiscloseOnError    (2),
        denyDiscloseOnError     (3),
        grantRead       (4),
        denyRead        (5),
        grantRemove     (6),
        denyRemove      (7),
        --可仅与条目组件联合使用的许可
        grantBrowse     (8),
        denyBrowse      (9),
        grantExport     (10),
        denyExport      (11),
        grantImport     (12),
        denyImport      (13),
        grantModify     (14),
        denyModify      (15),
        grantRename     (16),
        denyRename      (17),
        grantReturnDN   (18),
        denyReturnDN    (19),
        --可能会与被保护项的除条目外的任意组件联合使用的许可
        grantCompare    (20),
        denyCompare     (21),
        grantFilterMatch    (22),
        denyFilterMatch     (23),
        grantInvoke     (24),
        denyInvoke      (25) }
```

```
AttributeTypeAndValue ::= SEQUENCE {
        type        ATTRIBUTE.&id ({SupportedAttributes}),
        value       ATTRIBUTE.&Type({SupportedAttributes}{@type}) }
--属性--
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 }
--客体标识符分配--
--属性--
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 }
--访问控制方案--
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 --基本访问控制
```

附 录 F
（规范性附录）
用 ASN.1 描述的 DSA 操作属性类型

本附录包含了第 23 章和第 24 章所定义所有 ASN.1 类型和值的定义，其形式为 ASN.1 模块 DSAOperationalAttributeTypes 。

```
DSAOperationalAttributeTypes {joint-iso-itu-t ds(5) module(1) dsaOperationalAttributeTypes(22) 5}
DEFINITIONS ::=
BEGIN

--EXPORTS All--
——本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供
  其使用，
——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。
——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行
  的扩展和修改。
IMPORTS
——来自 GB/T 16264.2—2008
        distributedOperations, id-doa, id-kmr, informationFramework, opBindingManagement,
        selectedAttributeTypes, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5 }

        ATTRIBUTE, MATCHING-RULE, Name
                FROM InformationFramework informationFramework

        OperationalBindingID
                FROM OperationalBindingManagement opBindingManagement
——来自 GB/T 16264.4 —2008
  AccessPoint, DitBridgeKnowledge, MasterAndShadowAccessPoints
    FROM DistributedOperations distributedOperations
——来自 GB/T 16264.6—2008
 bitStringMatch,  directoryStringFirstComponentMatch
    FROM SelectedAttributeTypes selectedAttributeTypes ;
--数据类型--
DSEType  ::=  BIT STRING {
        root            (0),    --根 DSE--
        glue            (1),    --仅表示一个名(称)的知识--
        cp              (2),    --上下文前缀--
        entry           (3),    --客体条目--
        alias           (4),    --别名条目--
        subr            (5),    --下级引用--
        nssr            (6),    --非特定下级引用--
```

```
        supr            (7),    --上级引用--
        xr              (8),    --交叉引用--
        admPoint        (9),    --管理点--
        subentry        (10),   --子条目--
        shadow          (11),   --影像拷贝--
        immSupr         (13),   --直接上级引用--
        rhob            (14),   --rhob 信息--
        sa              (15),   --别名条目的下级引用--
        dsSubentry      (16),   --DSA 特定子条目--
        familyMember    (17),   --家族成员--
        ditBridge       (18),   --DIT 桥接引用--
        writeableCopy   (19) }  --可写拷贝--

SupplierOrConsumer ::= SET {
        COMPONENTS OF   AccessPoint, --提供者或使用者--
        agreementID [3] OperationalBindingID }

SupplierInformation ::= SET {
        COMPONENTS OF   SupplierOrConsumer, --提供者--
        supplier-is-master [4] BOOLEAN DEFAULT TRUE,
        non-supplying-master [5] AccessPoint OPTIONAL }

ConsumerInformation ::= SupplierOrConsumer --使用者--

SupplierAndConsumers ::= SET {
        COMPONENTS OF   AccessPoint,  --提供者--
        consumers       [3] SET OF AccessPoint }
--属性类型--
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 }
```

```
ditBridgeKnowledge  ATTRIBUTE  ::=  {
        WITH SYNTAX                    DitBridgeKnowledge
        EQUALITY MATCHING RULE         directoryStringFirstComponentMatch
        NO USER MODIFICATION           TRUE
        USAGE                          dSAOperation
        ID                             id-doa-ditBridgeKnowledge }
--匹配规则--
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 }
--客体标识符分配--
--dsa 操作属性--
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}
id-doa-ditBridgeKnowledge   OBJECT IDENTIFIER  ::=  {id-doa 8}
--知识匹配规则--
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  --DSA 操作属性类型
```

附 录 G
（规范性附录）
用 ASN.1 描述的操作绑定管理

本附录包含了本目录规范中关于操作绑定相关的所有 ASN.1 类型、值和信息客体类的定义，其形式为 ASN.1 模块 OperationalBindingManagement 。

```
OperationalBindingManagement {joint-iso-itu-t ds(5) module(1) opBindingManagement(18) 5}
DEFINITIONS ::=
BEGIN

--EXPORTS All--
——本模块中定义的所有类型与值都可被输出到本系列目录规范所包含的其他的 ASN.1 模块中，供其使用。
——也可以被其他应用所使用，这些应用将使用本模块中的定义来访问目录服务。
——其他应用可能将本模块中的定义用于其自身目的，但是不会限制为了维护和改进目录服务而进行的扩展和修改。
IMPORTS
——来自 GB/T 16264.2—2008
        directoryAbstractService, directoryShadowAbstractService, distributedOperations,
        directoryOSIProtocols, enhancedSecurity, hierarchicalOperationalBindings,
        commonProtocolSpecification
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        OPTIONALLY-PROTECTED-SEQ
                FROM EnhancedSecurity enhancedSecurity

        hierarchicalOperationalBinding, nonSpecificHierarchicalOperationalBinding
                FROM HierarchicalOperationalBindings hierarchicalOperationalBindings
——来自 GB/T 16264.3—2008
        CommonResultsSeq, directoryBind, securityError, SecurityParameters
                FROM DirectoryAbstractService directoryAbstractService
——来自 GB/T 16264.4—2008
        AccessPoint
                FROM DistributedOperations distributedOperations
——来自 GB/T 16264.5—2008
        id-err-operationalBindingError, id-op-establishOperationalBinding,
        id-op-modifyOperationalBinding, id-op-terminateOperationalBinding,
        OPERATION, ERROR
                FROM CommonProtocolSpecification commonProtocolSpecification
        APPLICATION-CONTEXT
                FROM DirectoryOSIProtocols directoryOSIProtocols
```

```
——来自 ISO/IEC 9594-9
        shadowOperationalBinding
                FROM DirectoryShadowAbstractService directoryShadowAbstractService ;
--绑定和解绑定--
dSAOperationalBindingManagementBind OPERATION  ::=  directoryBind
--操作、变量和结果--
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,
                --对称的,Role A 发起,或 Role B 发起--
        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,
        --对称的,Role A 响应,或 Role B 响应--
        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,
        --对称的,Role A 发起,或 Role B 发起--
        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,
        --对称的,Role A 发起,或 Role B 发起--
        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 } } }
--错误和参数--
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 }
--信息客体类--
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   --操作绑定管理
```

附 录 H
（规范性附录）
增强的安全性

已知本模块中包含了无效的规范。因此本模块中的部分内容被反对。被反对的部分在 ASN.1 的注释项中标注了出来。将来的版本或者将此反对的部分规范删除，或者提供一个升级版本的规范。

```
EnhancedSecurity { joint-iso-itu-t ds(5) modules(1) enhancedSecurity(28) 5 }
DEFINITIONS IMPLICIT TAGS ::=
BEGIN

--EXPORTS All--

IMPORTS
——来自 GB/T 16264.2—2008
        authenticationFramework, basicAccessControl, certificateExtensions, id-at, id-avc, id-mr,
        informationFramework, upperBounds
                FROM UsefulDefinitions { joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5 }

        Attribute, ATTRIBUTE, AttributeType, Context, CONTEXT, MATCHING-RULE, Name,
        objectIdentifierMatch, SupportedAttributes
                FROM InformationFramework informationFramework

        AttributeTypeAndValue
                FROM BasicAccessControl basicAccessControl
——来自 ISO/IEC 9594-8
        AlgorithmIdentifier, CertificateSerialNumber, HASH{}, SIGNED{}
                FROM AuthenticationFramework authenticationFramework

        GeneralName, KeyIdentifier
                FROM CertificateExtensions certificateExtensions

        ub-privacy-mark-length
                FROM UpperBounds upperBounds ;

OPTIONALLY-PROTECTED { Type } ::= CHOICE {
        unsigned        Type,
        signed          SIGNED {Type} }

OPTIONALLY-PROTECTED-SEQ { Type } ::=CHOICE {
        unsigned        Type,
        signed  [0]     SIGNED { Type } }
```

```
attributeValueSecurityLabelContext CONTEXT ::={
        WITH SYNTAX     SignedSecurityLabel--最多有一个安全标签上下文分配给一个属性值
        ID          id-avc-attributeValueSecurityLabelContext }

SignedSecurityLabel   ::=   SIGNED {SEQUENCE   {
            attHash         HASH {AttributeTypeAndValue},
            issuer          Name                OPTIONAL,--加标签的管理机构的名(称)
            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 ( {--无,最少须有一个组件存在-- } ) )

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,                          --标识了被保护的属性
        signer         Signer   OPTIONAL,              --管理机构或数据起源方的名(称)
        attribsHash    AttribsHash } }                 --被保护属性的散列值

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,   --签名将保护本条目中的所有属性值
```

```
        selectedTypes           [1]     SelectedTypes
                                    --签名将保护所选属性类型的所有属性值
        }

SelectedTypes   ::=   SEQUENCE SIZE (1..MAX) OF AttributeType

AttribsHash   ::=   HASH { SEQUENCE SIZE (1..MAX) OF Attribute }
                    --所选择范围内的属性类型和值以及相关的上下文值

attributeValueIntegrityInfoContext  CONTEXT   ::=   {
        WITH SYNTAX     AttributeValueIntegrityInfo
        ID      id-avc-attributeValueIntegrityInfoContext }

AttributeValueIntegrityInfo   ::=   SIGNED { SEQUENCE {
        signer          Signer OPTIONAL,        --管理机构或数据起源方的名(称)
        aVIHash         AVIHash } }             --被保护属性的散列值

AVIHash   ::=   HASH { AttributeTypeValueContexts }
                            --属性类型和值以及相关的上下文值

AttributeTypeValueContexts ::=SEQUENCE {
        type        ATTRIBUTE.&id ({SupportedAttributes}),
        value       ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
        contextList     SET SIZE (1..MAX) OF Context OPTIONAL }
--客体标识符分配--
--属性--
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}
--匹配规则--
--  id-mr-readerAndKeyIDMatch       OBJECT IDENTIFIER   ::=   {id-mr 43}
--上下文--
id-avc-attributeValueSecurityLabelContext   OBJECT  IDENTIFIER   ::=   {id-avc   3}
id-avc-attributeValueIntegrityInfoContext   OBJECT  IDENTIFIER   ::=   {id-avc   4}
END   --增强的安全性
```

附 录 I
（资料性附录）
树 的 数 学

一棵树是一些点的集合（这些点被称为*顶点*（vertices））以及一些直线的集合（这些直线被称为*弧*（arc））；每条弧 a 都是从一个顶点 V 导向另一个顶点 V'。例如，在图 I.1 的一棵树中，有 7 个顶点（标识为 V^1 到 V^7）以及 6 条弧（标识为 a^1 到 a^6）。

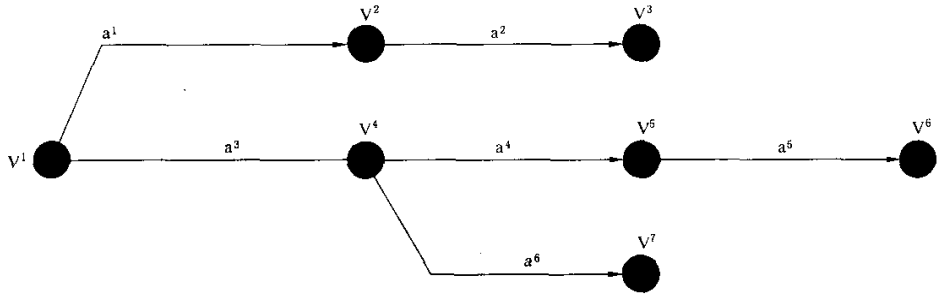


图 I.1 树的数学

两个顶点 V'和 V 分别被称为一条从 V 到 V'的弧 a 的*起始*（initial）顶点和*终结*（final）顶点。例如，图中的 V^2 和 V^3 分别是弧 a^2 的起始顶点和终结顶点。多个不同的弧可能会具有相同的起始顶点，但终结顶点不同。例如，弧 a^1 和 a^3 具有相同的起始顶点 V^1，但图中没有两条弧具有相同的终结顶点。

如果顶点不是任何一条弧的终结顶点，则该顶点常被称为*根*（root）顶点，或更非正式地被称为是树的“根”。例如，图 I.1 中，顶点 V^1 为根。

如果顶点不是任何一条弧的起始顶点，则该顶点常被称为*叶*（leaf）顶点，或更非正式地被称为是树图中的“叶”。例如，图中的顶点 V^3、V^6 和 V^7 是叶。

从一个顶点 V 到另一个顶点 V'的一条有向路径是弧（a^1，a^2，…，a^n）（$n \geqslant 1$）的集合，这样顶点 V 是弧 a^1 的起始顶点，V'是弧 a^n 的终结顶点，而对于 $1 \leqslant k < n$，则弧 a^k 的终结顶点也是弧 a^{k+1} 的起始顶点。例如，图中，从顶点 V^1 到顶点 V^6 的一条有向路径是弧（a^3，a^4，a^5）的一个集合。术语“路径”应当被理解为是标识了一条从根到某个顶点的有向路径。

附 录 J
（资料性附录）
名（称）设计准则

信息框架是非常通用的，并且考虑了 DIT 内的任意条目和属性种类。正如这里所定义的那样，由于名（称）与 DIT 内的路径密切相关，这就意味着名（称）可能也有任意种类。本附录提供了设计名（称）时应当考虑的准则。在设计 GB/T 16264.7—2008 中规定的名（称）格式时，已经使用了合适的准则。建议合适时在规定的名（称）格式不适用时，也建议使用该准则。

目前，仅提出了一个准则，即用户友好。

注：并不是所有的名（称）都需要用户友好。

本附录的剩余部分讨论了应用于名（称）的用户友好概念。

人类需要直接处理的名（称）应当是用户友好的。一个用户友好的名（称）是从人类用户的视角出发的，而不是从计算机视角出发的。它对用户来说，应当易于推导、记忆和理解，而不是易于被计算机解释。

用户友好的目标可以更精确地根据下述两个原则来规范：

——一般来说，人类应当能够根据客体所自然拥有的信息而正确地猜测出一个客体的用户友好名。例如，假设一个人能够通过正常的商务联络偶然得到某些信息，则他应当能够猜测出某位商务人士的名（称）；

——当一个客体的名（称）被指定时具有二义性，则目录应当能够意识到这个事实，而不是断定该名（称）标识了一个特定的客体。例如，如果两个人拥有同样的姓，这时候，单独的姓就被认为不足以用来标识任意一人。

从用户友好的目标可以推派生下述子目标：

a) 名（称）不得人为地删除某些自然的二义性。例如，如果有两个人拥有相同的姓“Jones”，则不能要求他们中的任何一个符合“Wjones”或“Jones2”。所替代的是，应当在命名惯例中提供用户友好的方式来区分不同的实体。例如，在命名惯例中除要求有姓外，还应当要求具有第一个名（称）和中间的大写首字母等。

b) 名（称）应当允许一般的缩写和一般的拼写变化。例如，如果某个人被 Conway 钢铁公司（Conway Steel Corporation）雇用，且他的名（称）中要包含其雇用者的名（称），则“Conway Steel Corporation”、“Conway Steel Corp.”、“Conway Steel”以及“CSC”中的任何一个都应当足以标识这个正在讨论的机构。

c) 在某些情况下，为了更加用户友好，或者为了缩减搜索的范围，可以使用别名来引导对某个特定条目的搜索。如下的示例举例说明了别名出于上述目的的一种用法。如图 J.1 所示，坐落于大阪（Osaka）的某分公司能够使用名（称）{ C = Japan, L = Osaka, O = ABC, OU = O-saka-branch }来标识。

d) 如果名（称）有多个部分，则必选部分的数量和可选部分的数量都应当相对少，以便易于记忆。

e) 如果名（称）有多个部分，则一般来说，这些部分出现在名（称）中的确切次序应当是不重要的。

f) 用户友好的名（称）不得包含计算机地址。

g) 在某些情况下，能够使用上下文来提供可替换的名（称）。例如，如图 J.2 所示，人员 Jones 能够使用{O = "XYZ", OU = "Research", CN = "Jones"}来标识，其上下文为 Language = English；另外还能够使用{O = "XYZ", OU = "Recherche", CN = "Jones"}来标识，而此时上下文为 Language = French。

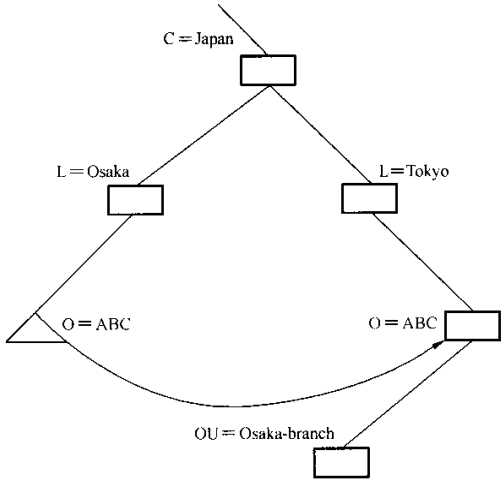


图 J.1 别名示例

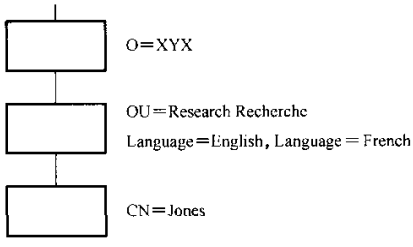


图 J.2 名(称)的上下文变化示例

附 录 K
（资料性附录）
模式的各方面的示例

K.1 属性层次结构的示例

图 K.1 显示了一个通用属性telephoneNumber 的值的简单层次结构，该属性的值被表示为包含在一个外部集合中。有两个特定的属性类型从该通用类型中派生出来，即workTelephoneNumber 和homeTelephoneNumber 。这些类型的值被表示为包含在内部集合中。

类型homeTelephoneNumber 的值既被包含在表示homeTelephoneNumber 的内部集合中，也被包含在表示telephoneNumber 的外部集合中，但不被包含在表示workTelephoneNumber 的内部集合中。

一个 DIT 结构规则能够被定义为允许条目可以包含图 K.1 中所显示的三种类型的值。而另外一个规则可能被定义为允许条目仅包含类型 telephoneNumber 的值。

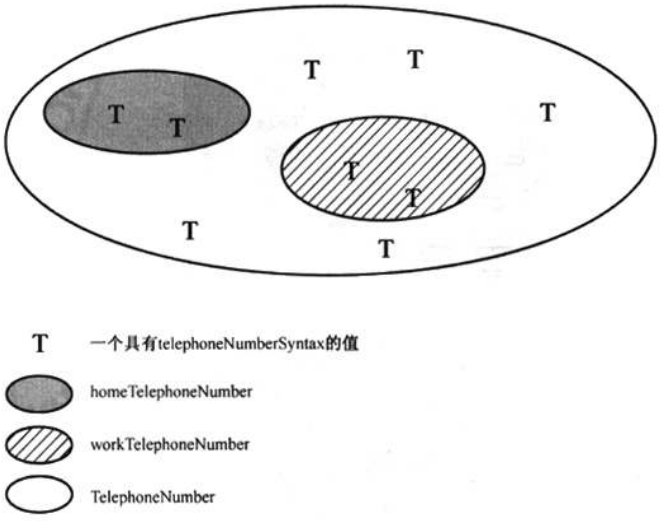


图 K.1 电话号码属性值的层次结构

K.2 子树规范的示例

下面是一个示例，举例说明了子树规范。考虑图 K.2 中表示的 DIT 的一部分。

子树 1 和子树 2 都与名(称)为 a 的管理点相关。而标识符 b1、c2、d3 等，则表示与管理点 a 相关的本地名(称)值。

子树 1 可被定义为：

