国 际 电 信 联 盟

ITU-T

国际电信联盟
电信标准化部门

X.518

(08/2005)

X系列：数据网、开放系统通信和安全性

号码簿

信息技术 — 开放系统互连 — 号码簿: 分布式操作规程

ITU-T X.518建议书

国际电信联盟

ITU-T X系列建议书

数据网、开放系统通信和安全性

公众数据网	
业务和设施	X.1-X.19
接口	X.20-X.49
传输、信令和交换	X.50-X.89
网络概貌	X.90-X.149
维护	X.150-X.179
管理安排	X.180-X.199
开放系统互连	
模型和记法	X.200-X.209
服务限定	X.210-X.219
连接式协议规范	X.220-X.229
无连接式协议规范	X.230-X.239
PICS书写形式	X.240-X.259
协议标识	X.260-X.269
安全协议	X.270-X.279
层管理对象	X.280-X.289
一致性测试	X.290-X.299
网间互通	
概述	X.300-X.349
卫星数据传输系统	X.350-X.369
以IP为基础的网络	X.370-X.379
报文处理系统	X.400-X.499
号码簿	**X.500-X.599**
OSI组网和系统概貌	
组网	X.600-X.629
效率	X.630-X.639
服务质量	X.640-X.649
命名、寻址和登记	X.650-X.679
抽象句法记法1(ASN.1)	X.680-X.699
OSI管理	
系统管理框架和结构	X.700-X.709
管理通信服务和协议	X.710-X.719
管理信息的结构	X.720-X.729
管理功能和ODMA功能	X.730-X.799
安全	X.800-X.849
OSI应用	
托付、并发和恢复	X.850-X.859
事务处理	X.860-X.879
远程操作	X.880-X.889
ASN.1的一般应用	X.890-X.899
开放分布式处理	X.900-X.999
电信安全	X.1000

欲了解更详细信息，请查阅*ITU-T*建议书目录。

国际标准 ISO/IEC 9594-4
ITU-T X.518建议书

信息技术 — 开放系统互连 — 号码簿:
分布式操作规程

摘　要

本建议书 | 国际标准规定了号码簿的分布式组件为了向其用户提供一致的服务而进行交互时所遵循的规程。

来　源

ITU-T 第 17 研究组（2005-2008 年）按照 ITU-T A.8 建议书规定的程序，于 2005 年 8 月 29 日批准了 ITU-T X.518 建议书。同一文本还以 ISO/IEC 9594-4 的形式发布。

前　言

国际电信联盟（ITU）是从事电信领域工作的联合国专门机构。ITU-T（国际电信联盟电信标准化部门）是国际电信联盟的常设机构，负责研究技术、操作和资费问题，并且为在世界范围内实现电信标准化，发表有关上述研究项目的建议书。

每四年一届的世界电信标准化全会（WTSA）确定 ITU-T 各研究组的研究课题，再由各研究组制定有关这些课题的建议书。

WTSA 第 1 号决议规定了批准建议书须遵循的程序。

属 ITU-T 研究范围的某些信息技术领域的必要标准，是与国际标准化组织（ISO）和国际电工技术委员会（IEC）合作制定的。

注

本建议书为简明扼要起见而使用的“主管部门”一词，既指电信主管部门，又指经认可的运营机构。

遵守本建议书的规定是以自愿为基础的，但建议书可能包含某些强制性条款（以确保例如互操作性或适用性等），只有满足所有强制性条款的规定，才能达到遵守建议书的目的。“应该”或“必须”等其它一些强制性用语及其否定形式被用于表达特定要求。使用此类用语不表示要求任何一方遵守本建议书。

知识产权

国际电联提请注意：本建议书的应用或实施可能涉及使用已申报的知识产权。国际电联对无论是其成员还是建议书制定程序之外的其它机构提出的有关已申报的知识产权的证据、有效性或适用性不表示意见。

至本建议书批准之日止，国际电联尚未收到实施本建议书可能需要的受专利保护的知识产权的通知。但需要提醒实施者注意的是，这可能并非最新信息，因此特大力提倡他们通过下列网址查询电信标准化局（TSB）的专利数据库：http://www.itu.int/ITU-T/ipr/。

目 录

引言

本建议书 | 国际标准连同本系列其他建议书|国际标准是为方便信息处理系统之间的互连以提供号码簿服务而制定的。所有这些系统的集合，连同它们所拥有的号码簿信息可被视为一个整体，被称为*号码簿*。号码簿所拥有的信息，总称为号码簿信息库(DIB)，典型地被用于方便对象之间的通信、与对象的通信或有关对象的通信等，这些对象如应用实体、个人、终端和分发表等。

号码簿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。

本建议书 | 国际标准规定了号码簿的分布式组件为了向其用户提供一致的服务而进行交互时所遵循的规程。

本建议书|国际标准提供了一个基础框架，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。在本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。目前本建议书|国际标准的第 5 版是原有第 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遵循第 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遵循第 5 版。

第 5 版详细定义了号码簿协议的第 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号码簿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 ITU-T X.519 建议书| ISO/IEC 9594-5 中定义的扩展规则调节。

附件 A，是本建议书 | 国际标准的一个组成部分，提供了号码簿分布式操作的 ASN.1 模块定义。

附件 B，不是本建议书 | 国际标准的一个组成部分，描述了分布式名字解析的一个示例。

附件 C，不是本建议书 | 国际标准的一个组成部分，描述了分布式操作环境中的鉴权。

附件 D，是本建议书 | 国际标准的一个组成部分，提供了本号码簿规范中引入的 ASN.1 信息对象类的定义。

附件 E，不是本建议书 | 国际标准的一个组成部分，举例说明了知识的维护。

附件 F，不是本建议书 | 国际标准的一个组成部分，列出了在本建议书 | 国际标准的本版本中合并的修正案和勘误。

国际标准
ITU-T 建议书

信息技术 — 开放系统互连 — 号码簿：分布式操作规程

第1部分 — 概述

1 范围

本建议书 | 国际标准规定了 DSA 在参与分布式号码簿应用中的行为。设计了允许的行为以便确保在一个跨多个 DSA 的广泛分布的 DIB 中，可以提供一致的服务。

号码簿并不是一个具有通用目的的数据库系统，尽管它可能是建立在这样的系统上的。假设号码簿中查询出现的频率比修改出现的频率要高很多。

2 规范性参考文献

下列建议书和国际标准所包含的条款，通过在本建议书中的引用而构成本建议书 | 国际标准的条款。在出版时，所指出的版本是有效的。所有的建议书和标准都面临修订，使用本建议书 | 国际标准的各方应探讨使用下列建议书和国际标准最新版本的可能性。IEC 和 ISO 的各成员有目前有效的国际标准的目录。国际电联电信标准化局有目前有效的 ITU-T 建议书的清单。

2.1 等同的建议书 | 国际标准

- ITU-T Recommendation X.200 (1994) | ISO/IEC 7498-1:1994, *Information technology – Open Systems Interconnection – Basic Reference Model: The basic model.*
- ITU-T Recommendation X.500 (2005) | ISO/IEC 9594-1:2005, *Information technology – Open Systems Interconnection – The Directory: Overview of concepts, models and services.*
- ITU-T Recommendation X.501 (2005) | ISO/IEC 9594-2:2005, *Information technology – Open Systems Interconnection – The Directory: Models.*
- ITU-T Recommendation X.509 (2005) | ISO/IEC 9594-8:2005, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks.*
- ITU-T Recommendation X.511 (2005) | ISO/IEC 9594-3:2005, *Information technology – Open Systems Interconnection – The Directory: Abstract service definition.*
- ITU-T Recommendation X.519 (2005) | ISO/IEC 9594-5:2005, *Information technology – Open Systems Interconnection – The Directory: Protocol specifications.*
- ITU-T Recommendation X.520 (2005) | ISO/IEC 9594-6:2005, *Information technology – Open Systems Interconnection – The Directory: Selected attribute types.*
- ITU-T Recommendation X.521 (2005) | ISO/IEC 9594-7:2005, *Information technology – Open Systems Interconnection – The Directory: Selected object classes.*
- ITU-T Recommendation X.525 (2005) | ISO/IEC 9594-9:2005, *Information technology – Open Systems Interconnection – The Directory: Replication.*
- ITU-T Recommendation X.530 (2005) | ISO/IEC 9594-10:2005, *Information technology – Open Systems Interconnection – The Directory: Use of systems management for administration of the Directory*.
- ITU-T Recommendation X.680 (2002) | ISO/IEC 8824-1:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Specification of basic notation.*

- ITU-T Recommendation X.681 (2002) | ISO/IEC 8824-2:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Information object specification.*
- ITU-T Recommendation X.682 (2002) | ISO/IEC 8824-3:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Constraint specification.*
- ITU-T Recommendation X.683 (2002) | ISO/IEC 8824-4:2002, *Information technology – Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications*.

2.2 其他参考文献

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

3 定义

本建议书 | 国际标准应用如下定义：

3.1 通信模型定义

下列术语在 ITU-T X.519 建议书 | ISO/IEC 9594-5 中规定：

a) *应用实体名称*。

3.2 基本号码簿定义

下列术语在 ITU-T X.500 建议书 | ISO/IEC 9594-1 中规定：

a) *号码簿*；
b) *号码簿信息库*。

3.3 号码簿模型定义

下列术语在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中规定：

a) *访问点*；
b) *别名*；
c) *识别名*；
d) *号码簿信息树*；
e) *号码簿系统代理（DSA）*；
f) *号码簿用户代理（DUA）*；
g) *相对识别名*。

3.4 DSA信息模型定义

下列术语在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中规定：

a) *种类*；
b) *公共可用的*；
c) *上下文前缀*；
d) *交叉引用*；
e) *DIB 片段*；
f) *DSA 信息树*；
g) *DSA 特定条目（DSE）*；
h) *DSE 类型*；
i) *直接上级引用*；
j) *知识信息*；
k) *知识引用种类*；
l) *知识引用类型*；

m) 命名上下文;
n) 非特定知识;
o) 非特定下级引用;
p) 操作属性;
q) 引用路径;
r) 特定知识;
s) 下级引用;
t) 上级引用。

3.5 抽象服务定义

下列术语在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中规定：

a) 流结果。

3.6 号码簿复制定义

下列术语在 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定：

a) 属性完备性;
b) 镜像操作绑定;
c) 下级完备性;
d) 复制单元。

3.7 分布式操作定义

下列术语在本建议书 | 国际标准中规定：

3.7.1 base object 基对象：指一个对象或别名条目，是发起者所发起的某个操作的目标。

3.7.2 bound DSA 绑定的 DSA：指一个 DSA，发起请求的 DUA 通过与该 DSA 执行一个绑定操作而与之绑定起来。

3.7.3 bound-DSA paged results 绑定 DSA 的分页结果：分页完全由 DUA 所绑定的 DSA 来执行。

注 — 这是遵循第 5 版之前的系统所支持的唯一一种分页模式。

3.7.4 chaining 链接：单链接或多链接的通用术语。

3.7.5 context prefix information 上下文前缀信息：上级 DSA 在一个 RHOB 中向下级 DSA 所提供的关于下级上下文前缀的上级 DIT 顶点的操作信息和用户信息。

3.7.6 distributed name resolution 分布式名字解析：名字解析在多个 DSA 中执行的过程。

3.7.7 DSP paged results DSP 分页结果：当执行的 DSA 与绑定的 DSA 不同时进行的 DSP 协议指配，据此来完成初始执行者的分页结果。

3.7.8 error 错误：由执行者向请求者发送的信息，信息中携带了一个对之前接收到的请求的否定结果。

3.7.9 hard error 硬错误：一个明确的错误，该错误指示如果没有外部的干预，则操作目前不能被执行。

3.7.10 hierarchical operational binding 分等级操作绑定（HOB）：指两个拥有命名上下文的主 DSA 之间的关系，其中一个是另一个的直接下级，在该关系中，上级 DSA 拥有一个指向下级 DSA 的下级引用。

3.7.11 initial performer 初始执行者：开始执行某个操作的第一个 DSA，即进入操作评估阶段的第一个 DSA。

3.7.12 modification operations 修改操作：指号码簿修改操作，即修改条目、增加条目、删除条目和修改 DN 等。

3.7.13 multi-chaining 多链接：一种交互模式，在该模式中，自己执行某个请求的 DSA 向其他 DSA 的一个集合发出多个请求，或者是并列的，或者是顺序的。

3.7.14 multiple entry interrogation operations 多条目查询操作：指号码簿搜索操作，即列表和搜索。

3.7.15 name resolution 名字解析：定位某个条目的过程，该过程是通过对声称名字的每个 RDN 与 DIT 的顶点进行顺序匹配而完成的。

3.7.16 non-specific hierarchical operational binding 非特定分等级操作绑定（NHOB）：指两个拥有命名上下文的主 DSA 之间的关系，其中一个是另一个的直接下级，在该关系中，上级 DSA 拥有一个指向下级 DSA 的非特定下级引用。

3.7.17 NSSR decomposition NSSR 分解：将非特定知识引用分解为多个子请求以便于其他 DSA 能够继续执行；这些子请求可能被执行该分解的 DSA 链接到其他 DSA；或者可能将标识了其他 DSA 的一个继续引用返回给请求者，由请求者来继续执行；或者执行分解的 DSA 可能继续执行某些子请求，而留下其他子请求交给请求者来继续执行。

3.7.18 operation progress 操作进展：一系列的值，指示了名字解析所完成的程度。

3.7.19 originator 发起者：发起一个特定的（分布式）操作的 DUA。

3.7.20 paging 分页：以一页或多页的分段方式返回的**搜索**或**列表**操作的结果，每页都由有限数量的条目组成。

3.7.21 performer 执行者：接收了某个请求的 DSA（即将执行某个操作）。

注 — 执行者也是初始执行者，除非对于可能包括多个 DSA 进行评估的操作。

3.7.22 procedure 规程：关于一个 DSA 如何将一个给定的输入变量集及其 DSA 信息树映射为一个结果的一个（非正式）规范。

注 — 输入变量和结果可以对应为从一个被请求的操作中接收到的信息和在一个答复中发送的信息，或者它们可能表示根据一个被请求的操作而对答复进行计算的中间阶段。在 14.2 节，前面一种输入变量和结果类型被称为是外部的。

3.7.23 relevant hierarchical operational binding (RHOB) 相关的分等级操作绑定（RHOB）：或者是指一个 HOB，或者是指一个 NHOB，依赖于上下文。

3.7.24 referral 提名：自己不能执行某个操作的 DSA 返回的一种结果，标识了一个或多个可以执行此操作的其他 DSA。

3.7.25 reply 答复：一个结果或一个错误。

3.7.26 request 请求：由一个操作代码和相关变量所组成的信息，表示从请求者向执行者所发起的一个号码簿操作。

3.7.27 request decomposition 请求分解：将一个请求分解为多个子请求以便于其他 DSA 能够继续执行；这些子请求可能被执行分解的 DSA 链接到其他 DSA；或者可能将标识了其他 DSA 的继续引用返回给请求者，由请求者来继续执行；或者执行分解的 DSA 可能继续执行某些子请求，而留下其他子请求交给请求者来继续执行。

3.7.28 requester 请求者：发送一个请求以便执行（即调用）某个操作的一个 DUA 或 DSA。

3.7.29 single entry interrogation operations 单条目查询操作：指号码簿阅读操作，即阅读和比较操作。

3.7.30 soft error 软错误：一个错误，可能是瞬时的，也可能指示了一个局部的问题，在这种情况下，使用一个不同的知识引用或访问点可能会获得一个结果或一个硬错误。

3.7.31 subordinate DSA 下级 DSA：共享一个 HOB 或一个 NHOB 的两个 DSA 中，其中拥有下级命名上下文的那个 DSA。

3.7.32 subrequest 子请求：通过请求分解而产生的一个请求。

3.7.33 superior DSA 上级 DSA：共享一个 HOB 或一个 NHOB 的两个 DSA 中，其中拥有上级命名上下文的那个 DSA。

3.7.34 superior, subordinate DSA 上级,下级 DSA：两个拥有命名上下文的主 DSA，其中一个是另一个的直接下级；这两个 DSA 之间的关系可以通过一个 HOB（或 NHOB）来显式地管理，或者依靠上级 DSA 来隐含存在，该上级 DSA 拥有一个指向下级 DSA 的下级引用（或非特定下级引用）。

3.7.35 target object name 目标对象名字：一个条目的名字，该条目或者是操作在名字解析的某个特定阶段所指向的对象，或者是包含在操作评估中。

3.7.36 uni-chaining 单链接：某个自身不能直接执行操作的 DSA 可选使用的一种交互模式。DSA 的链接是通过调用另一个 DSA 的某个操作，并且将结果再转发给初始请求者来完成的。

4 缩略语

就本建议书 | 国际标准而言，下列缩略语适用：

ASN.1 抽象语法标记 1

DISP 号码簿信息镜像协议

DMD 号码簿管理域

DOP 号码簿操作绑定管理协议

DSE DSA 特定条目

HOB 分等级操作绑定

NHOB 非特定的分等级操作绑定

NSSR 非特定下级引用

RHOB 相关的分等级操作绑定

5 约定

除少数例外，本号码簿规范是根据“*ITU-T | ISO/IEC 通用文本的表述准则，2001 年 11 月*”的要求制定的。

术语“号码簿规范（或本号码簿规范）”指的是本 ITU-T X.518 建议书 | ISO/IEC 9594-4。术语“系列号码簿规范”指的是 X.500 系列建议书和 ISO/IEC 9594 的所有部分。

本号码簿规范使用术语“*第 1 版系统*”来指遵循系列号码簿规范第 1 版的所有系统，即 1988 年版本的 CCITT X.500 系列建议书和 ISO/IEC 9594：1990 年版本。本号码簿规范使用术语“*第 2 版系统*”来指遵循系列号码簿规范第 2 版本的所有系统，即 1993 年版本的 ITU-T X.500 系列建议书和 ISO/IEC 9594：1995 年版本。本号码簿规范使用术语“*第 3 版系统*”来指遵循系列号码簿规范第 3 版的所有系统，即 1997 年版本的 ITU-T X.500 系列建议书和 ISO/IEC 9594：1998 年版本。本号码簿规范使用术语“*第 4 版系统*”来指遵循系列号码簿规范第 4 版的所有系统，即 2001 年版本的 ITU-T X.500、X.501、X.511、X.518、X.519、X.520、X.521、X.525、X.530 建议书和 2000 年版本的 ITU-T X.509 建议书以及 ISO/IEC 9594：2001 年版本的第 1 到第 10 部分。

本号码簿规范使用术语“*第 5 版系统*”来指遵循系列号码簿规范第 5 版的所有系统，即 2005 年版本的 ITU-T X.500、X.501、X.509、X.511、X.518、X.519、X.520、X.521、X.525 和 X.530 建议书，以及 ISO/IEC 9594：2005 年版本的第 1 到第 10 部分。

本号码簿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。

如果列表中的项以数字标识（而不是以“–”或字母标识），则表示这些项应当被认为是过程中的各个步骤。

第2部分 — 概述

6 概述

号码簿抽象服务允许对 DIB 中的号码簿信息进行查询、获取和修改。该服务按照 ITU-T X.511 建议书 | ISO/IEC 9594-3 中规定的抽象号码簿对象来进行描述。类似的，轻量级号码簿访问协议（LDAP）允许对 DIB 中的号码簿信息进行查询、获取和修改。该协议以及它所允许的服务在 RFC 3377 中规定。

必要地，抽象号码簿对象的规范没有以任何方式指定号码簿的物理实现：尤其是它没有指定号码簿系统代理（DSA）的规范，这些 DSA 存储了 DIB 并对 DIB 进行管理，并且通过 DSA 提供服务。另外，它没有考虑 DIB 是否是分布式的，即 DIB 是包含在一个单独的 DSA 内，还是分布在多个 DSA 内。因此，为了在一个分布式环境中支持抽象服务，需要 DSA 具有其他 DSA 的知识，能够导航到其他 DSA，并且与其他 DSA 进行合作等，这些需求也没有涵盖在服务描述中。

本号码簿规范对抽象号码簿对象进行了细化，这种细化通过一个或多个 DSA 对象集来表示，这些 DSA 对象共同构成了分布的号码簿服务。

另外，本号码簿规范规定了 DIB 可能被分布到一个或多个 DSA 内的允许的方式。对于某种受限情况，即 DIB 包含在一个单独的 DSA 内，这种情况下号码簿实际上是集中式的；对于 DIB 分布在两个或多个 DSA 内的情况，则规定了知识和导航机制以确保所有的拥有组成条目的 DSA 能够潜在地访问到整个 DIB。

DIB 的一部分内容也可能在多个 DSA 内被复制。本号码簿规范中描述的协议允许使用复制信息来提高分布式号码簿服务的可用性、性能和效率。复制信息的使用在某种程度上是在用户的控制之下的，通过使用服务控制选项来实现。本号码簿规范中描述的规程也指示了在使用复制信息时进行设计最优化的某些时机。

另外，也规定了请求处理的交互，使得特定的号码簿操作特性能够被它的用户所控制。尤其是当 DSA 要响应一个与其他 DSA 中拥有的信息相关的号码簿请求时，用户能够控制一个 DSA 是否拥有直接查询其他 DSA 的权力（即链接），或者它是否应当在响应中提供其他能够继续处理请求的 DSA 的信息（即提名）。

一般来说，一个 DSA 是进行链接还是提名的决定是根据用户所设置的服务控制，以及 DSA 的自身管理、操作或技术环境来决定的。

应当意识到的是，一般来说，号码簿将是分布式的，因此号码簿请求将由任意数量的合作 DSA 来满足，这些 DSA 根据上述的条件可能是任意地进行链接或提名，本号码簿规范规定了 DSA 在响应分布式号码簿请求时所采用的适当的规程。这些规程将确保分布式号码簿服务的用户能够感觉到规程既是用户友好的，又是协调一致的。

第3部分 — 分布式号码簿模型

7 分布式号码簿系统模型

号码簿抽象服务，如在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中定义的那样，将号码簿建模为一个对象，该对象向其用户提供了一系列的号码簿服务。号码簿用户通过一个访问点来访问它所提供的服务。号码簿可能拥有一个或多个访问点，且每个访问点都通过它所提供的服务以及提供这些服务的交互模式来描述其特性。

图 1 举例说明了分布式号码簿模型，该模型将作为规定号码簿分布特性的基础。它举例说明了由一个或多个 DSA 的集合所组成的号码簿。

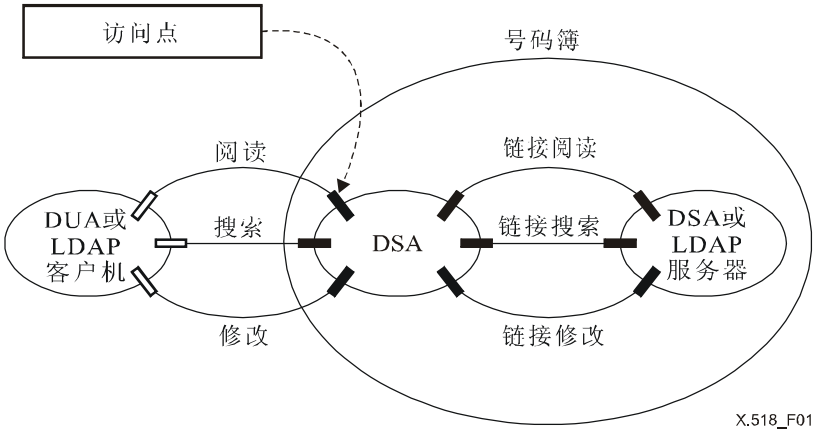


图 1－分布式号码簿模型中的对象

在本号码簿规范的后续章节中将详细规定 DSA。本节仅仅陈述一些它们的特性作为介绍性引言，同时建立本号码簿规范和其他号码簿规范之间的关系。

DSA 的定义是为了能够适应 DIB 的分布，且一系列的在物理上分布的 DSA 能够以一种预定义的、合作的方式进行交互来向号码簿用户（DUA 或 LDAP 客户机）提供号码簿服务。

图 1 举例说明了号码簿抽象服务和 DSA 抽象服务之间的关系。号码簿抽象服务在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中定义，是通过一系列的号码簿操作来提供的。为了实现该服务，组成号码簿的各 DSA 之间需要相互交互。这种交互的自然特性是根据一个 DSA 可能向另一个 DSA 所提供的服务来定义的，即 DSA 抽象服务。DSA 抽象服务是通过一系列的操作来提供的，这些操作被称为链接操作，每个操作都在号码簿抽象服务中拥有一个对应操作。因此，在号码簿抽象服务中给定的一个操作，如阅读，可能会要求提供服务的 DSA 通过使用链接操作来与其他的一个或多个 DSA 进行交互，如链接阅读。

注 — 对于作为 LDAP 请求者的 DSA 来说，进行链接操作也是可能的，例如，使用 LDAP 控制或扩展操作；然而，完成这些的规程和协议不在本号码簿规范的定义范围之内。

8 DSA交互模型

号码簿的一个基本特性是，给定一个分布式 DIB，一个用户应当潜在地能够被满足任何服务请求（在符合安全性、访问控制和管理策略等的前提下），而不论请求所发起的访问点。为了适应此需求，必要的是任何一个与满足某个特定服务请求相关的 DSA 都必须拥有被请求的信息位于何处的知识（在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中指定），并且，或者将这些知识返回给请求者，或者试图自己来满足此请求。（请求者可能是一个 DUA、一个 LDAP 客户机或其他的 DSA：在后一种情况下，两个 DSA 都必须支持 DSP。）

定义了三种 DSA 交互模式以便符合这些需求，这三种模式被称为“单链接”，“多链接”和“提名”。在本号码簿规范的后续部分，使用通用术语“链接”在适当的上下文中来表示单链接和/或多链接。“链接”指的是 DSA

为了满足某个请求，而向另一个 DSA 发送一个或多个链接操作；“提名”指的是向请求者返回知识信息，于是请求者可能自身再与知识信息中所标识的 DSA 进行交互。

单链接或一个提名交互可能源于一个单独的请求。可选的，一个请求可能在交互之前被分解为多个子请求。多链接或多提名交互，或两者的混合，可能源自一个分解后的请求。定义了两种分解类型：NSSR 分解和请求分解。

8.1 一个请求的分解

8.1.1 NSSR分解

NSSR 分解是将同样的请求准备成便于传送到（或者是顺序的，或者是并行的）多个下级 DSA 中去的过程，这是在名字解析的过程中遇到一个 NSSR 的结果。非特定下级引用中不包含被引用的下级命名上下文的 RDN，因此，引用的 DSA 不能够区分哪个下级 DSA 拥有哪个下级命名上下文。因此，在名字解析过程中，一个遇到 NSSR 的 DSA 必须向每个下级 DSA（在没有镜像的情况下）发送一个同样的请求。这可以是顺序执行的，也可以是并行执行的。典型地，仅有一个 DSA 能够继续执行名字解析；而其余 DSA 将返回一个问题为 **unableToProceed** 的 **serviceError**。在某种（很少）情况下，有可能有多个 DSA 将继续进行名字解析，由此导致了双重结果。

注 — NSSR 不能引用 LDAP 服务器。

8.1.2 请求分解

请求分解，另一种分解请求的方式，是在 DSA 与其他一个或多个 DSA 和/或 LDAP 服务器通信之前，由 DSA 内部执行的一个过程。一个请求被分解为多个可能不同的子请求，因此每个子请求完成原始任务的一部分。请求分解能够仅被用于列表或搜索操作的操作评估过程中。在请求分解完成后，每个子请求可能被链接到其他 DSA 和/或 LDAP 服务器以便继续任务的执行，或者可能会有一个局部结果（一个内嵌的提名）返回给请求者。同一个子请求被产生到不同的 DSA 和/或 LDAP 服务器中的一个示例为：某个条目具有下级引用和/或 NSSR，且共同引用了多个 DSA 或 LDAP 服务器。不同的子请求被产生到相同的或不同的 DSA 和/或 LDAP 服务器中的一个示例为：两个不同的条目在某个搜索（子树）的操作中相遇，且每个都拥有一个下级引用。

8.2 单链接

当一个 DSA 具有另一个 DSA 所拥有的命名上下文的知识时，第一个 DSA 可能会使用这种交互模式（图 2 中显示）将一个请求传递给第二个 DSA。单链接可能会用于与某个单独的 DSA 联系，该 DSA 是在一个交叉引用、一个下级引用、一个上级引用、一个提供者引用或一个主引用中被指向的 DSA。

注 — 在图 2 中，交互的顺序由交互线上相关联的数字来定义。

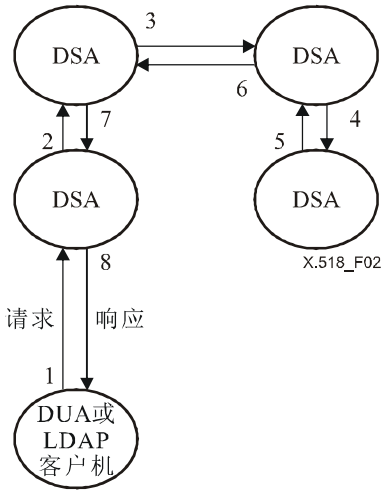


图 2－单链接模式

8.3 多链接

一个 DSA 使用这种交互模式来传递多个出请求，这些出请求来自于同一个入请求，或者是请求分解的结果，或者是 NSSR 分解的结果。

8.3.1 并行多链接

在并行多链接情况下，DSA 将多个出请求同步进行传递（见图 3a）。并行多链接可能会带来性能的提高，但同时它也可能会在某种环境下，如在镜像存在时，导致接收到重复结果。

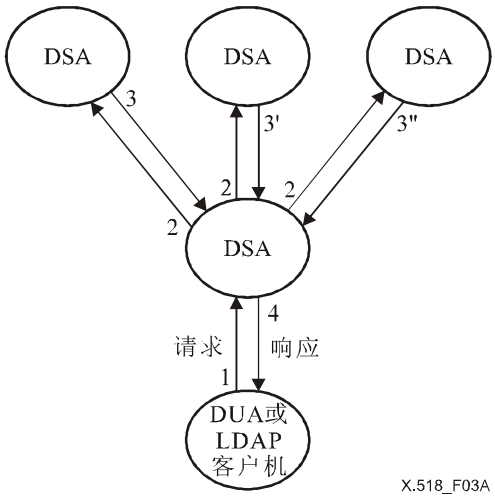
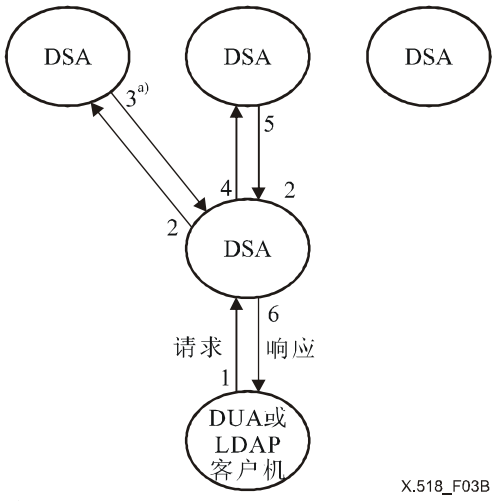


图 3a－并行多链接

8.3.2 顺序多链接

在顺序多链接情况下，DSA 在一个时间点传递一个出请求，并且在发送下一个请求前等待此请求的结果或错误（见图 3b）。顺序多链接可能不是一种最快的交互模式，但同时它也不可能导致接收到重复结果。

注 — 一个 DSA 可能联合使用并行多链接和顺序多链接。



a) 不能继续

图3b－顺序多链接
（NSSR分解的结果）

8.4 提名

一个 DSA 可以在响应从某个 DUA、LDAP 客户机或其他 DSA 发来的请求时，返回一个提名（在图 4a 和 4b 中描述）。提名可能会构成整个响应（在这种情况下，它被分类为一种错误）或者仅仅是响应的一部分。提名包含了一个知识引用，该知识引用可能是一个上级引用、下级引用、交叉引用、非特定下级引用，提供者引用或主引用等。

接收到提名的 DSA（见图 4a）可能会使用包含在提名中的知识引用来进行随后的链接或多播（依赖于引用的类型），将一个原始请求链接或多播到其他的 DSA。可选的，接收到提名的 DSA，可能会在其响应中依次传递提名。接收到提名的一个 DUA 或 LDAP 客户机（见图 4b）可能会使用该提名来与一个或多个其他 DSA 联系以便继续执行请求。

注 — 在图 4a 和 4b 中，交互的顺序由交互线上相关联的数字来定义。

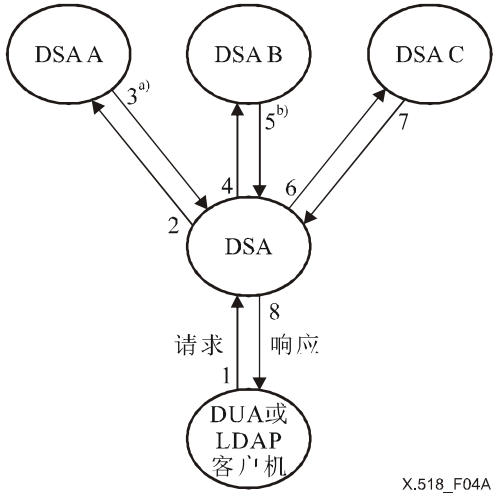


图 4a－提名模式（DSA根据提名执行动作）

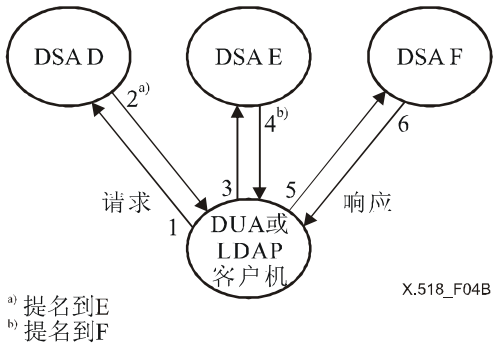


图 4b－提名模式（DUA根据提名执行动作）

8.5 模式的决定

如果一个 DSA 自身不能完全地解决一个请求，则它必须将该请求（或通过分解原始请求而形成的一个请求）链接到另一个 DSA，除非：

a) 用户通过服务控制禁止链接，在这种情况下，DSA 必须返回一个提名或一个问题为 **chainingRequired** 的 **serviceError**；或者

b) DSA 由于管理、操作或技术等方面的原因倾向于不进行链接，在这种情况下，DSA 必须返回一个提名。

注 1 — 不进行链接的一个“技术原因”是在知识引用中标识的 DSA 不支持 DSP。

注 2 — 如果服务控制 **localScope** 被设置，则 DSA（或 DMD）必须或者解决该请求，或者返回一个错误。

注 3 — 如果用户更希望提名，则用户必须设置 **chainingProhibited**。

第4部分 — DSA抽象服务

9 DSA抽象服务概述

号码簿服务在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中有全面描述。当这样的一个服务在分布式环境中提供时，如第 7 节中的建模，它可以被认为是通过一系列的 DSA 来提供的。如图 1 中所示。

对于号码簿服务中定义的每个操作，在 DSA 抽象服务中都定义了一个相应的“链接”操作，在完成号码簿服务操作中合作的 DSA 之间可以使用这些操作。这样，一个从 DUA 处接收到阅读操作的 DSA 可能会请求另一个 DSA 的协助（例如，另一个 DSA 中拥有目标条目或其拷贝）来满足此操作，因此会向此 DSA 发送一个链接阅读操作。

在 DSA 抽象服务中交互的信息类型在第 10 节定义。DSA 抽象服务的操作和错误在第 11 节到第 13 节定义。

10 信息类型

10.1 引言

本节标识了，同时在某些情况下定义了，在后续的 DSA 抽象服务的各种操作定义中所使用的一系列信息类型。这些涉及到的信息类型包括：对于多个操作而言是通用的信息类型，或者将来可能是通用的信息类型，或者是足够复杂或自包含的信息，需要与使用它们的操作分别定义。

在 DSA 抽象服务的定义中使用的多个信息类型实际上是在别处定义的。第 10.2 小节标识了这些类型，并且指示了它们的定义来源。第 10.3 小节到 10.10 小节分别标识并定义了一个信息类型。

10.2 其他地方定义的信息类型

下列信息类型在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中定义：

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

下列信息类型在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中定义：

(绑定)

- **DirectoryBind**

(操作)

- **Abandon**

(错误)

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

(信息对象类)

- **OPTIONALLY-PROTECTED**

(数据类型)

- **SecurityParameters**

下列信息类型在 ITU-T X.520 建议书 | ISO/IEC 9594-6 中定义：

— **PresentationAddress.**

10.3 链接变量

链接变量 **ChainingArguments** 出现在每个链接操作中，向某个 DSA 传递信息，这些信息是在该 DSA 成功执行全部任务中它需要完成的部分时所需的信息：

```
ChainingArguments ::= SET {
        originator                  [0]     DistinguishedName OPTIONAL,
        targetObject                [1]     DistinguishedName OPTIONAL,
        operationProgress           [2]     OperationProgress
                                            DEFAULT { nameResolutionPhase notStarted },
        traceInformation            [3]     TraceInformation,
        aliasDereferenced           [4]     BOOLEAN DEFAULT FALSE,
        aliasedRDNs                 [5]     INTEGER OPTIONAL,
                                    -- 仅出现在第一版本系统中
        returnCrossRefs             [6]     BOOLEAN DEFAULT FALSE,
        referenceType               [7]     ReferenceType DEFAULT superior,
        info                        [8]     DomainInfo OPTIONAL,
        timeLimit                   [9]     Time OPTIONAL,
        securityParameters          [10]    SecurityParameters DEFAULT { },
        entryOnly                   [11]    BOOLEAN DEFAULT FALSE,
        uniqueIdentifier            [12]    UniqueIdentifier OPTIONAL,
        authenticationLevel         [13]    AuthenticationLevel OPTIONAL,
        exclusions                  [14]    Exclusions OPTIONAL,
        excludeShadows              [15]    BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster         [16]    BOOLEAN DEFAULT FALSE,
        operationIdentifier         [17]    INTEGER OPTIONAL,
        searchRuleId                [18]    SearchRuleId  OPTIONAL,
        chainedRelaxation           [19]    MRMapping  OPTIONAL,
        relatedEntry                [20]    INTEGER  OPTIONAL,
        dspPaging                   [21]    BOOLEAN DEFAULT FALSE,
        nonDapPdu                   [22]    ENUMERATED { ldap (0) } OPTIONAL,
        streamedResults             [23]    INTEGER OPTIONAL
        excludeWriteableCopies      [24]    BOOLEAN DEFAULT FALSE }

Time ::= CHOICE {
        utcTime             UTCTime,
        generalizedTime     GeneralizedTime }
```

不同组件的含义定义如下：

a) 组件 **originator** 携带了请求的最初发起者的名字，除非已经在安全参数中指定。如果在 **CommonArguments** 中出现了 **requester**，则该变量可能被忽略。

注 1 — 当发起者具有通过上下文所区分的可替代名字时，则用于 **originator** 值的名字必须是主识别名，如果已知的话。否则，基于 **originator** 值的鉴权和访问控制可能不能按照期望的方式工作。

b) 组件 **targetObject** 携带了其号码簿条目是要被指向的对象的名字。该对象的角色依赖于相关的特定操作：它可能是其条目要被操作的对象，或者是某个包含多个对象的请求或子请求的基对象（例如 **chainedList** 或 **chainedSearch**）。当且仅当本组件的取值与链接操作中的对象或基对象参数的取值相同时，本组件才能够被忽略，在这种情况下，它的隐含值即为该参数值。

当 **targetObject** 所包括的 RDN 包含了属性类型和值对，且有多个通过上下文所区分的识别值时，被解析的 RDN 必须是主 RDN。

c) 组件 **operationProgress** 用于向 DSA 通报操作的执行进展，因此也通报了该 DSA 在整个执行过程中所期望承担的角色。本组件中携带的信息在 10.5 小节指定。

d) 组件 **traceInformation** 用于当操作中有链接时，防止在 DSA 间出现环回。一个 DSA 在向另一个 DSA 链接一个操作之前，将在踪迹信息中增加一个新的元素。当一个 DSA 被请求执行某个操作时，该 DSA 将通过检查踪迹信息来证实该操作没有形成一个环回。本组件中携带的信息在 10.6 小节指定。

e) 组件 **aliasDereferenced** 是一个布尔值，用于指示在分布式名字解析过程中，到目前为止所遇到的一个或多个别名条目是否被解除引用。缺省值 **FALSE** 指示没有别名条目被解除引用。

f) 组件 **aliasedRDNs** 指示了已经有多少个 **targetObject** 的 **Name** 中的 RDN 从一个（或多个）别名条目的 **aliasedEntryName** 属性中产生出来。当遇到一个别名条目并被解除引用后，该整数值被设置。当且仅当组件 **aliasDereferenced** 为 **TRUE** 时，本组件必须存在。

注 2 — 本组件是为了与号码簿第一版本的实现相兼容而提供的。根据号码簿规范的后续版本实现的 DUA（和 DSA）应当总是忽略后续请求中 **CommonArguments** 中的此参数。在这种方式下，如果别名解除引用到其他的别名时，号码簿不会发出错误信号。

g) 组件 **returnCrossRefs** 是一个布尔值，指示了在执行一个分布式操作的过程中所使用的知识引用是否被要求作为交叉引用，伴随一个结果或提名而传递回初始的 DSA。缺省值为 **FALSE** 指示了这样的知识引用将不被返回。

h) 组件 **referenceType** 向被要求执行操作的 DSA 指示，使用了什么类型的知识将请求路由到该 DSA。因此，该 DSA 能够在调用者所拥有的知识中检测到错误。如果有这样的错误被检测出，则它必须在 **serviceError** 中指示，且携带的问题为 **invalidReference**。**ReferenceType** 在 10.7 节中完整描述。

注 3 — 如果 **referenceType** 缺失，则假设值为 **superior**。

i) 组件 **info** 用于在多个 DSA 之间传递 DMD 特定的信息，这些 DSA 包含在某个公共请求的处理中。本组件的类型为 **DomainInfo**，是一个无限制的类型：

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

j) 组件 **timeLimit**，如果存在，则指示了操作应当完成的时间限制（见 16.1.4.1）。在 **Time** 的值用于任何一个比较操作之前，且 **Time** 的语法选择为 **UTCTime** 类型时，两位数字的年字段值必须被合理化为四位数字的年字段值，如下所述：

— 如果两位数字的值是 00 到 49（含），则最后的值应当加 2000。

— 如果两位数字的值是 50 到 99（含），则最后的值应当加 1900。

注 4 — 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。谁规定了本号码簿规范将应用的域，例如轮廓分组，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 年以后的日期。

k) 组件 **SecurityParameters** 在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中指定。它的缺失被认为与安全参数为空集是等价的。

l) 如果初始的操作是一个搜索操作，其中变量 **subset** 被设置为 **oneLevel**，且遇到一个别名条目是 **baseObject** 的直接下级，在这种情况下，组件 **entryOnly** 被设置为 **TRUE**。成功地对 **targetObject** 的名字执行了名字解析的 DSA 必须对这个唯一的命名条目执行对象评估。

m) 组件 **uniqueIdentifier** 是可选提供的，当它被要求证实发起者的名字时才提供。其数据类型 **UniqueIdentifier** 在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中描述。

n) 组件 **authenticationLevel** 是可选提供的，当它被要求指示鉴权执行的方式时才提供。其数据类型 **AuthenticationLevel** 在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中描述。

o) 组件 **exclusions** 仅对于搜索操作有意义；它如果存在的话，指示了 **targetObject** 的哪个下级条目子树必须从搜索操作的结果中排除出去（见 10.9）。

p) 组件 **excludeShadows** 仅对于搜索和列表操作有意义；它指示了搜索必须被应用于条目，而不能应用于条目拷贝。本可选组件可能被某个 DSA 使用，用以避免接收到复制的结果（见 20.1）。

q) 组件 **nameResolveOnMaster** 仅在名字解析时有意义，且仅在遇到 NSSR 时，才被设置。如果被设置为 **TRUE**，则它指示后续的名字解析，即匹配从 **nextRDNToBeResolved** 开始的剩余 RDN，不能够部署条目拷贝信息，其中包括多主实现中的可写拷贝；后续对每个剩余 RDN 的解析都必须在该 RDN 所标识的条目的主 DSA 中执行（见 20.1）。

r) 组件 **operationIdentifier** 简化了 DAP 操作与后续相关的 DSP 操作以及结果之间的相关性。它由第一个接收到 DAP 请求的 DSA 所分配，或者从要求链接的 DSP 请求的链接变量中复制。分配 **operationIdentifier** 的 DSA 在一段足够长的时间段内，不能重新使用已分配的整数值。相关的 DAP 和 DSP 请求及结果的相关性通过这样的方式被简化，即对于每个操作和结果，DSA 都将 **operationIdentifier** 以及分配它的 DSA（在链接请求的 **traceInformation** 中的第一个 DSA）的名字记入日志。这样的相关性可能在用于日志、审计、计费和结算等目的时是有用的。

s) 组件 **searchRuleId** 携带了某个搜索规则的唯一标识符。在下述情况下，它将包含在执行初始**搜索规程(I)**的 DSA 中，即如果该规程在一个服务特定的管理区内发起，且当在 DIT 中往下执行时，或者跟随着别名时，或者跟随着分等级分组指针时，搜索操作被传递到其他 DSA。

t) 组件 **chainedRelaxation** 可以在一个分布方式下为链接搜索操作执行放宽。如果某个 DSA 接收到一个链接搜索操作，且支持放宽策略，则该 DSA 能够使用所提供的 **chainedRelaxation** 组件来替代任何它可能实现的其他放宽策略，因此使得放宽可以在那些潜在地返回搜索结果的 DSA 间进行协调。

u) 当接收 DSA 被要求解析相关的条目时，元素 **relatedEntry** 必须存在。若存在时，接收 DSA 必须仅对 **SearchArgument** 中 **joinAttributes** 的值 **relatedEntry** 所指定的特定相关条目元素进行响应。这样，值为零的一个 **relatedEntry** 必须选择 **SearchArgument** 的 **joinAttributes** 序列中的第一个元素。该值永远不能超过一个低于 **SearchArgument** 中 **joinAttributes** 组件内元素个数的一个值。在某个指定相关条目的 DSP 操作中，如果 **ChainingArguments** 内的 **relatedEntry** 元素缺失，则表示正在被链接的分布式操作是基对象搜索，而不是搜索的相关条目部分。

注 5 — 如果一个将要被链接指向的 DSA 被要求既处理通常的搜索结果又处理相关条目结果，则通过向此 DSA 发送两个不同的 DSP 操作来实现。

当 **relatedEntry** 元素存在，则必须应用下列特殊的规则：

— 在评估 **SearchArgument** 的组件 **selection** 的子组件 **infoTypes** 时，**infoTypes** 必须被认为具有值 **attributeTypesAndValues**，而不论初始指定的值是什么；

— 在 **JoinAttPair** 的任意 **joinAtt** 组件中指定的所有属性必须被包含在选择中，不论之前是否被包含在内；

— 整理相关条目结果的 DSA 必须忽略值和未指定的变量，这样使得结果可以符合初始的用户请求。

在某个支持相关条目的 DSA 的后续的出 **ChainingArguments** 中，必须将变量 relatedEntry 传递出去。

v) 如果绑定 DSA 不同于初始执行者（见 15.5.5），且绑定 DSA 支持 DSP 分页结果，则它可能会设置 **dspPaging** 组件值为 **TRUE** 来指示初始执行者提供 DSP 分页结果。如果该组件取值为 **FALSE**（缺省值），则初始执行者不应执行 DSP 分页结果。一个支持 DSP 分页结果的初始执行者不应将该组件前向到它正在发送子请求的那些 DSA。

w) 组件 **nonDapPdu** 用于指示 PDU 在链接变量中是否被封装，该链接变量源自一个非 DAP 请求，如一个 LDAP 请求。

x) 组件 **streamedResults** 用作一个计数器，来决定在响应该操作时，是否要链接流结果。当且仅当该计数器存在，且 DSA 理解流结果，并且 DSA 希望为其链接操作接收流结果时，每个包含在名字解析中的 DSA 才将该计数器加 1。因此，该计数器被最终完成名字解析的 DSA 使用来判断之前的每个 DSA 是否都准备要处理流结果。

y) 组件 **excludeWriteableCopies** 仅针对搜索和列表操作有意义；它指示搜索必须应用于条目的主拷贝，而不能应用于这些条目的可写拷贝。本可选组件可能被某个 DSA 使用作为一种方式来避免接收到复制结果（见 20.1）。

10.4 链接结果

链接结果 **ChainingResults** 出现在每个操作的结果中，并向调用该操作的 DSA 提供反馈信息。

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

不同组件的含义定义如下：

a) 组件 **info** 用于在处理一个公共请求过程中所包含的多个 DSA 之间传递 DMD 特定的信息。该组件的类型为 **DomainInfo**，是一个无限制类型。

b) 组件 **crossReferences** 一般不出现在 **ChainingResults** 中，除非相应请求中的 **returnCrossRefs** 组件取值为 **TRUE**。该组件由一个按序排列的 **CrossReference** 项所组成，每个项中都包含一个 **contextPrefix** 和一个 **accessPoint** 描述符（见 10.8）。

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

当某个操作的 **targetObject** 变量中的一部分与某个 DSA 的上下文前缀之一相匹配时，则该 DSA 可能会增加一个 **CrossReference**。DSA 的主管当局可能有策略规定不能返回这样的知识，在这种情况下，则不会在序列中增加相应项。

c) 数据类型 **SecurityParameters** 在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中指定。组件 **securityParameters** 的缺失被认为与安全参数为空集时是等价的。

d) 组件 **alreadySearched**，如果存在，指示了作为 **targetObject** 下级的哪个下级 RDN 已经被作为链接搜索操作的一部分被处理过了，因此在后续的子请求中将被排除出去。

注 — 处于 **contextPrefix** 或 **alreadySearched** 中的名字必须是主识别名，且不应包含可替代识别名。

10.5 操作进展

OperationProgress 的值描述了执行一个操作的进展状态，该操作中有多个 DSA 参与。

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

不同组件的含义定义如下：

a) 组件 **nameResolutionPhase** 指示了在处理某个操作的 **targetObject** 名字时，已经到达了哪个阶段。当它指示名字解析尚未开始时（值为 **notStarted**），则表示迄今为止尚未接触到这样的 DSA，即在命名上下文中包含该名字的初始 RDN 的 DSA。如果名字解析是正在进行中（值为 **proceeding**），则表示名字的初始部分已经被识别，尽管尚未到达拥有目标对象的 DSA。组件 **nextRDNToBeResolved** 指示已经识别了名字的多少部分（见 10.5 节的 b 项）。如果名字解析已经完成（值为 **completed**），则表示已经到达了拥有目标对象的 DSA，且适当的操作正在执行中。

b) 组件 **nextRDNToBeResolved** 向 DSA 指示了在 **targetObject** 名字中的哪个 RDN 是下一个将被解析的对象。它的格式是一个整数，取值范围从 1 到名字中 RDN 的个数。该组件仅在组件 **nameResolutionPhase** 取值为 **proceeding** 时，才存在。

10.6 踪迹信息

踪迹信息 **TraceInformation** 的值中携带了包含在某个操作执行过程中的 DSA 的记录。它可用于判断由于知识不一致或 DIT 中出现别名环回而可能导致的环回的存在，或者避免环回的出现。

```
TraceInformation ::= SEQUENCE OF TraceItem

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

每个将操作散播到其他 DSA 的 DSA 将在 **TraceItem** 序列的尾部增加一个新的项。每个这样的 **TraceItem** 包含：

a) 增加该项的 DSA 的名字；

b) **targetObject** 的名字是增加该项的 DSA 在入请求中接收到的名字。如果被链接的请求来自一个 DUA（在这种情况下，它的隐含值为 **XOperation** 中的 **object** 或 **baseObject**），或者如果它的值与出请求中的 **ChainingArgument** 内的 **targetObject** 值相同（实际相同或隐含相同），则该参数被忽略；

c) **operationProgress** 的值是增加该项的 DSA 在入请求中接收到的值。

dsa 必须是主识别名，且不应包含可替代识别名。被处理的 **targetObject** 中的每个 RDN 都必须是一个主 RDN。在 RDN 的 **AttributeTypeAndDistinguishedValue** 内的组件 **valuesWithContext** 中可以包含带有上下文的可替代识别名。

10.7 引用类型

引用类型 **ReferenceType** 的值指示了在 ITU-T X.501 建议书 | ISO/IEC 9594-2 中定义的各种类型的引用之一。

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

10.8 访问点信息

有三种类型的访问点：

a) 一个 **AccessPoint** 值标识了一个特定的点，在该点可以对号码簿，更明确来说是对一个 DSA 或一个 LDAP 服务器，进行访问。当标识的是一个 DSA 时，则访问点必须具有相关 DSA 的名字 **Name**，并且可能有一个表示层地址 **PresentationAddress**，用于该 DSA 的 OSI 或 IDM 通信，在这种情况下，不能出现 labeledURI。当标识的是一个 LDAP 服务器时，则访问点必须有一个 **LabeledURI**，用于与该 LDAP 服务器的 LDAP 通信中。若 **LabeledURI** 项存在，则 **Name** 和 **PresentationAddress** 都必须被忽略，且 **SET OF protocolInformation** 不应当出现。

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

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

b) 一个 **MasterOrShadowAccessPoint** 值标识了号码簿的一个访问点。访问点的 **category**，取值或者为 **master**，或者为 **shadow**，依赖于它是指向一个命名上下文，还是指向一个公共可用的复制区。组件 **chainingRequired** 指示对于此 DSA，链接是否是需要的，即不应当向该 DSA 返回一个提名。

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

c) 一个 **MasterAndShadowAccessPoints** 值标识了号码簿的一系列访问点，即一系列相关的 DSA 和/或 LDAP 服务器。这些访问点共享特性，每个访问点都标识了一个 DSA 或 LDAP 服务器，这些 DSA 或 LDAP 服务器都拥有来自一个公共命名上下文的条目信息（或者如果值是属性 **nonSpecificKnowledge** 的一个值时，则是来自一个 DSA 作为属主所拥有的命名上下文的一个公共集合中）。**MasterAndShadowAccessPoints** 的一个值指示它所包含的每个 **AccessPoint** 值的 **category**。命名上下文的主 DSA 或 LDAP 服务器的访问点不必包含在此集合中。

注 — 实现者应当认识到对于一个 LDAP 服务器来说，这种情况是可能的，即使它被标识为 **shadow**，它也可以根据接收到的 LDAP 更新操作来更新条目。

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

一个**AccessPointInformation**值标识了一个或多个号码簿的访问点。

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

若产生一个 **AccessPointInformation** 值的 DSA 是按照第一版本实现的 DSA，则该集合中的可选组件不存在。若解释 **AccessPointInformation** 值的 DSA 是按照第一版本实现的 DSA，则出现的任何 **MasterAndShadowAccessPoints** 值都被忽略。

若 DSA 是按照第二版本和后续版本实现的，则为一个 **AccessPointInformation** 值而产生的 **MasterOrShadowAccessPoint** 值组件的种类可能是属主或镜像，由产生该值的 DSA 的知识选择规程来决定。它可能被看作是一个建议的访问点，由产生该值的 DSA 向接收该值的 DSA 提供。可选的，还可能为一个 **AccessPointInformation** 值产生一个 **MasterAndShadowAccessPoints** 值。它由附加的信息组成，这些信息可能由接收 DSA 的知识选择规程来部署，用于决定一个替代的访问点。

10.9 DIT桥接知识

一个 **ditBridgeKnowledge** 值标识了一个特定的点，在该点可以对另一个 DIT，更明确来说是对一个 DSA 或一个 LDAP 服务器，进行访问。**ditBridgeKnowledge** 指定了一个访问点 **accessPoint**，在这点上可能访问到 DSA 或 LDAP 服务器。

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

domainLocalID 包含一个可读的描述符，标识了包含在引用中的 DIT。

10.10 排除

正如在 10.3 节中定义的那样，**ChainingArguments** 中的 **exclusions** 组件被用于限制某个搜索操作的范围，这是通过标识一系列的作为目标对象下级的条目以及它们的所有下级来实现的，所标识的这些条目都不能包含在该搜索操作的处理过程中。组件 **exclusion** 被定义为 ASN.1 数据类型 **Exclusions** 的一个值。

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

Exclusions 集中的每个 **RDNSequence** 值都应当标识一个作为目标对象下级的命名上下文的上下文前缀。如果一个 DSA 接收到一个 **search** 请求，且携带的某个 **RDNSequence** 值不符合本限制，则该 DSA 可能会忽略此值。**RDNSequence** 是相对于目标对象的，因此不是上下文前缀的识别名。

Exclusions 必须是主识别名。可替代识别名和上下文信息也可能包含在内。

除了可以作为某个用户请求中一部分，**Exclusions** 还能够被 DSA 使用，使得在镜像信息存在的情况下，从搜索子请求中返回的复制信息最少。

图 5 举例说明了 **Exclusions** 的一种用法示例。在这个例子中，一个 DSA 拥有两个复制区，其中一个在另一个的下面。一个起始于上下文前缀 X，另一个起始于上下文前缀 C。一个处于 Y 的条目拷贝拥有三个下级引用，分别指向命名上下文 A、B 和 C。

作为一个示例，如果该 DSA 执行了一个子树搜索，该搜索起始于命名上下文 X 内的一个基对象，则 DSA 能够从复制区 X 和 C 中提供信息。命名上下文 A 和 B 中的信息必须通过下级引用才能够被提供。在执行请求分解时，可用于 **partialResults** 或链接中的继续引用，将指定 Y 为目标对象，而 C 为一个 **Exclusions** 集内的单独元素。

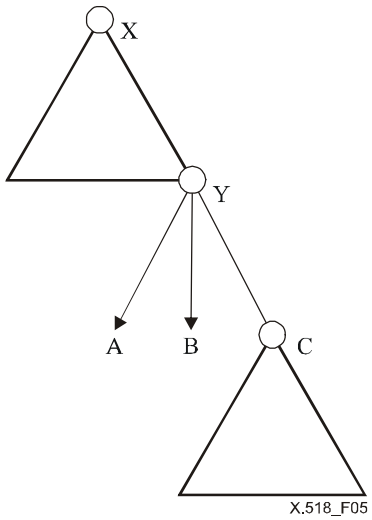


图 5－排除

10.11 继续引用

一个继续引用 **ContinuationReference** 描述了某个操作的全部或部分是如何在一个不同的 DSA、LDAP 服务器或它们的某种组合中继续执行的。典型地，当包含在内的 DSA 不能够或不愿意自己去传播请求时，继续引用是作为一个提名来返回的。