```
subtree1 SubtreeSpecification ::={
        specificExclusions { chopBefore b1 } }
```

子树 2 可被定义为：

```
subtree2 SubtreeSpecification   ::={
         base b1 }
```

假设图 K.2 中，所有以本地名(称) e1、e2 等标识的条目，是表示有组织的人员条目。可指定一棵子树精选来包括管理区内的这些所有的条目，如下：

```
subtree-refinement1 SubtreeSpecification ::={
```

```
        specificationFilter
                item                        id-oc-organizationalPerson }
```

还可以将其细化到仅包含子树 2 中的所有这些有组织人员，如下所述：

```
subtree2-refinement SubtreeSpecification  ::=  {
        base            b1,
        specificationFilter
                item            id-oc-organizationalPerson }
```

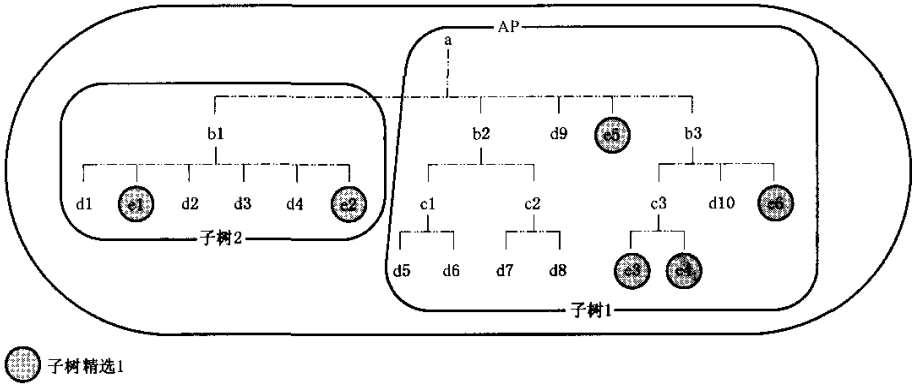


图 K.2　子树规范示例

K.3　模式规范

K.3.1　客体类和名(称)格式

下述客体类，在 GB/T 16264.7—2008 中定义，用于一个特定的子模式管理区：

——organization；

——organizationalUnit；

——organizationalPerson。

对于子模式内唯一的客体类为 organization 的管理条目而言，不要求名(称)格式。下述名(称)格式，在 GB/T 16264.7—2008 中定义，用于包含类 organizationalUnit 和 organizationalPerson 中的条目：

——orgNameForm；

——orgUnitNameForm；

——orgPersonNameForm。

K.3.2　DIT 结构规则

定义了如下结构规则，用以规范一棵如图 K.3 中的子树的结构。图 K.3 举例说明了在为 DIT 的不同点增加一个条目时可能会使用哪个规则。

```
    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 }
```

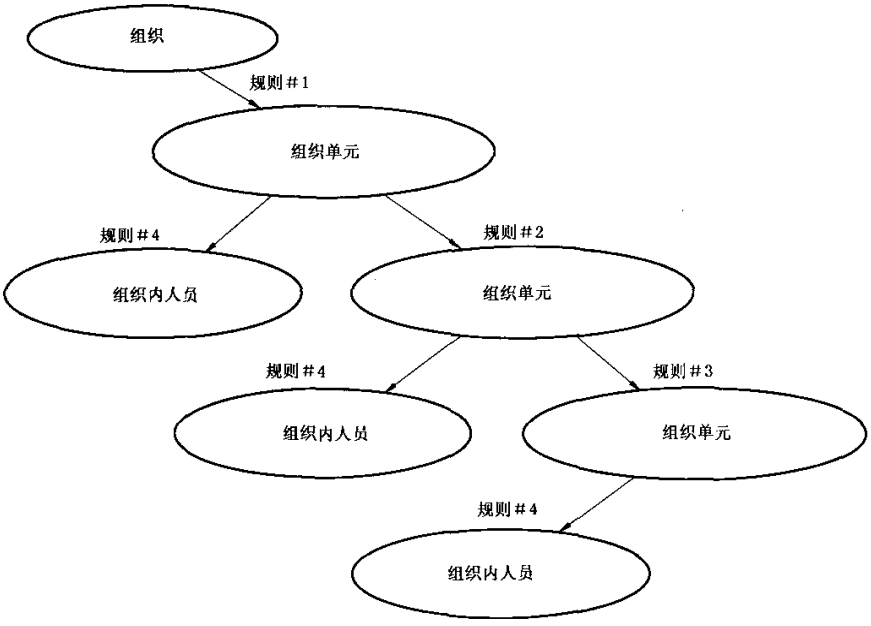


图 K.3　子模式示例

K.4　DIT 内容规则

假设子模式管理者有两种需求，需要为子模式管理区内的条目增加附加信息，需求如下：

——所有的 organizationalPerson 和 organizationalUnit 条目都应当具有属性 organizationalTelephoneNumber。当查询目录的电话号码时，该属性应当被返回；

——所有的 organizationalPerson 条目将具有新的属性管理者。

定义了如下属性类型来满足上述需求：

```
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 }
```

定义了如下 DIT 内容规则来满足上述需求：

```
organizationRule  CONTENT-RULE  ::=  {
        STRUCTURAL OBJECT CLASS    id-oc-organization }

organizationalUnitRule  CONTENT-RULE  ::=  {
        STRUCTURAL OBJECT CLASS    id-oc-organizationalUnit
        MAY CONTAIN                { organizationalTelephoneNumber } }

organizationalPersonRule  CONTENT-RULE  ::=  {
        STRUCTURAL OBJECT CLASS    id-oc-organizationalPerson
        MUST CONTAIN               { manager }
        MAY CONTAIN                { organizationalTelephoneNumber } }
```

K.5 DIT 上下文用法

子模式管理者需要实现某个国际化组织的策略，该策略要求必须使用 locale 上下文在该组织的管理区内来区分标题属性类型和描述属性类型的不同的值。此外，由于组织可能会以一种常规的方式来轮换职责，因此对某些人员来说，在条目中使用临时的标题上下文是需要的。

定义了如下 DIT 上下文规则来满足上述需求：