```
ContinuationReference ::= SET {
        targetObject          [0]    Name,
        aliasedRDNs           [1]    INTEGER OPTIONAL, -- 仅出现在第一版本系统中
        operationProgress     [2]    OperationProgress,
        rdnsResolved          [3]    INTEGER OPTIONAL,
        referenceType         [4]    ReferenceType,
        accessPoints          [5]    SET OF AccessPointInformation,
        entryOnly             [6]    BOOLEAN DEFAULT FALSE,
        exclusions            [7]    Exclusions OPTIONAL,
        returnToDUA           [8]    BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster   [9]    BOOLEAN DEFAULT FALSE }
```

不同组件的含义定义如下：

a) 组件 **targetObject** 指示了在继续执行操作中所要使用的对象名字。它可能不同于入请求中接收到的 **targetObject** 中的名字，例如在一个别名被解除引用，或者搜索中的基对象已经被定位的情况下。

   在 **targetObject** 中的 RDN 必须是主 RDN（对于已经处理过的 RDN）。带上下文的可替代识别值也可能包含在内。

b) 组件 **aliasedRDNs** 指示了在目标对象名字中有多少个 RDN（如果有的话）已经被别名解除引用处理过了。当且仅当别名已经被解除引用时该变量存在。

   注 — 提供本组件是为了与号码簿第一版本的实现相兼容。根据后续的号码簿规范版本实现的 DUA（和 DSA）应当总是忽略后续请求的 **CommonArguments** 中的此参数。在这种情况下，如果别名解除引用到其他别名时，号码簿不会发出错误信号。

c) 组件 **operationProgress** 指示了已经完成的名字解析的数量，该参数将控制指定 DSA 的后续操作的执行，假设接收 **ContinuationReference** 的 DSA 或 DUA 希望在其后继续执行。

d) 组件 **rdnsResolved** 的值（如果名字中的部分 RDN 不能进行完全的名字解析，但是被假设通过一个交叉引用而纠正了，在这种情况下，才需要出现该组件）指示了实际上有多少个 RDN 仅使用内部引用就已经被解析过了。

e) 组件 **referenceType** 指示了什么类型的知识被用于产生此继续引用。

f) 组件 **accessPoints** 指示了为了获得此继续引用而需要接触的访问点。仅在包含非特定下级引用时，才能够出现多个 **AccessPointInformation** 项。

g) 当初始操作是一个搜索操作，且 **subset** 变量被设置为 **oneLevel**，并且遇到一个别名条目是 **baseObject** 的直接下级时，在这种情况下，组件 **entryOnly** 被设置为 **TRUE**。针对 **targetObject** 名字成功地执行了名字解析的 DSA，必须对此唯一的命名条目执行对象评估。

h) 组件 **exclusions** 标识了一系列下级命名上下文，这些下级命名上下文不能由接收方 DSA 部署。

i) 元素 **returnToDUA** 是可选提供的，当创建此继续引用的 DSA 希望指示出它不愿意通过一个中介 DSA 来返回信息（如出于安全原因），并且希望指示出信息可以通过初始 DUA 或 LDAP 客户机和 DSA 之间的一个 DAP 或 LDAP 操作而直接可用时，则提供此组件。若 **returnToDUA** 被设置为 **TRUE**，则 **referenceType** 可能被设置为 **self**。

j) 元素 **nameResolveOnMaster** 是可选提供的，当创建此继续引用的 DSA 遇到 NSSR 时，则提供此组件。如果设置为 **TRUE**，则它指示后续的名字解析，即与从 **nextRDNToBeResolved** 开始的剩余 RDN 进行匹配时，不能使用条目拷贝信息，其中包括多主实现中的可写拷贝；对每个剩余 RDN 的后续解析必须在该 RDN 所标识的条目的主 DSA 内进行（见 20.1）。

11 绑定和解绑定

DSABind 和 **DSAUnbind** 分别被 DSA 用在访问另一个 DSA 的时间段的起始点和结束点。一个 DSP 连接的绑定和解绑定本身不能够导致在连接过程中所请求的任何分布式分页结果的丢失。

11.1 DSA绑定

一个 **DSABind** 操作被用于开始两个提供号码簿服务的 DSA 之间的合作阶段。

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

DSABind 中的组件与号码簿绑定 DirectoryBind（见 ITU-T X.511 建议书 | ISO/IEC 9594-3）中的对等部分是相同的，除了下列所述的不同：

— **DirectoryBindArgument** 中的 **Credentials** 允许标识始发 DSA 的 AE-Title 的信息能够被发送到响应 DSA。AE-Title 的格式必须为一个号码簿识别名。

— **DirectoryBindResult** 中的 **Credentials** 允许标识响应 DSA 的 AE-Title 的信息能够被发送到始发 DSA。AE-Title 的格式必须为一个识别名。

— DSA 的名字或 AE-Title 可能使用可替代识别名，并且可能包含上下文信息。

注 1 — 因为名字可被用于简单证书或强证书，因此可能使用可替代识别名，如果它们存在的话。然而，如果不使用主识别名，则基于名字的鉴权和访问控制可能不能按照期望的那样工作。随后对已鉴权的 BIND 操作的成功处理，无论在 BIND 操作中使用什么样的名字，从此以后被绑定的实体相互之间必须知道它们的主识别名，以便简化在 BIND 生效过程中访问控制的操作。

注 2 — 鉴权所要求的证书可能被安全交换服务元素（见 ITU-T X.519 建议书 | ISO/IEC 9594-5）所携带，在这种情况下，它们不出现在绑定变量或结果中。

11.2 DSA解绑定

两个提供号码簿服务的 DSA 之间合作阶段结束时的解绑定，如果用于 OSI 环境，则在 ITU-T X.519 建议书 | ISO/IEC 9594-5 的 7.6.4 节和 7.6.5 节规定，如果用于 TCP/IP 环境，则在 ITU-T X.519 建议书 | ISO/IEC 9594-5 的第 9.3.2 节规定。

12 链接操作

对于每个用于访问号码簿抽象服务的操作，在合作的 DSA 之间都有一个一一对应的对等操作。所选择的操作名字要体现出这种对应性，因此在合作的 DSA 之间使用的操作名字前都增加一个前缀“链接”。

链接操作的变量、结果和错误都是从号码簿抽象服务的相应操作的变量、结果和错误中系统构建而成的（见 12.1 的描述），只有一个例外。这个例外便是 **ChainedAbandon** 操作，它在语法上是等同于它相对应的号码簿抽象服务的（见 12.2 的描述）。

12.1 链接操作

如果一个 DSA 接收到来自一个 DUA 或 LDAP 客户机的操作，则它可能会选择构建该操作的一个链接形式来将此操作传播到另一个 DSA。而接收到一个操作的链接形式的 DSA，也可能会选择将此操作链接到另外一个 DSA。调用一个操作的链接形式的 DSA 可能会对操作的参数进行签名、加密、或签名并加密；而执行操作的 DSA，如果被这样请求的话，也可能会对操作响应者所返回的结果或错误进行签名、加密或签名并加密。从某个 LDAP 客户机处接收到一个操作的 DSA，或者从另一个 DSA 处接收到一个 LDAP 操作的 DSA，可能会选择将原始的 LDAP 客户机提供的操作传播到一个 LDAP 服务器。

一个操作的链接形式的规范是使用参数化类型 **chained { }**。

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

注 1 — 可能被用于 **chained { }** 中的实际参数的号码簿抽象服务的操作包括 **abandoned** 错误。在一个链接操作的一系列可能错误中，此错误的出现体现了 12.2 节讨论的可能性，即当一个链接的连接失效时，可以为 **chainedModify** 操作产生一个 **chainedAbandon**。

注 2 — 在附件 A 中对 DSA 抽象服务的明确规范中，应用了此参数化类型来构造所有的抽象服务的链接操作。

派生出的操作的变量具有下列组件：

a) **chainedArgument** — 这是 **ChainingArguments** 的一个值，包含了除始发 DUA 或 LDAP 客户机提供的变量之外的那些信息，为了让执行的 DSA 或 LDAP 服务器能够完成操作，需要这些信息。该信息类型在 10.3 节定义。

b) **argument** — 这是 **operation.&Argument** 的一个值，包含了 DUA 提供的原始变量，如同在 ITU-T X.511 建议书 | ISO/IEC 9594-3 的相应章节规定的那样，或者包含了 LDAP 客户机提供的原始变量，如同在 RFC 2251 的相应章节规定的那样。

注 3 — 如果认为适当的话，它还可能封装除了源自 DAP 或 LDAP 的 PDU 类型之外的其他 PDU 类型。完成上述工作机制的规范尚待研究。

如果请求成功，则派生操作的结果具有如下组件：

a) **chainedResult** — 这是 **ChainingResults** 的一个值，包含了除将要提供给始发 DUA 的信息之外的那些信息，在某个链接中，之前的 DSA 可能需要这些信息。该信息类型在 10.4 节中定义。

b) **result** — 这是 **operation.&Result** 的值，由该操作的执行者将要返回的结果组成，并且将要被放在返回给始发 DUA 的结果中而被传递回去。该信息在 ITU-T X.511 建议书 | ISO/IEC 9594-3 的适当章节中规定。

如果请求失败，除了 **dsaReferral** 要替代 **referral** 被返回之外，还将返回 **operation.&Errors** 错误集中的一个错误。可能会被上报的错误集在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中为相应操作描述。错误 **dsaReferral** 在 13.2 节中描述。

12.2 链接放弃操作

chainedAbandon 操作由一个 DSA 使用，向另一个 DSA 指示它不再对之前调用的某个分布式操作继续被执行感兴趣。这可能出于多种原因，其中示例如下：

— 让 DSA 初始发起链接的操作本身已经被放弃了，或者由于连接的中断而被隐含中止了；

— DSA 已经通过另外的方法获得了必要的信息，例如已经从一个包含在并行多链接中的某个更快的响应 DSA 处获得了信息。

一个 DSA 永远都不能被迫发起一个 **chainedAbandon**，或者实际上确实放弃了一个操作，如果被要求如此的话。

如果 **chainedAbandon** 真正成功停止了某个操作的执行，则将有一个结果被返回，且相应的操作将返回一个 **abandoned** 错误。如果 **chainedAbandon** 没有成功停止某个操作，则它本身将返回一个 **abandonFailed** 错误。

12.3 链接操作和协议版本

要求协议版本要高于 v1 的操作（例如带某些变量的 **modifyEntry** 操作），或者当与某个版本高于 v1 的协议使用时，返回不同结果的操作（例如带一个签名变量的 **modifyEntry** 操作），必须仅被链接到与传送请求的连接具有相同版本或更高版本的连接上。

13 链接错误

13.1 引言

在大多数情况下，在号码簿抽象服务中返回的相同错误都能够在 DSA 抽象服务中返回。例外情况是 **dsaReferral** 错误被返回（见 13.2），而替代了 **Referral**，同时下列服务问题具有相同的抽象语法但具有不同的语义：

a) **invalidReference** — 返回此错误的 DSA 在呼叫方 DSA 的知识中检测到一个错误，呼叫方 DSA 的知识在链接变量 **referenceType** 中指定。

b) **loopDetected** — 返回此错误的 DSA 在号码簿的知识信息中检测到一个环回。

可能出现的错误的优先级同号码簿抽象服务中的优先级相同，在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中指定。

如果在一个链接操作中出现了一个错误，则响应方 DSA 可能会对返回的错误进行签名、加密或签名并加密。

13.2 DSA提名

dsaReferral 错误由 DSA 产生，无论何种原因，当 DSA 不愿意将该操作链接到一个或多个其他 DSA 而继续执行该操作时，产生此错误。它可能会返回一个提名的环境在 8.3 节描述。

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

不同参数的含义如下所述：

a) **ContinuationReference** 包含调用者在将适当的请求传播到另一个 DSA 或一个 LDAP 服务器时所需的信息。该信息的类型在 10.11 节规定。

b) 如果该操作的 **ChainingArguments** 的 **returnCrossRefs** 组件取值为 **TRUE**，且该提名是基于一个下级引用或交叉引用，则可能可选地包含参数 **contextPrefix**。任意 DSA 的主管当局将决定哪种知识引用（如果有的话）能够以此种方式返回（例如，其他方式可能对该 DSA 是保密的）。

contextPrefix 或一个继续提名必须是主识别名。带上下文的可替代识别名可能包含在任意 RDN 的 **AttributeTypeAndDistinguishedValue** 的 **valuesWithContext** 组件中。

所提供的信息能够可选地通过使用 **CommonResults** 中的 **notification** 组件进行限定。

第5部分 — 分布式规程

14 引言

14.1 范围和限制

本节规定了 DSA 所执行的号码簿分布式操作的规程。每个 DSA 都独立地执行下面所描述的规程；所有 DSA 的集体行为将构成号码簿提供给用户的完整的服务集合。

14.2 一致性

本节对 DSA 规程的描述是基于 ITU-T X.501 建议书 | ISO/IEC 9594-2 中的第 8 节和第 9 节描述的模型，以及本号码簿规范的第 7 节和第 8 节描述的模型。流程图及其相应的文字描述是一种方式，将一个给定的向 DSA 输入的外部（DAP、LDAP 和/或 DSP）输入集合映射为该 DSA 所产生的一个或多个外部输出（即一个结果、错误、提名或链接请求等），依赖于该 DSA 所拥有的特定的 DSA 信息树。

号码簿很可能分布在各种不同的 DSA 之间，这些 DSA 可能根据号码簿规范的不同版本实现，或者仅支持 LDAP 的实现等。发起请求的 DUA 或 LDAP 客户机不会关注满足 DUA 或 LDAP 客户机请求的一个或多个 DSA 是按照哪个版本实现的。因此，要在这样一个异构环境下执行操作，DSA 必须根据 ITU-T X.519 建议书 | ISO/IEC 9594-5 的第 12 节中定义的扩展规则来实现。

注 1 — 仅支持 LDAP 而实现的 DSA，可能根据，也可能不根据该扩展规则来实现。

DSA 的实现必须在功能上与这里所描述的这些规程所规定的外部行为是等价的。某个特定的 DSA 在实现时，如何根据给定的输入以及拥有的 DSA 信息树而导出正确的输出，所使用的算法是没有被标准化的。

注 2 — 伴随规程的流程图是辅助性的，用来帮助对规程的理解。但不能认为它们是对文字描述的一种精确替代。对于某个特定的规程，如果在文字描述和流程图之间有分歧，则认为文字描述的优先级较高。

14.2.1 包含第一版本DSA的交互

如果修改操作是在跨 DSA 的边界进行评估的（即带有 **TargetSystem** 的 **addEntry** 操作，删除或重命名某个上下文前缀等操作），则本号码簿规范仅规定了两个第二版本或后续版本的 DSA 应当具有何种行为。两个第一版本的 DSA 之间的交互，以及第一版本的 DSA 和第二版本或后续版本的 DSA 之间的交互，不在本系列号码簿规范的定义范围之内。如果混合版本的 DSA 之间具有一个分等级的操作绑定，则彼此版本的知识可能会允许向用户给出一个一致性方面的错误。

14.3 概念模型

号码簿分布式操作的复杂性引发了这样的一个需求，即同时使用叙述性描述技术和图形描述技术来进行概念建模。然而，无论是叙述性方法还是图形图表方法都不能被认为是对分布式号码簿操作的一种正式描述。

14.4 DSA的单独操作和合作操作

该模型从两个不同的视角对 DSA 操作进行观察，这两个视角结合起来共同提供了一个完整的号码簿操作视图。

a) **以 DSA 为中心的视角** — 在这种视角下，支持号码簿的一系列规程都从一个单独 DSA 的视点进行描述。这就使得为每个规程都提供一个明确的规范成为可能，并且能够完整地解决它们之间的相互关系和全面的控制结构。第 16 节到 22 节便从一个以 DSA 为中心的视角描述了 DSA 规程。

b) **以操作为中心的视角** — 以 DSA 为中心的视点提供了完整的细节，但对理解单个操作的结构变得困难，这些操作可能会被多个 DSA 所执行。因此在后面的第 15 节采用了一个主要以操作为中心的视点来介绍应用到每个操作的执行阶段。

为了支持号码簿的分布式操作，每个 DSA 都必须执行两类动作，一类是实现每个操作的自身目的所需的动作，另一类是为了将这些实现在多个 DSA 间分布所需的附加动作。第 15 节研究了这两类动作之间的差异。第 16 节到 22 节对这两类动作都给出了详细规定。

14.5 DSA之间的合作协定

所有的 DSA 如果根据它们所拥有的命名上下文而处于一种下级/上级关系中，则这样的 DSA 之间具有分等级操作绑定和/或非特定的分等级操作绑定，依赖于这些 DSA 所拥有的知识引用的类型。

两个 DSA 之间的分等级操作绑定和非特定分等级操作绑定可能会通过使用 24 节和 25 节中定义的规程来进行管理，或者通过其他的途径（如电话）来进行管理。

一个 DSA，如果它拥有的条目处于其上级 DSA 的管理区内，则该 DSA 必须管理子模式，必须遵循控制搜索规则（如果有的话），且必须按照主管当局的要求对条目的访问进行控制。管理区内条目的调整可能会按照 ITU-T X.501 建议书 | ISO/IEC 9594-2 中定义的那样执行，也可能会通过本地机制来执行。

15 分布式号码簿行为

15.1 操作的合作实施

每个 DSA 都配备有相应的规程来完整地实施所有的号码簿操作。如果一个 DSA 包含完整的 DIB，则在这种情况下，实际上所有的操作都完全是在此 DSA 内部执行的。如果 DIB 是分布在多个 DSA 内的，则在这种情况下，一个典型操作的实现是分段的，在每个潜在的合作 DSA 内将仅执行该操作的一部分。

在分布式环境中，典型的 DSA 将每个操作都看作是一个临时事件：由一个 DUA、一个 LDAP 客户机或某些其他 DSA 所调用的操作；DSA 对相应对象进行处理，然后将其前向到另一个 DSA 以便作后续处理。

一个替代视点认为整个处理是在其实施过程中通过多个合作的 DSA 由一个操作所完成的。这种视角揭示了可以应用到所有操作的公共处理阶段。

15.2 操作处理的阶段

每个号码簿操作都可能被认为由三个不同的阶段构成：

a) *名字解析（Name Resolution）阶段*，在此阶段，某个特定操作将要执行的条目所在的对象名字被用来定位拥有此条目的 DSA；

b) 评估（*Evaluation*）阶段，在此阶段，某个特定的号码簿请求所指定的操作（如一个阅读操作）被真正地执行；

c) *结果合并阶段（Results Merging phase）*，在此阶段，某个指定操作的结果被返回到发出请求的 DUA 或 LDAP 客户机。如果选择了交互的链接模式，则结果合并阶段可能会包含多个 DSA，其中每个 DSA 都在之前的一个或两个阶段中将原始请求或子请求（如 15.3.1 节中的定义 — 请求分解）链接到另一个 DSA。

在操作为阅读、比较、列表、搜索、修改条目、修改 DN 和删除条目的情况下，名字解析针对的是在操作参数中所提供的对象名字。在操作为增加条目的情况下，名字解析的目标条目是操作参数所提供的条目的直接上级条目 — 这可以很容易地通过将操作参数所提供的名字中的最后一个 RDN 去掉而得到。（这是通过 18.3.1 节中 **FindDSE** 过程中的本地参数 **m** 而实现的。）

针对某个特定条目的一个操作可能会被初始发送到号码簿内的任意 DSA。该 DSA，使用它的知识，可能还会联合其他的 DSA 一起，通过上述三个阶段来处理此操作。

15.2.1 名字解析阶段

名字解析是一个将某个声称名字中的每个 RDN 与 DIT 的弧（或顶点）进行顺序匹配的过程，逻辑上起始于 DIT 的根，并在 DIT 内一直向下进行。然而，由于 DIT 是分布在任意的多个 DSA 内的，因此每个 DSA 可能仅能够执行名字解析过程中的一小部分。一个给定的 DSA 通过遍历它的本地 DSA 信息树来执行名字解析过程中属于它的那部分。该过程在第 18 节中描述，并且提供了相应的图（见图 9 到 12）。基于它的本地 DSA 信息树，以及包含在内的知识信息，一个 DSA 能够推断出名字解析是否可以被一个或多个其他 DSA 继续进行，或者名字是否是错误的。

如果服务控制选项 **manageDSAIT** 被设置，则名字解析阶段将被限制为仅在一个 DSA 信息树内工作。

15.2.2 评估阶段

当名字解析阶段完成后，所要求的实际操作（如阅读或搜索操作）就开始执行了。

包含一个单独条目查询的操作 — 阅读和比较 — 可能完全是在此条目所在的 DSA 内执行的。

包含多个条目查询的操作 — 列表和搜索 — 需要定位目标对象的下级，而这些下级可能处于同一个 DSA，也可能处于不同的 DSA。如果它们不是全部都处于同一个 DSA 内，则操作需要被指引到其他的 DSA 来完成评估过程，这些 DSA 是通过（适当的）下级引用、非特定下级引用、提供者引用、或主引用来指定的。

如果服务控制选项 **manageDSAIT** 被设置，则评估阶段将被限制为仅在一个 DSA 信息树内工作。类似的，如果评估过程起始于一个服务特定的管理区，则评估将被限制在该管理区内。

15.2.3 结果合并阶段

一旦评估阶段的某些结果可用，则进入到结果合并阶段。

如果操作仅受一个单独条目的影响，则在这种情况下，操作结果可以被简单地返回给请求 DUA 或 LDAP 客户机。如果操作受到多个 DSA 内的多个条目的影响，则在这种情况下，结果可以被组合。如果对结果执行了保护，则结果不能被组合。返回给 DUA 或 LDAP 客户机的结果必须是没有执行合并的结果。

在结果合并后，允许返回给请求者的响应包括：

a) 操作的完整结果；

b) 不完整的结果，由于 DIT 的某些部分仍是不可查询的（仅应用于列表和搜索）。这样的一个部分结果中可能包含为那些不能查询的 DIT 部分的继续引用；

c) 一个错误（提名是一种特殊的情况）；以及

d) 如果请求者是一个 DSA，则可能返回一个 **ChainingResults**。

15.3 管理分布式操作

在 DSA 可能被要求执行的每个操作的变量中包含的信息，指示了当操作在跨越号码簿的多个 DSA 时，该操作的执行进展情况。这使得对于每个 DSA 来说可以执行处理所要求的适当方面，并且在将操作向外前向到其他 DSA 前，记录此方面工作的完成。

在 DSA 中还包括附加的规程以便在物理上分布操作，同时支持由于它们的分布而引发的其他需求。

15.3.1 请求分解

请求分解是在 DSA 与其他一个或多个其他 DSA 和 LDAP 服务器通信之前，由 DSA 内部执行的一个过程。一个请求被分解为多个子请求，因此每个子请求完成原始任务的一部分。例如在搜索操作中，在基对象被发现后可以使用请求分解。在分解后，每个子请求可能被单链接或多链接到其他 DSA 和/或 LDAP 服务器上，以便继续执行任务。

如果操作是由一个 DUA 发起的，则一个链接请求（见 12.1 节）或子请求的 **argument** 必须是未经修改的操作变量，而如果操作是由一个 LDAP 客户机发起的，则该 **argument** 必须是未经修改的 LDAP 消息。一个接收到链接请求的 DSA 在做请求分解时不应当修改 **argument**。

注 — 下面的各小节规定了 **argument** 中的每个独立组件的需求。但不应当被解释为那些没有被显式提及的组件是可以修改的。

15.3.2 DSA作为请求响应者

接收请求的 DSA 可以使用 **operationProgress** 参数来检查该请求的进展情况。这可以判断操作是仍然处于名字解析阶段还是已经到了评估阶段，以及 DSA 必须准备满足操作的哪一部分。如果 DSA 不能够完全满足请求，

则它必须或者将请求（通过单链接或多链接）传递到其他的能够帮助实现请求的一个或多个 DSA 和/或 LDAP 服务器，或者返回另一个 DSA 或 LDAP 服务器的提名，或者终止该请求并返回一个错误。

15.3.3 操作的完成

每个发起操作的 DSA，或者将操作传播到一个或多个其他 DSA 和/或 LDAP 服务器的 DSA，都必须保持该操作的存在踪迹，直到每个其他的 DSA 和/或 LDAP 服务器都已经返回了一个结果或错误，或者操作的最大时间限制已经超时。此需求可应用到所有的操作、传播模式和处理阶段。它确保了传播到号码簿的分布式操作可以按顺序结束。

15.4 环回的处理

DIT 可能会处于某种状态而引起环回。作为一个示例，在名字解析过程中，当对一个或多个别名进行的解除引用使得解析又回到 DIT 的相同分支上时，则出现环回。另一个引起环回的潜在原因是由于错误地配置了知识引用。

在一个特定的号码簿操作的上下文内，如果在任何时间，操作返回到之前的某个状态，则出现环回，这里，状态通过下面的组件来定义：

— 当前处理操作的 DSA 的名字；
— 包含在操作变量中的 **targetObject** 的名字；
— 包含在操作变量中的 **operationProgress**，如 10.5 节的定义。

这并不意味着一个操作不能被某个特定的 DSA 多次处理。然而，它意味着 DSA 不能多次处理处于同一状态的同一个操作。

可以使用 10.6 节定义的 **traceInformation** 变量来控制环回，该变量记录了某个特定操作已经经历过的状态序列。定义了两个策略来判断是否出现了环回，或者将要出现环回。这两个策略是环回检测和环回避免，它们分别在 15.4.1 节和 15.4.2 节中描述。

环回检测是必选的，而环回避免是可选的。

15.4.1 环回检测

一旦接收到一个号码簿操作，DSA 最初必须对操作进行合法性判断以确保该操作是可被继续执行的。合法性判断的一个重要任务是检查环回，通过判断操作的当前状态是否已经存在于该操作的 **traceInformation** 变量中所记录的之前状态序列中。检查环回的这一步骤便是环回检测。

15.4.2 环回避免

环回避免要求某个 DSA，作为链接规程的一部分在将某个请求前向到另一个 DSA 之前，就要判断由此而产生的操作状态（这是一个 **traceItem**，接收的 DSA 在接收到它之后会将其增加到 **traceInformation** 中）是否已经存在于该初始入操作的 **traceInformation** 变量中记录的之前状态序列中。

若接收到提名或对提名进行操作，在这种情况下，就不能仅仅通过检查 **traceInformation** 来完成环回避免和环回检测。在这种情况下，每次 DSA 对提名进行操作时，它都必须存储由此而产生的操作状态（即 **traceItem**，接收的 DSA 在接收到该请求之后会将其增加到 **traceInformation** 中），并且将入请求记录也随之存储。在对某个提名进行操作或返回某个提名之前，DSA 都需要检查此列表，以便检查当准备为入操作服务时，相同的请求在之前并未发送过。

15.5 分布式操作的其他考虑

15.5.1 服务控制

在分布式环境中，为了操作能够按照所请求的方式执行，某些服务控制需要特殊的考虑。

a) **chainingProhibited** — 一个 DSA 在决定某个操作的传播方式时，需要考虑此服务控制。如果它被设置，则 DSA 将总是使用提名方式。然而，如果它未被设置，则 DSA 能够依赖于它自身的能力，选择是使用链接方式，还是使用提名方式。

b) **timeLimit** — 一个 DSA 需要考虑此服务控制以确保该 DSA 中没有超出时间限制。被某个 DUA 请求执行操作的 DSA，最初应注意 DUA 所表达的 **timeLimit**，该时间限制以秒计，作为完成该操作的可用的持续时间。如果要求链接，则 **timeLimit** 包含在将要传递到下一个 DSA 的链接变量中。在这种

情况下，同样的限制值将用于每个链接请求，且在此时间（UTC）之前，操作必须完成以满足最初规定的限制。在接收到指定了 **timeLimit** 的 **ChainingArguments** 时，接收的 DSA 应遵守此限制。

c) **sizeLimit** — 一个 DSA 需要考虑此服务控制以确保结果列表没有超出指定的尺寸限制。该限制包含在初始请求的公共变量中，在请求被链接时将被无改变地传递。如果要求进行请求分解，则相同的值包含在将要传递到下一个 DSA 的变量中，对每个子请求都应用完全的限制。当结果返回时，请求者 DSA 分析多个结果，并将此限制应用于全体结果以便确保仅有请求的数量被返回。如果超出了该限制，将在答复中指出。

d) **priority** — 在所有的传播方式下，每个 DSA 都有责任确保操作的处理过程是有序的，以便支持此服务控制，如果此服务控制存在的话。

e) **localScope** — 操作被限制在一个本地定义的范围内，且每个 DSA 都不能将请求传播到该范围之外。

f) **scopeOfReferral** — 如果 DSA 针对某个列表或搜索操作返回了一个提名或部分结果，则所包含的继续引用必须处于被请求的范围之内。

所有其他的服务控制都需要被遵守，但是它们的使用在分布式环境中不需要任何特别的考虑。

15.5.2 扩展

如果一个 DSA 在处理过程的名字解析阶段遇到一个扩展操作，且判断操作必须被链接到一个或多个 DSA，则它必须在链接操作中无变化地包含任何出现的扩展。

注 — 一个主管当局，如果不希望传播一个扩展时，他可能会判断返回一个问题为 **unwillingToPerform** 的 **serviceError** 是合适的。

如果一个 DSA 在处理过程的评估阶段遇到一个它不支持的扩展，则有两种可能性出现。如果扩展不是关键的，则 DSA 必须忽略此扩展。如果扩展是关键的，则 DSA 必须返回一个问题为 **unavailableCriticalExtension** 的 **serviceError**。一个关键扩展，如果扩展到一个多对象操作，可能会导致出现多样的结果和服务错误。合并这些结果和错误的 DSA 必须抛弃这些服务错误，并且使用 **PartialOutcomeQualifier** 的组件 **unavailableCriticalExtension**，正如在 ITU-T X.511 建议书 | ISO/IEC 9594-3 中描述的那样。

15.5.3 别名解除引用

别名解除引用是创建一个新的目标对象名字的过程，它使用别名条目的 **AliasedEntryName** 的属性值来替换原始目标对象名字的别名条目的识别名部分。在操作中的 **object** 名字不受别名解除引用的影响。

15.5.4 解析上下文变化的名字

在名字解析阶段，当 RDN 被处理时，通过确保 RDN 中的每个 **AttributeTypeAndDistinguishedValue** 都使用该属性的主识别值作为其 **value**，则可以创建一个新的目标对象名字。通过这种方法，目标对象名字被处理为一个主识别名。这样做是为了提供一致的名字处理，尤其是在名字解析中可能包含第三版本之前的 DSA 时。在操作中的 **object** 名字不受该替代的影响。

15.5.5 分页结果

当一个 DUA 在**搜索**或**列表**操作的请求（见 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 7.9 节）中包含 **PagedResultsRequest** 时，分页可能是由直接与 DUA 绑定的 DSA 来执行的，这种 DSA 又被称为绑定 DSA；或者可能是由拥有**搜索**或**列表**操作请求中的 **baseObject/object** 条目的 DSA 来执行的（可能在一个或多个别名解除引用之后），这样的 DSA 又被称为初始执行者。如果分页是由绑定 DSA 执行的，该 DSA 也可能是初始执行者，则这样的分页被称为绑定 DSA 的分页结果。如果分页是由初始执行者执行的，且初始执行者不同于绑定 DSA，则这样的分页被称为 DSP 分页结果。

一个支持 DSP 分页结果的 DSA 必须：

— 支持绑定 DSA 的分页结果；

— 作为一个绑定 DSA 时，支持 DSP 分页结果；

— 作为一个初始执行者时，支持 DSP 分页结果；且

— 支持 **PartialOutcomeQualifier** 的子组件 **entryCount**。

如果一个绑定 DSA 接收到一个包含 **PagedResultsRequest** 的**搜索**或**列表**请求，且绑定 DSA 不是该请求的初始执行者时，则绑定 DSA 可能会选择在 **ChainingArguments** 中包含参数 **dspPaging**。初始执行者可能会选择完成 DSP 分页结果。这种情况会通过在 **PartialOutcomeQualifier** 中包含一个 **queryReference** 参数而向绑定 DSA 发出信号。返回给 DUA 的 **queryReference** 将被用于检索下一页。

如果初始执行者不支持 DSP 分页结果，或者选择不去执行 DSP 分页结果，则绑定 DSA 可能会执行正常的绑定 DSA 分页。

一个作为执行者但不是初始执行者的 DSA，必须忽略 **chainingArguments** 中可能的 **dspPaging** 组件，并且它必须遵守服务控制 **sizeLimit**，如果该服务控制存在的话。

15.6 分布式操作的鉴权

号码簿的用户，以及提供号码簿服务的主管当局可能会根据它们自己的决定，要求号码簿操作被鉴权。对于任意一个特定的号码簿操作，鉴权过程的特性依赖于当时生效的安全策略。

有两个鉴权规程的集合可用，两者联合起来共同满足一定范围内的鉴权需求。一个规程集是由绑定提供的：这些规程简化了两个号码簿应用实体之间为了建立一个连接而所需的鉴权。绑定规程包含了一定范围的鉴权交互，从身份的简单交换到强鉴权。

除了绑定所提供的建立连接时对等实体之间的鉴权外，在号码簿内还定义了另外的规程使得独立的操作也可被鉴权。定义了两个不同的号码簿鉴权规程集。一个是发起者鉴权服务，由一个 DSA 对原始服务请求的发起者进行鉴权。第二个是结果鉴权服务，由发起者对返回的任意结果进行鉴权。

对于发起者鉴权，定义了两个规程，一个是基于身份的简单交换，被称为**基于身份的鉴权**；另一个是基于数字签名技术，被称为**基于签名的鉴权**。前一个规程从本质上说是最初级的，因为身份交换是基于识别名的交换，而识别名是明文传送的。

对于结果鉴权，定义了一个单独的**结果鉴权**规程，是基于数字签名技术的；由于一般从本质上来说结果核对是复杂的，因此没有定义简单的，基于身份的规程。

错误响应的鉴权可能被这些规程所支持。

下面所描述的服务被认为是对绑定服务所提供服务的增强；假设绑定规程在对号码簿操作进行鉴权之前就已经被成功实现了。

DSA 所实现的提供发起者鉴权和结果鉴权的规程在第 22 节规定。

16 操作调度程序

操作调度程序是 DSA 内主要的控制规程。它引导每个操作都经过请求处理的三个阶段。因此，**操作调度程序**使用一系列的规程来完全地处理请求，如图 6 所示。

16.1 通用概念

16.1.1 规程

操作调度程序所部署的每一个规程都包含一个概念性的接口定义，这是根据其参数来定义的，即变量、结果和错误等；同时还包含一个关于规程步骤本身的描述。这些规程的行为通过流程图和文字来描述。在一个流程图内所使用的符号具有如下语义（见图 7）。

16.1.2 公共数据结构的使用

所有的规程都使用一些数据结构，这些数据结构在**操作调用程序**内，在某个操作的处理过程中是可用的。这些数据结构用于协调**操作调用程序**内的数据流。大多数这些数据结构都直接与操作的变量和为操作而产生的结果相关联。变量和结果组件的表示都是在相关的 ASN.1 定义中使用它们的名字来完成的（例如链接变量的

operationProgress 组件）。如果这些结构中的任意一个是复合结构，则该结构的一个组件可被表示为 **compound.component**（例如 **operationProgress.nameResolutionPhase**）。

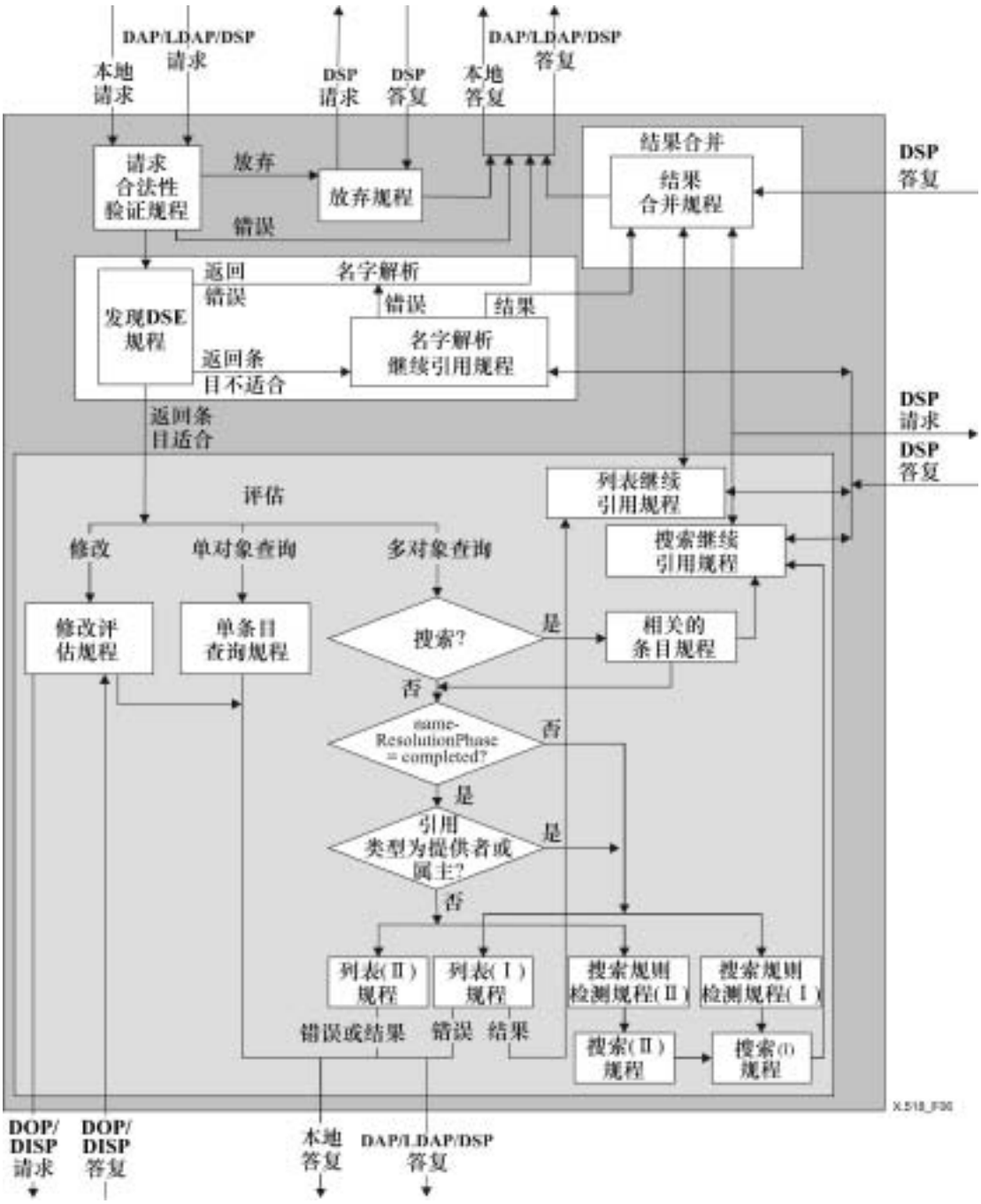


图 6－操作调度程序

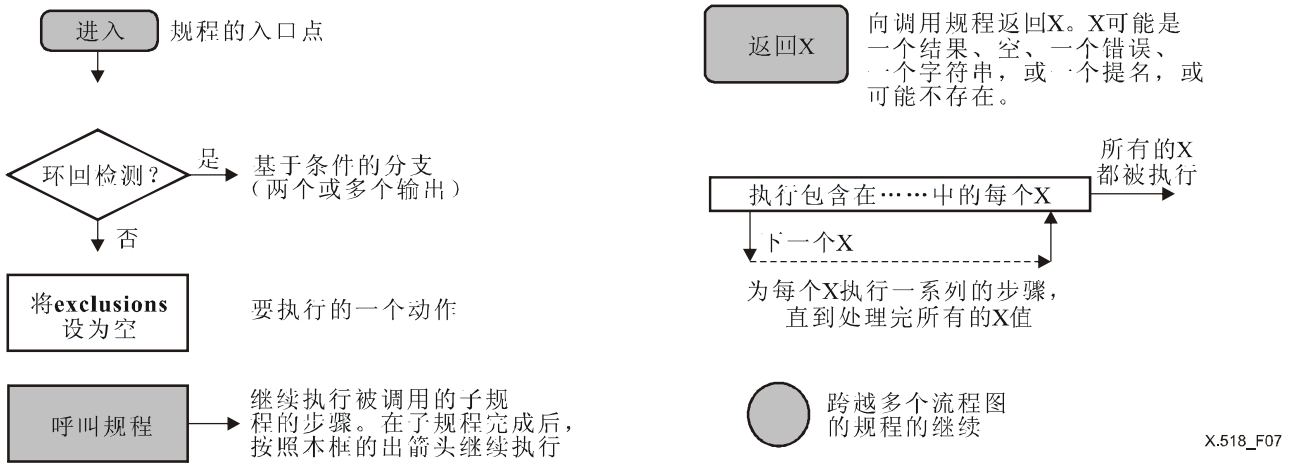


图 7－流程图中使用的符号

下面数据结构在**操作调用程序**中定义：

— **NRcontinuationList** — 所创建的继续引用的列表，在名字解析继续引用规程中使用。

— **SRcontinuationList** — 所创建的继续引用的列表，在**列表或搜索继续引用**规程中使用。

— **admPoints** — 名字解析过程中收集到的类型为管理点的 DSE 的引用列表。

— **referralRequests** — 由于运行提名而被链接的请求或子请求的列表。每个这样的请求/子请求都以 **TraceItem** 的形式进行概括。该列表被 15.4.2 节中定义的环回避免规程使用。

— **emptyHierarchySelect** — 一个布尔型的变量，其值可在**等级选择**规程中被设置。在一个搜索操作过程中，当第一次进入**等级选择**规程时，假设该变量被重置。

— **streamedResultsOK** — 一个布尔型的变量，其值在**名字解析**规程中被设置，用来指示该操作可能接收流结果。该变量的缺省值为 false。

另外，一个规程还可能使用本地定义的一系列变量。

16.1.3 错误

在处理的每个阶段，执行任何子规程时，都可能会检测到错误。在本子规程中标识的错误一般会作为一个相应的协议错误返回给请求者。在这种情况下，操作调度程序将被立即终止。如果在接收到多个错误的情况下，本地规程可能会选择返回其中的一个。

可选的，一个规程也可能会选择在操作处理的某个点来处理错误（例如，如果一个带有问题为 **busy** 的 **serviceError** 被返回给一个链接搜索子请求）。在这种情况下，规程将继续它的运行，而不会有错误返回给请求者。

可选的，DSA 可能会基于所选择的 DirQOP 和所请求的错误保护，对某个分布式操作中返回的错误进行签名、加密、或签名并加密。

16.1.4 异步事件

在操作调度程序内对某个操作请求的处理过程中，可能会发生多个异步事件。下面的段落规定了如何去处理时间限制或尺寸限制或管理限制的越限、连接的丢失、以及对一个正在进行的操作发起放弃请求等事件。所有其他异步事件的处理，如本地策略的决定等，不在本号码簿规范的定义范围之内。

16.1.4.1 时间限制

在 **CommonArguments** 中规定的 **timeLimit** 可能在操作过程中的任意时间点到期。在这种情况下，一般会有一个问题为 **timeLimitExceeded** 的 **serviceError** 返回给发出请求的 DUA、LDAP 客户机或 DSA，且操作调度程序将被终止。可选的，一个规程可能选择某种不同的方式来处理本事件（如在一个**搜索**请求的处理过程中）。

如果一个 DSA 接收到从另一个 DSA 发来的请求，且超越了时间限制，则它必须发送一个问题为 **timeLimitExceeded** 的 **serviceError**，而不再对请求进行任何后续处理。

如果一个 DSA 正在处理一个（子）请求，且当 **timeLimit** 到期时，尚无结果可用，则它必须向请求者返回一个问题为 **timeLimitExceeded** 的 **serviceError**。

如果一个 DSA 正在处理一个子请求，且当 **timeLimit** 到期时，有结果可用，则它必须向请求者返回一个结果，并具有如下内容：

a) 一直到 **timeLimit** 到期前，所有收集到的结果；

b) 结果参数 **partialOutcomeQualifier** 中的 **limitProblem** 组件必须被设置为 **timeLimitExceeded**；

c) 结果参数 **partialOutcomeQualifier** 中的 **unexplored** 组件必须为每个 DSA 集包含一个继续引用值，这些 DSA 是子请求发向的目的地，但是它们的结果没有被包含在返回给请求者的结果中，该组件中还应当包括本 DSA 不愿意发送子请求的那些 DSA 的继续引用值。

16.1.4.2 连接丢失

如果与请求者之间的连接丢失，则所有可能返回的结果都会丢失。DSA 可能会可选地为每个正在执行的查询（子）请求发送一个 **chainedAbandon** 请求，除非与正在讨论的 DSA 的连接也丢失了。这些 **chainedAbandon** 请求的所有响应，以及正在执行的（子）请求的所有响应都必须被丢弃。在 DSP 分页结果的情况下，绑定 DSA 应当取消当前的分页结果，这是通过选择 **PagedResultsRequest** 中的 **abandonQuery** 选项而产生一个新的分页结果请求来实现的。

如果与当前链接的子请求的其中一个的连接丢失，但与请求者的连接尚未丢失，则仅对于查询操作而言，DSA 可能会可选地对指向其他能够处理链接请求的 DSA 的任何可替代引用进行尝试（例如，当与主 DSA 的连接丢失后，可以尝试指向某个镜像 DSA 的引用）。如果这样也不成功，则 DSA 必须按照如下所述：

1) 如果 **operationProgress.nameResolution** 的值被设置为 **notStarted** 或 **proceeding**，则或者向请求者返回一个问题为 **unavailable** 的 **serviceError**，或者返回一个提名错误，其继续引用中包含能够继续处理操作的 DSA 集。如果在名字解析过程中使用了非特定下级引用，且不是所有的当前连接都丢失，则可选地可以尝试进行名字解析，但不包括连接丢失的 DSA。如果这样也失败，则返回一个问题为 **unavailable** 的 **serviceError**，或者返回一个包含 NSSR 完整集合的提名错误。

   如果使用本地知识的 DSA 已知需要链接到丢失了连接的 DSA 上，这可能体现在适当的 **MasterOrShadowAccessPoint** 值中，则它必须选择发送一个问题为 **unavailable** 的 **serviceError**，且数据结构 **CommonResults** 的 **notification** 组件中必须包含：

   — 一个 **dSAProblem** 通知属性，且值为 **id-pr-targetDsaUnavailable**；以及

   — 一个 **distinguishedName** 属性，且值为 DSA 的识别名。

2) 如果 **operationProgress.nameResolution** 的值被设置为 **completed**，且该请求是一个单对象操作，则向请求者返回一个问题为 **unavailable** 的 **serviceError**。

3) 如果 **operationProgress.nameResolution** 的值被设置为 **completed**，且该请求是一个多条目查询操作，则 DSA 必须在操作结果的 **partialOutcomeQualifier.unexplored** 中增加一个继续引用，并携带 **AccessPointInformation** 信息，用以标识可以继续处理该操作的 DSA 集，其中包括连接已经丢失了的那些 DSA。

16.1.4.3 放弃操作

在某个操作的处理过程中，可能会接收到对该操作的一个放弃请求。在这种情况下，在处理放弃请求的过程中，针对此要被放弃的操作，**Abandon** 规程被调用。

16.1.4.4 管理限制

可能会由本地 DSA 主管部门或者 DSA 实现本身施加某些限制，例如在处理一个请求时所花费的时间量，或者返回数据的最大尺寸等。如果这些限制中的任意一个被越限，则 DSA 必须或者返回一个问题为

administrativeLimitExceeded 的 **serviceError**，或者返回一个部分结果（从已经收集的结果集中得到），且 **limitProblem** 被设置为 **administrativeLimitExceeded**。

在通知属性 **dSAProblem** 中必须返回的附加信息包括如下：

a) 如果限制是由主管部门施加的，则通知属性 **dSAProblem** 必须取值为 **id-pr-administratorImposedLimit**；

注 — 这并不意味着要求某种实现对实施管理限制的主管部门具有客户化的能力。

b) 如果限制是由某个实现约束引起的，且问题被认为是一个永久性的问题，则通知属性 **dSAProblem** 必须取值为 **id-pr-permanentRestriction**；

c) 如果限制是由某个实现约束引起的，且问题被认为是一个临时性的问题，如临时拥塞，则通知属性 **dSAProblem** 必须取值为 **id-pr-temporaryRestriction**。

16.1.4.5 尺寸限制

在 **CommonArguments** 中规定的尺寸限制可能在列表或搜索操作处理过程中的任意时间点被越限。在这种情况下，必须向请求者返回一个部分结果（从已经收集的结果集中得到），且 **limitProblem** 被设置为 **sizeLimitExceeded**。另外，可能会使用 **unexplored** 组件来返回未访问到的 DSA 的继续引用。

如果这是一个搜索操作，且设置了 **entryCount** 搜索控制选项，则 DSA 必须作一个最佳估计，即在考虑访问控制但不是分等级选择时，如果没有尺寸限制，可能会有多少个条目被潜在地返回，然后如果没有**尚未访问到**的 DSA，则使用 **bestEstimate** 选择，并将此数值在 **PartialOutcomeQualifier** 的 **entryCount** 组件中返回，否则它必须使用 **lowEstimate** 选择。

然后操作调度程序被终止。

16.2 操作调度程序的规程

操作调度程序为处理每个接收到的请求（通过 DAP、LDAP 或 DSP）而执行的规程由下面的步骤来定义。由于有别名解除引用，该规程也可能调用自己（一个本地请求），在这种情况下，将返回一个本地答复（而不是一个 DAP、LDAP 或 DSP 答复）。

1) 对操作变量的多个方面进行合法性验证（**请求合法性验证**规程）。如果在合法性验证过程中遇到一个错误，则通过本地方式或通过 DAP/LDAP/DSP 返回此错误。

2) 如果接收到的操作是一个放弃操作，则调用**放弃**规程，并随后返回一个答复。

3) 通过执行**发现 DSE** 规程来对目标对象的名字进行解析（该规程中包括**目标发现**和**目标未发现**子规程）。如果被请求的条目被发现且适用（根据服务控制、链接变量和本地策略决策等的设置），则继续第 6 步的**评估阶段**。如果在名字解析过程中遇到一个错误，则被返回。如果条目被发现但不适用，继续执行步骤 4。

4) 调用**名字解析继续引用**规程来处理 **NRcontinuationList** 中存储的继续引用列表。为了处理这些继续引用，可能会向其他 DSA 发起链接请求（如果服务控制和本地策略决策允许的话）。

   在出现错误的情况下，该错误或者通过本地方式，或者通过 DAP/LDAP/DSP 被直接返回。如果链接请求产生了一个结果，则继续执行步骤 5。

5) 调用**结果合并**规程来将本地结果与接收到的链接结果合并起来。如果链接结果中包含内嵌的继续引用，则它们可能首先被解析，若服务控制和本地策略允许或要求的话。

   这样可能会导致发起附加的链接请求（其链接结果也可能包含内嵌的继续引用）。

   合并后的结果被返回给调用者，同时对请求的处理停止。

   如果对结果执行了保护，则不能对结果执行合并。

6) 如果操作是一个修改操作，继续执行步骤 7。

   如果操作是一个单独条目查询操作，则继续执行步骤 8。

   如果操作是一个多条目查询操作，则继续执行步骤 9。

7) 在执行一个修改规程时，作为操作执行的结果，操作绑定可能需要被建立、修改或终止，或者镜像可能需要被更新等。这些工作与初始操作的执行是同步进行的还是异步进行的，依赖于不同的修改操作（以及本地策略）。一个本地或 DAP/LDAP/DSP 结果或错误被返回给调用者。

8) 单独条目查询操作的结果，作为一个本地或 DAP/LDAP/DSP 结果被直接返回给调用者。

9) 如果操作是一个多条目查询操作，则检查操作的 **nameResolutionPhase**。如果它的值不是 **completed**，则分别调用**列表(I)** 规程或**搜索(I)**规程，否则分别调用**列表(II)**或**搜索(II)**规程。

10) 调用**列表(II)**规程的输出（结果或错误）以及调用**列表(I)**规程的输出（在这种情况下，该结果为一个错误）能够直接返回给调用者（作为一个本地结果或 DAP/LDAP/DSP 结果）。

如果被调用的规程是**列表(I)**规程，则结果中可能包含继续引用，该继续引用必须被解除引用（依赖于服务控制和本地策略）。这可能会导致链接列表操作被发送到不同的 DSA。为了合并结果，继续执行第 5 步骤，并呼叫**结果合并**规程。

11) 如果操作是一个搜索操作，则任何继续引用都被**搜索继续引用**规程所解析（如果被要求或被允许的话）。这可能会导致链接搜索请求被发送到不同的 DSA。**结果合并**规程（见第 5 步骤）被调用来合并搜索结果，同时可能对包含的继续引用进行解除引用，如果有的话。

16.3 规程概述

本节对**操作调度程序**所部署的规程的基本功能给出了概述，这些规程在第 17 节到 22 节中定义。

16.3.1 请求合法性验证规程

该规程在第 17 节描述，用于在执行本地名字解析之前执行环回检测、限制检测和安全检测等。如果请求是来自 DUA 或 LDAP 客户机，且 DAP 或 LDAP 客户机没有提供 **ChainingArgument** 中的某些参数的值时，该规程还为这些参数提供缺省的设置。另外，该规程挑选任意的 **abandon** 请求，并且将其通知给**操作调度程序**。

16.3.2 放弃规程

该规程在第 20.5 节描述，试图发现那些被放弃的操作，并且终止这些操作。如果有任意一个正在进行的子请求，则可能在发出请求后发送链接放弃操作。该规程可能向调用者返回一个空结果，或者返回一个错误指示（例如，问题为 **tooLate** 的 **abandonError**）。

16.3.3 发现DSE规程

该规程在第 18.2 节和 18.3 节描述，对目标对象的名字组件与本地拥有的 DSE 进行匹配，以便解析目标对象名字。如果遇到一个别名 DSE，则该别名被解除引用（如果允许的话），且该规程被重新执行来解析新的名字。

如果目标对象没有被发现，则该规程继续执行**目标未发现**子规程。如果目标对象被发现，则该规程继续执行**目标发现**子规程。

注 — **目标未发现**和**目标发现**是**发现 DSE** 规程的继续。

该规程可能会引起各种错误，在这种情况下，相关的协议错误将返回给请求者，且**操作调度程序**被终止。

16.3.3.1 目标未发现子规程

该规程在第 18.3.2 节中定义，对已定位的中介 DSE 执行评估，并且基于在**发现 DSE** 规程中检测出的知识引用集，在 **NRcontinuationList** 中创建一个继续引用集。然后，该引用集在**名字解析继续引用**规程中被进行后续处理。

该规程可能会引起各种错误，在这种情况下，相关的错误将返回给请求者，且**操作调度程序**被终止。

16.3.3.2 目标发现子规程

该规程在第 18.3.3 节中定义，检查已经发现的 DSE 是否适合于所请求的操作，即在它是镜像信息的情况下。在多对象操作的情况下（如子树搜索），这可能会包括对目标对象下的整个镜像信息子树的适宜性进行检查。

如果被定位的条目是适合的，则将调用适当的操作评估规程。否则，一个指向信息的提供者（或属主）的 **ContinuationReference** 将在 **NRcontinuationList** 中创建，且**名字解析继续引用**规程被调用。

16.3.4 单条目查询规程

该规程在第 19.2 节描述，被调用来实际地执行那些仅影响一个单独条目的操作，如阅读和比较操作。在操作完成后，该规程产生的一个答复（结果或错误）将被返回给发起请求的 DSA/DUA/LDAP 客户机。

16.3.5 修改规程

这些规程在第 19.1 节描绘，被调用来处理修改操作，即增加条目、删除条目，修改条目和修改 DN 等。这是通过执行为每个这样的操作定义的特定子规程来完成的。在这些子规程过程中（或结束后），可能会向其他 DSA 发起 DOP 和 DISP 请求。在成功完成后，一个结果（由子规程创建）被返回给发起请求的 DSA/DUA/LDAP 客户机。

16.3.6 多条目查询规程

这些规程在第 19.3 节描述，被调用来处理影响多个条目的操作，这些条目可能位于，也可能不位于同一个 DSA 内。这是通过执行为了完成请求分解，而为每个搜索和列表操作定义的特定子规程来完成的。这些规程创建了操作评估的一个本地结果，并且可选地，在 **SRcontinuationList** 中创建了一系列继续引用。如果在本规程结束时，**SRcontinuationList** 为空，则被创建的结果将直接返回给请求的 DSA/DUA/LDAP 客户机。如果这是一个搜索操作，且结果为空并且变量 **emptyHierarchySelect** 被设置，则在 **PartialOutcomeQualifier** 的 **notification** 组件中返回：

— 一个 **searchServiceProblem** 通知属性，且取值为 **id-pr-emptyHierarchySelection**。

如果 **SRcontinuationList** 不为空，则根据操作类型，通过调用**列表**或**搜索继续引用**规程来处理这些继续引用。

16.3.7 名字解析继续引用规程

该规程在第 20.4.1 节中描述，对在名字解析阶段创建的 **NRcontinuationList** 中的继续引用进行处理。这些继续引用或者被用来发起链接子请求，或者在一个提名错误中被返回。在链接的情况下，从链接请求返回的结果或错误被返回，以便由**结果合并**规程作进一步的处理。

16.3.8 列表和搜索继续引用规程

这些规程在第 20.4.2 节和 20.4.3 节中描述，对在多条目查询规程中创建的 **SRcontinuationList** 中的继续引用进行处理，或者通过发起链接子请求来完成对它们的解析，或者是在 **partialOutcomeQualifier.unexplored** 内创建相应继续引用。当接收到所有正在进行的子请求的结果或错误时，它们将被返回，以便由**结果合并**规程作进一步的处理。

16.3.9 结果合并规程

该规程在第 21 节描述，或者检查从链接请求返回的结果，或者将本地操作结果与从链接子请求中接收到的结果进行组合。如果一个子请求返回了一个错误，则该规程将决定此错误将被如何处理。

如果在结果中还留有继续引用，则它们将（如果本地策略允许这样，且服务控制要求这样）相应地被**名字解析继续引用**、**列表继续引用**或**搜索继续引用**等规程来解除引用。如果没有被签名的话，重复的信息将从结果中删除。

合并后的结果（包括所有的合并结果和未解析的继续引用）被返回给发起请求的 DUA/LDAP 客户机/DSA。

如果对结果执行了保护，则不应对结果执行合并。

17 请求合法性验证规程

17.1 引言

请求合法性验证规程是操作调度程序对于来自 DUA、LDAP 客户机和 DSA 的输入的入口点，为这些输入进行名字解析处理作准备。该规程的功能包括检测放弃操作；执行安全检查；调整从 DUA 或 LDAP 客户机接收到的

输入，使得它们可能与从 DSA 接收到的输入一样以相同的方法进行处理；检查请求中变量的合法语法和语义；执行环回检测；以及执行其他各种检查等。**请求合法性验证**的流程如图 8 所示。

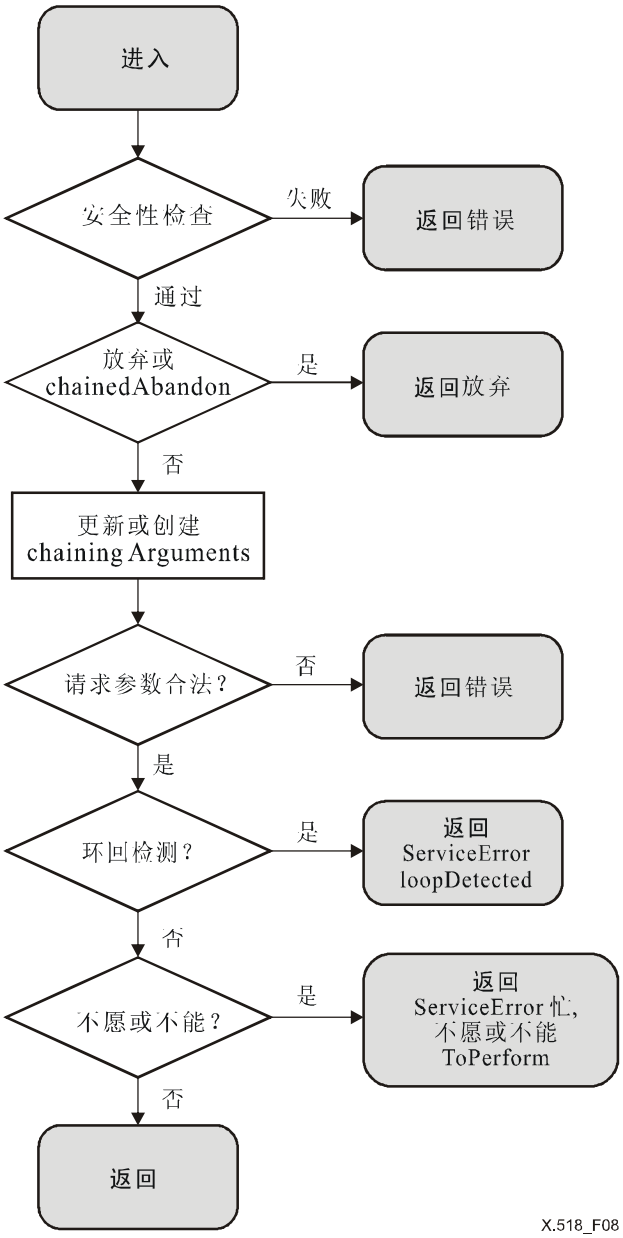


图 8－请求合法性验证规程

17.2 规程参数

17.2.1 变量

请求合法性验证的输入变量包括 **ChainingArguments**（除了在 **chainedAbandon** 操作的情况下），请求是否是从 DSA 处接收到，以及请求的发起者所给出的变量等。

17.2.2 结果

请求合法性验证的输出结果包括五种可能性。

a) 如果安全检查失败，则向请求者返回一个错误。

b) 如果输入是一个 **abandon** 或 **chainedAbandon** 操作，则输出为该操作的变量。

c) 如果请求的变量不合法，则向请求者返回一个错误。依赖于本地策略，DSA 可能选择或者返回一个 **serviceError**，或者返回一个 **securityError**。

d) 如果检测到一个环回，则向请求者返回一个问题为 **loopDetected** 的 **serviceError**。

e) 基于资源问题或策略方面的考虑，如果 DSA 不能或不愿意执行操作，则向请求者返回一个 **serviceError**（问题为 **busy**，**unavailable**，或 **unwillingToPerform**）。如果相关，则可能返回一个问题为 **dataSourceUnavailable** 的 **serviceError**。

f) 在所有其他情况下，如果输入是从 DUA 或 LDAP 客户机接收到的，则合法性验证后的输入通过加入 **ChainingArguments** 而被传送；如果输入是从 DSA 接收到的，则合法性验证后的输入通过更新 **ChainingArguments.traceInformation** 而被传送，这些合法性验证后的输入是本规程的输出，并且随后将作为**名字解析**规程的输入。

17.3 规程定义

17.3.2 节中描述的安全检查被执行。这可能会导致返回一个错误，并且终止该操作调度程序。

如果输入是一个 **abandon** 或 **chainedAbandon** 操作，则后续仅执行 17.3.1 节中定义的步骤，否则 17.3.3-17.3.5 节中定义的步骤都被执行。17.3.5 小节描述了环回检测规程，该规程可能会导致返回一个错误，并且终止该操作调度程序。

其次，17.3.6 节中描述的检查被执行。它们可能会导致返回一个错误，并且终止该操作调度程序。

如果 17.3.2-17.3.6 节中描述的检查没有导致操作调度程序的终止，则 17.3.7 节中定义的步骤被执行，且其结果被传送到**名字解析**规程，本规程终止。

17.3.1 放弃处理

一个 **abandon** 或 **chainedAbandon** 的变量被传递到**放弃**规程（见 20.5），来处理放弃请求。

17.3.2 安全检测

如果操作的变量被签名、加密或签名并加密，则签名可能被检测。如果签名不合法，或者加密失败，或者必须存在的时候没有存在，则会向请求者返回一个错误。可选地，一个 DSA 还可能会执行任何其他本地定义的动作。

17.3.3 输入准备

17.3.3.1 DUA或LDAP客户机请求

如果操作是从一个 DUA 或 LDAP 客户机处接收到的，则被创建的值 **ChainingArguments** 如下所述：

a) **ChainingArguments.originator** 按照 10.3 节的描述被设置。

b) **ChainingArguments.operationProgress** 被设置为 **CommonArguments.operationProgress** 的值。

c) **ChainingArguments.traceInformation** 被设置为一个序列，该序列包含一个单独的 **TraceItem** 值。该值的构造如下所述。**TraceItem.dsa** 被设置为执行**请求合法性验证**的 DSA 的名字。**TraceItem.targetObject** 必须被忽略。**TraceItem.operationProgress** 被设置为输入值。

d) 如果操作的服务控制规定了一个时间限制（操作完成的可用时间段，以秒计数），则 **ChainingArguments.timeLimit** 被设置为该（UTC）时间，到该时间为止操作必须被完成以满足用户指定的时间限制。

e) **ChainingArguments.AuthenticationLevel** 和 **ChainingArguments.UniqueIdentifier** 根据本地安全策略被设置。

f) **ChainingArguments.nameResolveOnMaster** 拷贝自 **CommonArguments.nameResolveOnMaster**。

g) **ChainingArguments.exclusions**, **ChainingArguments.entryOnly** 和 **ChainingArguments.referenceType** 拷贝自 **CommonArguments.exclusions**, **CommonArguments.entryOnly** 和 **CommonArguments.referenceType**，如果它们都存在的话，否则它们被忽略。

h) 如果 **manageDSAIT** 选项在 **ServiceControls** 中被设置，则：

— **operationProgress** 的组件 **nameResolutionPhase** 必须被设置为 **completed**；

— **operationProgress** 的组件 **nextRDNToBeResolved** 必须被忽略；

— **referenceType** 必须取值为 **self**；

— **entryOnly** 必须取值为 **FALSE**；

— **nameResolveOnMaster** 必须取值为 **FALSE**；且

— **ServiceControls** 中的 **chainingProhibited** 选项必须被设置；

— **ChainingArguments** 中的剩余可选元素被忽略，如果指定则假设为缺省值。

i) 如果在 **ServiceControls** 中没有设置 **manageDSAIT** 选项，则 **ChainingArguments** 中的剩余可选元素被忽略，如果指定则假设为缺省值。

j) **ChainingArguments.SecurityParameters.ProtectionRequest** 被用来指示准备应用于结果的保护级别（签名、加密、或签名并加密）。

17.3.3.2 LDAP请求

如果操作是从一个 LDAP 客户机处接收到的，则创建的 **ChainingArguments** 值如 17.3.3.1 节中所描述的，但有一个例外，即 **ChainingArguments.operationProgress** 必须被设置为 **nameResolutionPhase notStarted**，且 **ChainingArguments.exclusions**，**ChainingArguments.entryOnly**，和 **ChainingArguments.referenceType** 的值必须被忽略。

17.3.3.3 DSA请求