```
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 } }
```

附 录 L
（资料性附录）
基本访问控制许可概述

L.1 概述

本附录是资料性附录，为操作、被保护项及许可种类之间的各种不同组合的含义提供了一个概述。如果本概述和本目录规范的正文所提供的规范有差异，则正文提供的标准文本具有权威性，须以其为准。

表 L.1 将目录操作与条目和属性的访问控制关联起来，对许可种类进行了概述，这些许可种类是为了使操作成功而务必被准予的。

表 L.2 对许可种类 returnDN 和discloseOnError 提供了一个概述，并概述了如何将准予和拒绝与不同协议元素相关联起来。

表 L.3 对与条目访问控制的准予和拒绝相关的语义提供了一个概述。

表 L.4 对与属性访问控制的准予和拒绝相关的语义提供了一个概述。

L.2 操作所需的许可

表 L.1 根据目录操作所需的目录信息许可

目录操作	所需的条目被保护项的许可	所需的属性和属性值被保护项的许可
比较	*阅读(Read)*	*比较(Compare)*：用于正在比较的属性 *比较(Compare)*：用于正在比较的属性值
阅读	可辨别名的*阅读(Read)*和*返回 DN(ReturnDN)*：针对识别名	*阅读(Read)*：用于返回的任何属性类型信息 *阅读(Read)*：用于返回的任何属性值
列表	*浏览(Browse)*和*返回 DN(ReturnDN)*：用于所返回的 RDN 的所有下级条目	无
搜索	用于的搜索范围内候选条目*浏览(Browse)*； 返回 *DN (ReturnDN)* 用于每个被返回的识别名	如果有的话，*过滤器匹配(FilterMatch)*：用于评价一个过滤项是 TRUE 或 FALSE 的属性类型和属性值信息。 *阅读(Read)*：用于返回的任何属性类型信息。 *阅读(Read)*：用于返回的任何属性值
增加条目	*增加(Add)*	*增加(Add)*：用于指定的所有属性类型 *增加(Add)*：用于指定的所有属性值
删除条目	*删除(Remove)*	无
修改条目	*修改(Modify)*	*增加(Add)*：用于被增加的所有属性。 *增加(Add)*：用于被增加的所有属性值。 *删除(Remove)*：用于被删除的所有属性。 *删除(Remove)*：用于被删除的所有属性值
修改 DN	*初始位置的重命名(Rename)*：如果仅有最后一个 RDN 被修改 *输出(Export)*：从初始位置删除一棵子树 *输入(Import)*：在目标位置重新安置一棵子树	无

L.3 影响错误的许可

表 L.2 对错误和名(称)的返回产生影响的许可

	受影响的协议元素	含 义
返回 *DN* *(ReturnDN)*	EntryInformation CompareResult ListResult SearchResult NameError ContinuationReference	如果被准予,则可返回实际的识别名。 如果被拒绝,则禁止返回实际的识别名。 根据本地策略,可能会返回一个替代的合法别名
在错误中泄漏 *(DiscloseOnError)*	NameError UpdateError AttributeError SecurityError	如果被准予,则允许返回一个错误,该错误中显示了该被保护项的存在。 如果被拒绝,则要求目录隐藏该被保护项的存在

L.4 条目级许可

表 L.3 条目级许可及其含义

许 可	含 义
阅读(Read)	如果被准予,则允许对条目执行目录阅读或比较操作,但是它自己不能够批准可以返回该条目的任何属性信息。 如果被拒绝,则不允许对条目执行阅读或比较操作
浏览(Browse)	如果被准予,则在列表或搜索操作的范围内,允许条目作为候选条目。 如果被拒绝,则在列表或搜索操作的范围内,将排除该条目
增加(Add)	如果被准予,则允许条目本身被增加,但不包括它的属性。增加仅是在作为预定 ACI 时才有意义。 如果被拒绝,则不允许增加条目
修改(Modify)	如果被准予,则允许对条目执行修改操作。 如果被拒绝,则不允许对条目执行修改操作
删除(Remove)	如果被准予,则允许删除条目,而不必考虑任何属性。 如果被拒绝,则不允许删除条目
重命名(Rename)	如果被准予,则允许修改条目的 RDN,或者可选的,删除一个旧值并增加一个新值,而不必考虑可能应用于该条目的对属性或属性值的保护,修改是通过 ModifyDN 操作完成的,该操作必须符合 Import 和 Export 的适当的许可。 如果被拒绝,则不允许修改条目的 RDN
输入(Import)	如果被准予,在一个 ModifyDN 操作中,将条目及其所有的下级都重新安置到 DIT 内的一个指定位置上。输入(Import)操作仅在作为预定 ACI 时才有意义。 如果被拒绝,则不允许使用 ModifyDN 操作将条目及其所有的下级都重新安置到 DIT 内的一个指定点
输出(Export)	如果被准予,则允许使用 ModifyDN 操作将条目及其所有的下级都重新安置到 DIT 内的另外一个指定的位置。请求者必须在目标位置拥有输入许可。 如果被拒绝,则不允许使用 ModifyDN 操作重新安置条目及其所有的条目
返回 DN *(ReturnDN)*	如果被准予,则允许条目的识别名置于操作结果中返回。 如果被拒绝,则不允许返回识别名。根据本地策略,可能会返回一个替代的合法别名

表 L.3（续）

许　可	含　义
在错误中泄漏 *(DiscloseOnError)*	如果被准予，则允许在返回的错误中显示条目的存在。 如果被拒绝，则要求目录隐藏条目的存在。DiscloseOnError 本身不能够拒绝具有检测到条目的能力，这种检测是通过其他被准予的适当方式获得的

L.5　属性级许可

表 L.4　属性级许可及其含义

许　可	被保护项类别	含　义
阅读(Read)	属性类型	如果被准予，则允许关于该属性类型的信息在阅读或搜索操作的结果中返回。尽管它是作为阅读该属性值的先决条件，但它本身不能够对该属性类型的任何值的访问进行准予。 如果被拒绝，则不允许在阅读或搜索操作的结果中返回关于该属性类型的信息。结果是，它同样拒绝了所有的属性值
阅读(Read)	属性值	如果被准予，则允许一个属性类型的指定值可以在阅读或搜索操作的结果中返回。它不能对属性类型本身的访问进行准予。为了能够阅读一个属性值，还要求对相应的属性类型具有阅读许可。 如果被拒绝，则不允许在阅读或搜索操作的结果中返回该属性类型的指定属性值。它本身不能够拒绝对其他值的访问，或者拒绝对属性类型本身的访问
比较(Compare)	属性类型	如果被准予，则允许比较操作对属性类型进行检测。尽管它是作为该类型的属性值比较的先决条件，但它本身不能够对该属性类型的任何值的比较进行准予。 如果被拒绝，则不允许比较操作对该属性进行检测。结果是，它同样拒绝了对所有属性值的检测
比较(Compare)	属性值	如果被准予，则允许比较操作对指定属性类型的指定值进行检测。它不能对属性类型本身的比较进行准予。为了能够比较一个属性值，还要求对相应的属性类型具有比较许可。如果被拒绝，则不允许比较操作对该指定属性值进行检测
过滤器匹配 *(FilterMatch)*	属性类型	如果被准予，则允许该属性类型用于一个搜索过滤项的评价中。尽管它是作为该类型的属性值包含在过滤项评价中的先决条件，但它本身不能够对该属性类型的任何值进行准予。 如果被拒绝，则不允许在一个过滤项的评价中使用该属性类型及其属性值
过滤器匹配 *(FilterMatch)*	属性值	如果被准予，则允许该属性值用于一个搜索过滤项的评价中。为了能够成功评价，则还要求对相应的属性类型具有过滤器匹配许可。 如果被拒绝，则不允许在一个过滤项的评价中使用该属性值
增加(Add)	属性类型	如果被准予，则允许增加指定的属性类型。但其本身不能够对该属性类型的任何值的增加进行准予。 如果被拒绝，则不允许增加指定的属性类型，结果是也不允许增加其所有的属性值

表 L.4（续）

许　可	被保护项类别	含　义
增加(Add)	属性值	如果被准予，则允许增加指定的属性值。它不能对属性类型本身的增加进行准予。相反的，为了增加一个已经存在的属性的属性值，不要求对相应的属性类型具有增加许可。 如果被拒绝，则不允许增加指定的属性值
删除(Remove)	属性类型	如果被准予，则允许在一个修改操作中，将指定的属性类型及其所有的属性值都删除。但它本身不能够对该属性类型的任何单个属性值的删除进行准予。 如果被拒绝，则不允许在一个修改操作中删除指定的属性类型
删除(Remove)	属性值	如果被准予，则允许在一个修改操作中删除指定的属性值。为了能够删除最后一个属性值，还要求对相应的属性类型具有删除许可。 如果被拒绝，则不允许在一个修改操作中删除指定的属性值
在错误中显示(DiscloseOnError)	属性类型	如果被准予，则允许在返回的错误中可能显示属性的存在。 如果被拒绝，则要求目录隐藏属性的存在。DiscloseOnError 本身不能够拒绝具有检测到属性类型的能力，这种检测是通过其他被准予的适当方式获得的
在错误中显示(DiscloseOnError)	属性值	如果被准予，则允许在返回的错误中可能显示属性值的存在。 如果被拒绝，则要求目录隐藏该属性值的存在。DiscloseOnError 本身不能够拒绝具有检测到属性值的能力，这种检测是通过其他被准予的适当方式获得的

附 录 M
（资料性附录）
访问控制示例

M.1 引言

本附录仅作为资料性附录做指南之用。它提出了三个主要的主题：在基本访问控制机制的体系结构中非常重要的设计原则；基本访问控制的一个扩展示例；基于规则的访问控制的一个短示例。基本访问控制和基于规则的访问控制的详细信息在本目录规范的第 18 章和第 19 章以及 GB/T 16264.3—2008 中提供。

M.2 基本访问控制的设计原则

本章介绍了几个在基本访问控制的体系结构中使用的非常重要的设计原则。为了便于引用，每个设计原则都被赋予了标号，如 PR-1。

PR-1：一般来说，与具有更高特异性的UserClasses 相关的许可，其优先级要高于与具有较低特异性的UserClasses 相关的许可。当许可具有相同的优先级时，应用此原则。在本原则中，特异性测量了一个请求者的名(称)与一个特定的UserClasses 规范是如何明确相关的；其中，allUsers 具有最低的特异性，而name 是非常特定的。本原则体现在 18.8.4 的 2)中。当缺省许可的策略(表示为具有较低特异性的UserClasses)被与某个具有更高特异性的UserClasses 规范相关的许可所覆盖时，在这种情况下，本原则将简化这种情况。

PR-2：一般来说，与具有更高特异性的ProtectedItems 相关的许可，其优先级要高于与具有较低特异性的ProtectedItems 相关的许可。当许可具有相同的优先级以及相同的UserClasses 特异性时，应用此原则。在本原则中，特异性测量了ProtectedItems 规范与正在访问的某个确定的项是如何明确相关的。例如，当目标被保护项是一个特定的属性值，则allAttributeValues 和 allUserAttributeTypesAndValues 比attributeValue 的特异性要低。本原则体现在 18.8.4 的 3)中。当缺省许可的策略(表示为具有较低特异性的ProtectedItems)被与某个具有更高特异性的 ProtectedItems 规范相关的许可所覆盖时，在这种情况下，本原则将简化这种情况。

PR-3：基本访问控制被建模为完全独立于名(称)解析过程，别名解除引用的情况除外。除了别名解除引用外，访问控制决策仅出现在目录已经成功地查找到一个适当的包含目标被保护项的 DSA 之后。一个必然推派生的原则便是：基本访问控制不会对目录如何产生子请求产生影响，也不会对目录如何执行与子请求相关的名(称)解析产生影响(除了别名解除引用的情况之外)。

PR-4：Precedence 可以被用于加强上级管理机构与下级管理机构之间的关系，这样上级可以覆盖下级所设置的控制。例如：让 SE1 表示一个 ACSA(称为 ACSA-1)管理条目的子条目；类似的，让 SE2 表示 ACSA-1 内的一个 ACIA 管理条目的子条目。ACSA-1 管理机构可能会指定出现在 SE2 中的Precedence 的限制，这样 SE2 中的prescriptiveACI 就不能够取消 SE1 中设置的预定 ACI。同样，也能够为 entryACI (在 ACSA-1 中)指定Precedence 的限制，这样entryACI 就不能够取消设置在 SE1 中的预定控制。本原则简化了部分授权的管理机构的实现。

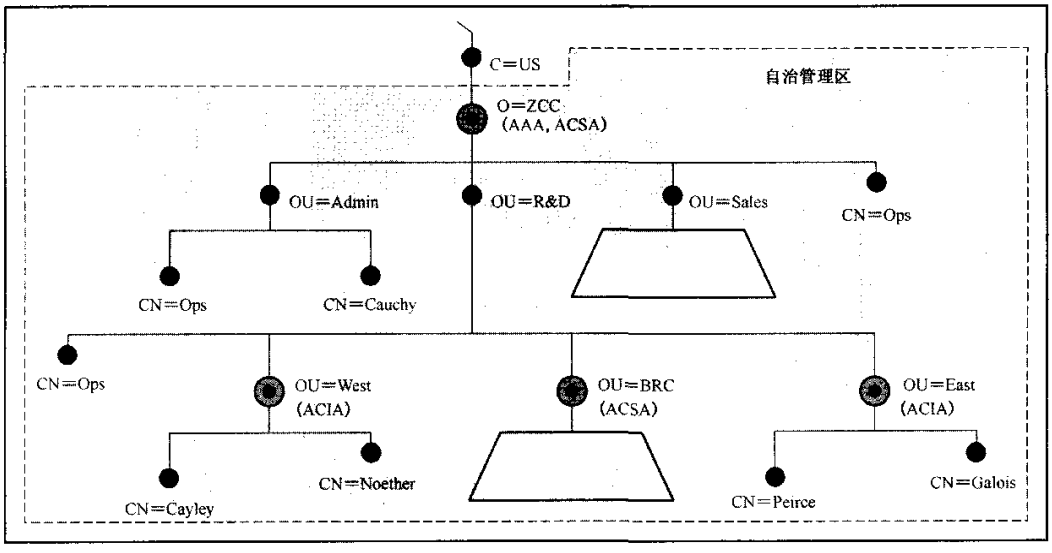
注：本目录规范假定与内部域相关的优先级限制的方法可以实现。然而，本目录规范不定义(或不描述)优先级是如何被限制的。

PR-5：基本访问控制从来不会被动地准予访问；每个准予访问的决定都是基于明确指定的访问控制信息。一个必然推派生的原则便是：准予一种形式的访问从来都不会隐含着准予执行另一种形式的访问。这些原则与一个更通用的被称为“最小特权”的安全设计原则是一致的。

PR-6：如果作为决策基础的prescriptiveACI、entryACI或subentryACI中的任意一个都不存在的话，则ACDF将拒绝访问。如果所有其他的决策参数都相等的话，则拒绝将覆盖准予（例如，如果有ACIItems准予，也有其他ACIItems拒绝，而Precedence和特异性都相同，则在这种情况下，拒绝将获胜。）

M.3 对示例的介绍

图M.1描述了一棵DIT子树，该子树表示一个假想的公司"Z计算机公司（Z Computer Corporation或ZCC）"，该子树用于整个示例。图M.1中的命名结构遵循GB/T 16264.7—2008中附录B的建议。可辨别名为{C=US, O=ZCC}的结点是一个管理条目，且是ZCC的自治管理点；因此它定义了一个自治管理区（AAA）的开始。自治管理区AAA的内容是一棵隐含定义的子树，该子树起始于自治管理点，终止于任一个叶结点或者终止于遇到的另一个自治管理点。在图M.1中，由于在{C=US, O=ZCC}结点之下，没有其他的自治管理点，因此该自治管理区AAA包含了结点{C=US}之下的所有结点。{C=US, O=ZCC}的结构客体类为organization；另外它还有一个辅助客体类certificationAuthority。辅助客体类用于在需要时帮助支持强鉴别。



图M.1 Z计算机公司（ZCC）的DIT分支

在自治管理点之下，有3棵子树：主管部门（Admin）、研发部门（R&D）和销售部门。每棵子树的根是一个结构客体类为organizationalUnit和辅助客体类为certificationAuthority的条目。R&D子树包含结构客体类organizationalUnit的条目，这些条目对应于远端地点，在其之下是结构客体类organizationalPerson的叶客体。在图中仅显示了类organizationalPerson的具有代表性的客体。结构客体类organizationalUnit的所有客体都有一个辅助客体类certificationAuthority。结构客体类organizationalPerson的所有客体都有一个辅助客体类strongAuthenticationUser。这些辅助客体类用于在需要时帮助支持强鉴别。

可辨别名为{C=US, O=ZCC, OU=Admin, CN=Ops}的客体属于结构客体类groupOfUniqueNames；它的属性uniqueMember的值包含了名（称）空间的管理者。它所包含的一个名（称）为{C=US, O=ZCC, OU=Admin, CN=Cauchy}。还有另外两个这样的客体：{C=US, O=ZCC, OU=R&D, CN=Ops}，其拥有的成员负责维护R&D子树内的条目；{C=US, O=ZCC, CN=Ops}，其拥有的成员负责{C=US, O=ZCC}的直接下级条目。可辨别名为{C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley}的用户是后两个组中的成员。

图 M.1 中的两个梯形表示局部子树，该局部子树的细节对本例不重要。

M.4 影响特定区和内部区定义的策略

为了支持基本访问控制，在 AAA 内可能建立两种类型的管理区：访问控制特定区(ACSA)和访问控制内部区(ACIA)。任何类型的一个管理区的建立都是通过为管理条目中的属性administrative-role赋予适当的值来实现的，此管理条目是该管理区的根顶点。一个 ACSA 的内容是一棵隐含定义的子树，该子树起始于根顶点，并向下扩展至叶客体或者遇到另一个 ACSA 的根为止。此外，一个 ACSA 的边界从来不会扩展至超出包含它的 AAA 的下游边界。在 ACIA 的情况下，其下游边界将出现在两种情况下，一种是遇到一个叶条目，另一种是遇到包含它的 ACSA 的边界。嵌套的 ACIA 具有相同的下游边界，且该边界与包含它们的 ACSA 的下游边界相同。

ZCC 已经建立了影响 AAA 内所需管理区的数量和类型的策略。第一个这样的策略是：名为基础研究协会(BRC)的组织单元被授予完全的权力，为了控制根顶点为{C=US, O=ZCC, OU=R&D, OU=BRC}的子树中的条目，它们可以建立所惯例的访问控制属性。为了简化该策略的实现，根{C=US, O=ZCC, OU=R&D, OU=BRC}被指定为一个管理条目，其管理角色为*id-ar-accessControlSpecificArea*。所形成的 ACSA 的下游边界由于叶条目的出现而被隐含地定义。

注：一个 ACSA 将权力的完全授权概念具体化，因为访问决策取决于出现在 ACSA 内部的 ACI，该 ACSA 包含了目标被保护项，并且不被出现在该 ACSA 之外的 ACI 所影响。

此外，上面所描述的 ACSA 是 ZCC 内访问控制权力被完全授权的唯一一个实例。然而，该目录管理模型的一个推论是，如果在 AAA 中至少有一个 ACSA 时，则 AAA 中的每一个条目都必须且仅必须包含在一个 ACSA 中。这个需求可通过集合理论来描述得更加清晰，即每个 ACSA 以及相关的 AAA 都被视为条目的一个集合：每对 ACSA 集合的交集都是空集，且所有 ACSA 的集合合集等于该 AAA。因此，在图中的示例中，至少需要一个另外的一个 ACSA 来包含那些处于 AAA 内，但不包含在 BRC 子树内的客体。因为示例中，在 AAA 内仅有一个完全授权的实例，则 AAA 的根也应当是另一个 ACSA 的起始点，这另外一个 ACSA 就包含了那些处于 AAA 内，但不包含在 BRC 子树内的所有条目。

这样形成的 ACSA 被描述为图 M.2 中的 ACSA-1 和 ACSA-2。在图 M.2 中，还应当注意到由于管理区被隐含地定义为子树，则每个区都包括其根顶点。ACSA-1 的内容从它的根开始一直向下扩展至叶客体，或者遇到另一个 ACSA 的根顶点为止(在本例中，为{C=US, O=ZCC, OU=R&D, OU=BRC})。在本例中，在{C=US, O=ZCC}之下没有自治管理点，因此该 AAA 的下游边界完全由叶客体所定义。本例的剩余部分将关注于 ACSA-1 内的访问控制 (ACSA-2 不再被讨论)。另外，为了简化，本示例不讨论关于对{C=US, O=ZCC, OU=Sales}的下级的控制。

另外一个影响管理区定义的 ZCC 策略是：西部 R&D 组织单元被授予了关于访问控制操作属性的部分权力，这些访问控制操作属性影响了根顶点为{C=US, O=ZCC, OU=R&D, OU=West}的子树中的条目。通过将该 R&D 西部子树的根作为一个管理点，且其管理角色为*id-ar-accessControlInnerArea*，使得该策略得到了最好的实现。这意味着，一般来说，为该子树规定的访问控制将是该子树根的子条目内定义的控制和包含它的 ACSA(即 ACSA-1)的根的子条目内所定义的控制的组合。所形成的 ACIA 的内容是一棵隐含定义的子树，其根为{C=US, O=ZCC, OU=R&D, OU=West}，并且一直向下扩展直至遇到叶客体。由于一个 ACIA 是一棵子树，它的内容包含该子树的根顶点。

对于 R&D 东部组织单元也有一个类似的策略。相应的 ACIA 的根顶点为{C=US, O=ZCC, OU=R&D, OU=East}。图 M.3 描述了 ACSA-1 内的两个 ACIA。为 R&D 西部定义的 ACIA 标注为 ACIA-1；为 R&D 东部定义的 ACIA 标注为 ACIA-2。

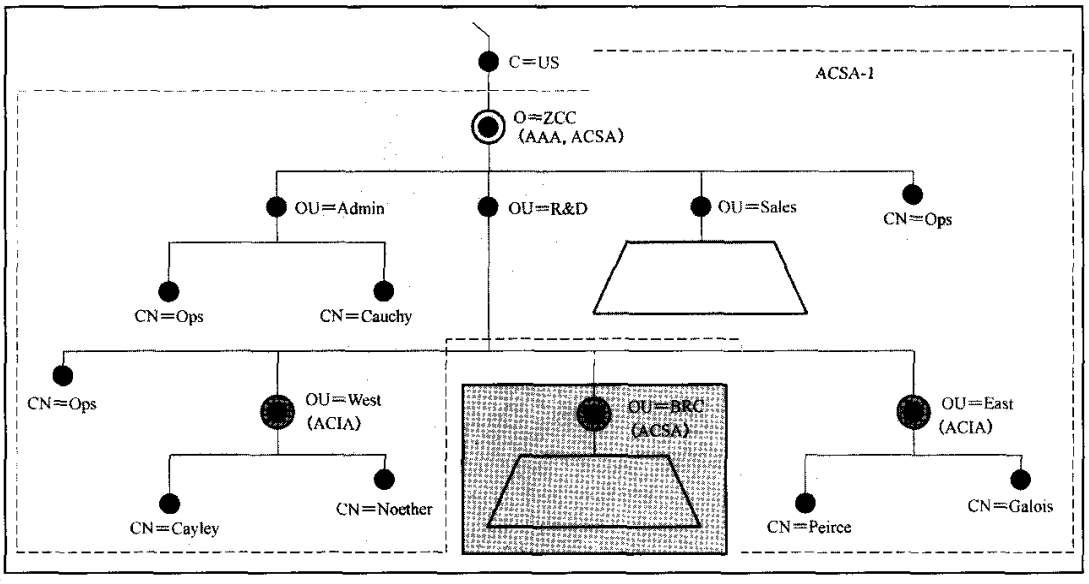


图 M.2 访问控制特定区

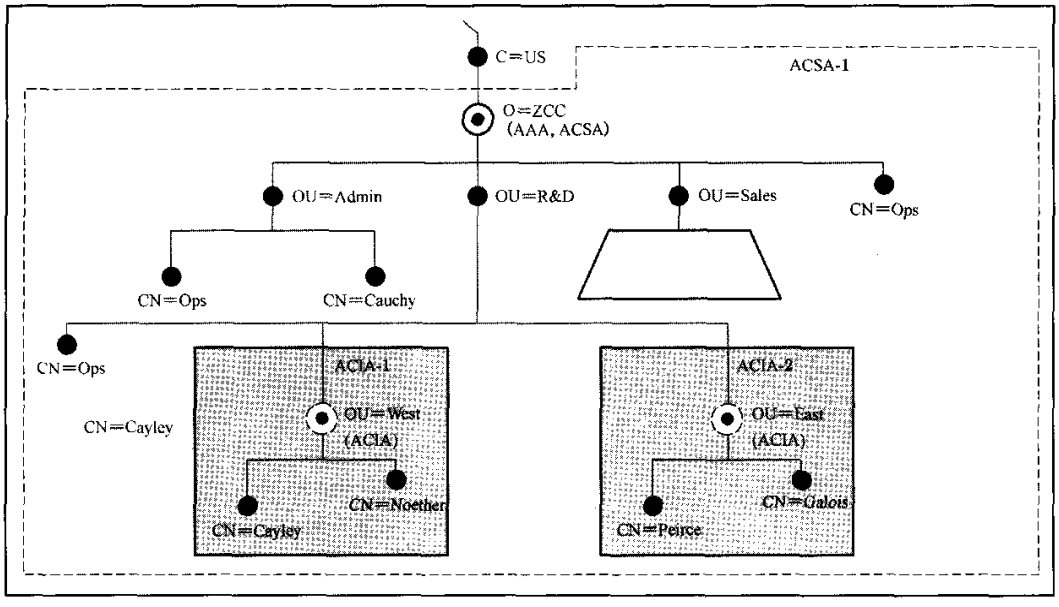