如果操作是从一个 DSA 处接收到的，则 **ChainingArguments.traceInformation** 的值被更新，这是通过在 **TraceItem** 序列的结尾处附加一个值实现的。该值的构成如下所述：

a) **TraceItem.dsa** 被设置为执行**请求合法性验证**的 DSA 的名字。

b) **TraceItem.targetObject** 被设置为 **ChainingArguments.targetObject** 的值，除非请求变量的 **object**（或搜索操作中的 **baseObject**）与 **ChainingArguments.targetObject** 相同，在这种情况下，**TraceItem.targetObject** 必须被忽略。

c) **TraceItem.operationProgress** 被设置为 **ChainingArguments.operationProgress** 的值。

如果操作是从一个 DSA 处接收到的，且如果 **ChainingArguments.streamedResults** 包含一个大于或等于 1 的值，则当且仅当 DSA 理解流结果，并且愿意为此操作接收流结果时，才为 **ChainingArguments.streamedResults** 的值加 1。

17.3.4 合法性声明

操作变量的语法和语义的合法性必须被检查，这是根据在定义每个操作的章节中所包含的规则来实施的（例如，必须检查 **nextRDNToBeResolved** 不能提供一个超出了 **targetObject** 中的 RDN 数量的数目）。如果请求被检测出包含了非法变量，则操作被终止，并向用户返回一个错误，依赖于所检测出的非法类型。

17.3.5 环回检测

如果 **ChainingArguments.traceInformation** 中的任意两个 **TraceItem** 值（如 17.3.3 节中准备的那样）是相同的，则操作的处理将返回到一个之前的状态，即检测出一个环回。在这种情况下，必须向请求者返回一个 **serviceError**（问题为 **loopDetected**），且**操作调度程序**终止。

17.3.6 不能或不愿执行

请求合法性验证可能会评估可用的资源，并决定操作不能被执行。它还可能会基于策略的考虑，决定操作不应当被执行。在这些情况下，可能会向请求者返回一个 **serviceError**（问题为 **busy**，**unavailable**，或 **unwillingToPerform**），且操作调度程序终止。

如果一个 DSA 通过本地方式能够判断出问题是与本地 DIB 资源的不可用相关的，则它必须发送一个问题为 **unavailable** 的 **serviceError**，且数据类型 **CommonResults** 的 **notification** 组件中必须包含：

— 一个 **dSAProblem** 通知属性，且取值为 **id-pr-dataSourceUnavailable**；以及

— 一个 **distinguishedName** 属性，取值为 DSA 的识别名。

17.3.7 输出处理

在**请求合法性验证**的最后一个阶段，如果输入是从 DUA 或 LDAP 客户机接收到的，则合法性验证后的输入将通过加入 **ChainingArguments** 中而被传送；如果输入是从 DSA 接收到的，则合法性验证后的输入将通过更新 **ChainingArguments.traceInformation** 而被传送，这些合法性验证后的输入被返回，并且随后将作为**名字解析**规程的输入。

18 名字解析规程

18.1 引言

本节描述了**名字解析**规程，以及它的变量、结果和可能的错误条件。如图 6 中所示（**操作调度程序**），**名字解析**规程包含两个规程：

— **发现 DSE** 规程；

— **名字解析继续引用**规程。

发现 DSE 规程在三个流程图中描述，分别为**发现 DSE**，**目标发现**和**目标未发现**。**发现 DSE** 规程将目标条目名字与本地存储的 DSE 进行匹配，一个组件接着一个组件地进行。如果目标条目在本地发现，则**发现 DSE** 继续执行**目标发现**子规程，然后调用**检查适宜性**规程来检测所发现的 DSE 是否适合评估。如果目标条目没有在本地发现，则**发现 DSE** 规程继续执行**目标未发现**子规程，准备要加入到 **NRcontinuationList** 中的继续引用，然后由**名字解析继续引用**规程来调度。

注 1 — 当判断一个匹配时，**名字解析**必须对多个由上下文所区分的识别值执行名字匹配，如在 ITU-T X.501 建议书 | ISO/IEC 9594-2 的 9.4 节中描述的那样。

注 2 — 如果一个第三版本之前的上级 DSA 拥有一个下级引用，其指向的条目由一个后续版本的 DSA 所拥有，且该条目的 RDN 中包含有上下文，则**名字解析**可能会失败。当一个可替代名字被用作声称的名字，而镜像条目由一个第一版本或第二版本的 DSA 所拥有时，则针对此条目的镜像拷贝的**名字解析**会失败。

18.2 发现DSE规程参数

18.2.1 变量

本规程使用如下变量：

a) **ChainingArguments.traceInformation;**
b) **ChainingArguments.aliasDereferenced;**
c) **ChainingArguments.aliasedRDNs;**
d) **ChainingArguments.excludeShadows;**
e) **ChainingArguments.nameResolveOnMaster;**
f) **ChainingArguments.operationProgress (nameResolutionPhase, nextRDNToBeResolved);**
g) **ChainingArguments.referenceType;**
h) **ChainingArguments.targetObject;**
i) **ChainingArguments.relatedEntry;**
j) **ChainingArguments.streamedResults;**
k) 操作类型；
l) 操作变量。

注 — 如果没有实际值存在时，将使用缺省的或隐含的值，如 10.3 节中的规定。

18.2.2 结果

从**发现 DSE** 有两类成功的输出（分别由**条目适合**或**条目不适合**来指示）：

第一种成功的情况是在 **NRcontinuationList** 中返回继续引用（从**目标未发现**子规程中返回），然后该继续引用被传递到**名字解析继续引用**规程，继续执行名字解析阶段。

第二种成功的情况是返回一个指向 DSE 的引用（从**目标发现**子规程中返回），该引用被继续传递到某个评估规程。

18.2.3 错误

下列错误可能被返回：

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

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

18.2.4 全局变量

本规程使用下列全局变量：

— **NRcontinuationList** 列表存储了在**名字解析继续引用**规程中继续进行名字解析所需要的继续引用。

— **StreamedResultsOK** 存储了一个决定，即该 DSA 是否可能在此操作的响应中链接流结果。

18.2.5 本地和共享变量

该规程使用下列本地变量：

a) **i** 索引，用于标识正在被操作的目标名字的组件。

b) **m** 目标对象名字的长度，在名字解析中使用。对于那些名字要解析到其父条目的操作，即增加条目操作，**m** 被设置为（目标对象中 RDN 的个数）－1。对于所有的其他操作，**m** 被设置为目标对象中的 RDN 的个数。

c) **lastEntryFound** 索引，表示此 DSE(lastEntryFound)是最后一个匹配的类型为 **entry** 的 DSE。

d) **lastCP** 索引，表示此 DSE(lastCP)是所遇到的最后一个被镜像的上下文前缀。

e) **candidateRefs** 继续引用的集合。

共享变量 **admPoints**（在操作调度程序中定义）也被使用。为了简便，目标对象名字中的组件 **i** 表示为 **N(i)**。

18.3 规程

注 — 在流程图中有一些仅与特定操作相关的文字描述。这没有在流程图中显示，但是在相应的文字中进行了描述。

18.3.1 发现DSE规程

见图 9。

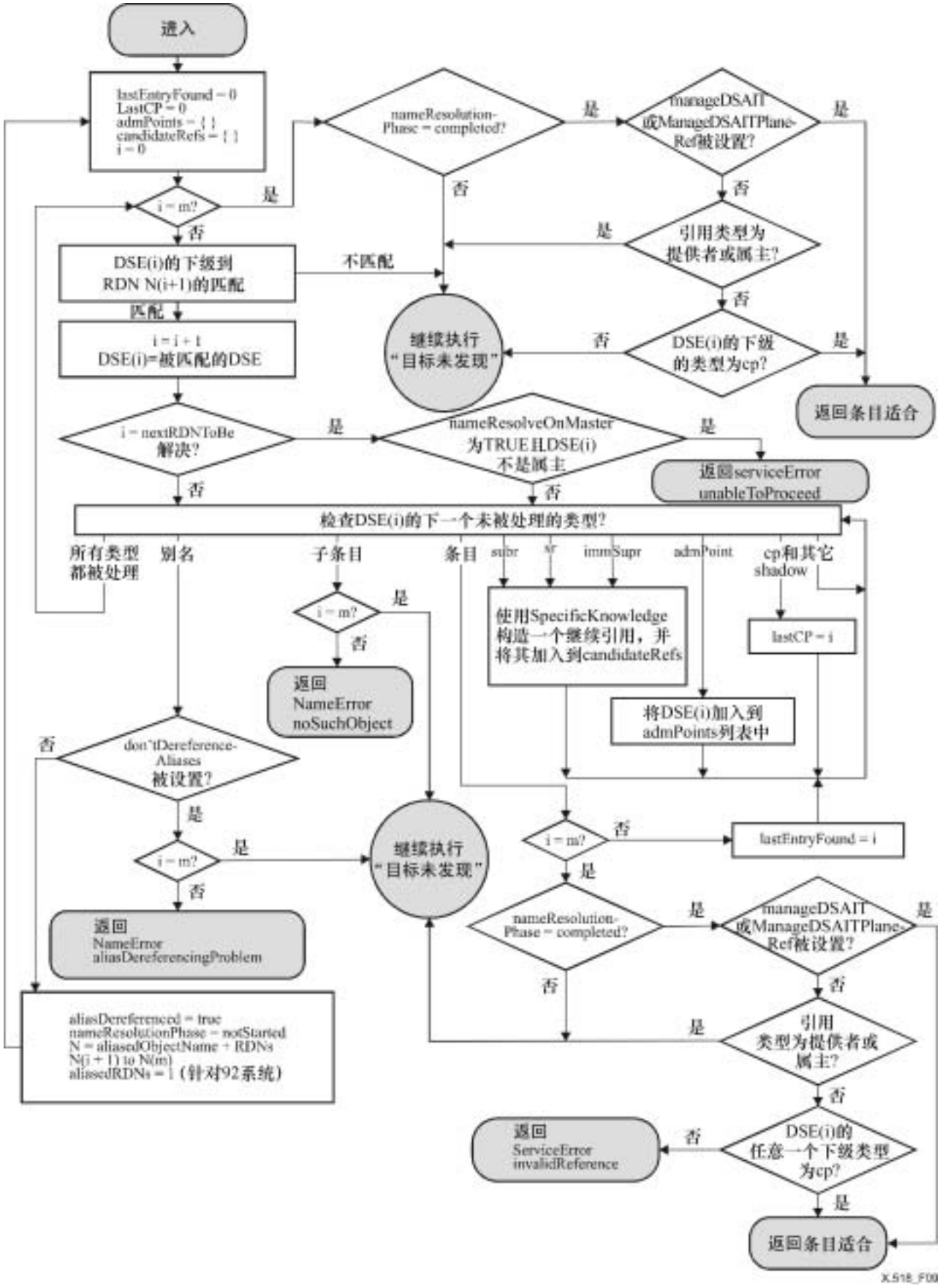


图 9－发现DSE规程

目标对象名字的判断如下所述：

a) 如果在 **ChainingArguments** 中存在 **targetObject**，则使用该组件的值。

b) 如果在 **ChainingArguments** 中存在 **relatedEntry**，而不是 **targetObject**，则使用 **relatedEntry** 所标识的 **JoinArgument** 中的 **baseObject** 组件。

注 1 — 这仅与某个受保护的搜索请求相关。

c) 如果在 **ChainingArguments** 中既没有存在 **relatedEntry**，也没有存在 **targetObject**，则使用操作变量中的 **base**（**baseObject**）组件。

本规程试图在本地解析目标对象名字。

1) 初始化本地变量 **lastEntryFound** 和 **lastCP** 为 **0**；**admPoints** 和 **candidateRefs** 为空集，且初始化 **i** 为 **0**。

2) 比较 **i** 和 **m**。如果它们不相等，则继续执行步骤 5)。

3) 如果它们相等，则检查 **nameResolutionPhase** 是否为 **completed**。如果不是 **completed**，则继续执行**目标未发现**子规程。

如果 **nameResolutionPhase** 的值为 **completed**，且 **manageDSAIT** 关键扩展被设置，则返回**条目适合**。

4) 如果 **nameResolutionPhase** 的值为 **completed**，则检查 DSE(i)的任意一个直接下级是否是一个上下文前缀（类型为 **cp**）。

— 如果一个（或多个）直接下级 DSE 的类型为 **cp**，则返回**条目适合**。

注 2 — 这种情况针对**列表 (II)**和**搜索(II)**子请求。

— 如果没有直接下级 DSE 的类型为 **cp**，则继续执行**目标未发现**子规程。

5) 尝试在目标对象名字的第**(i + 1)**个组件与最后匹配的 DSE 的某个下级名字之间找到一个匹配。在 i = 0 的情况下，尝试与作为根 DSE 直接下级的某个 DSE 进行匹配。如果没有发现匹配，则继续执行**目标未发现**子规程。如果发现了一个单独的匹配，则 **i** 增加 1，并且在已发现的 DSE 的向量中增加该匹配的 DSE，作为第 **i** 个元素。

注 3 — 名字匹配包括对已知的，由上下文所区分的多个识别名的处理， 如在 ITU-T X.501 建议书 | ISO/IEC 9594-2 的 9.4 节中的描述。

如果发现了多个匹配，则返回一个问题为 **contextProblem** 的 **nameError**。

注 4 — 例如，可能是这样一种情况，当一个声称的名字中的 **AttributeTypeAndDistinguishedValue** 包含了由上下文所区分的多个识别属性值，且这些不同的值与不同目标名字中的值相匹配。

6) 如果 **i** 等于 **nextRDNToBeResolved**，则检查下面的两个条件是否都满足：

— **ChainingArgument.nameResolveOnMaster** 取值为 **TRUE**；

— DSE(**i**)不是一个主条目。

如果这两个条件都满足，则返回一个问题为 **unableToProceed** 的 **serviceError**。

注 5 — 这指示了使用 **nameResolveOnMaster** 可以避免对同一目标对象出现多条路径。

7) 检查 DSE(**i**)中的所有 DSE 类型比特。对于每一个类型比特，需要某些潜在的处理。为每种发现的类型要执行的动作如下所述：

— 如果 **cp** 和 **shadow** 比特都被设置，则记住 **lastCP** 中的索引 **i**。

— 如果 **admPoint** 比特被设置，则检查 **administrativeRole** 操作属性。如果这是一个自治管理区的开始，则清空 **admPoints** 列表。如果这是一个或多个特定管理区的开始，则检查 **admPoints** 列表，并删除任何已存在的但不再相关的点（即它们的作用已经被新的管理点所替代）。在列表中存储 DSE (**i**)。

— 如果 **subr**，**xr**，**immSupr**，或 **ditBridge** 比特中的其中一个被设置，则产生一个继续引用，方法是使用 **specificKnowledge** 属性，其中 **operationProgress.nameResolutionPhase** 被设置为 **proceeding**，**nextRDNToBeResolved** 被设置为 **i**，**targetObject** 是由使用主 RDN（可替代识别值可能也包含在 RDN 中）的已解析组件与剩余的尚未解析的组件级联起来构成的，且 **accessPoints** 和 **referenceType** 都被适当地设置。在 **candidateRefs** 的继续引用列表中增加该继续引用。

— 如果 **entry** 比特被设置，则检查 **i** 是否等于 **m**（因此目标对象名字正在被完整匹配）。如果 **i** 不等于 **m**，则通过将 **lastEntryFound** 设置为 **i** 而记住此发现的条目，并且继续处理 DSE(**i**)的类型比特。如果 **i** 与 **m** 相等，则继续执行步骤 8)。

— 如果 **subentry** 比特被设置，则检查 **i** 是否等于 **m**（因此目标对象名字正在被完整匹配）。如果它们相等，则继续执行**目标发现**规程；如果它们不相等，则返回一个问题为 **noSuchObject** 的 **nameError**。

— 如果 **alias** 比特被设置，则检查 **dontDereferenceAliases** 是否被设置。

— 如果 **dontDereferenceAliases** 未被设置，则别名可被解除引用。因此，设置 **chainingArguments.aliasDereferenced** 为 **TRUE**，**nameResolutionPhase** 为 **notStarted**，且目标对象的名字为别名条目所提供的 **aliasedEntryName** 与前一个目标对象名字中的剩余未匹配组件级联起来组成（即，与前一个目标对象名字中的第 **(i + 1)**个到第 **m** 个组件级联起来）。第二版本和后续版本的 DSA 不设置 **aliasedRDNs**（反之，第一版本的 DSA 将 **aliasedRDNs** 设置为 **aliasedEntryName** 中的 RDN 的个数）。 再次开始名字解析，继续执行步骤 1)。

如果 **dontDereferenceAliases** 被设置，则别名不能被解除引用。通过比较 **i** 和 **m** 是否相等，来检查目标对象名字是否已经被完整地处理。如果它们相等（因此名字完全匹配），则继续执行**目标发现**子规程。如果它们不相等（因此名字不是完全匹配），则返回一个问题为 **aliasDereferencingProblem** 的 **nameError**。

— 对于其他所有可能的 DSE 类型，不需要任何其他动作。在内部标注已经处理过的 DSE 类型，并且继续处理 DSE(**i**)中尚未处理的 DSE 类型比特。

— 如果 DSE(**i**)的所有类型比特都被处理过，则继续步骤 2)。

8) 检查 **nameResolutionPhase** 是否为 **completed**。如果不是，则继续执行**目标发现**子规程。

9) 如果 **nameResolutionPhase** 已经为 **completed**，且 **manageDSAIT** 关键扩展被设置，则返回**条目适合**。

10) 否则，检查作为 DSE(**i**)直接下级的任何 DSE 是否是一个上下文前缀（即类型为 **cp**）。如果有一个或多个，则返回**条目适合**。如果没有直接下级条目的类型为上下文前缀，则返回一个问题为 **invalidReference** 的 **serviceError**。

注 6 — 这种情况是针对**列表(II)**和**搜索(II)**子请求。

18.3.2 目标未发现子规程

见图 10。

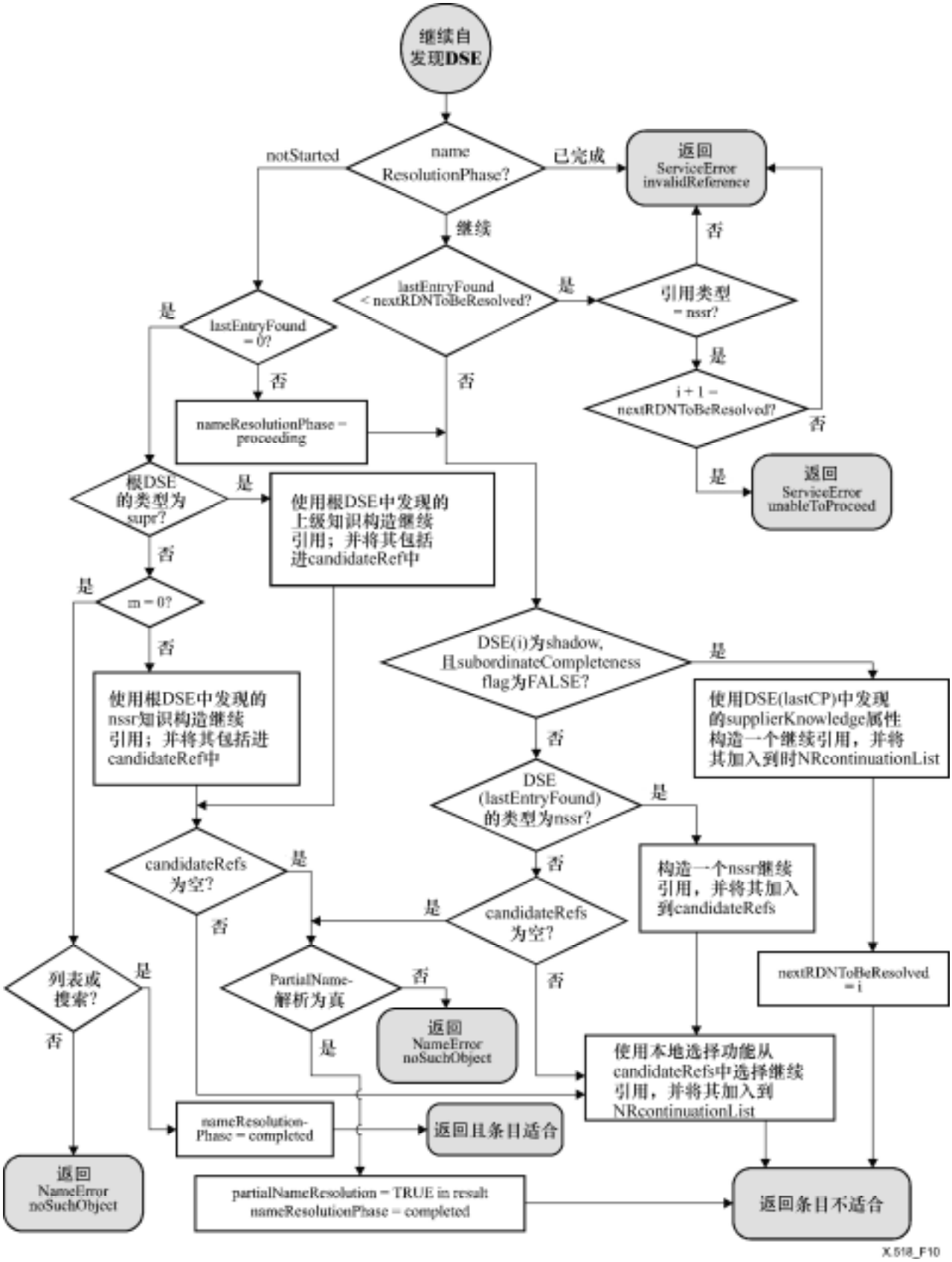


图 10－目标未发现子规程

当目标对象名字没有在本地 DSA 中发现时，调用本子规程。本子规程将决定用来继续进行名字解析的知识引用的最佳类型，除非检测到一个错误，在这种情况下，将返回错误。

1) 当继续自**发现 DSE** 规程时，要区分名字解析阶段的三个可能的阶段：

   — 如果 **nameResolutionPhase** 为 **notStarted**，则继续执行步骤 2)。
   — 如果 **nameResolutionPhase** 为 **proceeding**，则继续执行步骤 8)。
   — 如果 **nameResolutionPhase** 为 **completed**，则继续执行步骤 12)。

2) 如果发现了一个条目（**lastEntryFound** 不等于 **0**），则设置 **nameResolutionPhase** 为 **proceeding**，并且继续执行步骤 9)。

3) 如果没有发现条目（**lastEntryFound=0**），则检查该 DSA 是否为第一级的 DSA。

   如果它是第一级的 DSA，则根 DSE 不包含一个上级引用，因此其类型不是 **supr**。在这种情况下，继续执行步骤 4)。

   如果该 DSA 不是第一级的 DSA，则根 DSE 包含一个上级引用，因此其类型为 **supr**。在这种情况下，使用在根 DSE 中找到的上级知识产生一个继续引用，设置：

   — **targetObject** 为目标对象的名字，该名字是使用主 RDN（可替代识别值可能也包含在 RDN 中）的已解析组件与剩余的尚未解析的组件级联起来构成的；
   — **operationProgress.nameResolutionPhase** 为 **notStarted**；
   — **referenceType** 为 **superior**；以及
   — 适当的 **accessPoints**。

   在 **candidateRefs** 的继续引用列表中增加该继续引用。继续执行步骤 6)。

4) 检查该操作是否被直接引导到根条目 (**m = 0?**)。如果是，则继续执行步骤 5)。如果不是，则使用根 DSE 中找到的任意 NSSR 知识产生一个继续引用，设置：

   — **targetObject** 为目标对象的名字，该名字是使用主 RDN（可替代识别值可能也包含在 RDN 中）的已解析组件与剩余的尚未解析的组件级联起来构成的；
   — **operationProgress.nameResolutionPhase** 为 **proceeding**；
   — **operationProgress.nextRDNToBeResolved** 为 **1**；
   — **referenceType 为 nonSpecificSubordinate**；以及
   — 适当的 **accessPoints**。

   在 **candidateRefs** 的继续引用列表中增加该继续引用。继续步骤 6)。

5) 在第一级的 DSA 中，仅有列表或搜索操作可能将根条目作为基对象来执行。因此，如果操作不是一个列表或搜索操作，则返回一个问题为 **noSuchObject** 的 **nameError**。如果是一个列表或搜索操作，则设置 **nameResolutionPhase** 为 **completed**，并返回，且为**条目适合**。

6) 检查在 **candidateRefs** 中是否有任意一个继续引用。如果 **candidateRefs** 为空且 **partialNameResolution** 为 **FALSE**，则返回一个问题为 **noSuchObject** 的 **nameError**。如果 **candidateRefs** 为空且 **partialNameResolution** 为 **TRUE**，则在结果中，设置 **partialName** 为 **TRUE**，**nameResolutionPhase** 为 **completed**，并返回，且为**条目适合**。否则继续执行步骤 7)。

7) 使用一个本地选择功能从 **candidateRefs** 的继续引用列表中选择一个继续引用，并将其增加到 **NRcontinuationList** 的继续引用列表中，并返回，且为**条目不适合**。

8) 如果 DSA 不能继续执行名字解析（在这种情况下，**lastEntryFound** 小于 **nextRDNToBeResolved**），则继续执行步骤 11)。否则继续下一步骤。

9) 如果 DSE(**i**)是一个镜像 DSE，且具有不完整的下级知识（**subordinateCompletenessFlag** 取值为 **FALSE**），则根据从 DSE（**lastCP**）中找到的 **supplierKnowledge** 属性产生一个继续引用。设置：

   — **targetObject** 为目标对象的名字，该名字是使用主 RDN（可替代识别值可能也包含在 RDN 中）的已解析组件与剩余的尚未解析的组件级联起来构成的；

— **operationProgress.nameResolutionPhase** 为 **proceeding**；

— **operationProgress.nextRDNToBeResolved** 为 **lastEntryFound**；

— **referenceType** 为 **supplier**；以及

— 适当的 **accessPoints**。

在 **NRcontinuationList** 的继续引用列表中增加该继续引用，并返回，且为**条目不适合**。

10) 如果最后发现的条目包含一个 NSSR（即 **DSE(lastEntryFound)**的类型为 **nssr**），则根据从 **DSE(lastEntryFound)**中发现的 NSSR 知识产生一个继续引用。设置：

— **targetObject** 为目标对象的名字，该名字是使用主 RDN（可替代识别值可能也包含在 RDN 中）的已解析组件与剩余的尚未解析的组件级联起来构成的；

— **operationProgress.nameResolutionPhase** 为 **proceeding**；

— **operationProgress.nextRDNToBeResolved** 为 **lastEntryFound+1**；

— **referenceType** 为 **nonSpecificSubordinate**；以及

— 适当的 **accessPoints**。

在 **candidateRefs** 的继续引用列表中增加该继续引用。继续步骤 7)。

如果 **DSE(lastEntryFound)**不是类型 **nssr**，则继续步骤 6)。

11) 如果 **chainingArguments.referenceType** 是类型 **nssr**，则继续步骤 13)，否则继续步骤 12)。

12) 返回一个问题为 **invalidReference** 的 **serviceError**。

13) 如果 **i + 1** 等于 **nextRDNToBeResolved**，则根据一个 NSSR，请求被引导到一个不能继续名字解析的 DSA；在这种情况下，返回一个问题为 **unableToProceed** 的 **serviceError**；否则继续执行步骤 12)。

18.3.3 目标发现子规程

当目标对象名字与本地的某个条目 DSE 相匹配，则进入本子规程。本子规程检查所发现的条目是否适合在本地对请求进行处理（如图 11 所示）：

1) 调用检查适宜性规程。

2) 如果条目是适合的（即**条目适合**），则执行下述动作：

— 设置 **nameResolutionPhase** 为 **completed**；

— 比较 **ChainingArguments.streamedResults** 的值(如果存在)与 **ChainingArguments.traceInformation** 中的元素个数；如果相等，则将 StreamedResultsOK 设置为真；并且

— 返回**条目适合**。

3) 如果条目是不适合的（即**条目不适合**），则根据从 **DSE(lastCP)**中发现的 **supplierKnowledge** 属性产生一个继续引用。设置：

— **targetObject** 为目标对象的名字，该名字是使用主 RDN（可替代识别值可能也包含在 RDN 中）的已解析组件与剩余的尚未解析的组件级联起来构成的；

— **operationProgress.nameResolutionPhase** 为 **proceeding**；

— **operationProgress.nextRDNToBeResolved** 为 **m**；

— **referenceType** 为 **supplier**；以及

— 适当的 **accessPoints**。

在 **NRcontinuationList** 的继续引用列表中增加该继续引用。返回**条目不适合**。

注 — 如果设置了服务控制选项 **localScope**，然而，基于本地策略，DSA 能够决定将此条目认为是适合的，并按照第 2）步骤来继续执行。

4) 如果不支持关键扩展（**不支持的关键扩展**），则返回一个问题为 **unavailableCriticalExtension** 的 **serviceError**。

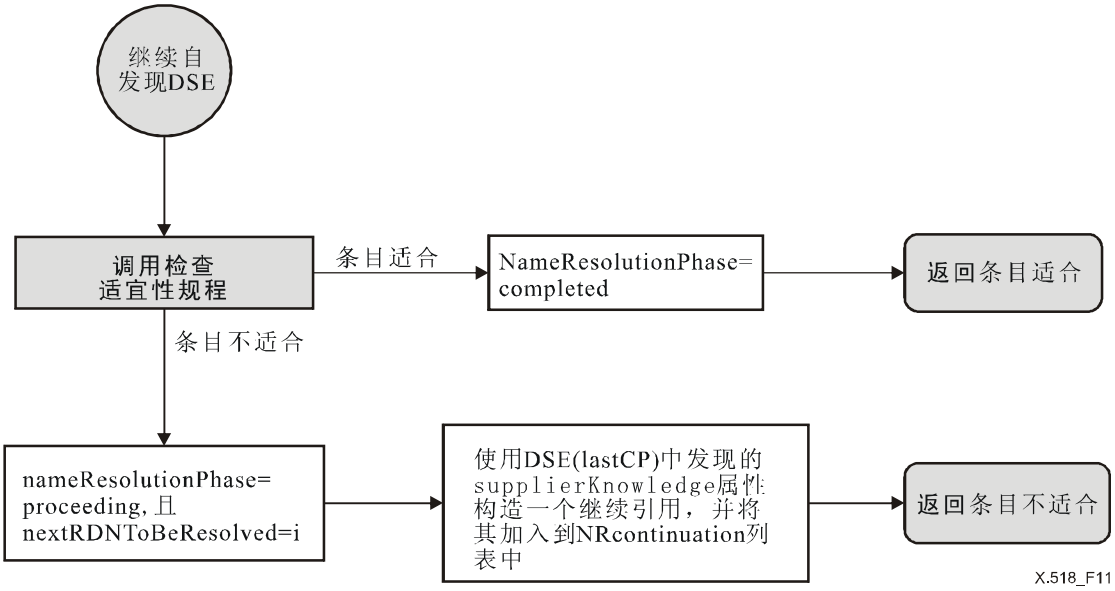


图 11－目标发现子规程

18.3.4 检查适宜性规程

调用该规程是用来决定一个发现的 DSE 是否适合执行所请求的操作（见图 12）。它考虑 **ChainingArguments**，**ServiceControls**，用户所提供的变量，操作类型，以及 DSE 的特性（镜像，下级知识，存在的属性等）。

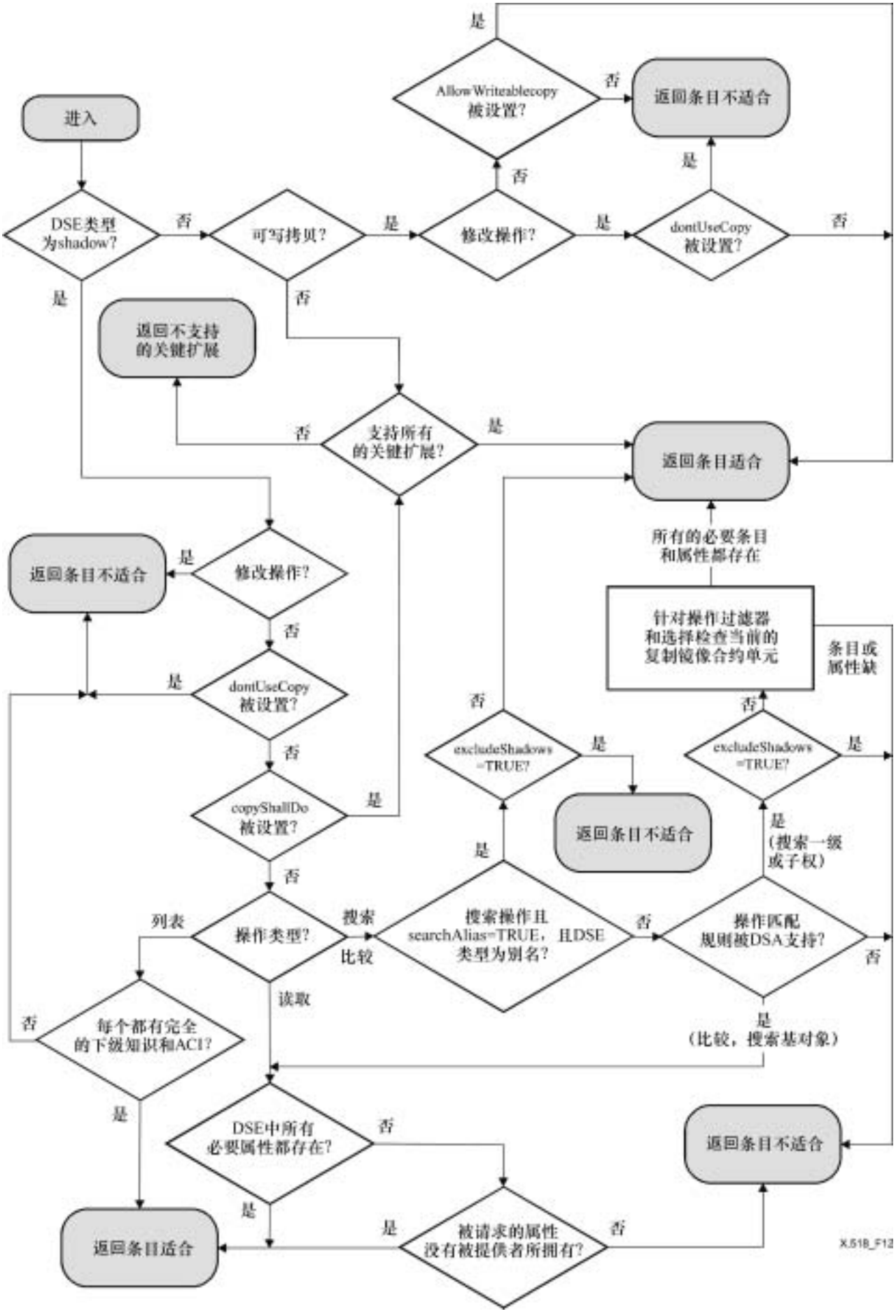

进入
DSE类型为shadow?
否
可写拷贝?
是
修改操作?
是
dontUseCopy被设置?
否
AllowWriteablecopy被设置?
否
返回条目不适合
是
否
是
返回不支持的关键扩展
否
支持所有的关键扩展?
是
返回条目适合
所有的必要条目和属性都存在
针对操作过滤器和选择检查当前的复制镜像合约单元
条目或属性缺
否
excludeShadows=TRUE?
是
返回条目不适合
是
(搜索一级或子权)
操作匹配规则被DSA支持?
否
是
(比较，搜索基对象)
返回条目不适合
返回条目不适合
是
修改操作?
否
是
dontUseCopy被设置?
否
是
copyShallDo被设置?
否
列表
操作类型?
搜索
比较
搜索操作且searchAlias=TRUE，且DSE类型为别名?
否
读取
否
每个都有完全的下级知识和ACI?
是
DSE中所有必要属性都存在?
否
是
返回条目适合
是
被请求的属性没有被提供者所拥有?
否
X.518_F12

图 12－检查适宜性规程

18.3.4.1 规程参数

本规程的输入变量为：

— 一个指向某 DSE 的引用；

— 操作类型，为了此操作类型，DSE 的适宜性将被检查；

— **ChainingArguments**；以及

— 操作变量。

输出可以是**条目适合**，**条目不适合**，或者**不支持的关键扩展**。

1) 如果 DSE 不是 **shadow** 类型，也不是 **writeableCopy** 类型，则检查是否所有的 **criticalExtensions** 都被支持。如果是，则返回条目适合，否则返回不支持的关键扩展。

2) DSE 的类型为 **shadow**。如果下面的任意一条为真，则返回条目不适合：

— 被请求的操作类型为一个修改操作。

— 服务控制 **dontUseCopy** 被设置。

否则，继续下一步骤。

3) 如果 DSE 的类型为 **writeableCopy**，且下面的任意一条为真，则返回条目不适合：

— 被请求的操作类型为一个修改操作且服务控制 **dontUseCopy** 被设置。

— 被请求的操作类型为一个查询操作且服务控制 **allowWriteableCopy** 没有被设置。

否则，返回条目适合。

4) 如果服务控制 **copyShallDo** 被设置，则检查是否所有的 **criticalExtensions** 都被支持。如果是，则返回**条目适合**，否则返回**不支持的关键扩展**。

5) 如果服务控制 **copyShallDo** 没有被设置，则检查是否所有的 **criticalExtensions** 都被支持。如果是，则转到步骤 5)，否则返回**条目不适合**。

6) 区分几类操作类型：

如果为列表操作，则继续执行步骤 6)。

如果为阅读操作，则继续执行步骤 7)。

如果为搜索或比较操作，则继续执行步骤 8)。

7) 如果条目具有完整的下级知识，则列表操作可以被执行。在这种情况下，返回**条目适合**，否则返回**条目不适合**。

8) 如果所有被请求的属性都出现在 DSE 中，则返回**条目适合**。如果某些属性缺失，则通过本地方式判断镜像拷贝是否保存了属主所拥有的所有属性（例如，通过参考镜像合约可以得知）。如果是，则条目是适合的（返回**条目适合**）。否则，提供者可能拥有镜像中不存在的那些被请求的属性；在这种情况下，请求必须被链接（返回**条目不适合**）。

9) 如果操作为 **search**，且 **searchAliases** 被设置为 **TRUE**，同时 DSE 的类型为 **alias**，则如果 **chainingArguments.excludeShadows** 为 **FALSE**，则返回**条目适合**，如果为 **TRUE** 则返回**条目不适合**。

10) 如果 DSA 支持所要求的用于比较或搜索的匹配规则，且操作为 **compare** 或 **search** 操作，其中 **subset** 为 **baseObject**，则继续执行步骤 7)。如果 DSA 支持匹配规则，且操作为 **search**，其中 subset 为 **oneLevel** 或 **subtree**，则继续执行步骤 10)。否则返回**条目不适合**。

11) 如果 **chainingArguments.excludeShadows** 为 **TRUE**，则返回**条目不适合**。否则，根据操作过滤器和选择来检查镜像信息规范的本地理解。如果所有的必要条目和属性都存在，则返回**条目适合**。如果有任意条目或属性缺失，则返回**条目不适合**。

19 操作评估

本节定义了如果（在名字解析阶段）已经在本地找到了某个操作的目标条目，则一个 DSA 必须遵循的规程。根据操作的类型，下面的规程之一将被调用：

— 对于一个 **addEntry**, **chainedAddEntry**, **removeEntry**, **chainedRemoveEntry**, **modifyEntry**, **chainedModifyEntry**, **modifyDN** 或 **chainedModifyDN** 等操作，必须遵循 19.1 节中定义的规程。

— 对于一个 **read, chainedRead, compare** 或 **chainedCompare** 操作，必须遵循 19.2 节中定义的规程。

— 对于一个 **search, chainedSearch, list** 或 **chainedList** 操作，必须遵循 19.3 节中定义的规程。

19.1 修改规程

根据修改操作的类型，必须遵循 19.1.1 节到 19.1.4 节中定义的相应规程。

19.1.1 增加条目操作

1) DSA 必须检查发起者是否有足够的访问权限，如同 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 11.1.5 节定义的那样。如果没有，则返回一个适当的错误。

2) DSA 必须确保要被加入的条目的名字所对应的条目尚不存在。否则，必须返回一个问题为 **entryAlreadyExists** 的 **updateError**。如果上级 DSE 具有附加的类型 **nssr**，则 DSA 必须遵循 19.1.5 节（修改操作和 NSSR）中定义的规程，以便确保新条目的名字是无二义性的。如果要加入的条目的名字中，最后一个 RDN 的某些属性包含了由上下文所区分的多个识别值，则 DSA 必须确保在可能构建的可替代 RDN 中，没有一个会产生和已经存在的条目名字相同的名字（不考虑上下文）。

3) 如果 **targetSystem** 存在，且 **AccessPoint** 不是当前 DSA 的访问点，则转到步骤 4)。如果 **targetSystem** 不存在，或存在但 **AccessPoint** 是当前 DSA 的访问点，则转到步骤 5)。

4) 如果该条目是一个子条目，则 DSA 必须返回一个问题为 **affectsMultipleDSAs** 的 **updateError**。如果该条目不是一个子条目，则 DSA 有一个本地选择，即它是否愿意与指定的 DSA 之间建立一个 HOB。如果不愿意，则 DSA 必须返回一个问题为 **unwillingToPerform** 的 **serviceError**，否则 DSA 必须与指定的下级 DSA 之间建立一个分等级的操作绑定（HOB）。如果支持 DOP，则必须遵循 24.3.1.1 节中定义的规程。否则，将使用本地方式建立该 HOB。如果下级 DSA 不愿意建立此操作绑定，则 **addEntry** 操作将返回一个问题为 **unwillingToPerform** 的 **serviceError**。如果 HOB 被成功建立，则继续执行步骤 7)。

   注 1 — 规程中的本步骤不能应用于在一个下级 DSA 中创建自治管理区。

5) DSA 必须确保新的条目符合子模式，或者新的子条目或其他类型的 DSE 符合系统模式（例如一个子条目的直接上级 DSE 的类型为 **admPoint**）。如果不符合，则 DSA 必须返回一个适当的 **updateError** 或 **attributeError**，否则它必须加入此新的 DSE。如果是条目，则继续执行步骤 7)。如果是子条目，则继续执行步骤 6)。否则，应当执行其他 DSE 类型的适当的知识管理规程。见第 6 部分。

6) DSA 必须在一个适当的时间点，将一个修改操作绑定前向到所有相关的下级 DSA，本 DSA 与这些下级 DSA 之间具有分等级操作绑定或非特定的分等级操作绑定。相关的绑定是与作为上级 DSE 的下级的命名上下文相关的绑定。上下文前缀符合自治管理点的命名上下文是不相关的。如果支持 DOP，则必须遵循 24.3.2.1 节和 25.3.2 节中规定的规程。如果不支持 DOP，则必须使用本地方式来修改 RHOB。

   注 2 — 一个适当的时间由 DSA 管理者来指定，其范围可能包括从操作结果刚刚返回（甚至在返回之前）一直到某个周期性策略（例如在某个指定的小时）。时间可能会根据修改的原因而变化，例如对 ACI 的更新会立即生效，而对模式的修改可能会周期性进行。

7) 如果增加的条目或子条目是在一个或多个镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

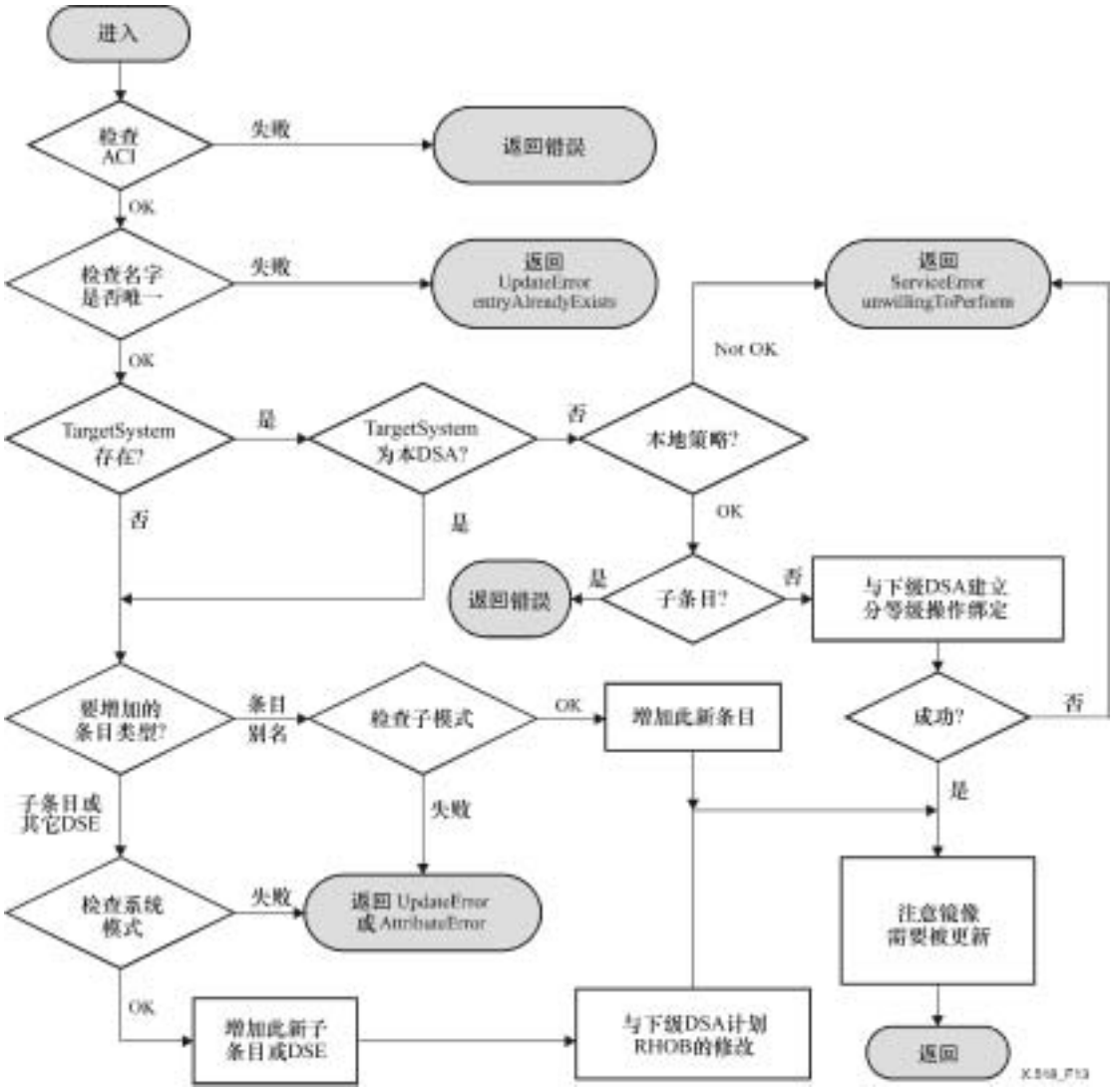


图 13－增加条目规程

19.1.2 删除条目操作

1) DSA 必须检查发起者是否有足够的访问权限，如同 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 11.2.5 节定义的那样。如果没有，则返回一个适当的错误。

2) DSA 必须确保要删除的条目是一个叶条目。否则，DSA 必须返回一个问题为 **notAllowedOnNonLeaf** 的 **updateError**。

3) 要删除的条目的 DSE 类型被检查。如果是 **subentry**，则继续执行步骤 5)。如果是 **cp**，则继续执行步骤 6)。如果是 **entry** 或 **alias**，则继续执行步骤 4)。否则，应当执行其他 DSE 类型的适当的知识管理规程。见第 6 部分。

4) 删除条目或别名条目，并继续执行步骤 7)。

5) 删除子条目。在一个适当的时间点，修改所有相关的下级 DSA 的操作绑定，本 DSA 与这些下级 DSA 之间具有分等级操作绑定或非特定的分等级操作绑定。相关的绑定是与作为上级 DSE 的下级的命名上下文相关的那些绑定。

   上下文前缀符合自治管理点的命名上下文是不相关的。如果支持 DOP，则必须遵循 24.3.2.1 节和 25.3.2 节中规定的规程。否则必须使用本地方式。继续执行步骤 7)。

6) 删除命名上下文。如果该 DSA 有一个此命名上下文的分等级操作绑定，则它必须终止与其直接上级 DSA 之间的分等级操作绑定。如果该 DSA 有一个此命名上下文的非特定分等级操作绑定，且这是非特定分等级操作绑定的最后一个命名上下文，则它必须终止与其直接上级 DSA 之间的非特定分等级操作绑定。如果支持 DOP，则必须遵循 24.3.3.2 节和 25.3.3.2 节中规定的规程。否则，必须使用本地方式来终止 RHOB。

7) 如果已删除的命名上下文、条目、别名条目或子条目等是在一个或多个镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

如果直接上级 DSA 内（其 RHOB 被终止）已删除的下级引用或非特定下级引用，是在一个或多个镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

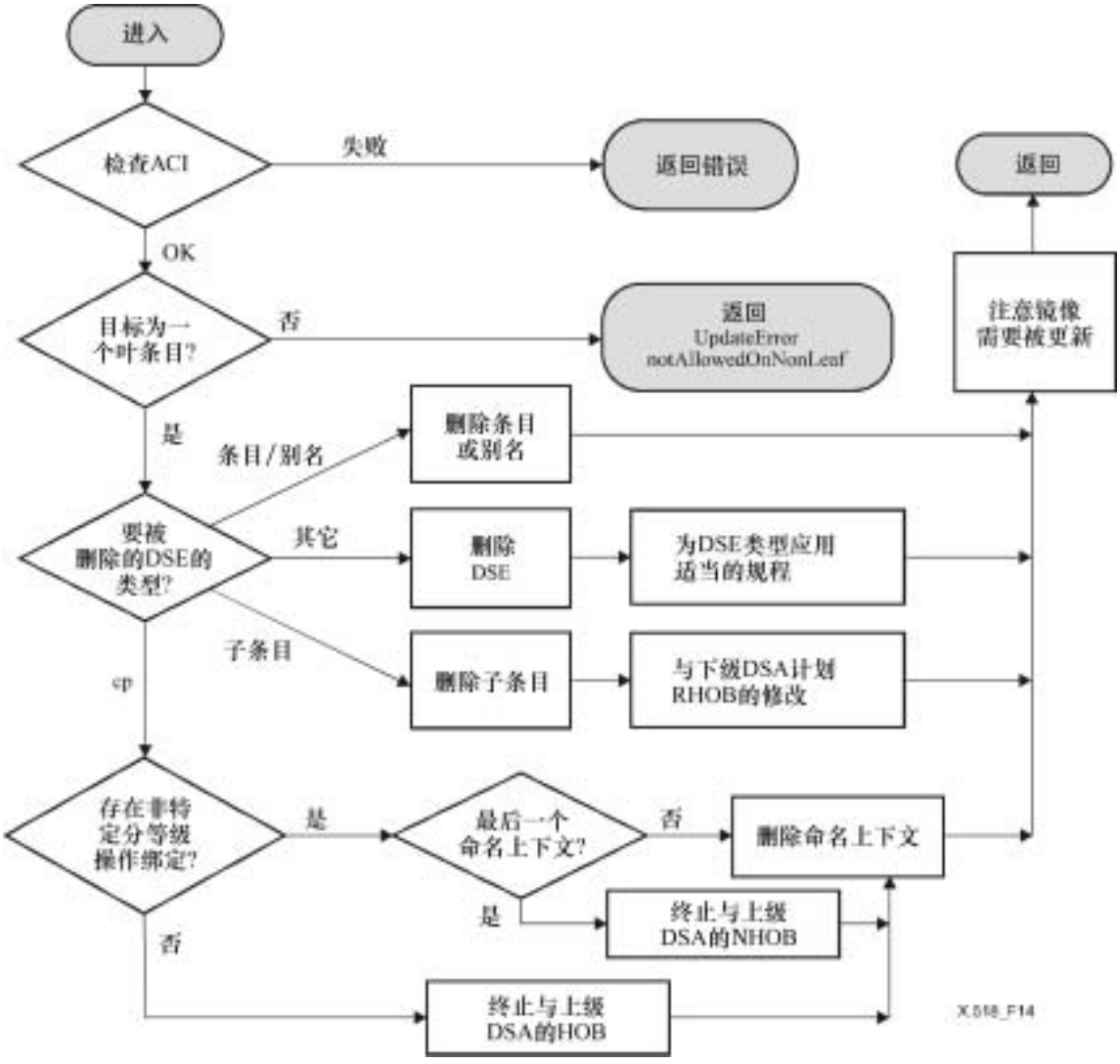


图 14－删除条目规程

19.1.3 修改条目操作

1) DSA 必须检查发起者是否有足够的访问权限，如同 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 11.3.5 节定义的那样。如果没有，则返回一个适当的错误。

2) 对条目或别名的修改必须符合子模式。对 DSE 的其他类型的修改，包括子条目，必须符合系统模式。否则，DSA 必须返回一个适当的 **updateError** 或 **attributeError**。在执行了修改之后，如果目标 DSE 的类型为 **subentry**，则继续执行步骤 3)；如果目标 DSE 的类型为 **entry** 或 **alias**，则继续执行步骤 4)；否则，应当执行其他 DSE 类型的适当的知识管理规程。见第 6 部分。

3) DSA 必须在一个适当的时间点，修改与所有相关的下级 DSA 的操作绑定，本 DSA 与这些下级 DSA 之间具有分等级操作绑定或非特定的分等级操作绑定。相关的绑定是与作为管理点的下级的那些命名上下文相关的绑定，被修改的子条目位于此管理点之下。上下文前缀符合自治管理点的命名上下文是不相关的。如果支持 DOP，则必须遵循 24.3.2.1 节和 25.3.2 节中规定的规程。否则，将使用本地方式。

4) 如果被修改的条目、别名条目或子条目等是在一个或多个镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

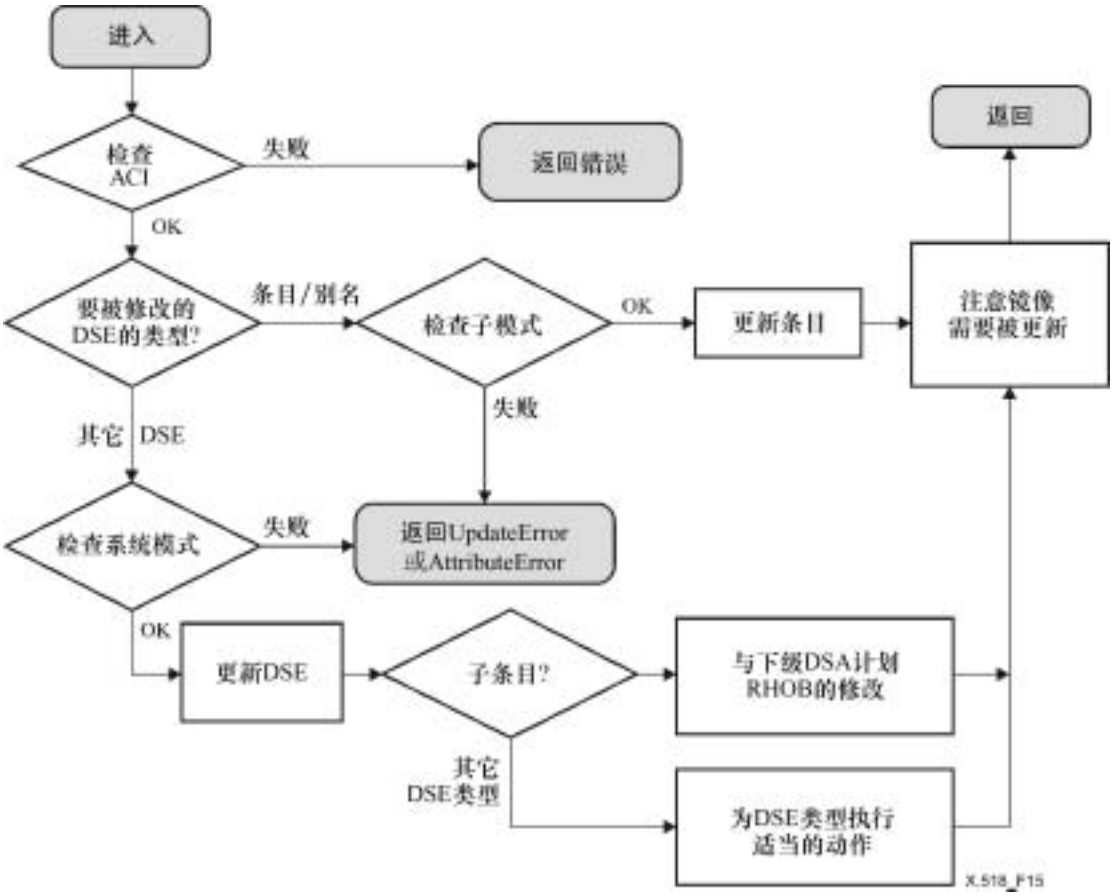


图 15－修改条目规程

19.1.4 修改DN操作

1) DSA 必须检查发起者是否有足够的访问权限，如同 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 11.4.5 节定义的那样。如果没有，则返回一个适当的错误。

2) 如果操作是移动一个条目，或者是既移动条目又修改其相对识别名，则转到步骤 3)。如果操作仅仅是修改一个条目的相对识别名，则转到步骤 4)。

3) 操作必须按照 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 11.4.1 节中的定义来执行。如果旧的上级、新的上级、本条目或其任意下级中的一个不在本 DSA 内，或者如果新的上级有 NSSR，则该操作必须被拒绝，且返回问题为 **affectsMultipleDSAs** 的 **updateError**。DSA 必须确保不存在具有此新名字的其他条目。否则，它必须返回一个问题为 **entryAlreadyExists** 的 **updateError**。DSA 必须确保条目的新名字符合子模式。否则，它必须返回一个适当的 **attributeError** 或 **updateError**。如果没有发生上述这些问题，则移动该条目（如果需要，则修改 RDN），并且转到步骤 9)。

4) 下面的文字应用于修改条目的相对识别名，该条目可能是，也可能不是一个叶条目；而且可能在一个或多个 DSA 中具有或不具有一个或多个下级。对要重命名的条目的 DSE 类型进行检查。如果为 **subentry**，则继续执行步骤 7)。如果为 **cp**，则继续执行步骤 6)。如果为 **entry** 或 **alias**，则继续执行步骤 5)。

5) DSA 必须确保具有此新名字的其他条目尚不存在。否则，必须返回一个问题为 **entryAlreadyExists** 的 **updateError**。如果被重命名的条目的上级 DSE 具有附加的类型 **nssr**，则 DSA 必须遵循 19.1.5 节（修改操作和 NSSR）中定义的规程，以便确保条目的新名字是无二义性的。如果新名字中某个 RDN 的某些属性包含了由上下文所区分的多个识别值，则 DSA 必须确保在可能构建的 RDN 中，没有一个会产生和已经存在的条目名字相同的名字（不考虑上下文）。DSA 必须确保条目的新名字符合子模式。否则，它必须返回一个适当的 **attributeError** 或 **updateError**。重新命名条目或别名条目。如果条目是一个非叶条目，且有下级在其他的 DSA 中，则继续执行步骤 8)，否则继续执行步骤 9)。

6) DSA 必须确保命名上下文的新名字符合子模式；否则，它必须返回一个适当的 **attributeError** 或 **updateError**。

   如果 DSA 与上级 DSA 之间有一个 HOB，则下级 DSA 必须在响应修改 DN 操作之前就尝试修改 HOB。上级 DSA 必须在接受修改之前，确保没有具有新名字的其他条目存在。如果支持 DOP，则必须遵循 24.3.2.2 节中规定的规程。如果不支持 DOP，则 HOB 如何被修改，且如何检查新名字的唯一性等属于本地事务。如果 HOB 被成功修改，且命名上下文具有下级命名上下文在其他 DSA 中，则转到步骤 8)；否则转到步骤 9)。如果 HOB 不能被修改，则返回一个问题为 **affectsMultipleDSAs** 的 **updateError**。

   如果 DSA 与其上级 DSA 之间有一个关于该命名上下文的 NHOB，则如何检测出重复的条目不在本号码簿规范的定义范围之内。重命名此条目。如果命名上下文具有下级命名上下文在其他 DSA 中，则转到步骤 8)；否则转到步骤 9)。

7) DSA 必须确保子条目的新名字符合系统模式。否则，它必须返回一个适当的 **attributeError** 或 **updateError**。DSA 必须确保没有具有新名字的其他子条目已经存在。否则，它必须返回一个问题为 **entryAlreadyExists** 的 **updateError** 。

8) DSA 必须在一个适当的时间点，修改与所有相关的下级 DSA 的操作绑定，本 DSA 与这些下级 DSA 之间具有分等级操作绑定或非特定的分等级操作绑定。相关的绑定指的是与作为重命名条目的下级的所有命名上下文相关的那些绑定，或者是与作为管理点的下级的命名上下文相关的那些绑定，此管理点的子条目被重命名。上下文前缀符合自治管理点的命名上下文是不相关的。如果支持 DOP，则必须遵循 24.3.2.1 节和 25.3.2 节中规定的规程。否则，将使用本地方式来更新 RHOB。

9) 如果被重命名的命名上下文、条目或其任意下级、别名条目或子条目等是在 DSA 所拥有的一个或多个镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

   如果条目、别名条目或子条目等是在 DSA 所拥有的一个或多个镜像合约的 **UnitOfReplication** 内，但重命名的条目、别名条目或子条目等的上级不在此 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿镜像服务规程；在这种情况下，被镜像的条目及其所有下级都必须被删除。

   如果条目、别名条目或子条目等不在 DSA 所拥有的一个或多个镜像合约的 **UnitOfReplication** 内，但重命名的条目、别名条目或子条目等目前在此 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿镜像服务规程；在这种情况下，被镜像的条目及其所有下级都必须被镜像。

   如果处于直接上级 DSA 内的被重命名的下级引用（其 HOB 在上述的第 6）步被修改）是在一个或多个它的镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

   如果与某个下级 DSA 绑定的一个 RHOB 的组件（在上述的第 8）步被修改）是在下级 DSA 所拥有的一个或多个镜像合约的 **UnitOfReplication** 内，则镜像消费者必须被更新，方法是使用 ITU-T X.525 建议书 | ISO/IEC 9594-9 中规定的号码簿信息镜像服务规程。

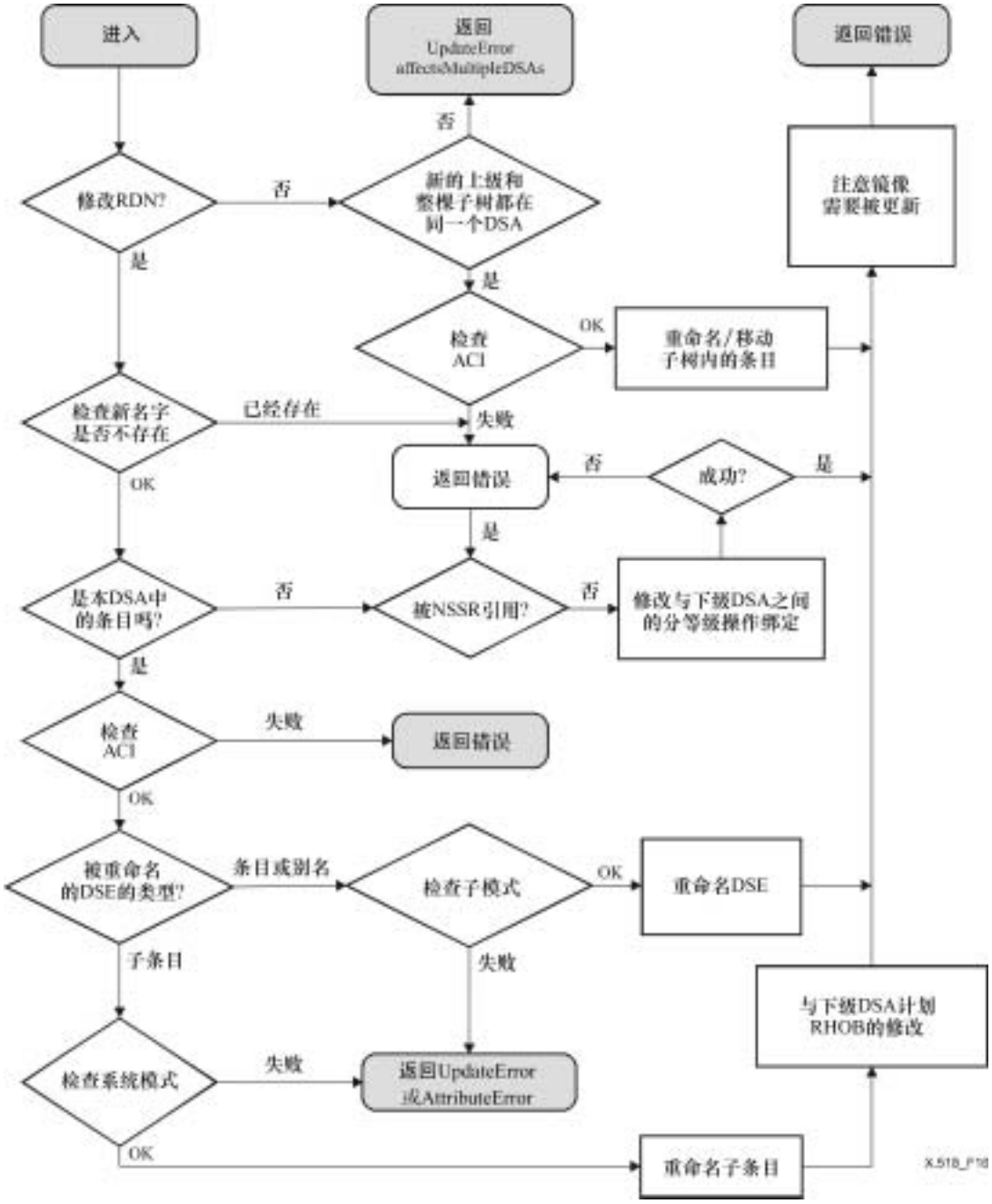


图 16－修改DN规程

19.1.5 修改操作和非特定下级引用

如果一个 DSA 具有 NSSR 且不知道某个条目的所有下级的完整名字集合，对于该 DSA，或者：

a) 有一个 **addEntry** 操作已经被指向到该 DSA；或者

b) 有一个 **modifyDN** 操作已经被指向到该 DSA。

则该 DSA 可能在执行操作前执行下列的规程集。

1) 如果 **chainingProhibited** 服务控制选项在 **addEntry** 或 **modifyDN** 操作中被设置，则返回一个问题为 **affectsMultipleDSAs** 的 **updateError**。

2) 如果该 DSA 不愿意或不能够多链接出请求，则分别返回一个问题为 **unwillingToPerform** 或 **unavailable** 的 **serviceError**。

3) DSA 必须将一个 **chainedReadEntry** 操作多链接到 NSSR 的 **accessPointInformation** 集合中的每个主 DSA 上。（由于镜像所引起的临时不一致性，DSA 必须仅使用来自每个 **MasterAndShadowAccessPoints** 的主 DSA。）**ReadArgument** 的参数必须按照如下所述来设置：

object 或者设置为要增加的条目的名字（在 **addEntry** 的情况下），或者设置为某个存在条目的声称的名字（在 **modifyDN** 的情况下）。

selection 对象类属性。

CommonArguments 的参数必须按照如下所述来设置：

— 设置 **dontDereferenceAliases** 服务控制选项；

— 设置 **OperationProgress.nameResolutionPhase** 为 **completed**。

ChainingArguments 的参数必须按照如下所述来设置：

— 设置 **originator** 为发起者的名字；

— **targetObject** 被忽略；

— 设置 **OperationProgress.nameResolutionPhase** 为 **proceeding**，且设置 **nextRDNToBeResolved** 为（对象名字中的 RDN 数量）– 1；

— 设置 **traceInformation** 为一个空序列；

— 设置 **referenceType** 为 **nonSpecificSubordinate**；

— **timeLimit**，根据入请求进行适当地设置。

其他参数，例如 **SecurityParameters**，可能被适当地设置，例如由本地策略来设置。

4) DSA 等待响应的完整集合。如果有任意一个响应是一个 **ReadResult**，则必须返回一个错误，如下面的 6)中所述。

5) 如果所有的响应是问题为 **unableToProceed** 的 **serviceError**，则操作评估可能会继续进行。

6) 如果返回了一个 **ReadResult**，则必须为原始操作返回一个问题为 **entryAlreadyExists** 的 **updateError**；

7) 如果有任意一个其他的错误返回给 **readEntry** 请求，则必须返回一个问题为 **unwillingToPerform** 的 **serviceError**。

接收到 **chainedRead** 请求的 DSA 必须根据条目的存在与否，以及它的访问控制策略来给出一个响应。

19.2 单条目查询规程

阅读、ChainedRead、比较和 **ChainedCompare** 等操作被划分为单条目查询规程组。这些规程仅包含下列三个步骤：

1) 检查访问控制，如同 ITU-T X.511 建议书 | ISO/IEC 9594-3 第 9 节中的描述。如果该操作不允许，则返回相应的安全错误。

2) 在发现的 DSE 上执行操作，如同 ITU-T X.511 建议书 | ISO/IEC 9594-3 第 9 节中的描述。

3) 准备答复，然后返回。

19.3 多条目查询规程

根据查询操作的类型（**列表**或**搜索**），必须遵循 19.3.1 节和 19.3.2 节中定义的相应规程。

19.3.1 列表规程

本子节规定了特定于 **list** 和 **ChainedList** 操作的评估规程。

当列表请求的 **operationProgress.nameResolutionPhase** 组件被设置为 **notStarted** 或 **proceeding**，且当 DSA 在执行完名字解析后，发现它拥有基对象时，必须遵循**列表(I)**规程。当列表请求的 **nameResolutionPhase** 组件被设置为 **completed** 时，必须遵循**列表(II)**规程。

19.3.1.1 规程参数

19.3.1.1.1 变量

本规程使用的变量为：

— **ListArgument**；

— 目标 DSE **e**；

— **chainingArgument** 的 **operationProgress**。

19.3.1.1.2 结果

如果本规程被成功执行，则它返回：

— 一系列 **e** 的下级，在 **listInfo.subordinates** 中；

— **limitProblem**，在 **partialOutcomeQualifier** 中指示；

— 一系列的继续引用，在 **SRcontinuationList** 中。

19.3.1.2 规程定义

19.3.1.2.1 列表(I)规程

列表(I) 规程包含下列步骤，如图 17 所示：

1) 如果服务控制 **subentry** 被设置，则转到步骤 5)；否则转到步骤 2)。

2) 如果 DSE **e** 的类型为 **nssr**，则向 **SRcontinuationList** 中增加一个继续引用，其组件被设置如下：

— **targetObject** 被设置为 DSE **e** 的主识别名（可替代识别值可能也包含在该 RDN 中）；

— **aliasedRDNs** 缺失；

— **nameResolutionPhase** 的 **operationProgress** 被设置为 **completed**，且 **nextRDNtoBeResolved** 缺失；

— **rdnsResolved** 缺失；

— **referenceType** 被设置为 **nonSpecificSubordinate**；

— **accessPoints** 被设置为一系列 **accessPointInformation**，每个值都源自 DSE **e** 的 **nonSpecificKnowledge** 属性的一个值。

3) 对于作为 DSE **e** 直接下级的每个 DSE **e'**，执行下列步骤：

a) 检查 **e'** 中的 ACI 是否可用。如果 ACI 不允许列出 **e'**的 RDN，则跳过该 DSE。如果 ACI 不可用（例如在下级引用和粘接的情况下），则由本地策略来决定是否继续。

b) 检查 **e'**的所有 DSE 类型。

i) 如果 **e'**的类型为 **subr**，则有两种情况。在第一种情况下，下级条目的 ACI 和对象类是本地可用的，在这种情况下，基于本地策略和 ACI 的准许，将 **e'**的 RDN 加入到 **listInfo.subordinates** 中，其中 **aliasEntry** 被设置为 **TRUE**，如果 **e'**的类型为 **sa**，则 **fromEntry** 被设置为 **FALSE**。另一种情况是条目的 ACI 在 **e'**中不可用，在这种情况下，向 **SRcontinuationList** 中增加一个继续引用，其组件被设置如下：

— **targetObject** 被设置为 DSE **e** 的主识别名（可替代识别名可能包含在 RDN 中）；

— **aliasedRDNs** 缺失；

— **nameResolutionPhase** 的 **operationProgress** 被设置为 **completed**，且 **nextRDNtoBeResolved** 缺失；

— **rdnsResolved** 缺失；

— **referenceType** 被设置为 **subordinate**；

— **accessPoints** 被设置为包含在 DSE **e'**的 **specificKnowledge** 属性中的值。

ii) 如果 DSE **e'**的类型为 **entry** 或 **glue**，则将 **e'**的 RDN 加入到 **listInfo.subordinates** 中，其中 **aliasEntry** 被设置为 **FALSE**，同时根据 **e'**是否是一个拷贝，而设置 **fromEntry** 的值。

注 — 在 **e'** 的类型是 **glue** 的情况下，它必须拥有一个或多个下级，这意味着它不能是主 DSA 中的一个别名。另外，任何与列表操作相关的 ACI 都存储在此 DSE 中，通过镜像协议来提供。

iii) 如果 DSE **e'**的类型为 **alias**，则将 **e'**的 RDN 加入到 **listInfo.subordinates** 中，其中 **aliasEntry** 被设置为 **TRUE**，同时根据 **e'**是否是一个拷贝，而设置 **fromEntry** 的值。

c) 检查时间、尺寸或管理限制是否被超越。如果被超越，则在 **partialOutcomeQualifier** 中设置相应的 **limitProblem** 并返回。

d) 继续执行步骤 3)中的 a)，直到所有的下级 DSE 都被处理过。

4) 如果所有的下级 DSE 都被处理，则返回到**操作调度程序**。

5) 对于作为 DSE **e** 直接下级的每个子条目 **e'**，执行下列步骤：

a) 检查 **e'**中的 ACI。如果 ACI 不允许列出 **e'**的 RDN，则跳过此 DSE。否则，将 **e'**的 RDN 加入到 **listInfo.subordinates** 中，其中 **aliasEntry** 被设置为 **FALSE**，同时根据 **e'**是否是一个拷贝，而设置 **fromEntry** 的值。

b) 检查时间、尺寸或管理限制是否被超越。如果被超越，则在 **partialOutcomeQualifier** 中设置相应的 **limitProblem** 并返回。

6) 返回到**操作调度程序**。

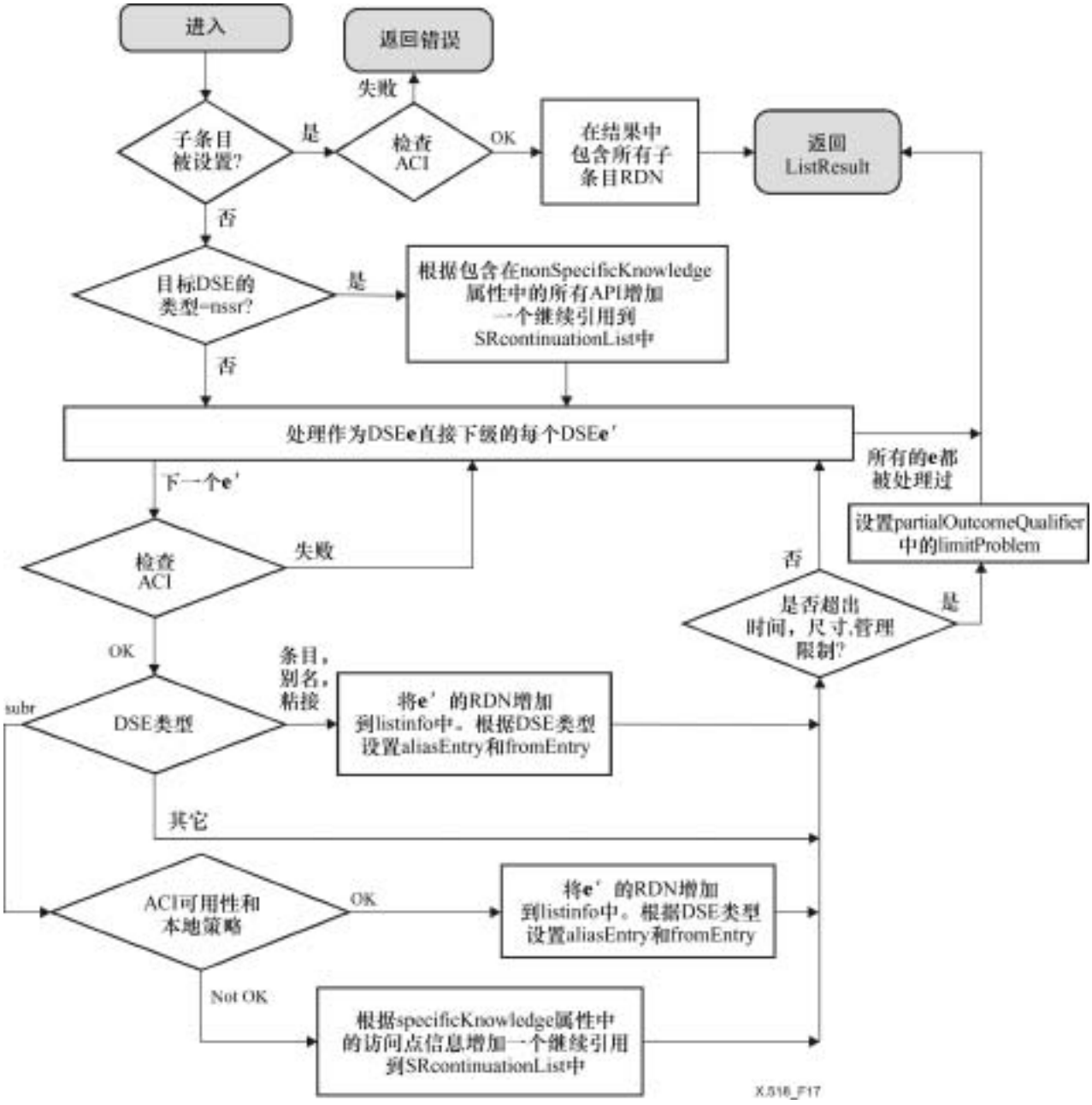


图 17－列表(I)规程

19.3.1.2.2 列表(II)规程

列表(II)规程包含下列步骤，如图 18 所示：

1) 对于作为 DSE **e** 直接下级的每个 DSE **e'**，执行步骤 1)中的 a)到 1)中的 d)：
   a) 如果 **e'** 不是一个条目或别名，则继续下一个直接下级。
   b) 检查 **e'**中的 ACI。如果 ACI 不允许该操作，则继续 **e** 的下一个直接下级。
   c) 将 DSE **e'**的 RDN 加入到 **listInfo.subordinates** 中，其中根据 **e'**是否是一个别名来设置 **listInfo.subordinates** 的 **aliasEntry** 组件的值，同时根据 **e'**是否是一个拷贝，而设置 **fromEntry** 组件的值。如果 **excludeShadows** 为 **TRUE**，则忽略那些类型为 **shadow** 或 **writableCopy** 的 DSE。
   d) 检查时间、尺寸或管理限制是否被超越。如果被超越，则在 **partialOutcomeQualifier** 中设置相应的 **limitProblem** 并返回。
   e) 继续执行步骤 1) 中的 a)直到所有的下级 DSE 都被处理过。
2) 如果所有的下级 DSE 都已经被处理，则检查该子请求是否来自 DAP 或 DSP。如果该子请求是通过 DAP 提交的，且 **ListResult** 为空，则向操作调度程序返回一个问题为 **invalidReference** 的 **serviceError**。否则，返回 **ListResult**。

   注 — 当用户不能访问到上级条目时，**invalidReference** 被用作一个安全预警。如果上级的条目 ACI 可用（通过 RHOB 来提供），则如果允许的话，可能返回一个空结果。

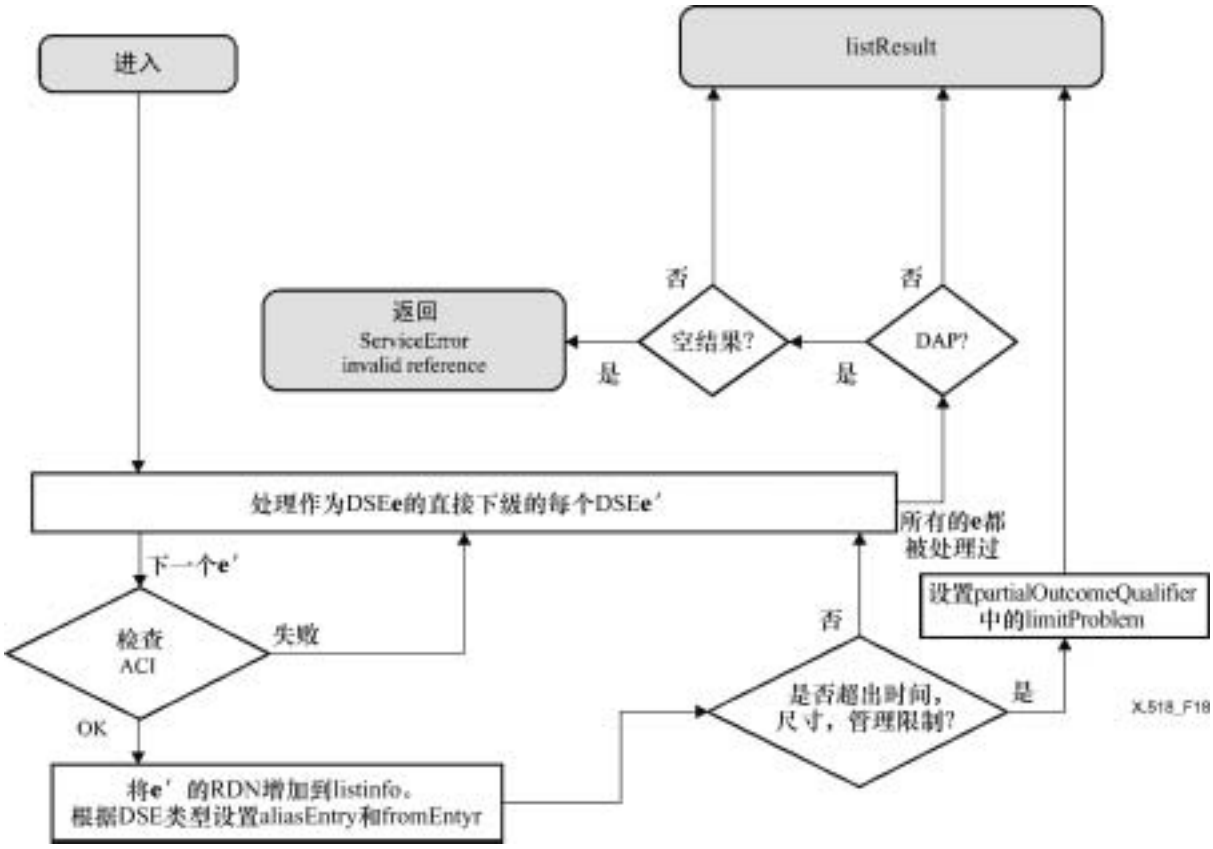


图 18－列表(II)规程

19.3.2 搜索规程

本子节规定了特定于 **Search** 和 **Chained Search** 操作的评估规程。

当搜索请求的 **operationProgress.nameResolutionPhase** 组件被设置为 **notStarted** 或 **proceeding**，且当 DSA 在执行完名字解析后，发现它拥有一个目标对象时，必须遵循**搜索规则检查(I)**规程。如果该规程返回了一个错误，则返回此错误。否则，必须遵循 **Search (I)** 规程。

当搜索请求的 **nameResolutionPhase** 组件被设置为 **completed** 时，必须遵循**搜索规则检查(II)**规程。如果该规程返回了一个错误，则返回此错误。否则，必须遵循 **Search (II)** 规程。

注 — 若 **nameResolutionPhase** 为 **completed**，则期望目标对象是一个上下文前缀的直接上级。

19.3.2.1 规程参数

19.3.2.1.1 变量

本规程所使用的变量包括：

— **SearchArgument**；

— 目标 DSE **e**；

— **ChainingArguments** 的 **operationProgress**；

— **ChainingArguments** 的 **exclusions**（要从搜索中排除出去的 RDN 列表）；

— **ChainingArguments** 的 **traceInformation**；

— **ChainingArguments** 的 **searchRuleId**；

— **ChainingArguments** 的 **chainedRelaxation**；以及

— **ChainingArguments** 的 **relatedEntry**。

19.3.2.1.2 结果

如果本规程被成功执行，它将返回：

— 一系列匹配的条目，在 **searchResult.entryInformation** 中；

— **alreadySearched**，在 **ChainingResults** 中；

— 根据条件的一个数目，在 **partialOutcomeQualifier.entryCount** 中；以及

— 一系列的继续引用，在 **SRcontinuationList** 中。

19.3.2.2 规程定义

19.3.2.2.1 相关的条目变量规程

只有当**搜索**请求有一个 **joinArguments** 组件，且 **ChainingArguments**（如果有的话）没有 **relatedEntry** 组件时，本规程才相关。

1) 如果**搜索**请求被保护，则为 **joinArguments** 组件中的每个元素产生一个 DSP 请求，每个请求都包含了初始的 DAP 请求或 LDAP 消息。**ChainingArguments** 必须如下所述：

   — 如果入请求有一个 **ChainingArguments**，且带了 **originator** 组件，则该组件的值将被拷贝到所产生的请求的 **originator** 组件中；否则该组件的使用由本地安全策略所决定；

   注 — 接收端 DSA 可能不能够使用本组件中给定的名字，如果它是来自一个不同的 DIT 的话。

   — 组件 **operationProgress** 必须被忽略或者被设置为缺省值；

   — 组件 **traceInformation**、**aliasDereferenced**、**aliasedRDNs**、**returnCrossRefs**、**entryOnly**、**exclusions**、**nameResolutionOnMaster**、**searchRuleId**、**chainedRelaxation** 等必须被忽略；且

   — 组件 **relatedEntry** 被设置为一个值，该值相对应于应用到 DSA 的 **JoinArgument** 的相对位置，而该 DSA 是请求要前向的 DSA；当第一个 **JoinArgument** 被给定值为 0 时，下一个值则为 1，依此类推。

2) 如果入请求没有被保护，则为 **joinArguments** 组件中的每个元素产生一个 DSP 请求，此时 **SearchArgument** 必须按照如下所述产生：

   — 组件 **baseObject** 的值必须拷贝自相应的 **JoinArgument** 中的 **joinBaseObject** 组件；

   — 组件 **subset** 必须拷贝自相应的 **JoinArgument** 中的 **joinSubset** 组件；

   — 组件 **filter** 必须拷贝自相应的 **JoinArgument** 中的 **filter** 组件；且