图 M.3 访问控制内部区

M.5 影响 DACD 定义的策略

惯例的访问控制在访问控制管理条目的子条目(客体类为accessControlSubentry)内定义。每个这样的子条目都有一个相关联的属性subtreeSpecification,该属性定义了子条目范围内的一个条目集合。

包含在此范围内的条目可能会组成一棵子树,或者组成一棵子树精选。在基本访问控制的上下文内,一个访问控制子条目的范围被称为一个目录访问控制域 DACD)。使用基本访问控制的安全机构应当注意不要混淆管理区的概念和 DACD 的概念。在本条的开始,将对管理区和 DACD 之间的区别及关系进行分析,然后继续讨论产生每个 DACD 的 ZCC 策略。

管理区和 DACD 之间的区别可以摘要描述如下:

——一个管理区是一棵隐含定义的子树，其根为一个管理条目，并一直向下扩展，如 M.4 中所描述的那样。这样的一个区被称为是隐含定义的，是因为在目录中没有标准化的属性来规定区的边界；DIT 在逻辑上被检查以确定一个管理区的边界。一个管理区一定不是一棵子树精选。

注 1：管理区的定义方法引出了一个推论，即对于每个被基本访问控制所影响的条目，都须有一个且仅有一个 ACSA 来包含该条目（即使该条目没有包含在 ACSA 内的任何 DACD 中）。

——一个 DACD 是一棵子树或子树精选，由客体类为accessControlSubentry 的子条目的属性subtreeSpecification 显式地定义。

——ACSA 和 ACIA 被 ACDF 使用，以确定哪一个惯例的访问控制（即哪个访问控制子条目）会潜在地影响一个给定访问控制决策的结果。ACSA 被用来实现访问控制权力的完全授权。而 ACIA 被用来实现访问控制权力的部分授权。

——一个 DACD 被用来规定哪一个条目（或潜在的条目）可能会被相关的访问控制子条目所影响。

——管理区和 DACD 的其他重要方面以及它们之间是如何相互关联的等，包括下述一些观察点。

每个 DACD 在一个特定管理条目的子条目内定义，该管理条目是某个管理区的根顶点。DACD、子条目、管理条目以及管理区之间的联系，对于一个给定的 DACD 来说，可以用来决定“相关联的管理区”（见 M.5.1）。包含在 DACD 中的条目集合可能是包含在相关联的管理区内的条目集合的一个完全子集或不完全子集。

注 2：术语“完全子集”和“不完全子集”是从数学的集合理论中借用的。说集合 A 是集合 B 的完全子集，当且仅当 A 中的每个元素都是 B 中的元素，且 B 中至少有一个元素不是 A 中的元素。说集合 A 是集合 B 的不完全子集，当且仅当两个集合中都包含了完全相同的元素。

——当包含在 DACD 中的条目集合是包含在相关联的管理区内的条目集合的一个不完全子集时，在这种情况下，DACD 和管理区被称为是“重叠的（congruent）”。然而，即使是这种重叠存在时，DACD 和管理区也是服务于根本不同的目的（管理区决定哪一个子条目被允许潜在地影响一个特定访问控制决策的结果，而 DACD 明确规定了哪个条目被一个给定子条目内的惯例的访问控制所影响。）

——DACD 永远不能够包含处于相关联的管理区之外的条目。

——ACDF 被设计为稳健的，即使在一个定义了 DACD 的subtreeSpecification 的范围内，还拥有处于相关联的管理区之外的条目，与这些条目相关的访问控制决策也不会受到影响。这种稳健性的方面体现在 ACDF 过程中，用以决定哪个子条目潜在地影响一个给定的决定（见 18.3.2 和 18.8.1 中的 d））。

——在同一个管理区的子条目内定义的不同 DACD 可以在共同相关联的管理区内自由地重叠。

——ACSA 之间永远都不能重叠；每个 ACIA 都被完全地嵌套在一个 ACSA 中。完全嵌套的含义是被包含的域内的条目构成了用来包含的域内条目的完全子集。另外，一个 ACIA 可能会包含一个或多个完全嵌套的 ACIA。

——若管理区是嵌套的，则与包含的区相关联的 DACD 可以同与被包含的区相关联的 DACD 之间自由重叠。用来包含的区可能是一个 ACSA 或一个 ACIA，而被包含的区总是一个 ACIA。

每个 DACD 都与策略的某一方面相关，这些方面影响了一个或多个条目或潜在条目。被策略的某个特定方面所影响的条目构成了一个 DACD。与策略的某个特定方面相关的 DACD 应当与负责实施该方面策略的管理机构所控制的管理区相关联。

在本例中，有多个策略方面被控制 ACSA-1 的管理机构所实施。例如，有应用于 ACSA-1 内所有客体的“缺省”控制。这些控制被赋予了一个优先级和一个特异性级别，允许它们容易地被其他预定的控制或 entryACI 属性所覆盖。还有一个策略是仅应用于{C=US，O=ZCC}的直接下级（在 ZCC 内，这些条目被称为是管理级条目）。还有一个策略是仅应用于具有结构客体类organizationalPerson 的条目。

ACSA-1 内的所有条目都被包含在与缺省控制相关的一个 DACD 内。因此，该 DACD 被定义为一棵子树，(以{C=US, O=ZCC}为其 base 顶点)以及一个 chop 规范(该规范将根为{C=US, O=ZCC, OU=R&D, OU=BRC}的子树排除在外)。所形成的 DACD 与 ACSA-1 是重叠的，在图 M.4 中被描述为 DACD-1。

注 3：本上下文中的"重叠"的含义见 18.3.2 中的 g)。

还是在 ACSA-1 内，控制organizationalPerson 条目的 DACD 是一棵子树精选(以{C=US, O=ZCC}为其 base 顶点)以及一个仅包含具有organizationalPerson 的客体类条目的specificationFilter(见 K.2 中的subtree-refinement1)。该 DACD 在图 M.4 中被描述为 DACD-2。

在 ACSA-1 内的第三个 DACD 是与用来控制的管理级条目相关的(即此组织根条目的除子条目外的其他直接下级)。该 DACD 是一个(被切除(集)的)子树(以{C=US, O=ZCC}为其 base 顶点)以及一个 chop 规范，该 chop 规范仅包含了顶点{C=US, O=ZCC}的除子条目外的其他直接下级。该 DACD 在图 M.4 中被描述为 DACD-5。

对于 ACIA-1，需要一个 DACD 来处理策略的某个方面，该方面的策略已经授权给控制该内部区的管理机构。被授权的管理机构仅影响{C=US, O=ZCC, OU=R&D, OU=West}的下级，因此该 DACD 与 ACIA-1 不重叠。该 DACD 在图 M.4 中被描述为 DACD-3。

对于 ACIA-2，仅需要一个 DACD；然而，被授权的管理机构影响 ACIA-2 内的所有条目，因此，该 DACD 与 ACIA-2 是重叠的。该 DACD 在图 M.4 中被描述为 DACD-4。

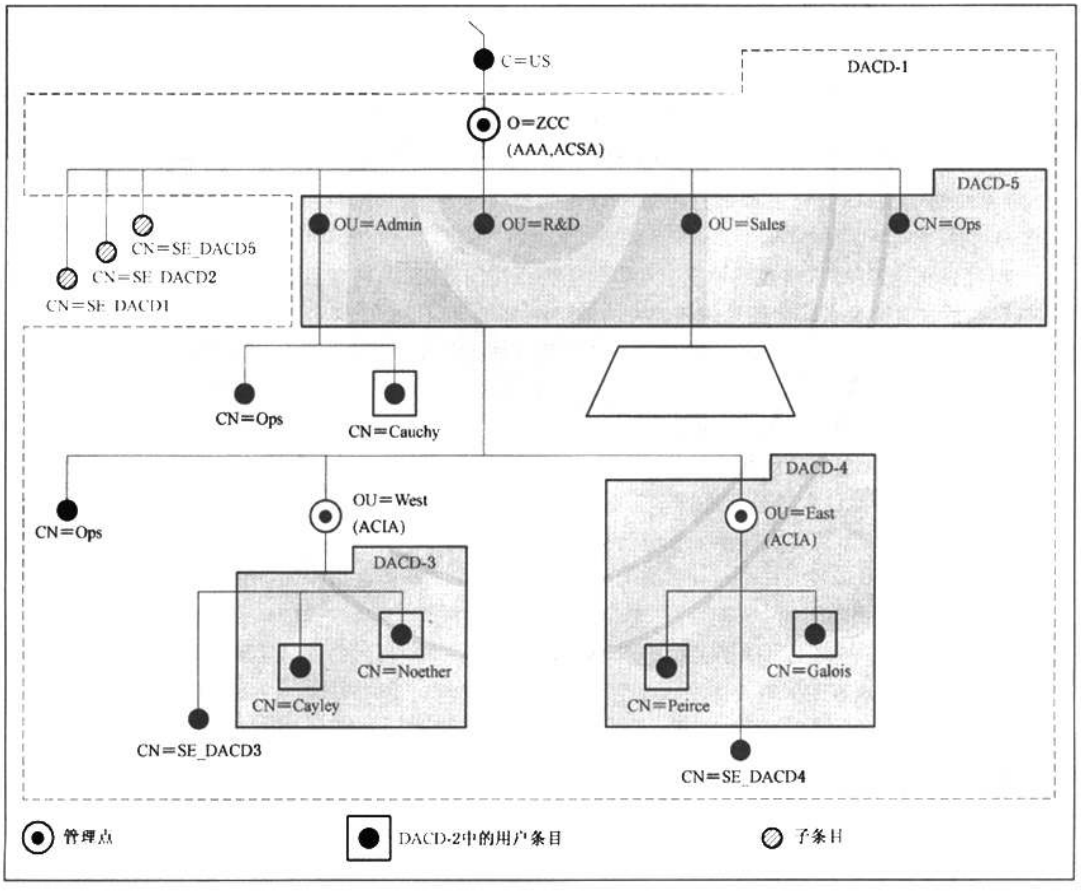


图 M.4 目录访问控制域

M.5.1 与每个 DACD 相关联的管理区

本例中使用的每个子条目都显示在图 M.4 中。本条总结了每个子条目的位置，并且指出了与每个 DACD 相关联的管理区。

DACD-1、DACD-2 和 DACD-5 定义在顶点{C=US，O=ZCC}的子条目中，该顶点是 ACSA-1 的根顶点。因此，这三个 DACD 被称为与 ACSA-1 相关联。定义 DACD-1 的子条目的名(称)为{C=US，C=ZCC，CN=SE_DACD1}。其他子条目具有类似的名(称)，都指示了它们所定义的 DACD。

DACD-3 定义在顶点{C=US，O=ZCC，OU=R&D，OU=West}的子条目中，该顶点是管理条目，是 ACIA-1 的根顶点。因此 DACD-3 与 ACIA-1 相关联。

DACD-4 定义在顶点{C=US，O=ZCC，OU=R&D，OU=East}的子条目中，该顶点是管理条目，是 ACIA-2 的根顶点。因此 DACD-4 与 ACIA-2 相关联。

M.6 属性 prescriptiveACI 所表示的策略

本章详细描述了可应用于 ACSA-1 内每个 DACD 的访问控制策略。本例中讨论的策略应当被认为是一个被简化的部分策略，以便易于表示。特别的，没有讨论关于口令是如何被控制的，因为一般来说，口令表示的是访问控制的一种特殊情况；另外，也没有讨论关于DiscloseOnError 或ReturnDN 的许可。

本章中讨论的策略用术语"策略片段"来表示，以便于理解属性prescriptiveACI 是如何被集合起来共同用于实施整个策略。每个片段被赋予一个引用标签，后续部分中可以使用这些标签；标签的格式为 PF-n，其中 n 是一个连续的整数。对于每个 DACD，还指示了可应用的策略片段是如何使用一个或多个子条目(包含属性prescriptiveACI)来表示的。

M.6.1 DACD-1 的 prescriptiveACI

DACD-1 的一个主要目的是施加与"缺省"访问控制相关的策略片段。当没有其他的具有更高优先级和特异性的控制时，这样的策略片段就可以提供替代控制了。特异性在 M.2 的设计原则 PR-1 和 PR-2 中讨论。

ZCC 已经根据缺省策略规则规定了它们关于公众访问的策略，这些策略可能会由于某些条目而被覆盖，这些条目需要更多限制的控制。缺省的策略在 PF-1 和 PF-2 中规定。注意：根据 ZCC 策略，那些实现了策略的部门都有责任确保任何背离了缺省规则的规则都应当比缺省规则具有更严格的限制。

PF-1：员工应当从普通的公众中区分出来。一般来说，公众的访问权限应当根据下面的 a)和 b)来限制；然而，对于某些特定的条目来说，公众访问可能会具有更严格的限制(永远不会具有更少的限制)。

a) 条目可能会根据通用名(称)来定位。对通用名(称)的搜索允许容纳近似匹配和替代名(称)。特别的，不允许普通公众进行基于电话号码的搜索，但允许那些处于组织中的人员进行该类搜索。搜索结果可能会显示 commonName 的所有值；

b) 仅有的公众属性是commonName 、telephoneNumber 和postalAttributeSet 中的组件以及 facsimileTelephoneNumber 。

PF-2：普通公众的访问可能不会被鉴别，但是应当提供一个身份。

ZCC 还使用了缺省的策略规则来表示它们关于雇员访问的通用策略。背离了缺省规则的规则比缺省规则可能具有更严格的限制，也可能具有更少的限制。缺省的策略在 PF-3 和 PF-4 中定义。

PF-3：一般来说，雇员对大部分条目的大部分属性都有兴趣进行阅读和搜索访问。

PF-4：对于雇员不修改 ACSA-1 内容的访问(在任何方式下)，需要简单鉴别。

还有一些应用于 DACD-1 的策略片段不能被作为缺省值。这样的两个示例在 PF-5 和 PF-6 中给出；它们与条目的管理相关。

PF-5：{C=US，O=ZCC，CN=Cauchy}是"超级用户"，被授权可以访问所有的数据，并可以执行任何必需的操作。

PF-6：对 ACSA-1 的内容进行任何修改时，都要求强鉴别。

可以使用{C=US，O=ZCC}的一个或多个子条目来实现 DACD-1 的策略片段。每个这样的子条目都应该具有相同的subtreeSpecification（该subtreeSpecification 以顶点{C=US，O=ZCC}为其base）以及一个chop 规范（该规范将 OU=BRC 的子树排除在外）。每个这样的子条目还应当包含一个属性prescriptiveACI 来实现该 DACD-1 策略片段的某个子集。为了本例的目的，假设使用一个单独的子条目（没有强制性的技术原因来使用更多的子条目）来获取所有的与 DACD-1 相关的预定控制。为了简化引用，该子条目在引用时被称为 SE_DACD1。在 SE_DACD1 中的属性prescriptiveACI 具有多个值；本条的后续部分将讨论每个值是如何设计的。

出现在prescriptiveACI 属性中的值的数量，部分地取决于策略片段是如何为了简便的目的而被分组为itemFirst 和userFirst 值的（这两个类型中的任一个都可能用于任何一个给定的情况中）；另外它还依赖于预定的访问控制本身是如何被处理的。

例如，PF-1 实现中的一部分要求公众用户（即allUsers）被准予如下所有的许可：

a) 浏览（Browse）：对于被保护项entry；

b) 过滤器匹配（FilterMatch）和阅读（Read）：对于被保护项attributeType {commonName}；

c) 过滤器匹配（FilterMatch）和阅读（Read）：对于被保护项allAttributeValues {commonName}。

这些许可对于实现 PF-1 是必要的（但不是充分的见注 1）。由于有 3 种被保护项（entry，attributeType 和allAttributeValues）且仅有一个用户类（allUsers），因此，看起来使用userFirst 类型的一个单独的ACIItem 是最自然的，而不是使用itemFirst 类型。

注 1：如果下述两个条件都同时满足的话，则上述所讨论的许可对于允许针对commonName 的搜索来说也是足够的：

a) 没有其他的具有更高优先级或特异性的相关 ACIItems 来拒绝上述所列的浏览（Browse）或过滤器匹配（FilterMatch）许可；以及

b) 在 SE_DACD1 内没有其他的prescriptiveACI 属性值，来拒绝上述所列的浏览（Browse）或过滤器匹配（FilterMatch）许可。

替代地，能够使用三个不同的ACIItem：分别针对每个被保护项。这种替代允许每个ACIItem 都拥有不同的访问控制；每个都有一个唯一的identificationTag（不同于同一个属性 prescriptiveACI 的其他值的identificationTag），且能够在另一个 ACIItem 中被引用，在该 ACIItem 中，被保护项是 attributeValue，且相关的属性值断言中规定了被保护值的identificationTag。注意：在本方法中，使用attributeValue 利用了为属性prescriptiveACI 而定义的特殊的相等匹配规则（见 18.5.1）。保护 ACI 的示例将在本例的随后部分给出详细讨论。

为了本例的目的，使用了 SE_DACD1 中的属性prescriptiveACI 的 6 个值来实现策略片段 PF-1 到 PF-4。每 3 个值的设计摘要描述如下。

注 2：在下述的设计摘要中，每个被保护项都被分配了一个标签以简化后续的引用。标签格式为圆括号内的斜体字（如 A1、A2、B1）。

注 3：本例中使用了 4 个优先级，分别是 10、20、30 和 40。

identificationTag： “公众访问 — 可以对通用名（称）的列表和搜索操作进行条目访问”

Precedence： 10

UserClasses： { allUsers }

authenticationLevel: none

ProtectedItems： { (A1) entry }

grantsAndDenials： { grantBrowse }

---

identificationTag: "公众访问 — 可以对搜索操作进行过滤器访问"
Precedence: 10
UserClasses: { allUsers }
authenticationLevel: none
ProtectedItems: { (B1) attributeType { commonName },
(B2) allAttributeValues { commonName },
(B3) attributeType { objectClass },
(B4) allAttributeValues { objectClass } }
grantsAndDenials: { grantFilterMatch }

---

identificationTag: "公众访问 — 可以对阅读和比较操作进行条目访问"
Precedence: 10
UserClasses: { allUsers }
authenticationLevel: none
ProtectedItems: { (C1) entry }
grantsAndDenials: { grantRead }

---

identificationTag: "公众访问 — 可以对查询操作进行属性访问"
Precedence: 10
UserClasses: { allUsers }
authenticationLevel: none
ProtectedItems: { (D1) attributeType {commonName,
postalAttributeSet,
telephoneNumber,
facsimileTelephoneNumber },
(D2) allAttributeValues { commonName,
postalAttributeSet,
telephoneNumber,
facsimileTelephoneNumber } }
grantsAndDenials: { grantRead, grantCompare }

---

identificationTag: "雇员访问 — 可以对程序操作进行属性访问"
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: "雇员访问 — 可以对搜索操作进行过滤器访问"
Precedence: 10
UserClasses: subtree with base { C=US, O=ZCC } and chop to exclude O=BRC subtree
authenticationLevel: simple
ProtectedItems: { (F1) allUserAttributeTypesAndValues }
grantsAndDenials: { grantFilterMatch }

注 4：对雇员的许可是对公众的许可和对雇员特定的许可的合集。上述的为雇员访问的 ACIItem 值与为公众访问的值具有密切的联系(强耦合)。这种强耦合在必要时能够被避免，方法是通过重复为公众访问的每个值(每个重复的值都应当具有一个新的 UserClasses，每个用户类仅规定一个雇员)。

该属性还有另外两个值与实现某个策略相关，该策略是关于条目是如何被管理的(见 PF-5 和 PF-6)。为了简化，本例假设访问控制属性是出现在 AAA 中的唯一的操作属性。这两个值的设计摘要描述如下：

identificationTag: "Cauchy 是超级用户(第 1 部分)"
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 是超级用户(第 2 部分)"
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 }

注意：要让 Cauchy 作为超级用户，上述的两个值是必要的，但不是充分的。它们不是充分的是因为它们不能够让 Cauchy 控制 ACSA-1 的管理点子条目；有两个原因来解释为什么会这样。首先，预定的 ACI 不会应用于它所在的子条目。第二，不能使用置于某个子条目(被称为子条目-1)内的预定 ACI 来控制作为子条目-1 兄弟的子条目。因此，有必要在与 ACSA-1 的管理点相应的条目内放置 subentryACI，因此 Cauchy 被允许管理该管理点下的所有子条目。必要的 subentryACI 在 M.7 中讨论。

还应当注意在上述预定 ACI 的两个值中准予的权力允许 Cauchy 能够对与作为 ACSA-1 管理点下级的管理点的相关子条目进行全权管理。

M.6.2 DACD-2 的 prescriptiveACI

DACD-2 在 ACSA-1 的管理条目的子条目内定义。DACD-2 与管理客体类为organizationalPerson 的控制条目相关。下述策略片段是与之相关的。

PF-7:仅有命名空间管理组{C=US, O=ZCC, OU=Admin, CN=Ops}内的成员才能够增加、删除或重命名用户条目。然而,它们仅被允许向一个新的条目内增加必选属性(仅包含必选属性的一个条目被称为一个*最小条目 minimal entry*)。

SE_DACD2 的属性prescriptiveACI 的下述两个值可以实现 PF-7。

注:在 PF-7 的上下文中,重命名一个条目被理解为重命名时不修改其直接上级。为了简化,该示例不对更复杂的情况进行讨论,即在重命名时修改被重命名条目的直接上级的情况;在这种情况下,必须考虑 Import 和 Export 许可。