— 剩余的组件必须与原始请求中相应组件的值相同，除了 **joinArguments** 和 **joinType** 组件必须被忽略。

ChainingArguments 必须与上面被保护的请求的要求相同，除了 **relatedEntry** 组件必须被忽略以外。

3) 为每个要本地继续执行的请求调用**操作调度程序**。

4) 如果**操作调度程序**返回一个 **referral** 错误、或忙、或不可用错误等，则在 **SearchResult** 的 **partialOutcomeQualifier** 中增加（或产生并增加）一个继续引用，并返回。

5) 如果**操作调度程序**返回其他错误，则丢弃并返回。

6) 如果**操作调度程序**返回一个 **SearchResult**，则：

i) 如果结果被签名、加密或签名并加密，则将该结果增加到 **SearchResult** 的 **uncorrelatedSearchInfo** 中。

ii) 如果结果没有被签名、加密或签名并加密，则根据 ITU-T X.511 建议书 | ISO/IEC 9594-3 中的规定执行合并过程。

19.3.2.2.2 搜索规则检查规程(I)

当且仅当 DSA 支持服务特定的管理区时，本规程才相关。

如果 **searchRuleId** 组件出现在 **ChainingArguments** 中，则该操作是之前的评估阶段中的一个别名解除引用规程的结果。然后，如果目标 DSE 处于一个具有不同 **dmdId** 的服务特定管理区内，或者如果目标 DSE 处于某个服务特定管理区之外，则返回一个 **unwillingToPerform** 服务错误。否则基于 **searchRuleId** 中的信息选择适当的搜索规则，并返回。

注 1 — 服务管理已经被定义为一个关键扩展。当一个不支持服务管理的 DSA 接收到一个链接搜索请求，且带有一个 **searchRuleId** 组件时，它将返回一个问题为 **unavailableCriticalExtension** 的 **serviceError**。

如果 **searchRuleId** 组件不存在，且目标 DSE 处于某个服务特定管理区之外；或者如果它处于这样的一个管理区内但没有子条目与此管理区相关联，则返回。

如果目标 DSE 处于某个服务特定管理区内，且 **traceInformation** 显示该操作已经在之前的评估阶段出现过，则返回一个 **unwillingToPerform** 服务错误。

注 2 — 这是这样一种情况，即一个搜索操作已经在某个服务特定管理区外启动了它的初始评估，而目前准备扩展到一个不同的服务特定管理区内。

否则，必须遵循下述规程：

1) 查找定位所有的与目标 DSE 相关联的搜索规则，即在服务子条目中，其子树规范内具有目标 DSE 的所有搜索规则（例如通过使用 **searchRulesSubentry** 操作属性）。这些搜索规则在后续被称为*候选搜索规则*。如果没有这样的搜索规则，则产生一个问题为 **requestedServiceNotAvailable** 的服务错误，并在 **CommonResults** 的 **notification** 组件中，包括一个取值为 **id-pr-unidentifiedOperation** 的 **searchServiceProblem** 属性，并返回。

2) 如果在搜索请求中包含了 **serviceType** 和/或 **userClass** 服务控制，则从候选搜索规则中除去所有的与这些服务控制不符合的搜索规则。如果这样使得剩余的列表为空，则产生一个问题为 **requestedServiceNotAvailable** 的服务错误；并在 **CommonResults** 的 **notification** 组件中，包括下列详述的信息，并返回：

— 一个取值为 **id-pr-unidentifiedOperation** 的 **searchServiceProblem** 属性；

— 如果在搜索请求中包括了 **serviceType** 服务控制，则返回一个取值为该服务控制的 **serviceType** 属性。

3) 将候选搜索规则划分为四个列表（某些列表可能是空的）：

— 一个 **GoodPermittedSR** 列表，包含了所有这样的候选搜索规则，即请求者具有对这些搜索规则的调用许可，且根据 ITU-T X.511 建议书 | ISO/IEC 9594-3 中第 13 节规定的搜索合法性验证规程，搜索请求符合这些搜索规则；

注 3 — 如果此列表非空，则没有理由去创建其他列表。

— 一个 **MatchProblemSR** 列表，包含了所有这样的候选搜索规则，即请求者具有对这些搜索规则的调用许可，且除了在一个或多个请求属性表中的 matchingUse 外，搜索请求符合这些搜索规则；

— 一个 **BadPermittedSR** 列表，包含了所有这样的候选搜索规则，即请求者具有对这些搜索规则的调用许可，但搜索请求不符合这些搜索规则；

— 一个 **DeniedSR** 列表，包含了所有这样的候选搜索规则，即请求者不具有对这些搜索规则的调用许可。

4) 如果 **GoodPermittedSR** 列表中包含了一个或多个空搜索规则，则使用本地算法选择这些空搜索规则中的一个作为控制搜索规则，然后返回。

5) 如果 **GoodPermittedSR** 列表非空，则除了那些具有最高 **userClass** 指示的搜索规则外，丢弃所有其他的搜索规则。

6) 使用本地算法，在 **GoodPermittedSR** 列表中，选择剩余的搜索规则中的一个作为控制搜索规则，然后返回。

注 4 — 如果在上述列表中有多个搜索规则可供选择，则在实现时应当将事件记入日志，以便为管理之用，因为搜索规则的定义可能需要改写。

7) 如果 **MatchProblemSR** 列表非空，则根据与上述 5)和 6)中规定的算法类似的一种算法选择其中一个搜索规则；产生一个服务错误以及如 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 13.4 节详述的相关信息，并且返回。

8) 如果 **DeniedSR** 列表为空，继续执行步骤 10)；否则，将那些搜索请求不符合的所有搜索规则从列表中丢弃，并且丢弃所有的空搜索规则。如果此时列表为空，则继续执行步骤 10)；否则产生一个问题为 **requestedServiceNotAvailable** 的服务控制；同时在 **CommonResults** 的 **notification** 组件中包含下面详述的子组件，并且返回：

— 一个取值为 **id-pr-unavailableOperation** 的 **searchServiceProblem** 属性；

— 如果在 **DeniedSR** 列表中的所有剩余搜索规则的 **serviceType** 组件都具有相同的值，则包括一个取值为该值的 **serviceType** 属性。

9) 如果 **BadPermittedSR** 列表为空，则产生一个问题为 **requestedServiceNotAvailable** 的服务错误；同时在 **CommonResults** 的 **notification** 组件中包含下面详述的子组件，并且返回：

— 一个取值为 **id-pr-unidentifiedOperation** 的 **searchServiceProblem** 属性。

10) 对于 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 13.1 节定义的规程中的每个按顺序标号的项，根据 **BadPermittedSR** 中剩余的搜索规则检查搜索请求，并且对于每个项：

— 如果搜索操作符合该项中的某些搜索规则，但不是所有的搜索规则，则丢弃那些不符合的搜索规则；

— 如果 **BadPermittedSR** 中目前仅拥有一个搜索规则，则执行 ITU-T X.511 建议书 | ISO/IEC 9594-3 的第 13 节规定的规程，然后返回；

— 否则，检查下一项。

11) 根据到此为止的规程，如果 **BadPermittedSR** 中目前仅含有搜索操作不符合的那些搜索规则，则产生一个问题为 **requestedServiceNotAvailable** 的服务错误；同时在 **CommonResults** 的 **notification** 组件中包含下面详述的子组件，并且返回：

— 一个取值为 **id-pr-unidentifiedOperation** 的 **searchServiceProblem** 属性；

— 如果 **BadPermittedSR** 中的所有搜索规则都规定了同一个服务类型，则包含一个取值为该服务类型的 **serviceType** 属性。

12) 对于 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 13.2 节定义的规程中的每个按顺序标号的项，根据 **BadPermittedSR** 中剩余的搜索规则检查搜索请求，并且对于每个项：

— 如果搜索操作符合该项中的某些搜索规则，但不是所有的搜索规则，则丢弃那些不符合的搜索规则；

— 如果 **BadPermittedSR** 中目前仅拥有一个搜索规则，则执行 ITU-T X.511 建议书 | ISO/IEC 9594-3 的第 13 节规定的规程，然后返回；

— 否则，检查下一项。

13) 对于 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 13.3 节定义的规程中的每个按顺序标号的项，根据 **BadPermittedSR** 中剩余的搜索规则检查搜索请求，并且对于每个项：

— 如果搜索操作符合该项中的某些搜索规则，但不是所有的搜索规则，则丢弃那些不符合的搜索规则；

— 如果 **BadPermittedSR** 中目前仅拥有一个搜索规则，则执行 ITU-T X.511 建议书 | ISO/IEC 9594-3 的第 13 节规定的规程，然后返回；

— 否则，检查下一项。

14) 产生一个问题为 **requestedServiceNotAvailable** 的服务错误；同时在 **CommonResults** 的 **notification** 组件中包含下面详述的子组件，并且返回：

— 一个取值为 **id-pr-unidentifiedOperation** 的 **searchServiceProblem** 属性；

— 如果 **BadPermittedSR** 列表中的所有搜索规则都指定了同一个服务类型，则包含一个取值为该服务类型的 **serviceType** 属性。

19.3.2.2.3 搜索规则检查规程(II)

当且仅当 DSA 支持服务特定的管理区时，本规程才相关。

如果 **searchRuleId** 不存在，且目标 DSE 的所有直接下级条目（上下文前缀）都是服务特定管理点，则返回一个问题为 **unwillingToPerform** 的 **serviceError**。然而，如果某些下级条目不是服务特定管理点，则为搜索评估选择相应的命名上下文并且返回。

如果 **searchRuleId** 存在，目标 DSE 的每个下级条目都被检查以验证它是否与目标 DSE 处于同一个服务特定管理区内。如果不是，则相应的命名上下文从搜索中排除。如果还有搜索操作可以继续进行的剩余命名上下文（包含正在执行的 DSA 中的那些命名上下文），在这些上下文中，则选择在 **searchRuleId** 中指定的搜索规则并且返回。如果没有搜索操作可以继续进行的剩余命名上下文，则产生一个问题为 **unwillingToPerform** 的 **serviceError** 并且返回。

注 — 如果知识信息在 DSA 与拥有上级命名上下文的 DSA 之间是一致的，则后一种情况不可能发生。

19.3.2.2.4 条目信息选择

对于已经匹配的条目，以及被选择作为分等级选择中的一部分的条目而言，所选择的属性信息为下列信息的交集：

a) 由 **searchArgument.selection** 所规定的信息，可能被缺省的上下文规范修改，对于已经匹配的条目，还包括由 **searchArgument.matchedValuesOnly** 所规定的信息；

b) 由控制搜索规则（如果有的话）所决定的信息。

该条目信息被加入到 **searchResult.entryInformation** 中的条目列表中。

仅将尺寸（类型和所有的值）不大于 **attributeSizeLimit** 的属性加入。

19.3.2.2.5 搜索(I)规程

这是一个递归规程，应用于一个起始于某个给定的目标条目 **e** 的**搜索**请求。它搜索目标条目 **e**，然后处理作为 **e** 的直接下级的那些 DSE。当整个子树都需要被搜索时，该规程被自己递归调用。该规程包括如下步骤，如图 19 所示：

1) 如果 DSE **e** 的类型为 **cp**（即一个作为上下文前缀的 DSE），则检查 **exclusions** 变量中是否有任意一个元素是 **e** 的 DN 的一个前缀。

a) 如果是，则返回。

b) 否则，调用检查适宜性规程。

i) 如果 **e** 不适合，则按照如下所述构建一个 **continuationReference**，并且将其加入到 **SRContinuationList** 中：

— **targetObject** 被设置为 DSE **e** 的 DN；

— **nameResolutionPhase** 的 **operationProgres** 被设置为 **proceeding**，且 **nextRDNtoBeResolved** 被设置为 **e** 中的 RDN 的数量；

— **continuationReference** 中的所有其他组件都不变。

然后返回。

注 1 — 当一个**搜索**子请求被链接到一个镜像提供者时，这是唯一的情况。换句话说，这样一个链接子请求的目标对象总是一个上下文前缀。

ii) 否则，将 **e** 的识别名加入到 **ChainingResults** 的 **alreadySearched** 中。

注 2 — **alreadySearched** 仅包含上下文前缀。

2) 如果 **e** 的类型为 **alias**，且 **SearchArgument** 中的 **searchAliases** 为 **TRUE**，则调用**搜索别名**规程，然后返回。

3) 如果 **subset** 是 **oneLevel**，则继续执行步骤 6)。

注 3 — **e** 在这个点上，不可能是不完整的下级，因为针对上下文前缀检查适宜性已经保证了这种情况不会发生。

4) 如果 **subset** 为 **baseObject**，或者如果 **entryOnly** 为 **TRUE**，则继续本步骤；否则转到步骤 5)。

如果下列情况之一是正确的：

a) **e** 的类型为 **subentry**，且服务控制 **subentry** 被设置；或者

b) **e** 的类型不是 **subentry**，且服务控制 **subentry** 未被设置，则执行下列步骤：

i) 检查 ACI。如果本操作不被允许，则返回。

ii) 将 **SearchArgument.filter** 中指定的过滤器变量应用到 DSE **e**。确保对过滤器中使用的所有属性的访问是被准予的，如同 ITU-T X.501 建议书 | ISO/IEC 9594-2 中的定义。如果过滤器匹配，且根据分等级选择，条目未被排除，则按照 19.3.2.2.3 节中的规定增加属性信息。

iii) 如果在**搜索**请求中包含 **hierarchySelection** 搜索控制（可能被某个搜索规则规范所修改），同时本条目是一个拥有多个成员的分等级分组中的一部分，且不仅 **self** 指示被设置，则调用**分等级选择(I)**规程。

然后返回。

5) 如果 **subset** 为 **subtree**（且 **entryOnly** 不为 **TRUE**），并且下列中的一个是正确的：

a) **e** 的类型为 **subentry**，且服务控制 **subentry** 被设置；或者

b) **e** 的类型不是 **subentry**，且服务控制 **subentry** 未被设置，则执行下列步骤：

i) 检查 ACI。如果本操作不被允许，则转到步骤 6)。

ii) 将 **SearchArgument.filter** 中指定的过滤器变量应用到 DSE **e**。确保对过滤器中使用的所有属性的访问是被准予的，如 ITU-T X.501 建议书 | ISO/IEC 9594-2 中的定义。如果过滤器匹配，且根据分等级选择，条目未被排除，则按照 19.3.2.2.3 节中的规定增加属性信息。

iii) 如果在**搜索**请求中包含 **hierarchySelection** 服务控制（可能被某个搜索规则规范所修改），同时本条目是一个拥有多个成员的分等级分组中的一部分，且不仅 **self** 指示被设置，则调用**分等级选择(I)**规程。

iv) 继续执行步骤 6)。

6) 如果 **e** 的类型为 **nssr**，则向 **SRcontinuationList** 中增加一个继续引用，其组件如下所述：

— **targetObject** 被设置为 DSE **e** 的主识别名（可替代识别值可能包含在 RDN 中）；

— **aliasedRDNs** 缺失；

— **nameResolutionPhase** 的 **operationProgress** 被设置为 **completed**，且 **nextRDNtoBeResolved** 缺失；

— **rdnsResolved** 缺失；

— **referenceType** 被设置为 **nssr**；

— **accessPoints** 被设置为 **AccessPointInformation**，该值根据 **nonSpecificKnowledge** 属性中发现的值推导而来。

7) 对处于目标 DSE **e** 的直接下级的所有 DSE **e'**进行处理，直到所有的下级 DSE 都处理完成。如果 **e** 是处于某个服务特定管理区内，则仅对那些作为同一服务特定管理区内的直接下级 DSE 才进行处理。如果 **e** 处于服务特定管理区外，则不应当对那些作为某一个服务特定管理区内的直接下级 DSE 进行处理。在这个环回过程中，如果在 **searchResult.entryInformation** 中的已匹配条目的列表超出了尺寸限制，或者时间限制或管理限制被超越，则在 **partialOutcomeQualifier** 中设置相应的 **limitProblem**，并返回。

注 4 — 在每次 **searchResult** 被更新时，都隐含应用了对尺寸限制的检查。

a) 如果 DSE **e'**的类型为 **subr**，不是 **cp**，且不表示一个下级条目，该下级条目是一个服务特定的管理点，则向 **SRcontinuationList** 中增加一个继续引用，其组件如下所述：

— **targetObject** 被设置为 DSE **e** 的主识别名（可替代识别值可能包含在 RDN 中）；

— **aliasedRDNs** 缺失；

— **nameResolutionPhase** 的 **operationProgress** 被设置为 **completed**，且 **nextRDNtoBeResolved** 缺失；

— **rdnsResolved** 缺失；

— **referenceType** 被设置为 **subr**；

— **accessPoints** 被设置为包含在 DSE **e'**的 **specificKnowledge** 属性中的访问点信息。

注 5 — 如果 **e'** 的类型既是 **cp** 又是 **subr**，则能够潜在地生成一个搜索子请求，或者根据下级引用，或者根据上级知识，但不会两个都是。本规程使用后一种（即根据 cp 中发现的提供者引用）。

b) 对于所有的情况：

i) 如果 **subset** 为 **oneLevel**，则设置 **entryOnly** 为 **TRUE**。

ii) 为目标 DSE **e'**递归执行 **Search (I)**规程。

8) 如果所有的下级都已经被处理，则返回到操作调度程序以便执行进一步的处理。

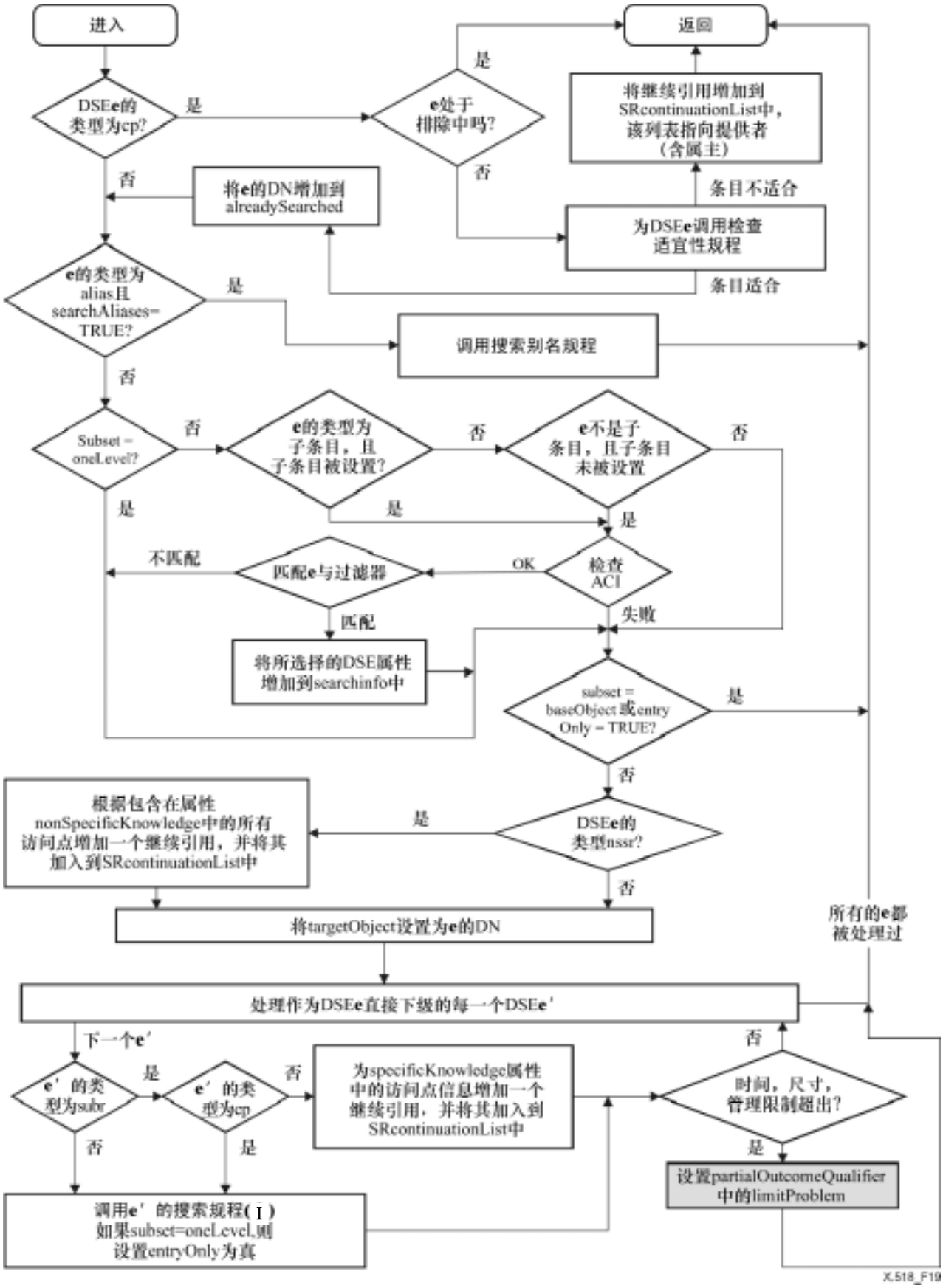

进入
返回
DSE e的类型为cp?
是
否
e处于排除中吗?
是
否
将继续引用增加到SRcontinuationList中，该列表指向提供者（含属主）
条目不适合
为DSE e调用检查适宜性规程
条目适合
将e的DN增加到alreadySearched
e的类型为alias且searchAliases=TRUE?
是
否
调用搜索别名规程
Subset = oneLevel?
否
是
e的类型为子条目，且子条目被设置?
否
是
e不是子条目，且子条目未被设置
否
是
检查ACI
OK
匹配e与过滤器
不匹配
匹配
失败
将所选择的DSE属性增加到searchinfo中
subset = baseObject或entry Only = TRUE?
是
否
DSE e的类型nssr?
是
否
根据包含在属性nonSpecificKnowledge中的所有访问点增加一个继续引用，并将其加入到SRcontinuationList中
将targetObject设置为e的DN
所有的e都被处理过
处理作为DSE e直接下级的每一个DSE e'
下一个e'
e' 的类型为subr
是
否
e' 的类型为cp
否
是
为specificKnowledge属性中的访问点信息增加一个继续引用，并将其加入到SRcontinuationList中
时间，尺寸，管理限制超出?
否
是
设置partialOutcomeQualifier中的limitProblem
调用e' 的搜索规程(I)
如果subset=oneLevel,则设置entryOnly为真
X.518_F19

图 19－搜索(I)规程

19.3.2.2.6 搜索(II)规程

如果一个搜索请求是源自某个 DSA 处的一个请求分解，而该 DSA 是接收请求的 DSA，则当该请求被处理时，应用本规范。本规程处理目标 DSE **e** 之下的 DSE，并且为每个对象条目调用**搜索(I)**规程：

1) 对处于目标 DSE **e** 的直接下级的所有 DSE **e'**进行处理，直到所有的下级 DSE 都处理完成。如果所有的下级都已经被处理，则返回到操作调度程序以便执行进一步的处理。

2) 如果 DSE 的类型不是 **cp**，则忽略该 DSE。返回到步骤 1)。

3) 调用**检查适宜性**。如果条目适合，则转到步骤 4)；否则忽略此 DSE 并返回到步骤 1)。

4) 对 DSE **e'**执行**搜索规程(I)**，如 19.3.2.2 节中的描述。如果 DSE 的类型为 **alias** 且 **subset** 参数的值被设置为 **oneLevel**，则在调用**搜索(I)**规程时，设置 **ChainingArguments.entryOnly** 为 **TRUE**。返回到步骤 1)。

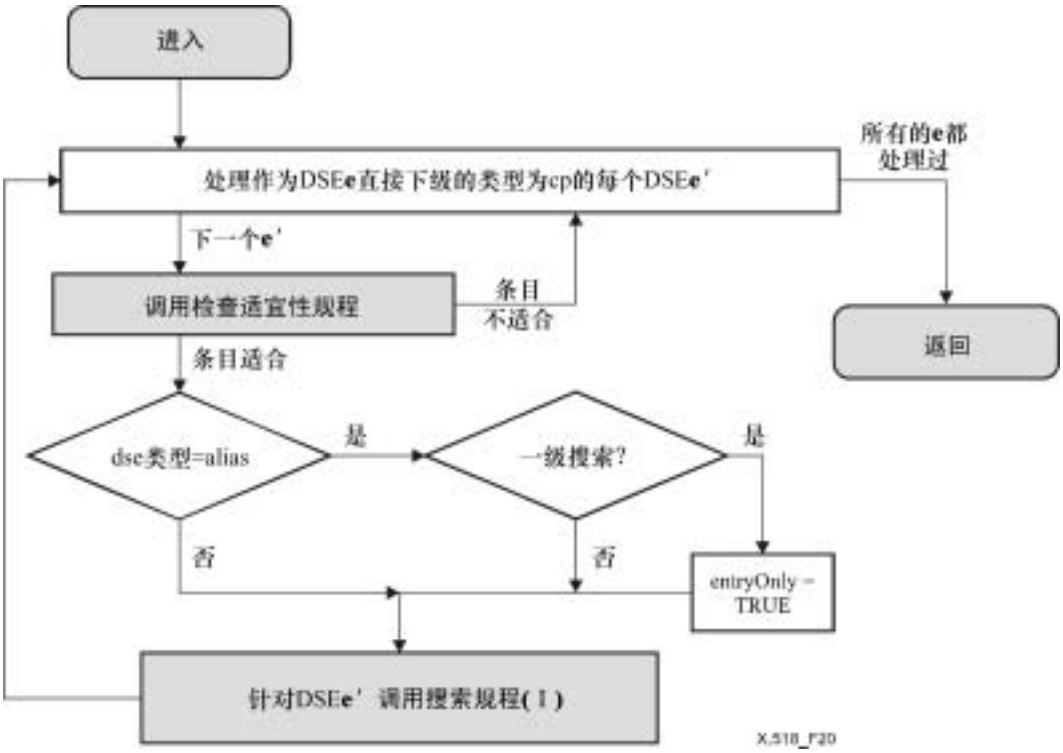


图 20－搜索 (II) 规程

19.3.2.2.7 搜索别名规程

在一个**搜索**请求的处理过程中，当遇到一个类型为 **alias** 的 DSE 时，运行本规程（见图 21）：

1) 如果 **subset** 为 **baseObject** 或 **oneLevel**，则转到步骤 4)。

2) 如果 **aliasedEntryName** 是 **targetObject** 或 **baseObject** 的一个前缀，或者是 **ChainingArguments.traceInformation** 中的 **targetObject** 的任意一个之前值，则该别名被排除出搜索操作，因为这会引起一个递归搜索导致重复的结果。

3) 如果 **targetObject** 或 **baseObject** 或 **ChainingArguments.traceInformation** 中的 **targetObject** 的任意一个之前值是 **aliasedEntryName** 的一个前缀，则不需要对此别名作任何特殊的处理，因为无论如何被起别名的子树都将被搜索。

   注 — 对于上述的两种情况，由于别名解除引用，**baseObject** 可能不是 **targetObject** 的前缀。

4) 如果该搜索是在一个服务特定管理区内执行的，且如果服务特定管理点不是 **aliasedEntryName** 的一个前缀，则不需要对此别名作任何特殊的处理，因为被起别名的条目是在服务特定管理区之外。

5) 构建一个 **DSP** 请求，其中 **targetObject** 被设置为 **aliasedEntryName**。如果 **subset** 为 **oneLevel**，则设置 **entryOnly** 为 **TRUE**。然后为该请求调用**操作调度程序**，在本地继续执行。

6) 如果**操作调度程序**返回了一个 **referral** 错误、或忙、或不可用错误，则在 **SearchResult** 的 **partialOutcomeQualifier** 中增加（或构建并增加）一个继续引用，然后返回。

7) 如果**操作调度程序**返回了其他错误，则丢弃并返回。

8) 如果**操作调度程序**返回了一个 **SearchResult**，则：

i) 如果结果被签名、加密、或签名并加密，则将其加入到 **SearchResult** 的 **uncorrelatedSearchInfo** 中。

ii) 如果结果没有被签名、加密、或签名并加密，则将其加入到 **SearchResult** 的 **searchInfo** 中。然后返回。

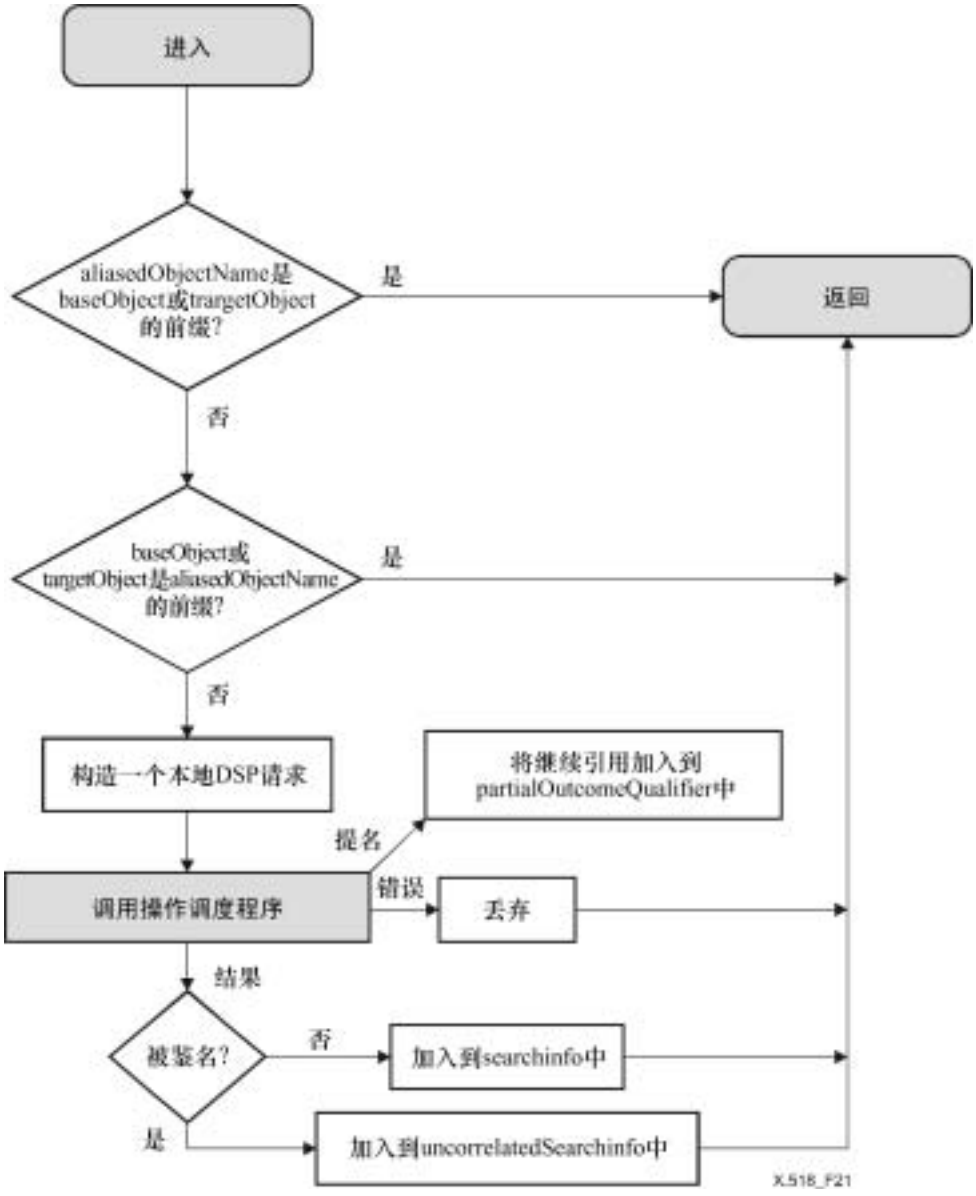


图 21－搜索别名规程

19.3.2.2.8 分等级选择规程(I)

在一个规定了分等级选择的搜索请求的处理过程中，当遇到一个分等级分组中的某个成员时，运行本规程。

a) 如果 DSA 不支持的一个分等级选择存在，则返回如下信息：

— 一个问题为 **requestedServiceNotAvailable** 的 **serviceError**；

— 一个取值为 **id-pr-unavailableHierarchySelect** 的通知属性 **searchServiceProblem**；

— 一个通知属性 **serviceType**，其取值与搜索规则中的 **serviceType** 组件的值相同；以及

— 一个指示了非法选择的通知属性 **hierarchySelectList**。

b) 否则，按照 19.3.2.2.4 节中定义的那样，加入分等级选择所定义的所有条目。如果这样导致没有任何条目被加入，即分等级选择仅仅指定了一些不存在的条目，则设置全局变量 **emptyHierarchySelect**。

20 继续引用规程

调用本节中的规程可以处理由其他规程所创建的继续引用列表（**NRcontinuationList** 或 **SRcontinuationList**）。

继续引用规程由图 24、25 和 26 中显示的步骤组成。第一阶段是从继续引用列表中标识那些具有公共目标对象组件的继续引用集。这些引用集合是根据与 DIT 中同一个条目相关的下级引用集或非特定下级引用集而创建的。在每个引用集中，可能有的继续引用不止出现一次。这些集合必须被扫描，并且发现的任何重复引用都必须被丢弃。

这些集合（每个都拥有一个不同的 **targetObject** 组件）可能独立地被 DSA 处理，或者是顺序处理、或者是并行处理，因为不会有从任意两个集合中返回同一个结果的风险存在。然而，在一个集合中对每个继续引用的处理，在一个继续引用中对每个 **AccessPointInformation** 的处理，以及在一个 **AccessPointInformation** 中对每个访问点的处理，都必须被控制，否则可能会出现重复结果，如 20.1 节中的描述。

在 **APInfo** 规程中采纳的规程是一个接一个地处理包含在一个单独 **AccessPointInformation** 中的访问点集合。这些都指向同一个命名上下文（或其拷贝）（或者在 NSSR 的情况下，可能是指向一个 DSA 所拥有的一个命名上下文集）。如果第一个访问点产生了一个结果或一个硬错误，则后续的访问点不需要再被处理。然而，如果错误是一个软错误，即一个 **serviceError**（问题可能为 **busy**、**unavailable**、**unwillingToPerform**、**invalidReference** 或 **administrativeLimitExceeded**），则作为一个本地选项，该 DSA 可能会从集合中选择另一个访问点来处理。

对一个继续引用集中的 **AccessPointInformation** 值的处理应当以一种统一的方式进行，而不考虑继续引用源自何处。（这是因为处于一个单独条目之下的两个类型为 **subr** 的 DSE 将产生两个继续引用，其中每个都包含一个 **AccessPointInformation** 值，然而对于一个类型为 **nssr** 的 DSE，如果 **nssr** 指向同样的这两个下级 DSE，假设它们由不同的 DSA 所拥有，则将产生一个继续引用，该引用包含了两个 **AccessPointInformation** 值。）

accessPointInformation 值的处理可能是按顺序的，也可能是并列的，在 20.1 节中描述。并列策略更易于产生重复结果。重复结果必须总是被丢弃。

20.1 镜像存在时的链接策略

镜像存在时，当某个 DSA 必须将一个请求多链接到多个 DSA 时，该 DSA 可能会在两个不同的策略之间进行选择。如果 DSA 必须处理针对同一个 **targetObject** 的多个继续引用时，总是会发生此选择。这可能会在两种情况下发生，一种是在名字解析过程中由于 NSSR 分解而引起的多链接情况（如图 22 所示），另一种是在一个多对象操作的评估过程中由于请求分解而引起的情况（如图 23 所示）。

这些策略的目的是为了解决在请求的多链接中使用镜像信息时所出现的重复结果和重复处理的问题（由于 NSSR 分解或者请求分解而引起）。例如，在图 22 中，由于在 DSE B 中所拥有的 NSSR，DSA 1 将一个请求多链接到 DSA 2 和 DSA 3。如果允许使用镜像信息，则 DSA 2 和 DSA 3 可能将链接操作应用到分别起始于 X 和 Y 的两棵子树上。

类似的，在图 23 中（由于请求分解的结果）DSA 1 将请求多链接到两个下级引用，这两个下级引用分别由 DSE X 和 DSE Y 所拥有。同样的，如果允许使用镜像信息，则 DSA 2 和 DSA 3 可能将链接操作应用到分别起始于 X 和 Y 的两棵子树上。

为了处理这种重复问题，当多链接到多个 DSA 的请求具有同一个 **targetObject** 时，一个 DSA 可能选择下面的策略之一。

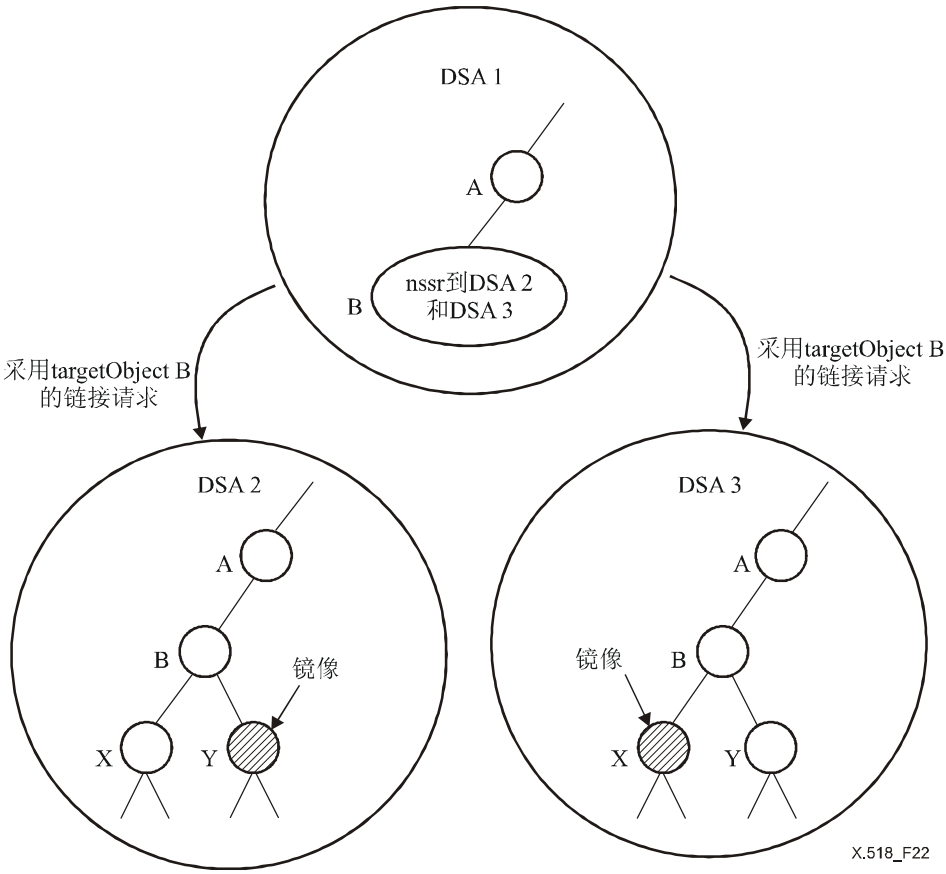


图 22－在名字解析阶段由于NSSR引起的多链接

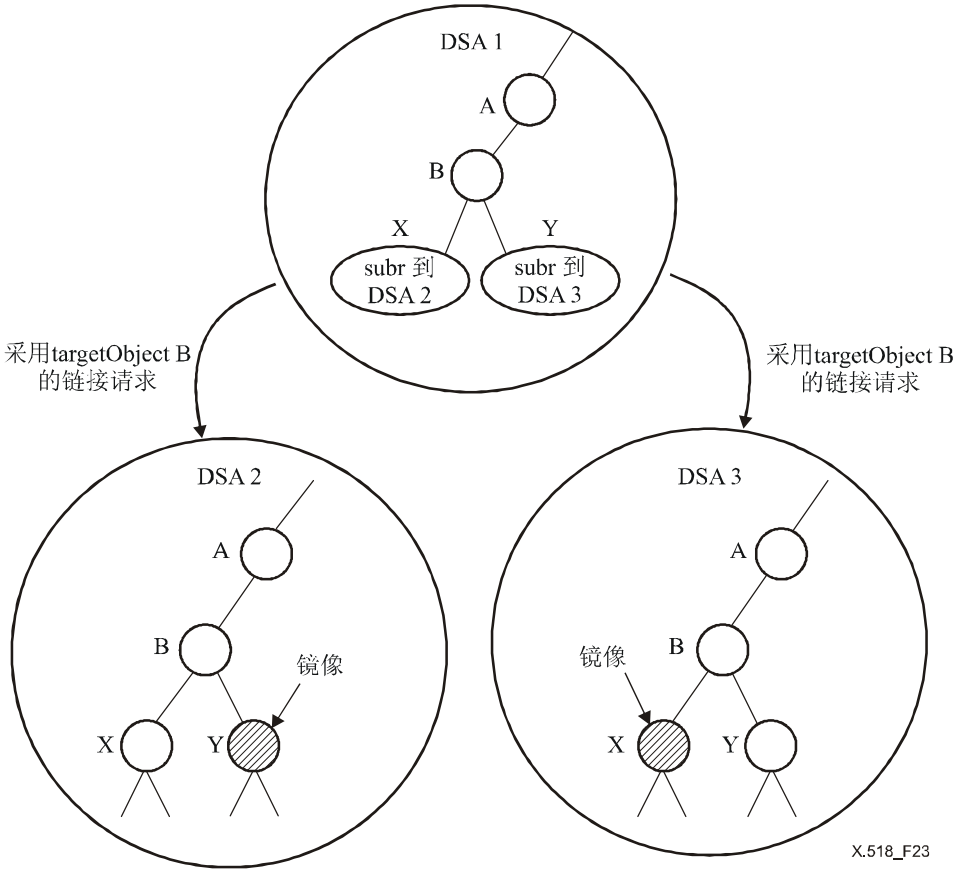


图 23－使用下级引用的多链接请求分解

20.1.1 仅使用属主策略

在某个搜索或列表评估过程中，当执行一个由于 NSSR 分解或请求分解而引起的并行或顺序多链接时，一个 DSA 可能会选择本策略来防止使用镜像信息。为了使用本策略，在某个搜索或列表操作评估过程中，**ChainingArguments** 的组件 **excludeShadows** 被设置为 **TRUE**。如果在名字解析过程中遇到了 NSSR，则 DSA 可能会设置 **nameResolveOnMaster** 为 **TRUE** 来确保仅遵循一个单独的路径。如果遇到了 NSSR，且操作是号码簿修改操作中的一种，则 **nameResolveOnMaster** 必须被设置为 **TRUE**。在任何一种情况下，仅有拥有与操作相关的主条目的 DSA 才应当执行此操作。在并行和顺序多链接过程中都可以使用此“仅使用属主”策略。