```
identificationTag:    "最小的叶条目管理(第 1 部分)"
    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:    "最小的叶条目管理(第 2 部分)"
    Precedence:    20
    UserClasses:    userGroup { C=US, O=ZCC, OU=Admin, CN=Ops }
    authenticationLevel:  strong
    ProtectedItems:    { (K1 ) entry }
    grantsAndDenials:  { grantRename }
                ----------------------------------------------
```

M.6.3 DACD-3 的 prescriptiveACI

DACD-3 在 ACIA-1 的管理条目的子条目内定义。它实现了关于某些策略的策略片段,这些策略已经被部分授权给 ACIA-1。一个例子是 ACIA-1 的关于telephoneNumber 的策略与 DACD-1 内提供的缺省策略不同。在 DACD-3 内,telephoneNumber 不再被认为是一个公众可访问的项。这个可由下面的策略片段所反映。

PF-8: ACIA-1 内仅有的公众属性是commonName、postalAttributeSet 中的组件以及facsimileTelephoneNumber。

子条目{C=US, O=ZCC, OU=R&D, OU=West, CN=SE_DACD3}的属性 prescriptiveACI 的下述取值可以实现 PF-8。

```
identificationTag:  "公众访问的已授权控制"
Precedence:    10
UserClasses:    { allUsers }
authenticationLevel:  none
ProtectedItems:    { (L1 )attributeType { telephoneNumber } }
```

grantsAndDenials: { denyRead, denyCompare, denyFilterMatch }

R&D 西部组织也被授权实现对客体类为 organizationalPerson 的条目进行自管理。该策略可由下面的策略片段所反映。

PF-9: R&D 西部的雇员可能会管理其自身目录条目内的属性值,相应的属性类型包括:telephoneNumber、commonName 和facsimileNumber;然而,他们可能不能修改或删除由主管部门提供的电话号码值。

PF-9 的第 1 部分由下面的两个ACIItem 所反映。对属性telephoneNumber 的某个特定值进行删除的限制通过使用在 M.8 中描述的entryACI 来实现。

```
identificationTag:    "对 R&D 西部雇员条目的自管理(第 1 部分)"
Precedence:    20
UserClasses:    thisEntry
authenticationLevel:    strong
ProtectedItems:    { (M1 ) entry }
grantsAndDenials:    { grantModify }
```

```
identificationTag:    "对 R&D 西部雇员条目的自管理(第 2 部分)"
Precedence:    20
UserClasses:    thisEntry
authenticationLevel:    strong
ProtectedItems:    { (N1 ) attributeType {    commonName,
                                              postalAttributeSet,
                                              telephoneNumber,
                                              facsimileTelephoneNumber },
                     (N2)allAttributeValues {    commonName,
                                                 postalAttributeSet,
                                                 telephoneNumber,
                                                 facsimileTelephoneNumber } }
grantsAndDenials:    { grantAdd, grantRemove }
```

PF-10:若有一个分组,其成员在{C=US, O=ZCC, OU=R&D, CN=Ops}内标识,则该分组有责任为 ACIA-1 内条目的用户属性进行常规维护;然而,他们可能不能修改位于 ACIA-1 内的子条目。

该策略的第 1 部分在下述 ACIItem 中反映:

```
identificationTag:    "R&D 常规管理(第 1 部分)"
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 常规管理(第 2 部分)"
Precedence: 20
UserClasses: userGroup { C=US, O=ZCC, OU=R&D, CN=Ops }
authenticationLevel: strong
ProtectedItems: { (Q1)allUserAttributeTypesAndValues }
grantsAndDenials: { grantAdd, grantRemove, grantRead, grantFilterMatch, grantCompare}

关于子条目的限制是这样处理的,即在允许访问的 ACIA-1 的管理条目内不包含任何 subentryACI 值。

M.6.4 DACD-4 的 prescriptiveACI

DACD-4 在 ACIA-2 的管理条目的子条目内定义。正因为如此,它实现了某些策略的策略片段,这些策略已经被部分授权给 ACIA-2。

为了简化,DACD-4 不再详细讨论。

M.6.5 DACD-5 的 prescriptiveACI

DACD-5 在 ACSA-1 的管理条目的子条目内定义。使用该 DACD 来控制对本组织中根的除子条目以外的所有直接下级的访问。特别地,将应用下述策略。

PF-11:执行分组{C=US, O=ZCC, CN=Ops}负责管理所有的作为{C=US, O=ZCC}的直接下级的条目。

PF-11 由下述ACIItem 值来表示。

identificationTag: "管理级条目的控制(第 1 部分)"
Precedence: 40
UserClasses: userGroup { C=US, O=ZCC, CN=Ops }
authenticationLevel: strong
ProtectedItems: { (R1) entry }
grantsAndDenials: { grantRead, grantBrowse, grantRemove, grantAdd, grantRename, grantModify }

---

identificationTag: "管理级条目的控制(第 2 部分)"
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 属性 subentryACI 所表示的策略

M.7.1 ACSA-1 管理条目的 subentryACI

PF-5 体现在 prescriptiveACI 和subentryACI 的组合中,相关的prescriptiveACI 已经在 M.6.1 中

描述。为了让 Cauchy 能够管理 ACSA-1 管理点的子条目(以及任何一个作为 ACSA-1 管理点下级的子条目),有必要将下述subentryACI 值置于与 ACSA-1 管理点相应的条目内。

identificationTag: "Cauchy 是超级用户(第 3 部分)"
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 是超级用户(第 4 部分)"
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 ACIA-1 管理条目的 subentryACI

将属性 subentryACI 置于 ACIA-1 的根顶点内,可以实现下述策略片段。

PF-12: 拥有通用名(称)的用户 Cayley 有责任管理 ACIA-1 内定义的所有 prescriptiveACI。

属性 subentryACI 的下面两个值可以实现 PF-12。

identificationTag: "Cayley 管理 ACIA-1 内的子条目(第 1 部分)"
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 管理 ACIA-1 内的子条目(第 2 部分)"
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 属性 entryACI 所表示的策略

PF-9 要求每个 R&D 西部雇员都被允许管理在其目录条目内的 telephoneNumber 的所有值，但有一个限制是他们不能修改或删除由主管部门所提供的一个特定值。为了实施该限制，在受限制的电话号码被加入到条目中的时候，主管部门将为每个条目增加一个 entryACI 值。entryACI 值摘要描述如下：

identificationTag： “电话号码的受限的自管理”
Precedence： 30
UserClasses： thisEntry
authenticationLevel： none
ProtectedItems： { (V1) attributeValue { telephoneNumber = 主管部门提供的值 } }
grantsAndDenials： { denyRemove }

注意：由于用户不能修改 entryACI 属性(它不是 PF-9 中所定义的自管理的一部分)，上述的控制不能被用户覆盖。

下述策略片段是使用 entryACI 来实现一个分组条目自管理的例子。

PF-13： 条目{C=US, O=ZCC, OU=Admin, CN=Ops}是一个“自管理的”分组条目；这意味着该分组中的每个成员可能会从分组中删除它们的名(称)，或者修改它们在分组中的名(称)。但它们可能不会删除或重命名分组本身。

PF-13 通过条目{C=US, O=ZCC, OU=Admin, CN=Ops}中的属性 entryACI 实现，该属性有两个值简要描述如下。

identificationTag： “管理执行分组的自管理(第 1 部分)”
Precedence： 30
UserClasses： userGroup { C=US, O=ZCC, OU=Admin, CN=Ops }
authenticationLevel： strong
ProtectedItems： { (W1) entry }
grantsAndDenials： { grantModify }

---

identificationTag： “管理执行分组的自管理(第 2 部分)”
Precedence： medium
UserClasses： userGroup { C=US, O=ZCC, OU=Admin, CN=Ops }
authenticationLevel： strong
ProtectedItems： { (X1) selfValue { uniqueMember } }
grantsAndDenials： { grantRemove, grantAdd }

M.9 ACDF 示例

M.9.1 公众访问阅读

一个可辨别名为{C=GB, O=XC, CN=Smith}的普通公众成员,试图发起一个阅读操作来请求获取用户 Cayley 的电话号码值。该操作的访问控制决策在 GB/T 16264.3—2008 中定义。假设在名(称)解析中不包括别名解除引用,则第一个决策点是判断对目标条目的 Read 许可是否准予;该决策基于 ACDF 的下列输入:

——被请求的许可:Read;

——请求发起者:{C=GB, O=XC, CN=Smith},无唯一标识符;

——鉴别级别:无;

——被保护项:条目{C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley};

——表 M.1 中显示的元组。

表 M.1

用户	项	许　可	准予或拒绝	优先级	鉴别级别
allUsers	(*A1*)entry	*Browse*	G	10	None
allUsers	(*B1*)commonName 类型	*FilterMatch*	G	10	None
allUsers	(*B2*)commonName 值	*FilterMatch*	G	10	None
allUsers	(*B3*)objectClass 类型	*FilterMatch*	G	10	None
allUsers	(*B4*)objectClass 值	*FilterMatch*	G	10	None
allUsers	(*C1*)entry	*Read*	G	10	None
allUsers	(*D1*)commonName 类型	*Read*	G	10	None
allUsers	(*D1*)postalAttributeSet 类型	*Read*	G	10	None
allUsers	(*D1*)telephoneNumber 类型	*Read*	G	10	None
allUsers	(*D1*)facsimileTelephoneNo 类型	*Read*	G	10	None
allUsers	(*D2*)commonName 值	*Read*	G	10	None
allUsers	(*D2*)postalAttributeSet 值	*Read*	G	10	None
allUsers	(*D2*)telephoneNumber 值	*Read*	G	10	None
allUsers	(*D2*)facsimileTelephoneNo 值	*Read*	G	10	None
allUsers	(*L1*)telephoneNumber 类型	*Read*	D	10	None
allUsers	(*L1*)telephoneNumber 类型	*Compare*	D	10	None
allUsers	(*L1*)telephoneNumber 类型	*FilterMatch*	D	10	None

被保护的目标条目在 DACD-1、DACD-2 和 DACD-3 范围内(见图 M.4)。它没有entryACI。这 3 个 DACD 将表 M.1 中显示的元组(可应用于指定的请求发起者)用于 18.8 描述的 ACDF 过程。

在去掉不相关的行后,ACDF 最后只剩下两行要考虑:第 4 行允许对条目进行阅读,第 13 行拒绝对条目进行阅读。因此 ACDF 将拒绝该访问。

注:为了简化,本例中没有讨论与错误条件相关的许可和过程。然而,在上述拒绝访问的情况下,响应方 DSA 的行为应当被 18.2.3 或 18.4.1 中所描述的来支配,并且将包括使用 ACDF 来判断对于目标条目而言,*DiscloseOnError* 是否被准予。

M.9.2 公众访问搜索

一个可辨别名为{C=GB, O=XC, CN=Smith}的普通公众成员,试图发起一个搜索操作来请求获取基客体{C=US, O=ZCC, OU=R&D, OU=West}的所有下级用户(wholeSubtree)的所有属性的所有值;filter 规定了FilterItem equality 为:objectClass = organizationalPerson。该操作的访问控制决策点在 GB/T 16264.3—2008 的 10.2.6 中定义。

M.9.2.1 在搜索范围内检查每个条目是否有正确的条目许可

对于在搜索范围内的每个条目,假设在名(称)解析中没有别名解除引用,则第一个决策点是判断对

目标条目的*Browse* 是否准予。对于第一个这样的条目，ACDF 的输入为：

——被请求的许可：Browse；

——请求发起者：{C=GB，O=XC，CN=Smith}；

——唯一标识符：无；

——鉴别级别：无；

——被保护项：条目{C=US，O=ZCC，OU=R&D，OU=West}；

——表 M.2 中显示的元组。

由于被检查的条目仅包含在 DACD-1 中，则由 ACDF 收集的最初元组集合在表 M.2 中显示。注意这里没有考虑entryACI 。

从表 M.2 中去掉行的 ACDF 过程，将导致仅有第一行被保留下来；因此 ACDF 将准予被请求的访问。

类似的，ACDF 将对搜索范围内的每个条目都准予进行 *Browse* 操作。

表 M.2

用户	项	许　可	准予或拒绝	优先级	鉴别级别
allUsers	(*A1*)entry	*Browse*	G	10	None
allUsers	(*B1*)commonName 类型	*FilterMatch*	G	10	None
allUsers	(*B2*)commonName 值	*FilterMatch*	G	10	None
allUsers	(*B3*)objectClass 类型	*FilterMatch*	G	10	None
allUsers	(*B4*)objectClass 值	*FilterMatch*	G	10	None
allUsers	(*C1*)entry	*Read*	G	10	None
allUsers	(*D1*)commonName 类型	*Read*	G	10	None
allUsers	(*D1*)*postalAttributeSet* 类型	*Read*	G	10	None
allUsers	(*D1*)telephoneNumber 类型	*Read*	G	10	None
allUsers	(*D1*)facsimileTelephoneNo 类型	*Read*	G	10	None
allUsers	(*D2*)commonName 值	*Read*	G	10	None
allUsers	(*D2*)postalAttributeSet 值	*Read*	G	10	None
allUsers	(*D2*)telephoneNumber 值	*Read*	G	10	None
allUsers	(*D2*)facsimileTelephoneNo 值	*Read*	G	10	None

M.9.2.2　检查是否满足过滤器

对于搜索范围内的被准予 Browse 的每个条目，下一个决策点是判断对于属性objectClass 的Filter-Match 是否被准予。对于第一个这样的条目，ACDF 的输入包括：

- 被请求的许可：Browse；
- 请求发起者：{C=GB，O=XC，CN=Smith}；
- 唯一标识符：无；
- 鉴别级别：无；
- 被保护项：条目{C=US，O=ZCC，OU=R&D，OU=West}；
- 表 M.2 中显示的元组。

ACDF 将去掉表 M.2 中除第 4 行之外的所有行；因此访问将被准予。下一步，搜索操作将检查属性objectClass 是否有任意一个值与organizationalPerson 相等。由于被检查的条目是一个组织单元条目，则Filter 将被评估为FALSE 。

类似的，Filter 将对 CN=SE_DACD3 的条目评估为FALSE 。

对于在搜索范围内的另外两个条目(CN=Cayley 和 CN=Noether)，Filter 将被评估为TRUE 。对于每一个这样的条目，下一个访问控制决策是判断使Filter 被评估为TRUE 的属性值的*FilterItem* 是否

被准予。因为这些条目都包含在 DACD-1、DACD-2 和 DACD-3 中，则输入到 ACDF 的最初的元组集合为表 M.1。表 M.1 中的第 5 行对两个条目的访问请求都准予。

因此，搜索结果中包含了来自条目 Cayley 和 Noether 的信息。这两个条目的附加访问控制决策从本质上来说与 M.9.1 中显示的公众阅读示例是相同的。

M.10 基于规则的访问控制

为了举例说明基于规则的访问控制的用法，确定了下述的安全规则示例(注意：仅是为了举例说明的目的，并不表示任何完全的现实世界的策略)。

可能的安全标签值是一个分等级的集合，包括：未标注，不保密，受限，机密，保密，绝密。

许可证的值也是一个分等级的一组值，最多包括：未标注，不保密，受限，机密，保密，绝密。

注：这些规则可能通过在团体中包括更多的特权信息来扩展，这些特权信息在专用标注或安全类别中承载。

访问规则为：

a) 如果许可证级别高于或等于标签级别，则访问被准予；

b) 如果许可证级别低于标签级别，则访问被拒绝。

附 录 N
（资料性附录）
DSE 类型组合

表 N.1 规定了在没有影像时，将 DSA 信息模型应用到 DSA 时可能出现的 DSE 类型的多种组合（即 dseType 属性中被命名的比特的组合）。提供该表可以帮助将 DSA 信息模型阐述清楚。本目录规范中不强制要求支持这些或其他 DSE 类型的组合。

表 N.1 不存在影像时定义的 DSE 类型的组合

DSE 类型	*admPoint*	*cp*	*supr*	*nssr*	*sa*	家族成员	注释
根 (Root)			√	√			DSA 的根 DSE。如果 DSA 设置了 nssr 比特，则为第一级 DSA 的根 DSE。如果 DSA 设置了 supr 比特，则为非第一级的根 DSE
粘接 (Glue)							粘接 DSE
条目 (Entry)	√	√		√		√	客体条目 DSE；如果设置了 admPoint 比特，则其也是一个管理点；如果设置了 cp 比特，则其也是上下文前缀；如果设置了 nssr 比特，则其也是 nssr
别名(Alias)							别名条目 DSE
子条目 (Subentry)		√					子条目 DSE
下级引用 (subr)					√		下级引用 DSE；如果设置了 sa 比特，则为别名的下级引用点
直接上级引用 (immSupr)						√	直接上级引用
交叉引用 (xr)							交叉引用 DSE
注：DSE 类型 subr 和 immSupr 也可能会出现(可能在附加的比特 admPoint 中)，尽管在表中不方便出现。RHOB 所维护的子条目和管理点信息通过比特 rhob 的出现而指示。							

表 N.1 中的第一列标明了 DSE 类型，这些 DSE 类型不需要与其他 DSE 类型组合便可用于表示 DSE 的功能。

例如，可能会有一个 DSE 仅设置了比特 entry。第二列至第六列用勾号(√)标明了除第一列中标明的比特外，还可能被设置的附加 DSE 类型比特。这些比特可能被独立地设置。例如，一个 entryDSE 可能还设置了 nssr 比特、admPoint 比特和 cp 比特等，还可能是 admPoint、cp 和 nssr 比特的各种其他组合等。最后一列说明了表中该行所表示的各种 DSE 类型的组合。

表 N.2 规定了当存在影像时，可能出现的附加 DSE 类型的多种组合。类似于前一个表，表的第一列标明了 DSE 类型，在一个影像 DSA 中，这些 DSE 类型不需要与其他 DSE 类型组合便可用于表示 DSE 的功能。第二列至第六列用勾号(√)标明了除第一列中标明的比特外还可能被设置的附加 DSE 类型比特。这些比特可能被独立地设置。

表 N.2　当存在影像时定义的 DSE 类型的组合

DSE 类型	admPoint	cp	supr	nssr	sa	家族成员	注释
根 (Root)			√	√			第一级影像 DSA 的根 DSE，且设置了 nssr 比特
条目 (Entry)							对象条目 DSE；如果设置了 admPoint 比特，则其也是一个管理点；如果设置了 cp 比特，则其也是上下文前缀；如果设置了 nssr 比特，则其也是 nssr
别名 (Alias)	√	√		√		√	客体条目 DSE；如果设置了 admPoint 比特，则其也是一个管理点；如果设置了 cp 比特，则其也是上下文前缀；如果设置了 nssr 比特，则其也是 nssr
子条目 (Subentry)							别名条目 DSE
下级引用 (subr)		√					子条目 DSE
直接上级引用 (immSupr)					√		下级引用 DSE；如果设置了 sa 比特，则为别名的下级引用点
管理点 (admPoint)						√	直接上级引用
上下文前缀 (Cp)							没有用户属性的管理点 DSE(条目未被影像)；如果设置了 cp 比特，则其也是上下文前缀；如果设置了 nssr 比特，则其也是 nssr
非特定下级引用(nssr)							NssrDSE(条目未被影像)
注：表中所有的情况下，都设置了shadow 比特(因此在表中没有显式地表示出来)。如同在表 N.1 中的情况，DSE 类型 subr、immSupr 和shadow 也可能出现(可能在附加的比特admPoint 中)。最后，对于设置了subr 和/或immSupr 比特的 DSE，当被影像的条目信息被 RHOB 或影像维护的知识信息所覆盖时，则比特entry 和shadow 也可能出现。							

附 录 O
（资料性附录）
知识的建模

下述示例举例说明了一个假设的 DIT 及其到 3 个 DSA 的潜在映射以及 DSA 为了支持此映射而必须维护的信息(包括知识信息)。

在下面的图 O.1 和图 O.2 中，使用了下述符号。

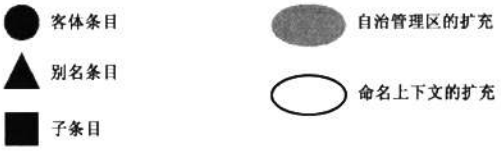


图 O.1 描述了一个假设的 DIT。它被分割为 4 个自治管理区：单个条目的退化情况{C=WW}和{C=VV}以及两棵根分别为{C=WW，O=ABC}和{C=VV，O=DEF}的子树。还有一个条目{C=VV，O=DEF，OU=K}是客体条目{C=WW，O=ABC，OU=I}的一个别名。

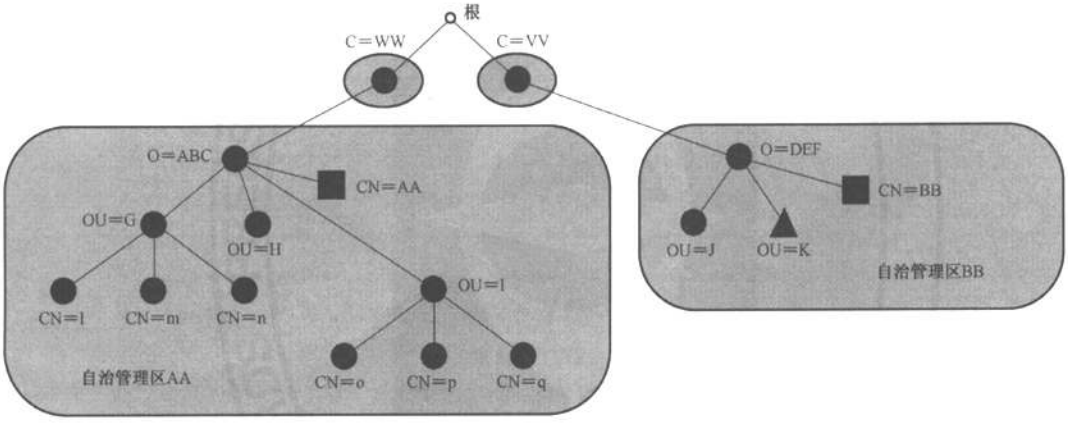


图 O.1 假设的 DIT

图 O.2 描述了将假设的 DIT 分割为 5 个命名上下文(A、B、C、D 和 E)以及它们与 3 个 DSA (DSA1、DSA2 和 DSA3)之间的映射。在图中，DSA1 拥有上下文 C，DSA2 拥有上下文 A、B 和 E，最后 DSA3 拥有上下文 D。

三个 DSA 拥有的知识如下所述：DSA1 将 DSA2 作为它的上级引用，且对处于 DSA2 内的{C=WW，O=ABC}的下级信息有一个非特定的下级引用。DSA2 是第一级 DSA，并且维护一个到 DSA1 的关于上下文 C 的下级引用以及一个到 DSA1 的关于上下文 E 的直接上级上下文的直接上级引用。DSA2 维护一个到 DSA3 的关于上下文 D 的下级引用。DSA3 也将 DSA2 作为它的上级引用，并且到 DSA2 有一个关于上下文 E 的交叉引用。

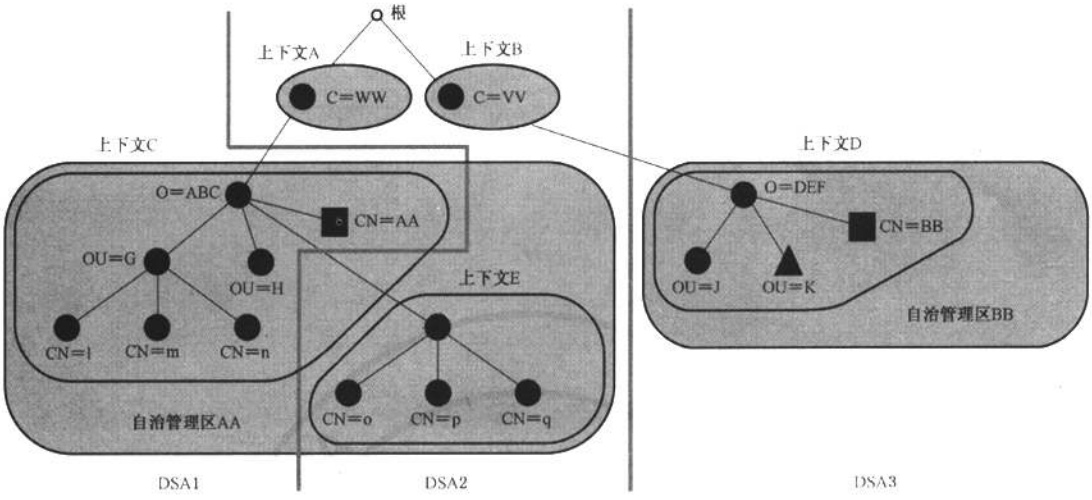


图 O.2　假设的 DIT 映射到三个 DSA

图 O.3 到图 O.6 描述了每个 DSA 内为支持该配置而拥有的信息(即每个 DSA 的 DSA 信息树)。下述符号用于这些图中。

图 O.3 举例说明了 DSA1 的 DSA 信息树。

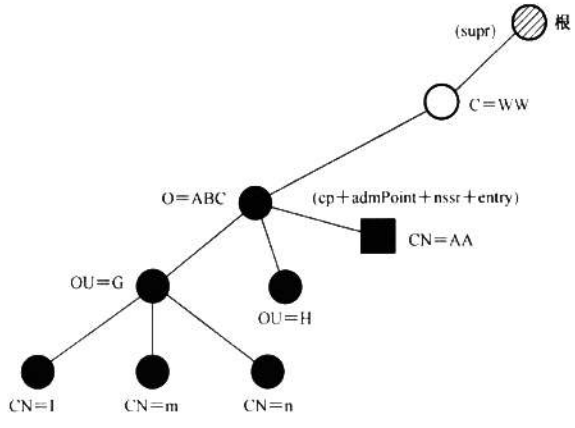


图 O.3　DSA1 的 DSA 信息树

由于 DSA1 不是第一级的 DSA,因此它的根 DSE 拥有一个上级引用,在本例中是 DSA2 的访问点。该 DSE 的类型为 root + supr。

DSA1 拥有一个粘接 DSE 来表示它的知识,名(称)为{C=WW}。

自治管理区 AA 被分割为两个命名上下文 C 和 E。其中上下文 C 由 DSA1 拥有。为了简化这个示例,假设与访问控制相关的特定管理区和子模式信息是一致的,且假设整个自治管理区内有一个单独的访问控制区和一个单独的子模式。上述假设的结果便是示例中的每个自治管理区仅需要一个单独的子条目(可能有多个目的)。

对于 DSA1，在{C=WW，O=ABC}处的 DSE，表示自治管理区 AA 的管理点、上下文 C 的上下文前缀以及到 DSA2 的一个非特定下级引用等，它的类型为 entry + cp + admPoint + nssr。区操作信息在子条目{C=WW，O=ABC，CN=AA}中拥有。

DSA1 拥有下述包含在上下文 C 中的条目：{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}以及{C=WW，O=ABC，OU=G，CN=n}。

图 O.4 举例说明了 DSA2 的一个可能的 DSA 信息树。

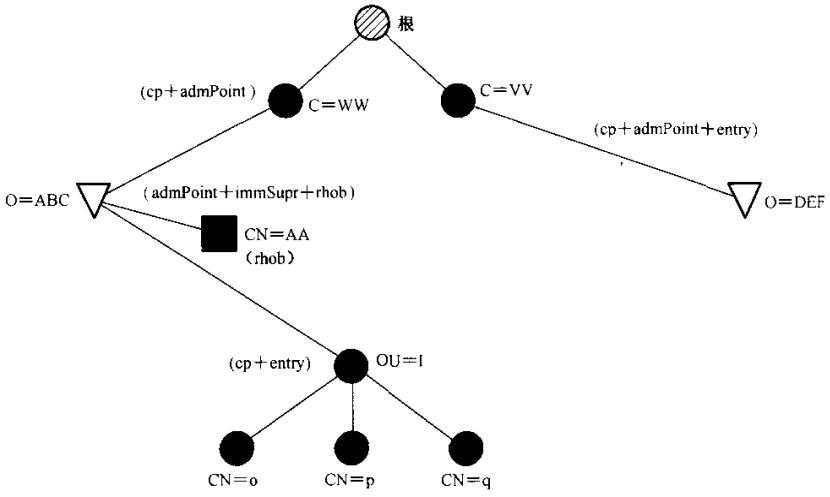


图 O.4　DSA2 的 DSA 信息树

在这个假设的情况下，DSA2 是第一级的 DSA，因此它的根 DSE 没有上级引用。

两个退化的自治管理区{C=WW}和{C=VV}，表示它们的 DSE 的类型为 cp + entry + admPoint。

DIT 的下级知识由两个下级引用 DSE 来表示：{C=WW，O=ABC}和{C=VV，O=DEF}。在前一种情况下，该 DSE 的类型为 subr + admPoint + immSupr + rhob，原因将在下面描述。

在图 O.4 中，DSA2 被配置为假设有一个单独的子条目拥有关于 AA 的区操作信息。这就要求在 DSA2 中有一个子条目的拷贝（为了合理的性能）。实现上述需求的一个方法可以是建立 DSA1 和 DSA2 之间的 NHOB，以此来维护此子条目的拷贝。在这种情况下，区操作信息存储于名为{C=WW，O=ABC，CN=AA}的 DSE 内，它的类型为 subentry + rhob。在名为{C=WW，O=ABC}的 DSE 内拥有的 administrative-role 属性由 DSA1 向 DSA2 提供，作为 NHOB 的一部分。由于这个原因，该 DSE 的类型为 admPoint + rhob。

最后，命名上下文 E 包含如下信息：上下文前缀 DSE {C=WW，O=ABC，OU=I}，它的类型为 cp + entry；其他三个条目 DSE{C=WW，O=ABC，OU=I，CN=o}、{C=WW，O=ABC，OU=I，CN=p}和{C=WW，O=ABC，OU=I，CN=q}。

配置 DSA2 的另一个替代方式在图 O.5 中举例说明。

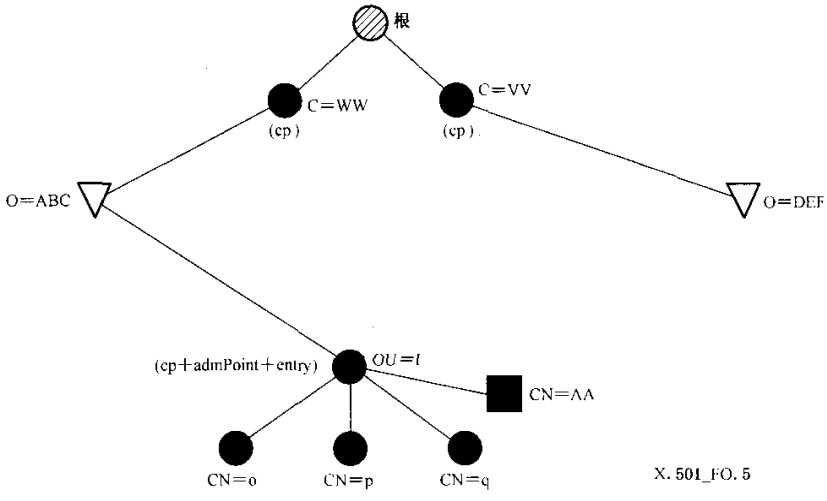


图 O.5　DSA2 的替代 DSA 信息树

与图 O.4 中所描述配置的不同之处仅在于对区操作信息的处理，这种配置的动机可能是希望避免维护与 DSA1 之间的 NHOB。

这种情况的策略是将 AA 分割为两个自治管理区（即分割域访问控制信息以及类似地分割子模式信息），其中一个与上下文 C 一致，另一个与上下文 E 一致。

在这种情况下，上下文前缀 DSE{C=WW，O=ABC，OU=I}也变为一个管理点，该 DSE 的类型为 cp + admPoint + entry。不再使用一个由 DSA1 提供的影像子条目作为 NHOB 的一部分，取而代之的是精简的区操作信息存储在子条目{C=WW，O=ABC，OU=I，CN=AA}中。

图 O.6 举例说明了 DSA3 的 DSA 信息树。

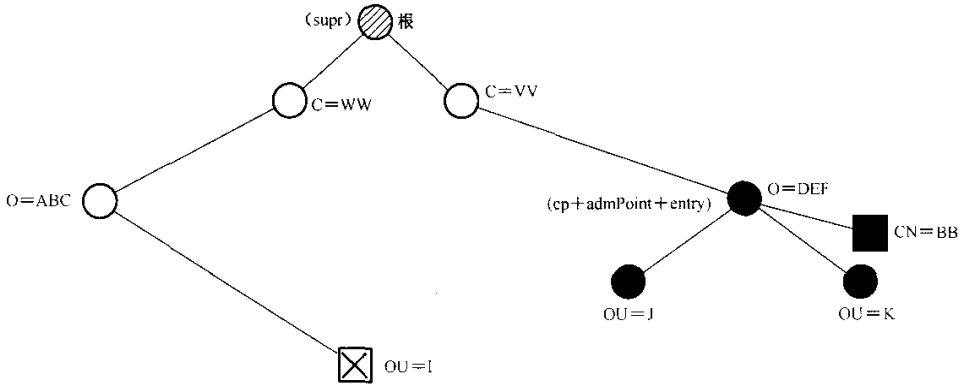


图 O.6　DSA3 的 DSA 信息树

同 DSA1 一样，DSA3 不是第一级的 DSA。它的根 DSE 拥有一个上级引用，本例中是 DSA2 的访问点。该 DSE 的类型为 root + supr。

DSA2 拥有一个 glue DSE 来表示其知识，名（称）为{C=VV}。

自治管理区 BB 与命名上下文 D 相一致。为了简化这个假设的示例，如同自治管理区 AA 的情况一样，假设与访问控制相关的特定管理区和子模式信息是一致的，且假设整个自治管理区内有一个单独的访问控制域和一个单独的子模式。上述假设的结果便是示例中的每个自治管理区仅需要一个单独的（但有多个目的的）子条目。

对于 DSA3，在{C=VV，O=DEF}处的 DSE，表示自治管理区 BB 的管理点以及上下文 D 的上下

文前缀,它的类型为 entry + cp + admPoint。区操作信息在子条目{C=VV, O=DEF, CN=BB}中拥有。

DSA3 拥有包含在上下文 D 中的一个客体条目和一个别名条目:{C=VV, O=DEF, OU=J}(类型为 entry)和{C=VV, O=DEF, OU=K}(类型为 alias,且包含一个属性 aliasedEntryName,其值为{C=WW, O=ABC, OU=I})。

最后,DSA3 拥有一个到上下文 E 的交叉引用,一个类型为 xr 的 DSE,名(称)为{C=WW, O=ABC, OU=I}。

附 录 P
（资料性附录）
作为属性值存储或作为参数使用的名（称）

当一个名（称）作为一个属性值存储于其他属性中时，或者作为一个属性值在某些交互中传递时（例如，一个别名指针），总会有这样的问题存在，即所存储的名（称）是一个可替代辨别名还是一个主辨别名？它是否能够包含替代值？它是否能够包含上下文信息等。在本系列目录规范的必要位置会提及这些特定的限制。一般来说，对于作为一个属性值存储的名（称）没有限制；但是，为了简化与较老系统的交互并提供可预知的结果，有下述一些建议：

当一个操作属性的值是某个客体的名（称）时（例如creatorsName），则此名（称）应当是该客体的主辨别名。可替代值和上下文信息不是必需的，但也可以包含在内。

当名（称）中的某个 RDN 的属性类型和值对包含了多个可辨别值时，这些可辨别值使用了valuesWithContext，则应当使用主可辨别值作为AttributeTypeAndDistinguishedValue 中的value，这样与较老的系统之间的交互是可预知的。

当一个用户属性的值是一个名（称）时（例如名（称）组中的成员），它可以是任意一个可替代辨别名、多可辨别名或所有可辨别名等，但建议使用主辨别名，这样与较老系统的交互是可预知的。另外，如果在这样的引用属性中包含了上下文和可替代值，这些上下文或可替代值一般是没什么作用的。

当属性是 DSA 信息树中的一部分，且用于名（称）解析中（如知识引用）时，则它须是主辨别名，且每个 RDN 都须在AttributeTypeAndDistinguishedValue 中为每个属性携带上下文和所有的可替代可辨别值，以便增强名（称）解析功能，并且与较老系统的交互也是可预知的。

附 录 Q
（资料性附录）
子过滤器

一个过滤器通过狄摩根(deMorgan)规则逐步展开后，可被转换为一系列的子过滤器。(狄摩根规则用于过滤器所使用的三值逻辑)。将一个过滤器看做一棵子树，这棵子树中，非叶结点对应于每个and{}、or{}、not{}，而每个叶结点是一个过滤项。子树中的每个弧表示 and{}、or{}、not{}中的一个元素；在 not{}情况下，仅有一条弧。

使用下面规则首先将每个 not{}发展成为叶结点：

not{and{x,y,z}}等同于 or{not{x}, not{y}, not{z}}

not{or{x,y,z}}等同于 and{not{x}, not{y}, not{z}}

not{not{x}}等同于 x 剩下的 not 直接应用于过滤项。

然后，再使用下面的规则来组合 and 和 or，并且朝着叶子的方向删除 and，使树再次得到简化：

and{and{x,y,z}, p, q}等同于 and{ x,y,z,p,q}

or{or{x,y,z}, p, q}等同于 or{ x,y,z,p,q}

and{or{x,y,z}, p, q}等同于 or{and{x,p,q}, and{y,p,q}, and{z,p,q}}

and (x,y,z)等同于 and{任何顺序的 x,y,z }

or (x,y,z)等同于 or{任何顺序的 x,y,z }

若 and{}为 TRUE，则 or{and{},x,y,z}总是 TRUE，且 and{and{},x,y,z}等同于 and{x,y,z}

若 or{}为 FALSE，则 and{or{},x,y,z}总是 FALSE，且 or{or{},x,y,z}等同于 or{x,y,z}

注—这里所使用的表示法，如{x,y,z}等的意思是零个、一个或多个成员的集合，如 x、y 和 z 等。

通过逐步地应用这些规则，过滤器最终将被转换为如下的一个规范格式：

or{and{p1, p2…,and{q1, q2… }…}其中，每个 pi 或 qi 或者是一个过滤项 F，或者是一个否定过滤项 not{F}。

则每个 and{p1, p2… }是原始过滤器的子过滤器。

附 录 R
（资料性附录）
复合条目的命名方式及其使用

本地成员名(称)的概念在 9.3 中做了介绍。本目录规范对于在结构规则之外如何分配名(称)不做任何限制。然而，在某些情况下，为家族成员建立一个命名方式对于取得所希望的效果来说是必要的。根据这个简单的方式，不同复合条目中的类似家族成员能够具有相同的本地成员名(称)。例如，一个拥有一个电话号码及其相关特性(如用法、话务量、呼叫限制等)的家族成员，能够在不同的复合条目中都具有相同的本地成员名(称)。当复合条目是层次结构组中的成员时(见 GB/T 16264.3—2008 中的 7.13)，这是非常必要的。还有另外一种方法来建立一个命名方式，即让一个家族成员的 RDN 反映出该成员所拥有的信息。例如，一个通信地址(例如一个电话号码或 Email 地址)能够拥有一个等于{ comAdressName = telephone1}或者{ comAdressName = emailAddress3 }的 RDN。这样，通过对 RDN 执行最初的子字符串匹配即可定位电话号码家族成员。

下述使用命名方式的例子也是使用控制属性的一个例子，该控制属性由搜索规则组件 additionalControl 所指向(见 16.10.8)。这个例子应当被明确地理解为仅是一个例子，而不是能够实现的一个规范，也不是其他规范能够正式引用的一个规范。它仅是一个给定的示例，显示了一个控制属性是如何被构造的以及什么规范可以与这样的控制属性相关联。

一个搜索规则控制了在 DIT 某个特定区内的一个搜索行为。这个服务对某些特定的访问用户是适合的。

然而，条目的“拥有者”，如由客户条目所表示的客户，对于这个与该特定条目相关联的服务应如何被限制和调整，可能会有各自不同的可能合法的需求。这些各自不同的需求可以是：

a) 一个条目中的信息可能以不同的语言来提供。然而，该条目的拥有者可能会请求，如选址信息应当以某种特定的语言返回，而不论访问用户在搜索请求中使用的是哪种语言，也不论访问用户可能请求的是什么。该功能不能被上下文功能提供；

b) 一个条目的所有者可能会请求返回一个假的地址或一个替代的地址，即使访问用户匹配到了一个真实的地址；

c) 当一个访问用户匹配到一个电话号码时，他/她将得到所有的或选择的电话号码以及相关的信息。

这些不同的限制和调整可以由控制属性示例 markingRules 来执行。该属性可由某个特定服务管理区内的一个条目或一个复合条目的祖先所拥有。它的定义如下：

```
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 }
```

控制属性 markingRules 的一个值表示了一个规则，该规则规定了如何对在搜索评估中已经匹配的复合条目中的成员加标注或不加标注以及如何从输出中删除已经匹配的非成员条目。

组件 searchRules 指示了该属性的某个特定值是应用于哪个搜索规则的。如果控制搜索规则拥有一个 id 等于该组件内的一个值，则须应用由该控制属性值所指定的标注。一个给定的搜索规则能够在本属性类型的多个值中表示。如果该组件缺失，则此标注规则应用于所有的搜索规则。

如果 familyGrouping 在匹配过程中取值为 strand 或者 multiStrand，则组件 markingStrands 是唯一相关联的。它指示了对于分支的某种可能标注，什么样的条件应当出现。该组件中的过滤器将针对这样的每个分支做评估，即分支中的所有成员都被标注为搜索过滤器匹配结果中的成员。如果至少有一个分支被评估为 TRUE，则此过滤器被评估为 TRUE。匹配所遵循的是 GB/T 16264.3—2008 的 7.8 中规定的规则。如果该组件缺失，则缺省值是一个总被评估为 TRUE 的过滤器。

如果 *familyGrouping* 在匹配过程中取值为 *strand* 或者 *multiStrand*，且 markingStrands 被评估为 TRUE，则组件 localName 是唯一相关联的。它通过选择零个或多个家族成员，指示了哪个分支中的家族成员应当被标注为参与成员。如果某个家族成员的本地成员名(称)的 RDN 个数与本组件中过滤项的个数相同，且每个过滤项与相应的 RDN 之间都一一匹配，则这样的家族成员将被选中。称一个过滤项与某个 RDN 匹配指的是该过滤器与 RDN 的一个 AVA 相匹配。如果任意一个分支穿过被选中的家族成员，则其所有的家族成员都被标注为参与。

组件 explicitUnmark 中规定了一个过滤器，如果与某个条目或家族成员匹配时，该过滤器将使得该条目或家族成员被显式地不标注。显式不标注仅针对于那些已经被选中在搜索结果中返回的条目和家族成员。

如果一个家族成员被显式不标注，且在搜索过滤器匹配中该家族分组不是 entryOnly，则该显式不标注的成员的所有家族条目下级也都将被显式不标注。对一个非家族成员进行显式不标注意味着将此条目从结果中删除，就好像从来没有匹配过一样。对一个家族成员进行显式不标注意味着这样一个家族成员不得被包含在结果中。

对控制属性 markingRules 的评估执行有两个阶段过程。

第一个阶段仅在这样的情况下执行，即在匹配过程中，familyGrouping 取值为 strand 或者 multiStrand，且在条目信息选择中 familyReturn 不是 contributingEntriesOnly。

在第一个阶段中，仅考虑在搜索过滤器评估中完全满足下述所有条件的复合条目：

a) 祖先拥有一个 markingRules 控制属性；

b) 一个或多个值可应用于控制搜索规则，且包含 localName 组件。

其他成员则被标注为参与成员，如上所述。

在第二个阶段，根据出现的控制属性类型 markingRules，所有当前已经被标注为参与成员的家族成员以及所有的非家族成员都被检测，然后再检测属性是否拥有一个或多个可应用于控制搜索规则的值。如果这样的话，组件 explicitUnmark 如果出现，则被评估。如果针对某个家族成员评估为 TRUE，则该家族成员被显式地不标注，即它既不被标注为参与，也不起作用。所有的下级家族成员也都被显式地不标注。如果针对的是非家族成员，则显式不标注的效果同搜索过滤器从来没有匹配过该条目一样。

附 录 S
（资料性附录）
命名概念和考虑

S.1 历史告诉我们……

由于本系列目录规范的第1版是在1998年第一次出版的，而到目前为止，在信息技术中发生了数不清的变化。这些变化中的一些是可预见和可预知的，而另一些是不可预见和预知的。因此相应的，在当前出版的本系列目录规范中的大部分内容与1998年出版时一样还是可应用的，而其他部分很明显不是。在本附录中，我们将标识出目前需要考虑的有关这两个集合的主要概念。

S.1.1 仍然有效的初始概念

幸运地，许多目前仍然有效的初始的目录概念正是初始规范中的那些非常基础的概念。特别是：

——目前仍然有效的是，目录是条目的一个集合，其中每个条目以属性的形式拥有信息，描述了一个特定的现实世界中的客体；

——目前仍然有效的是：将目录条目认为是已命名的实体，而且这些名(称)以一种层次关系进行组织，这种层次关系体现了相应的现实世界客体间可能被组织的某些合理的分类法；

——目前仍然有效的是：在命名中提供灵活性，并允许根据层次关系对命名机构进行授权；

——目前仍然有效的是：希望这些条目可以在目录服务器的集合内进行分布；

——目前仍然有效的是：如果给出了一个关于现实世界客体的任意数据内容，则希望目录能够快速发现一个描述了该客体本身的条目；以及

——目前仍然有效的是：这些给定的任意数据内容可以或者是条目的名(称)，或者是条目所包含的一些非命名属性。

S.1.2 不再有效的初始概念

尽管有上述所列的基础概念仍然有效，但也还有其他的基础概念根据过去几十年的经验来看已经不再有效。这些概念中的一些有的已经被本系列目录规范所修订，而另外一些还没有。这些已经改变了的内容包括：

——目前不再有效的是：希望任何给定的现实世界的客体都精确地被一个条目所描述(即目前存在相关条目)。

——目前不再有效的是：尽管有安全方面的考虑，但希望包含在目录中的命名知识可以足以获得目录内的所有已命名条目(即目前存在多DIT)。

——目前不再有效的是：包含在目录内的命名知识是获取一个特定的已命名条目的唯一方法(即目前可能部署一个目录外部的服务来帮助定位一个已命名条目)。

——目前不再有效的是：可辨别名总是唯一地标识了一个单独的条目(即目前同一个DN可能被用来对存储于两个或多个DIT内的条目进行命名)。

——当给定一个关于某个客体的任意的非命名数据(应该是一个实例中的数据)，而该客体可能处于多个目录服务器中的一个时，目前不再有效的是：用来定位一个所期望条目的唯一机制是分布式搜索(即目前的需求是可以拥有一个单独的本地服务器来明确地标识所关联的条目，而不论该条目是否被该服务器所拥有)。

S.2 关于名(称)解析的新考虑

因为命名对于一个目录服务的成功操作是如此重要，且由于关于目录服务特性的这些基本假设已经纳入考虑范围，因此本条将讨论关于名(称)解析的主题。本条首先介绍了关于名(称)解析的一些关

键性方面，然后提出当前的名(称)解析模型已经不再足以满足所有的目录需求。然后本条继续提出了一个替代方法来扩展模型以适应这些需求，同时保证后向兼容于已经存在的系统。

S.2.1 显式的知识模型

自从第一次出版后，本系列目录规范就提供了分布式的名(称)解析。从概念上说，在一个给定名(称)空间中参与的每个 DSA 都被要求维护最小的命名知识，以确保分布式的名(称)解析能够跨整个 DIT 以一种可预知的方式存在(当然，依赖于所有参与的服务器的真正可达到的能力)。明确地，最小的知识包括上级知识引用和下级知识引用，使得 DIT 被认为是具有“良好的连通性”，如果没有更好的术语来表达的话。使用此模型，在解析一个声称名时所包含在内的任何 DSA 都应当明确知道下面三个条件中的哪一个被满足：

——被声称名属于该 DSA 所拥有的一个命名上下文；

——被声称名属于该 DSA 已知的某个下级名(称)空间；或者

——上述两个都不是。

在第一个实例中，该 DSA 将通过标识出条目或判断出它的不存在而完成名(称)解析过程。在第二个实例中，名(称)解析过程将由到另一个 DSA 的下级引用来继续完成。在第三个实例中，名(称)解析过程将由上级引用来继续完成，如果这样的上级引用存在的话，否则解析过程将结束。一旦 DIT 被良好地关联，则名(称)解析将总是会得到一个明确的答案，即条目或者存在于某个特定的 DSA 内，或者不存在。

图 S.1 描述了一个示例场景，在该场景中名(称)解析过程是基于 DSA2 所拥有的一个条目的名(称)而进行的，如图所示。在图中，知识引用将以虚线箭头来表示。注意，DSA3 除了拥有一个作为 DSA2 下级的命名上下文外，它还拥有一个到 DSA1 的上级引用，而 DSA1 拥有根命名上下文。根据图中所包含的 DSA，名(称)解析过程如下所述：

——对于 DSA1，名(称)解析过程将沿着下级引用到达 DSA 2；

——对于 DSA2，名(称)解析将找到已命名的条目；

——对于 DSA3，名(称)解析将沿着上级引用到达 DSA 1，并按照上述的过程执行；

——对于 DSA4，名(称)解析将沿着上级引用到达 DSA 1，并按照上述的过程执行。

在所有情况下，名(称)解析都将找到已命名条目。

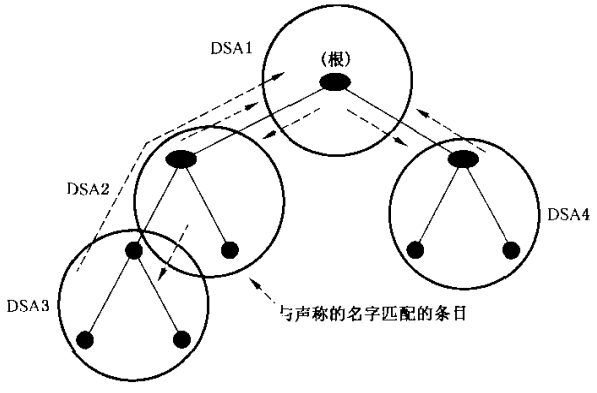


图 S.1

值得注意的是，尽管有一些优化是可应用的，但回答的成功不变。两个明显的优化包括：在 DSA3 中使用一个直接上级引用(避免穿过 DSA1 达到 DSA2 的需要)，在 DSA4 中使用一个交叉引用，允许名(称)解析的过程是从 DSA4 直接到 DSA2(再次避免了穿过 DSA1 达到 DSA2 的过程)。在任何情况下，本例中的名(称)解析将总是得到相同的答案。

不幸的是，正如上面所提到的，DIT 不再被假设为是一个具有良好连接的 DIT。存在着多 DIT，有

时还包括重复的 DN。目前先不考虑重复名(称)的可能性,我们假设这样一个情况,如图 S.2 所示。在本例中,我们有两个 DIT,每个 DIT 内部都是良好连接的,但是每个 DIT 都不拥有另一个的知识。第一个 DIT,如上例中的情况一样,包含了从 DSA1 到 DSA4 所拥有的条目。第二个 DIT 包含了从 DSA5 到 DSA6 所拥有的条目。注意,如果将其认为是一个单独的 DIT,也仍然是合理的,因为相对于一个概念上的根而言,所有的 DN 都是明确的。然而,该 DIT 与一个良好连接的单独 DIT 之间还是有区别的,是因为基于这样的事实,即 DSA1 与 DSA5 都缺乏对作为根下级的命名上下文的完整知识。

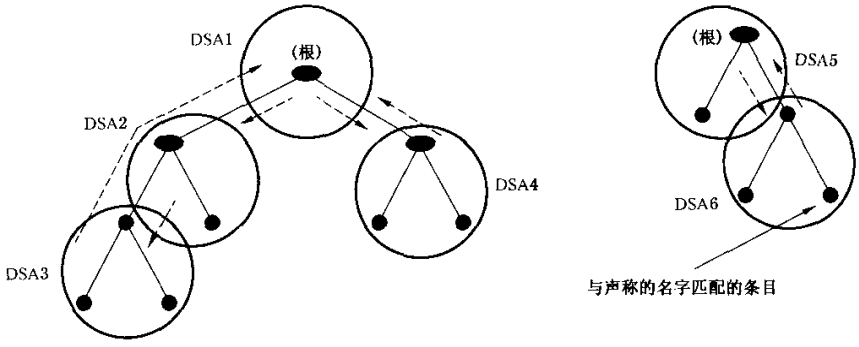


图 S.2

如图所示,当给定了所指示的条目名(称)时,名(称)解析工作如下所述:

——DSA1 到 DSA4 将不会找到该条目;

——DSA5 和 DSA 6 将成功找到该条目。

寻找条目失败可能是一个问题,也可能不是一个问题,取决于当时的需求。剩下的讨论将说明什么情况下这是一个问题。

在寻找一个解决方案时,首先部署使用交叉引用或某些类似的结构,看起来是合理的。例如,考虑使用一个交叉引用,该交叉引用给 DSA4 提供了关于 DSA6 中命名上下文的知识。上述讨论如图 S.3 举例说明。

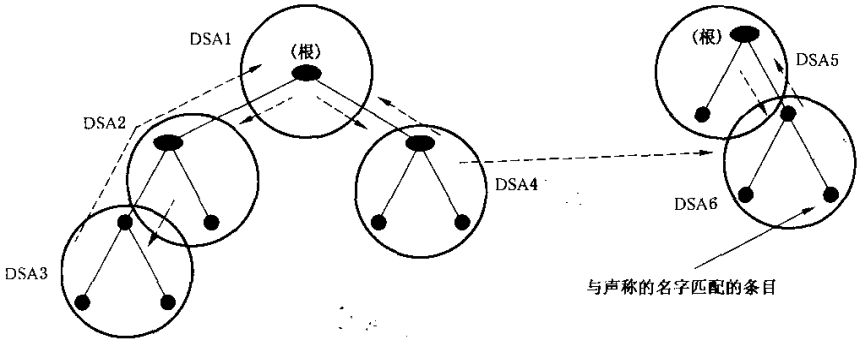


图 S.3

对这种方法的快速分析显示了如下场景:

——DSA1 到 DSA3 的名(称)解析将失败;

——DSA4 到 DSA6 的名(称)解析将成功。

尽管初看起来,这与上面所描述的相比,没有更受欢迎或更不受欢迎,但有一个关键的区别是:在一个良好定义的 DIT 范围内进行的名(称)解析,目前给出了不一致的结果。

为了给出一致的结果,有两种选择可以应用,都使用了当前存在的知识结构。一种方法是使用多个交叉引用,这样每个源 DSA 都拥有到目标命名上下文的交叉引用。这个概念在图 S.4 中显示。注意在这个场景中,位于左侧视图中的名(称)解析将始终如一地找到 DSA6 所拥有的命名上下文内的任何名

(称)。但这种方式下,在 DSA5 中的名(称)将找不到,而在 DSA1 到 DSA4 中的名(称)对于 DSA5 和 DSA6 来说是不可访问的。

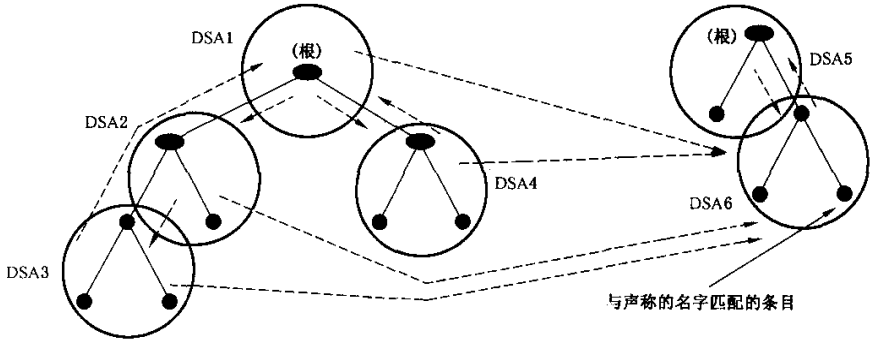


图 S.4

然而,当交叉引用在 DSA1 内实现时,可能会出现一个问题。即从 DSA1 的视角来看,在 DSA5 中引用的命名上下文认为是它所拥有的某个条目的命名上下文,但实际上可能是该条目的下级命名上下文。特定的,如果 DSA1 相信自己对根上下文有权限,则可能真的需要该交叉引用是一个下级引用,这样我们引入了第二种选择。

在左侧视图中,对 DSA6 的命名上下文提供一致的名(称)解析的第二种方法是:在 DSA1 中创建一个根级别的下级引用。这种方法在图 S.5 中举例说明。如果按照这种方法实现,则在 DSA2、DSA3 或 DSA4 中的任何交叉引用将是最优的。

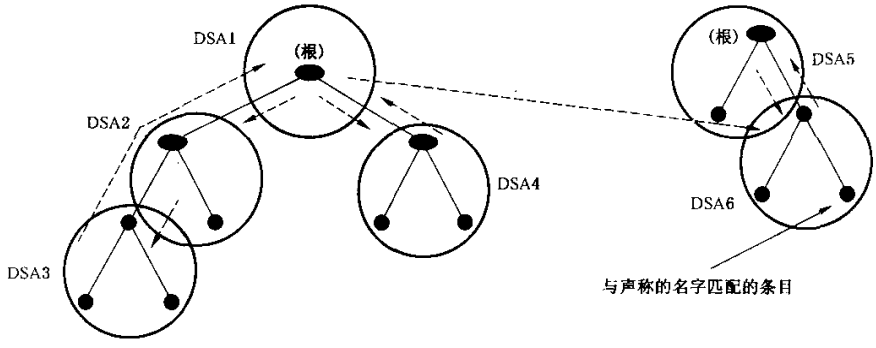


图 S.5

再将此概念扩展一步,如图 S.6 所示,又提出了一些有趣的问题。在本图中,DSA1 拥有 6 个 DSA 所拥有的所有命名上下文的完整下级知识,同时 DSA5 仅拥有 DSA5 或 DSA6 所拥有的命名上下文知识。注意,如果将 DSA2 和 DSA4 所拥有的命名上下文的下级知识给了 DSA5,则整个图将再次表示一个具有良好连接的 DIT。然而,本例中不是这种情况。从本质上来说,一个具有良好连接的单个 DIT 和多 DIT 之间的区分已经变得模糊,因此出现了一种情况,这种情况在当前的目录规范中无法充分地进行建模。根据非常实际的一种判断,本图是在许多环境中都被部署的一种情况。

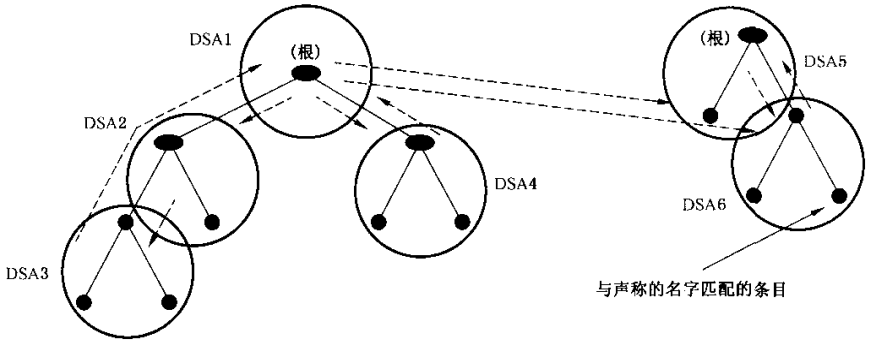


图 S.6

为了看看还有哪些复杂的情况在等待着我们，让我们先考虑这样一种情况：一个单独的 DN 存在于多个 DIT 中。一个简单的场景在图 S.7 中所示。在本例中，DSA5 中存在一个 DIT。这个新 DIT 的名(称)空间与之前示例中的名(称)空间有部分重叠，但是同时还引入了一些新的名(称)。特别的，箭头指向一个条目，该条目与其双亲一起，与 DSA2 中的某个条目共享了该条目的名(称)。这一对共享名(称)的条目可能拥有相同的信息，也可能不拥有，因此它们不应被认为是同一个条目。

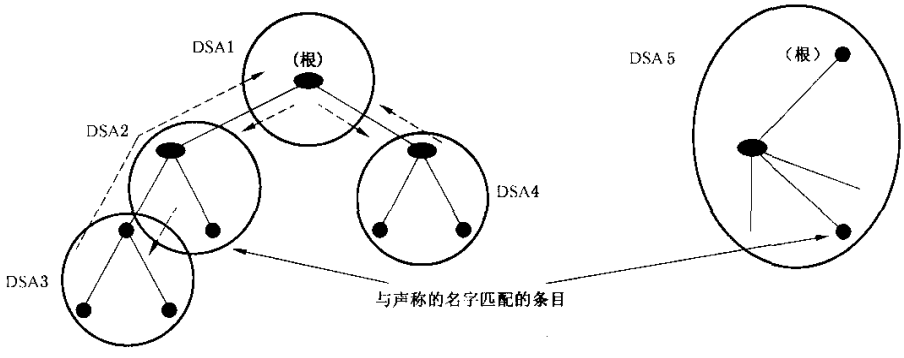


图 S.7

如果在这两个 DIT 间没有任何引用，则名(称)解析将非常有预知性。在一个特定的 DIT 内，将总是得出同样的结果。如果引入引用，将带来特殊的问题：

——将永远不会使用从 DSA2 到 DSA5 的交叉引用，或从 DSA5 到 DSA2 的交叉引用，因为它们每个都相信自己所拥有的命名上下文正是自己感兴趣的；

——从 DSA3 或 DSA4 到 DSA5 的交叉引用，其优先级高于上级引用；

——如果同时出现从 DSA3 到 DSA5 的交叉引用和从 DSA3 到 DSA2 的直接上级引用，则其行为不可确定；

——如果同时出现从 DSA1 到 DSA2 和 DSA1 到 DSA5 的下级引用，则其行为不可确定。

很明显，这些问题都是不希望发生的。另外还能够考虑到其他一些额外的场景，然而，上述所列的问题已经足够表明这种方法是不可接受的。不幸的是，导致这种特定类型的命名和知识分布场景的情况在现实世界中的出现频率还较高，不能被忽略。因此，需要某些形式的扩展。本条的剩余部分将讨论一种替代的方法。

S.2.2 具有隐含知识的名(称)解析

在上述所有的讨论中，名(称)解析都完全取决于 DSA 所拥有的显式的知识引用。在本系列目录规范之外(最显著的是在 IETF 内)，有一个概念是通过使用隐含知识来部分地解析名(称)，与这个概念相关的工作已经在许多年前就开始了。也就是说，有一个工作实体，在客户与某个 DSA 初始接触之前，就

使用了包含在 DN 本身内的信息来部分地解析名(称)。从概念上说,如果名(称)中包含了足够的信息,则第一个接触的 DSA 就能够提供一个确定的答案;它或者包含了这个已命名条目,或者明确已知该条目不存在。

这个概念在下面的图 S.8 中显示。本图与图 S.1 基本相同,所不同的是本图中的 DSA 没有包含知识引用。所替代的是,知识隐含在 DN 中,且通过使用本目录之外的其他服务来进行解析,这个服务在本图中显示为一个黑盒。注意,该黑盒能够提供指向除根以外的所有命名上下文的指针。根据这种方法无法确定根的位置,因为与根相关联的空 DN 缺少任何关于其位置的隐含信息。

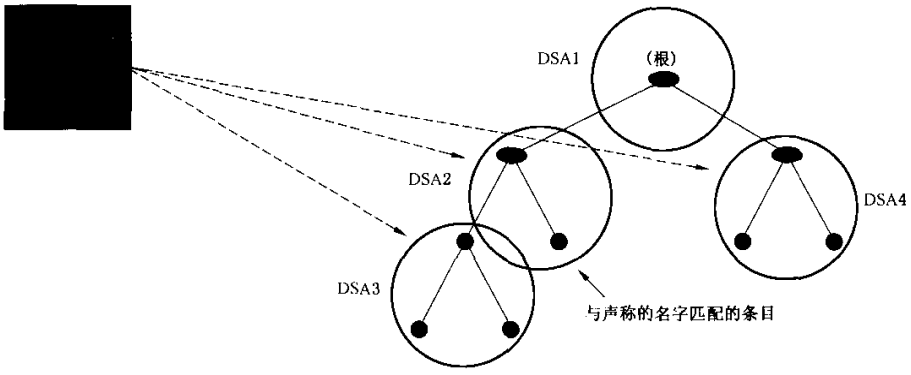


图 S.8

现在让我们考虑一下图 S.9 中的情况,该图相应于图 S.2 中所显示的 DIT。在这个场景中,假设同一个隐含知识模型,即同一个黑盒服务,也能够指向图中右侧新增加的命名上下文。与图 S.2 中的情况有所不同的是,DSA5 和 DSA6 中的命名上下文不会建立一个完全不同的视角。假设存在必要的连接,则这所有的 6 个 DSA 都处于相同的视角中,尽管在这 6 个 DSA 间并不存在显式的知识。

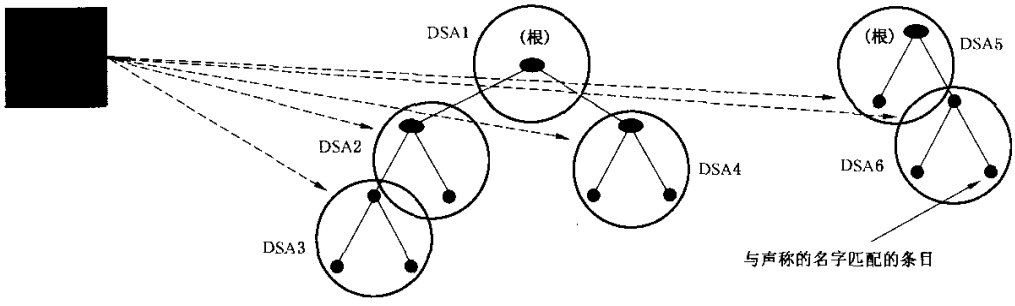


图 S.9

关于这个主题的最早出版的工作是 RFC 2247,该 RFC 在可辨别名和域名系统(DNS)之间定义了一个映射。此后,又出版了其他一些文稿,还有一些是正在进行中。到目前为止,关于此主题的已经出版的所有工作都是基于使用一个特殊的命名属性,已知为域组件(domainComponent 或 dc)属性。

为了对讨论进行简化,关于此主题的工作已经将一个概念进行了发展,即在最重要的 RDN 中使用 dc 属性构造的 DN 可以被隐含地解析到拥有该命名上下文的 DSA 中,而所使用的外部黑盒服务是 DNS。于是该 DSA 被联系,且在该 DSA 中完成了名(称)解析。

附 录 T
（资料性附录）
定义的英文字母顺序索引

本附录以英文字母表顺序列出了本目录规范中所定义的所有术语以及定义它们的章条的交叉引用。

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

U

中华人民共和国
国家标准
信息技术 开放系统互连 目录
第2部分:模型
GB/T 16264.2—2008/ISO/IEC 9594-2:2005
*
中国标准出版社出版发行
北京复兴门外三里河北街16号
邮政编码:100045
网址 www.spc.net.cn
电话:68523946 68517548
中国标准出版社秦皇岛印刷厂印刷
各地新华书店经销
*
开本 880×1230 1/16 印张 16.25 字数 491千字
2008年12月第一版 2008年12月第一次印刷
*
书号:155066·1-34752 定价 104.00元