注 — 设置 **nameResolveOnMaster** 为 **TRUE** 就排除了在名字解析过程中出现多路径的可能性，这是通过如下方式实现的：

1) 忽略镜像条目和条目的可写拷贝；以及
2) 确保仅有一个 DSA 可能继续进行名字解析过程，否则在一种复杂 DIT 分布的情况下，可能会允许多条路径来继续处理。

这是通过仅允许一个 DSA 来继续进行名字解析而实现的，该 DSA 拥有与目标对象名字中的第一个 **nextRDNToBeResolved** RDN 相应的主条目。其他任何 DSA 都不能够继续处理，即使它们可能拥有与更多的目标对象名字相匹配的主条目。

20.1.2 并行策略

使用本策略，一个 DSA 将所有的链接请求通过并行的多链接发出。在搜索或列表评估阶段，以及 NSSR 的名字解析阶段，可能会使用本策略。这将会允许使用镜像信息来处理链接请求，但是可能会引起对操作的重复处理和产生重复结果。如果一个 DSA 选择了本策略，则它必须从返回的操作结果中将重复结果删除。

如果请求的是一个签名结果，则删除重复结果是不可能的，因此如果在搜索评估阶段要求签名结果时，DSA 不能使用本策略，除非 **excludeShadows** 也被设置。

20.1.3 顺序策略

本策略通过使用顺序多链接来处理一个搜索分解或一个 NSSR 分解的链接请求（或链接子请求），可以避免产生重复结果。每个链接请求都被一个接着一个地处理。

在 NSSR 分解的情况下，如果某个请求有一个结果或一个硬错误返回，则后续的请求不必再被链接。如果有一个软错误返回，则可能会有一个后续的请求被链接，或者将该软错误返回给请求者，依赖于本地策略。

在搜索评估的情况下，**ChainingArguments** 的 **exclusions** 组件被设置为已经被处理过的的 RDN 集合。这是通过将 **ChainingResults.alreadySearched** 中的元素合并到下一个链接请求的 **exclusions** 变量中来实现的。这是唯一的一种策略可以在搜索评估阶段完全地避免重复。

没有为列表评估定义顺序策略（尽管也可能会使用顺序多链接），因为一个上级 DSA 没有办法将特定的下级从后续列表子请求的返回中排除出去（注意 **excludeShadows** 不排除特定的下级，但它是一种更粗糙的方式将所有的镜像和可写拷贝都排除出去）。

20.2 向一个远端DSA发起链接子请求

在发起一个子请求之前，当 DSA 必须建立一个与远端 DSA 之间的连接时，它必须先运行一个 **dSABind** 操作。对连接的管理不在本系列号码簿规范的定义范围之内。当连接不能够被建立，或者 DSA 由于本地原因决定不建立连接时，与另一个 DSA 之间的连接被认为是不可用的。在这种情况下，**dSABind** 失败。何时停止对连接建立的尝试并宣称一个连接是不可用的，由本地决策来决定。

当一个 DSA 尝试向另一个 DSA 发起 **dSABind**，但接收到一个 **directoryBindError** 时，则子请求的发起失败。

20.3 规程的参数

20.3.1 变量

这些规程使用如下变量：

— 要处理的继续引用的列表，在 **NRcontinuationList**（对于**名字解析继续引用**规程）和 **SRcontinuationList**（分别对于**列表继续引用**和**搜索继续引用**规程）中；

— 操作变量的 **CommonArguments**；

— **ChainingArguments**。

20.3.2 结果

这些规程产生如下结果：

— 如果选择了链接，则针对所发起的链接请求接收到的结果/错误列表；

— 一个更新后的未处理的继续引用的列表，在 **continuationList** 中。

20.3.3 错误

这些规程能够返回如下错误之一：

— 若一个提名已经被产生但它不在 **scopeOfReferral** 内时，则产生一个问题为 **outOfScope** 的 **serviceError**；

— 若一个非法的知识引用已经被检测到，则返回一个问题为 **ditError** 的 **serviceError**；

— 若所有的来自 NSSR 分解的子请求都返回 **unableToProceed** 时，则返回一个问题为 **noSuchObject** 的 **nameError**；

— 某个链接子请求返回的任何其他错误；

— 若链接未被选择且 **operationProgress.nameResolutionPhase** 被设置为 **notStarted** 或 **proceeding** 时，则返回一个 **referral**。

20.4 规程的定义

如果 **operationProgress.nameResolutionPhase** 被设置为 **notStarted** 或 **proceeding**，则必须遵循 20.4.1 节（**名字解析继续引用**规程）中的规程。多条目查询操作列表和搜索将分别调用 20.4.2 节和 20.4.3 节的规程。

20.4.1 名字解析继续引用规程

名字解析继续引用规程由图 24 中显示的步骤组成。本规程的基本原则是顺序地处理在名字解析阶段创建的继续引用集。对于 **NRcontinuationList** 中包含的每个继续引用 **C**，按照一个选定的顺序，必须运行下面的步骤，直到所有的继续引用都被处理过或者有一个错误或结果已经返回。如果所有的引用都已经被处理，则返回到**操作调度程序**继续执行，并且使用**结果合并**规程来处理所接收到的结果或提名。

1) 检查 **chainingProhibited** 是否被设置。如果已经被设置，则 DSA 不允许链接。根据本地策略，或者有一个问题为 **chainingRequired** 的 **serviceError**，或者一个**提名**被返回到**操作调度程序**。

2) 如果 **chainingProhibited** 没有被设置，则检查**本地策略**是否允许链接。如果链接不被允许，则返回一个**提名**。如果**本地策略**允许链接，则继续下一步步骤。

3) 处理在 **NRcontinuationList** 中发现的继续引用列表中的每一个继续引用。如果不再有未处理过的继续引用，则返回 **serviceError**。

4) 处理 **NRcontinuationList** 中的下一个继续引用 C。如果它是一个 NSSR，则继续执行步骤 5)。如果它不是一个 NSSR，则调用 **APInfo** 规程来处理它。区分对 **APInfo** 规程的调用可能会返回的两种结果：

   — 如果 **APInfo** 规程返回了一个**空结果**，则继续执行步骤 3)，处理下一个继续引用。

   — 如果 **APInfo** 规程返回了一个**错误**、**提名或结果**，则将其返回。

5) 在这种情况下，继续引用的类型为 NSSR，且 DSA 可以选择进行顺序的或并行的链接，依赖于策略的本地选择。如果 NSSR 将被顺序地处理，则继续执行步骤 6)。如果它将被并行地处理，则对于 NSSR 中的每个 **AccessPointInformation (API)**，**APInfo** 规程都被调用，因此它们被并行地执行。等待所有的 API 都被处理，即等待所有对 **APInfo** 规程的调用都返回。按照如下顺序检查所有的从 **APInfo** 规程的调用中接收到的结果：

   — 如果所有的调用都返回一个问题为 **unableToProceed** 的 **serviceError**，且 **partialNameResolution** 为 **FALSE**，则返回 **nameError**。

   — 如果所有的调用都返回一个问题为 **unableToProceed** 的 **serviceError**，且 **partialNameResolution** 为 **TRUE**，则在结果中设置 **partialName** 为 **TRUE**，**nameResolutionPhase** 为 **completed**，且设置为**条目适合**（这将是为 **lastEntryFound** 而设置），然后转到适当的操作评估。

— 如果接收到一个或多个**结果**，则**丢弃可能的重复**并且返回结果。

— 如果接收到一个不是 **serviceError** 的**错误**（例如，一个 **nameError**），则返回该**错误**。

— 否则根据本地选择，向**操作调度程序**返回一个**提名**或 **serviceError**。

6) 从 NSSR 的 **API** 集合中选择下一个未被处理的 **API**，然后继续执行步骤 7)。如果所有的 **API** 都已经被处理过了，则检查是否所有的对 **APInfo** 规程的调用都返回一个问题为 **unableToProceed** 的 **serviceError**。

— 如果是这样，且 **partialNameResolution** 为 **FALSE**，则不能发现条目且返回一个 **nameError**。如果是这样，且 **partialnameResolution** 为 **TRUE**，则在结果中设置 **partialName** 为 **TRUE**，**nameResolutionPhase** 为 **completed**，且设置为**条目适合**（这将是为 **lastEntryFound** 而设置），然后转到适当的操作评估。如果不是这样，则根据本地选择，返回一个**提名**或一个 **serviceError**。

7) 调用 **APInfo** 规程。区分对 **APInfo** 规程的调用可能会返回的结果：

— 如果接收到一个问题为 **unableToProceed** 的 **serviceError**，则尝试另一个访问点。继续执行步骤 6)。

— 如果接收到一个问题为 **busy**、**unavailable**、**unwillingToPerform** 或 **invalidReference** 的 **serviceError**，则被指示的错误可能是一个具有临时特性的错误，是否尝试将请求链接到另一个 DSA，是一个本地选择。如果选择要尝试另一个 DSA，则继续执行步骤 6)；否则根据本地选择，返回一个**提名**或一个 **serviceError**。

— 如果接收到一个错误，但不是问题为 **busy**、**unavailable**、**unwillingToPerform**、**invalidReference** 或 **unableToProceed** 的 **serviceError**，则该错误必须被返回到**操作调度程序**。如果 **serviceError** 为 **invalidReference**，则在返回给请求者之前必须被转换为 **ditError**。

— 如果接收到一个**结果**或一个**提名**，则将其返回到**操作调度程序**。

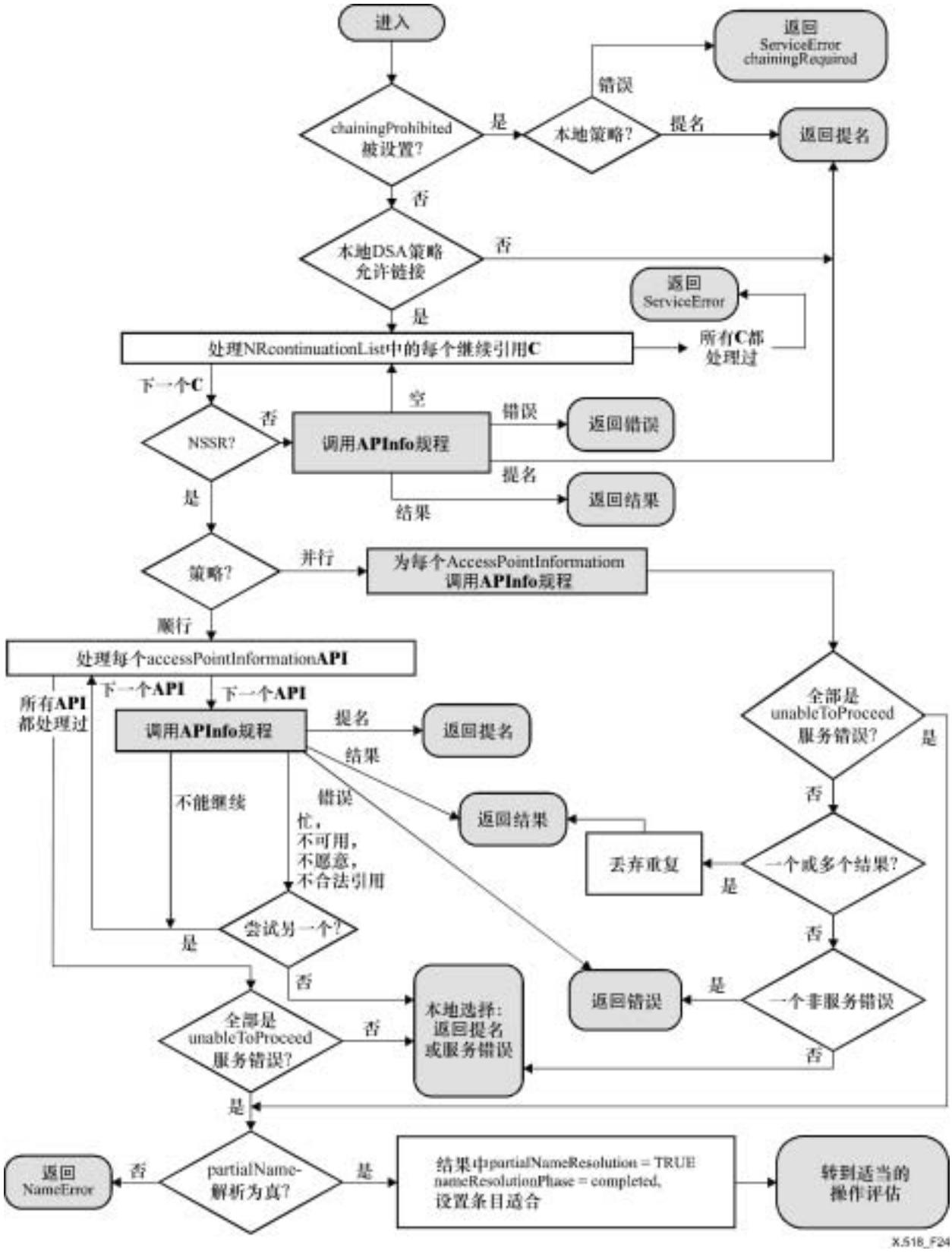


图 24－名字解析继续引用规程

20.4.2 列表继续引用规程

列表继续引用规程由图 25 中显示的步骤组成。当本地 DSA 不能够满足某个列表请求，且为了链接或提名已经有一系列的继续引用被加入到 **SRcontinuationList** 中时，本规程被调用。所有的这些继续引用 (CR) 都具有相同的 **targetObject**。这些具有 **referenceType** 为 **nssr** 的 CR 具有一个或多个 **AccessPointInformation** 值（**API**），而同时其他类型的 CR 仅具有一个 **API**。这些 **API** 中的每一个都被抽取出来，并考虑用作链接或提名。

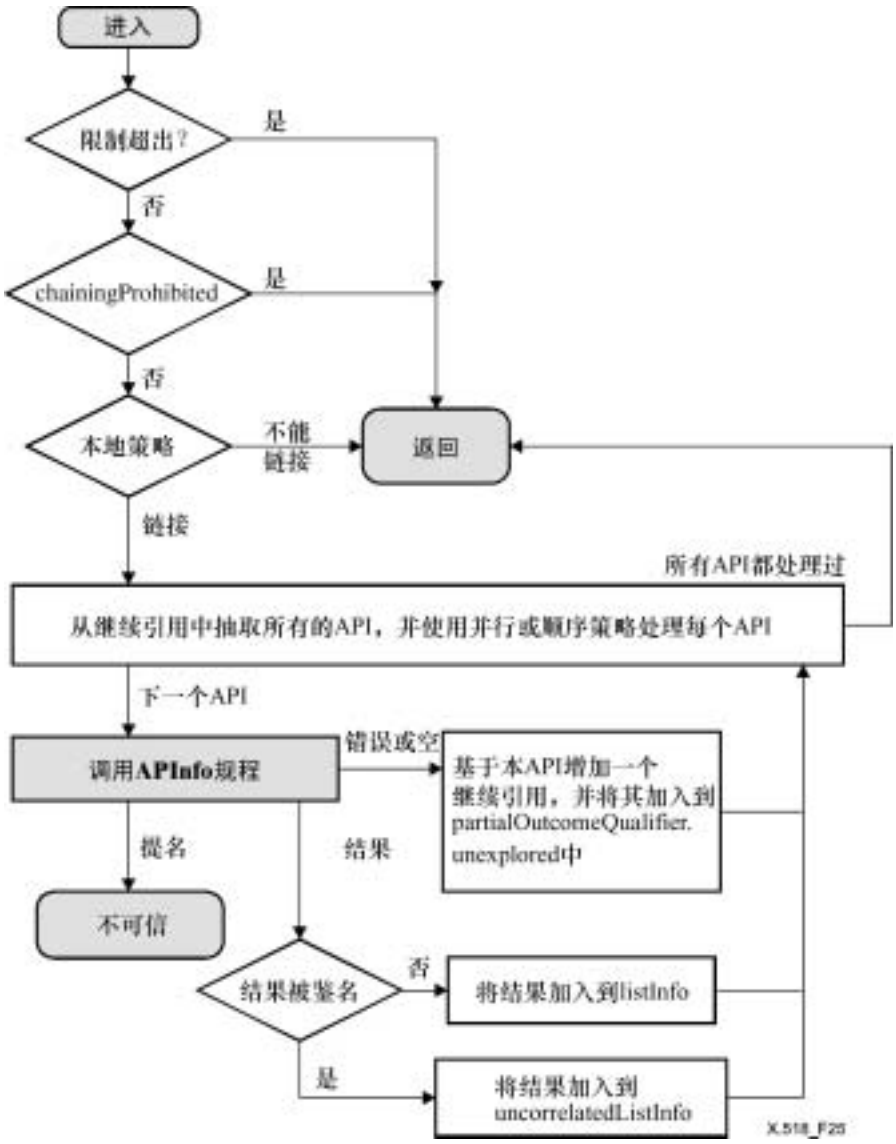


图 25－列表继续引用规程

下述步骤必须被执行：

1) 迄今为止，如果有任意一个限制问题被超越，则返回到**操作调度程序**来继续进行**结果合并**规程。

2) 如果 **CommonArguments.serviceControls** 中的 **chainingProhibited** 标志被设置，或者由于 DSA 的本地操作策略，DSA 决定不做任何链接，则 DSA 必须直接返回到**操作调度程序**来继续进行**结果合并**规程。

3) 根据 **SRcontinuationList** 中的每个继续引用的 **accessPoints** 组件，创建一个 **AccessPointInformation** 值的集合。

   使用并行或顺序策略来处理每个 **API**，如下所述：

   i) 对集合中的下一个 **API** 调用 **APInfo** 规程。

   ii) 如果有一个结果返回，且没有被签名的话，则将其加入到 **listInfo** 中，或者如果结果被签名的话，则将其加入到 **uncorrelatedListInfo** 中。

   iii) 如果返回的是一个错误或空结果，则它意味着 **APInfo** 已经尝试了 **API** 中的所有访问点而没有成功。则基于本地操作和安全策略，或者忽略并继续下一个 **API**，或者向 **partialOutcomeQualifier** 中增加一个基于此 **API** 的继续引用。

注 — 从 **APInfo** 中取回一个提名是不可信任的。任何“提名”必须以 **partialOutcomeQualifier** 中的 **unexplored** 形式出现。

4) 当所有的 **APIs** 都被处理后，返回到**操作调度程序**。

20.4.3 搜索继续引用规程

搜索继续引用规程由图 26 中显示的步骤组成。当本地 DSA 不能够满足某个搜索请求，且为了链接或提名已经有一系列的继续引用被加入到 **SRcontinuationList** 中时，本规程被调用。本规程非常类似于**列表继续引用**规程。区别是在本规程中，**SRcontinuationList** 中的继续引用 (CR) 可能具有不同的 **targetObject** 值。因此，这些继续引用被分类为不同的继续引用集合，每个集合拥有相同的 **targetObject**。另外，在链接变量中的 **exclusions** 的用法，以及链接结果中的 **alreadySearched** 的用法都被定义，因为对于搜索来说这是一个重要的策略。**exclusions** 和 **alreadySearched** 的用法被应用到对具有相同 **targetObject** 的每个继续引用集合的处理中。

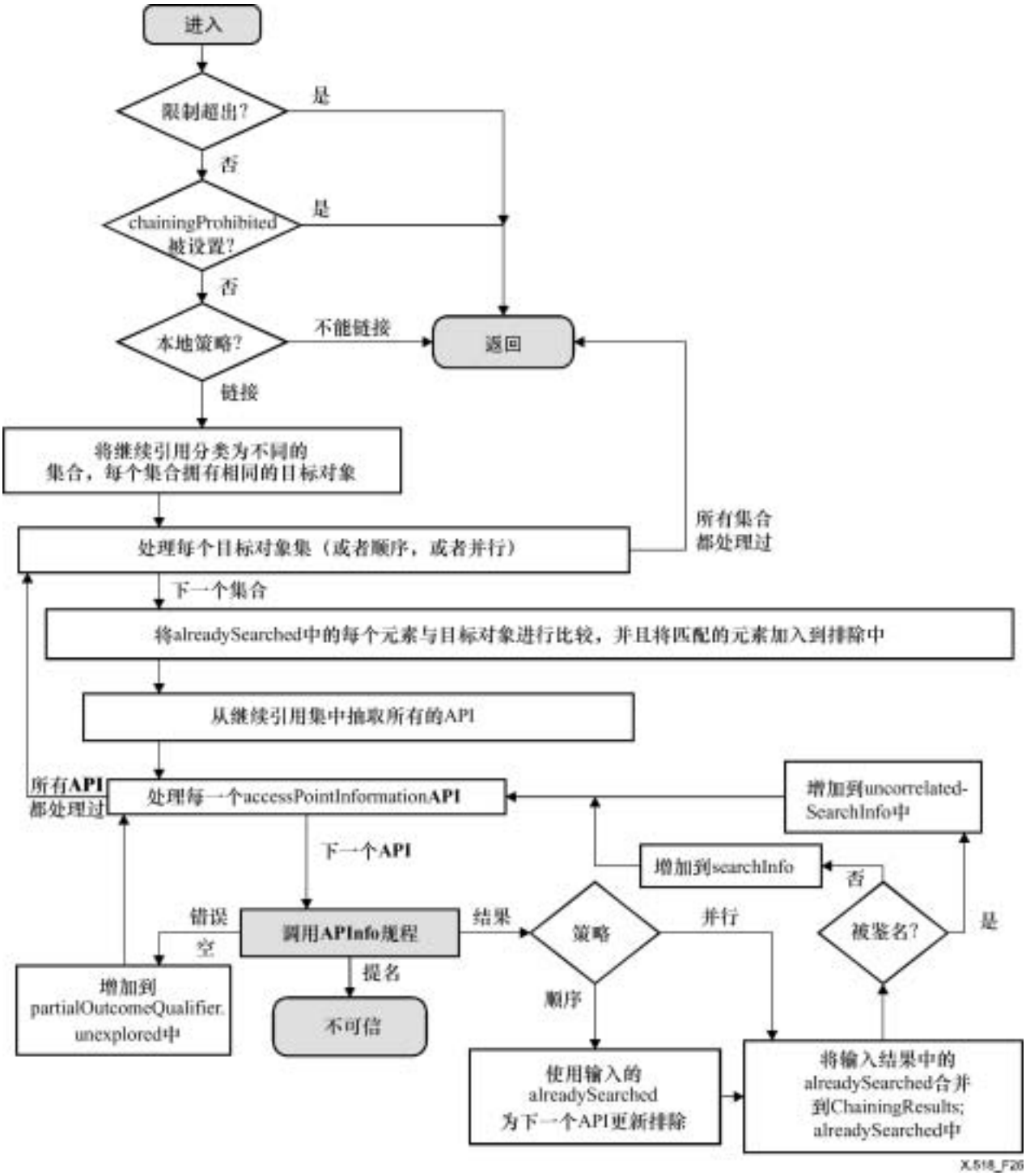


图 26－搜索继续引用规程

下述步骤必须被执行：

1) 迄今为止，如果有任意一个限制问题被超越，则返回到**操作调度程序**来继续进行**结果合并**规程。

2) 如果 **CommonArguments.serviceControls** 中的 **chainingProhibited** 标志被设置，或者由于 DSA 的本地操作策略，DSA 决定不做任何链接时，则 DSA 必须直接返回到**操作调度程序**来继续进行**结果合并**规程。

3) 将 **SRcontinuationList** 中的继续引用分类为不同的集合，每个集合具有相同的 **targetObject**。在这样的集合中不包括类型为 **ditBridge** 的继续引用，但每个这种类型的继续引用组成了自己的一个集合。在每个集合中，删除任何重复内容。

   注 1 — 如果有一个或多个 **targetObject** 的值不是一个主 RDN，则这种分类可能是不正确的。分类必须考虑可替代的识别 RDN，如果已知的话。

4) 对于继续引用的每个子集，根据子集中的每个继续引用的 **accessPoints** 组件，创建一个 **AccessPointInformation** 值集，并且选择顺序或并行策略来做后续处理。如果选择的是并行策略，则跳过下列标示为仅应用于顺序策略的那些步骤。

   a) 如果选择的是顺序策略，则为每个具有相同 **targetObject** 的继续引用集合维护一个本地变量 **localExclusions**。初始化时，**localExclusions** 被设置为入链接请求（如果存在的话）中的 **exclusions**，且所有本地搜索过的子树直接置于 **targetObject** 下。

   b) 如果选择的是顺序策略，则比较 **targetObject** 与 **localExclusions** 中的所有元素，并且删除那些没有将 **targetObject** 作为一个前缀的元素。这些是针对当前目标对象的相关排除。

   c) 从当前目标对象集合中的所有继续引用中抽取出所有的 **API**。

   d) 循环执行每个 **API**。对于每个 **API**：

      i) 调用 **APInfo**。

      ii) 如果返回一个结果，则若结果未被签名，则将其增加到 **searchInfo** 中，若结果被签名，则将其增加到 **uncorrelatedSearchInfo** 中。如果使用了顺序策略，则使用入答复中的 **alreadySearched** 更新 **localExclusions**，并且将入答复中的 **alreadySearched** 合并到该 DSA 的 **ChainingResults.alreadySearched** 中。然后继续下一个 **API**。

      iii) 如果返回一个错误或空结果，则它意味着 **APInfo** 已经尝试了 **API** 中的所有访问点而没有成功。则基于本地操作和安全策略，或者忽略并继续下一个 **API**，或者向 **partialOutcomeQualifier** 中增加一个基于此 API 的继续引用。

         注 2 — 从 **APInfo** 中取回一个提名是不可信任的。任何“提名”必须以 **partialOutcomeQualifier** 中的 **unexplored** 的形式出现。

   e) 当所有的 **APIs** 都被处理后，继续下一个具有相同 **targetObject** 的继续引用集合。

5) 当所有的继续引用都被处理后，返回到**操作调度程序**。

20.4.4 APInfo规程

本规程被调用来处理一个 **AccessPointInformation**，该信息中包含了一个或多个访问点（见图 27）。它们被一个接一个地处理直到返回一个结果或一个错误。如果错误是一个服务错误，因此尝试另一个访问点可能成功，因此只要本地操作策略允许，则可以一直尝试其他的访问点：

1) 执行环回检测。如果有一个环回被检测出，则返回一个问题为 **loopDetected** 的 **serviceError**。否则继续执行步骤 2)。

2) 根据访问点信息处理每个访问点。如果所有访问点都已经被处理，则返回一个**空结果**。如果还有任意一个访问点要处理，则继续执行步骤 3)。

3) 检查本地策略是否允许链接到此访问点。这种检查必须考虑到服务控制和链接变量的设置（例如 **chainingProhibited**、**preferChaining**，该访问点是否处于 **localScope** 内以及 **excludeShadows** 等）。如果本地设置或相应的服务控制设置不允许使用这个特定的访问点，则忽略此访问点并继续执行步骤 2)。如果能够使用此访问点，则继续执行步骤 4)。

4) 如果本地策略选择了“仅使用属主”策略，则将链接变量 **excludeShadows** 设置为 **TRUE**。

如果 **nameResolutionPhase** 不是 **completed**，且策略是继续执行主条目的名字解析，则将 **nameResolveOnMaster** 设置为 **TRUE**。

如果下列的任何一点为真，则链接变量 **nameResolveOnMaster** 必须被设置为 **TRUE**：

— 在入链接变量中，**nameResolutionPhase** 为 **proceeding**，且 **nameResolveOnMaster** 为 **TRUE**；或者

— 操作是修改操作之一，要发出的链接请求的 **referenceType** 为 NSSR，且使用了一个并行策略。

注 — 这种使用 **nameResolveOnMaster** 的方法是为了防止由于 NSSR 的存在，而多次应用修改操作。

5) 构建一个链接请求并尝试发起该请求：

a) 执行环回避免，方法是检查具有相同的 **targetObject** 和 **operationProgress** 的项是否出现在所接收到的 **ChainingArguments** 内的 **traceInformation** 中。如果接收到的请求（如步骤 5）的第 c）项所描述）将导致一个环回，则 DSA 或者向发起请求的 DUA/LDAP 客户机/DSA 返回一个问题为 **loopDetected** 的 **serviceError**，或者通过继续执行步骤 2)来忽略此访问点，并尝试下一个访问点。

b) 如果将要被链接的请求或子请求是执行一个提名的结果，则要求执行一个额外的环回避免检查。方法是检查具有相同的 **targetObject**、**operationProgress** 和目标 DSA 的项是否出现在 **referralRequests** 中。如果这样的话，则执行 a)项中指定的动作。如果没有，则向 **referralRequests** 中增加一个新的 **TraceItem**，并具有如下组件：

— **targetObject** 和 **operationProgress** 被设置为与被链接的请求/子请求的值相同；

— **dsa** 被设置为请求/子请求要被链接向的 DSA 的名字。

c) 在一个成功的绑定后，DSA 必须发起一个与被处理的操作具有相同操作类型的链接操作，并具有如下参数：

— 链接操作中的操作变量被设置为与接收到的操作变量相同；

— **ChainingArguments.originator** 被设置为与接收到的相同；

— **ChainingArguments.targetObject** 被设置为继续引用的 **targetObject**；

— **ChainingArguments.operationProgress** 被设置为继续引用的 **operationProgress** 的值；

— 如果继续引用的类型不是 **ditBridge**，则 **ChainingArguments.traceInformation** 被设置为由**请求合法性验证**规程更新过的踪迹信息，否则该组件必须缺失；

— **ChainingArguments.aliasDereferenced** 被设置为本地更新后的 **aliasDereferenced** 的更新后的值；

— **ChainingArguments.returnCrossRefs** 的设置是一个本地选择；

— **ChainingArguments.referenceType** 被设置为继续引用的 **referenceType** 的值；

— **ChainingArguments.timeLimit** 被设置为接收到的 **timeLimit** 的值；

— 如果被搜索继续引用规程所调用，则 **chainingArguments.exclusions** 被设置为当前目标对象的相关排除，或者如果 **APInfo** 规程是由名字解析或列表继续引用规程所调用，则该 **chainingArguments.exclusions** 缺失；

— **SecurityParameters** 被设置为接收到的 **SecurityParameters** 的值。

6) 如果请求不能被成功发出，则继续执行步骤 7)。如果能够被成功发出，则继续执行步骤 8)。

7) 是否继续执行由本地选择。如果 DSA 选择继续，则此错误被忽略，下一个访问点将被尝试。继续执行步骤 2)。如果 DSA 决定不再尝试下一个访问点，则本地策略将选择是向规程的调用者返回一个相应的**提名**还是一个 **serviceError**。

8) 如果请求能够被成功发出，则 DSA 必须等待答复，并对答复进行处理：

a) 如果接收到一个**结果**，则该**结果**被返回给规程的调用者。

b) 如果接收到一个问题为 **busy**、**unavailable**、**unwillingToPerform** 或 **invalidReference** 的 **serviceError**，则继续执行步骤 7)。

c) 如果接收到一个**提名**，且 **returnToDUA** 被设置为 **TRUE**，则接收方 DSA 不应当对该提名执行动作，而必须将提名返回给请求者。

d) 如果接收到一个**提名**，且 **returnToDUA** 被设置为 **FALSE**，则应用同步骤 3)同样的本地策略进行考虑（考虑服务控制、链接变量、链接策略等）。如果决定不对**提名**解除引用，则向调用者返回该**提名**。如果决定对**提名**解除引用，则清空 **NRcontinuationList**，将接收到的提名中的继续引用放置到 **NRcontinuationList** 中，并且调用**名字解析继续引用**规程。这可能会产生一个**结果、提名、服务错误**或其他**错误**。无论从**名字解析继续引用**规程的调用中接收到什么，都必须被返回给调用者。

e) 如果出现任何其他**错误**，则必须被返回给调用者。

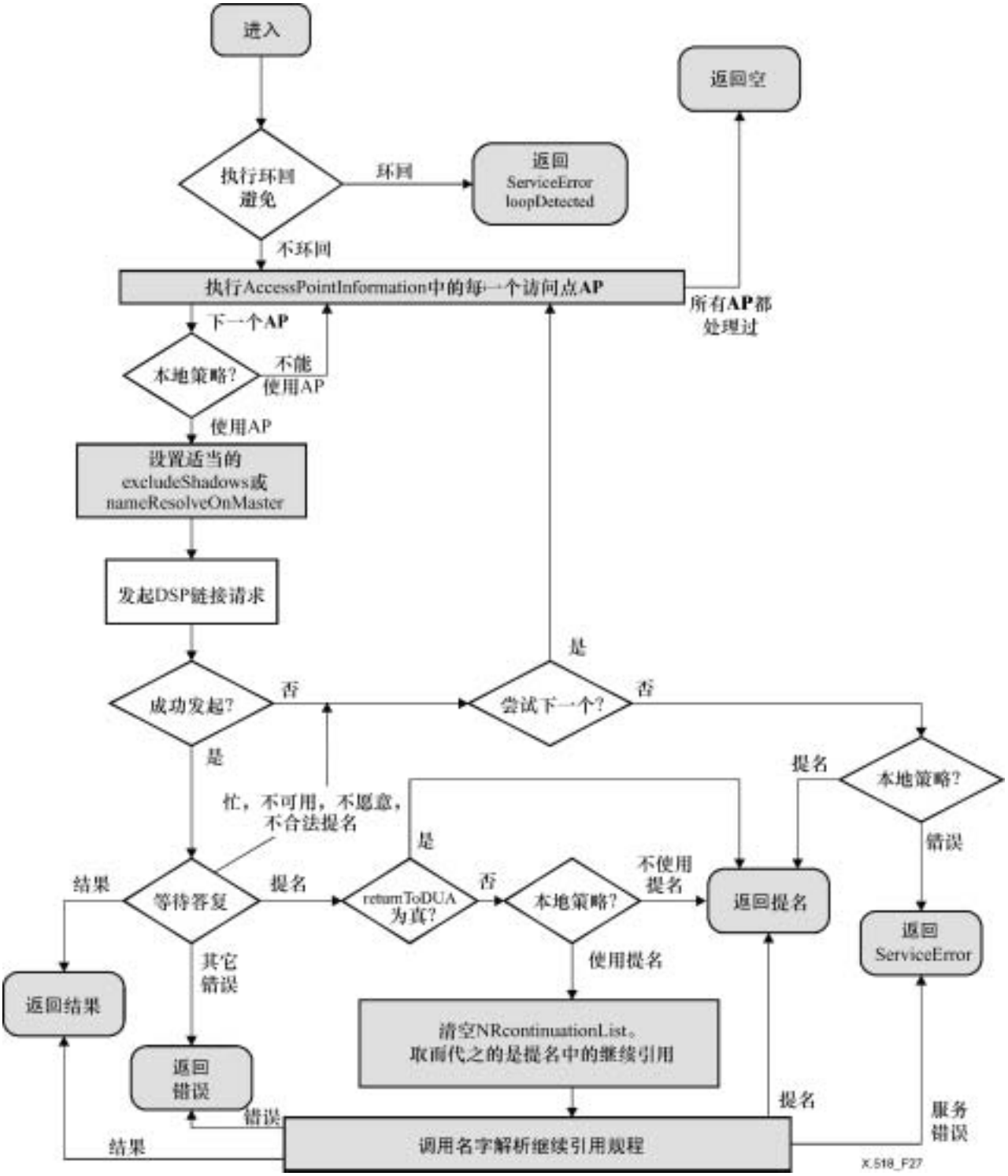


图 27－APInfo规程

20.5 放弃规程

如果接收到一个放弃请求，则本规程被调用。它由图 28 中显示的下列步骤组成。

1) 当接收到一个**放弃**请求，而该请求指向一个未知的操作，则必须向请求者返回一个问题为 **noSuchOperation** 的 **abandonFailed**。

2) 如果要被放弃的请求已经被答复，且 DSA 已经保留了需要知道的信息，则可能向请求者返回一个问题为 **tooLate** 的 **abandonError**。

3) 如果要被放弃的请求是不合法的，即要求放弃的请求不是一个查询请求，则必须向请求者返回一个问题为 **cannotAbandon** 的 **abandonFailed**。

4) 如果一个 DSA 在接收到一个合法的对原始请求的放弃请求时，还有链接请求（或子请求）正在进行中，而 DSA 决定尝试执行放弃，则它可能会向本操作的正在进行的请求（或子请求）的部分或全部发送放弃请求， 并且等待放弃请求和正在进行的请求（或子请求）的答复。在本操作执行过程中的任何时间点，DSA 都可能向请求者发送一个放弃结果和一个 **abandonFailed**，然后当所发起的放弃请求和正在进行的请求（或子请求）的答复到达时，放弃它们。

   如果 DSA 决定直到没有正在进行的请求（或子请求）时，才向请求者发送答复，则如果所有发起的 **abandon** 请求都被答复为具有 **abandonedFailed** 错误，且没有本地放弃操作被执行时，DSA 可能会可选地向请求者发送一个 **abandonedFailed** 错误。

   如果向请求者返回了一个 **AbandonedFailed** 错误，则原始请求必须被处理，就好像从来没有接收过放弃操作一样。

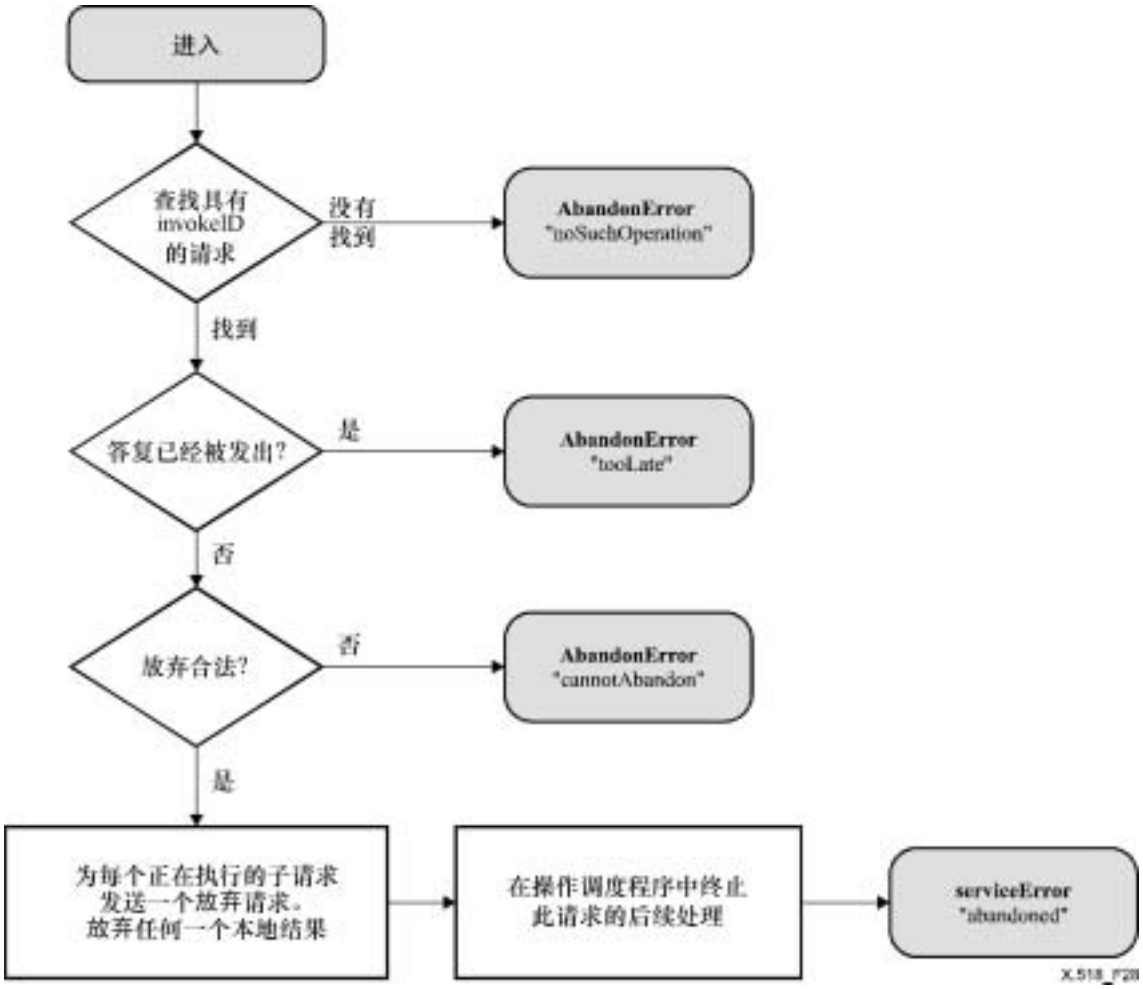


图 28－放弃规程

21 结果合并规程

在任意一个**继续引用**规程之后，图 29 中显示的**结果合并**规程被调用。如果结果没有被签名，且如果在 **partialOutcomeQualifier.unexplored** 中有附加的继续引用时，本规程将删除重复。因此如果本地操作策略允许的话，相关的继续引用规程将被调用：

1) 如果操作是一个列表操作，则继续执行步骤 2)；如果操作是一个搜索操作，则继续执行步骤 3)；否则，返回作为**结果合并**规程的输入参数所提供的结果。

2) 操作是一个列表操作。删除所有的重复，且属主信息的优先级高于镜像信息。

   如果操作结果是本地产生的，且它包含继续引用，因此这些结果不能被用于链接而是要返回给用户。在这种情况下，继续执行步骤 6)。

   如果接收到的操作结果是一个链接列表操作的结果，则该结果中可能包含继续引用。在这种情况下，检查 **preferChaining** 服务控制是否被设置。如果被设置为 **TRUE**，则 DSA 必须使用此继续引用来进行链接。继续执行步骤 4)。

3) 操作是一个搜索操作。删除所有的重复，且属主信息的优先级高于镜像信息。如果有一个限制问题，则返回此结果。否则继续执行步骤 4)。

4) 处理任意链接操作结果中 **partialOutcomeQualifier.unexplored** 内的每一个继续引用。如果本地策略决定不使用它来进行链接，则忽略它并选择另一个继续引用。如果本地策略允许使用该继续引用来进行链接，则执行如下动作：

   检查继续引用中提供的 **nameResolutionPhase**。如果它的值为 **notStarted** 或 **proceeding**，则将其加入到准备提供给**名字解析继续**规程的继续引用列表中（**NRcontinuationList**）。如果 **nameResolutionPhase** 的值为 **completed**，则将继续引用加入到提供给子请求继续规程的继续引用列表中（**SRcontinuationList**）。

   继续执行，直到所有的继续引用都被处理过。

5) 如果在 **SRcontinuationList** 中有要处理的继续引用，则检查操作类型。如果操作是一个列表操作，则调用**列表继续引用**规程，并继续执行步骤 2)。如果操作是一个搜索操作，则调用**搜索继续引用**规程，并继续执行步骤 3)。

   如果 **SRcontinuationList** 为空，则检查在 **NRcontinuationList** 中是否有继续引用。如果有，则调用**名字解析继续引用**规程，并继续执行步骤 3)。

   如果两个继续引用列表都为空，则继续执行步骤 6)。

6) 检查结果是否为空。如果非空，则返回该结果。如果为空，且访问控制和本地策略允许，则返回一个空结果，或者返回一个适当的错误。

当一个 DSA 从其他 DSA 处接收到搜索或列表结果，且这些结果中含有该 DSA 未知的参数时，则无关的结果必须被返回。否则，如果搜索结果未被签名，或者如果 DSA 是一个允许删除签名的初始执行者时（见 ITU-T X.511 建议书 | ISO/IEC 9594-3 的 7.9 节），则 DSA 必须执行合并。

如果一个 DSA 从一个不能执行合并的 DSA 处接收到未被签名的、不相关的结果，且该 DSA 对这些不相关的结果的所有参数都拥有正确知识时，该 DSA 必须执行合并。

如果一个 DSA 从另一个 DSA 处接收到未被签名的结果，同时它可能也拥有一个本地结果，则该 DSA 将产生一个条目数量，该条目数量将放在 DSA 产生的 **PartialOutcomeQualifier** 的 **entryCount** 中被返回，这个条目数量将是所有接收到的 **entryCount** 的值，本地结果，以及从没有返回 **entryCount** 值的 DSA 处接收到的条目数量的总和，然后再将重复结果抵消后的值。如果该 DSA 是初始执行者，且分页结果被请求时，则它还必须包括从其他 DSA 处接收到的签名结果中的条目数量。

如果分页结果被请求，且任何 DSA 都没有遇到限制问题，则 DSA 必须为 **entryCount** 参数执行 **exact** 选择。必须为每个返回的页给出同样的值。

如果一个或多个 DSA 遇到了某个限制问题，则：

— 如果所有遇到限制问题的 DSA 都已经返回了一个 **entryCount**，且具有 **exact** 或 **bestEstimate** 选择，则如果仅有一个 DSA 能够执行该选择，则它必须执行 **bestEstimate** 选择；否则它必须执行 **exact** 选择；

— 如果只有一个 DSA 遇到一个限制问题，且返回了一个选择为 **lowEstimate** 的 **entryCount**，或不返回一个 **entryCount**，则它必须执行 **lowEstimate** 选择。

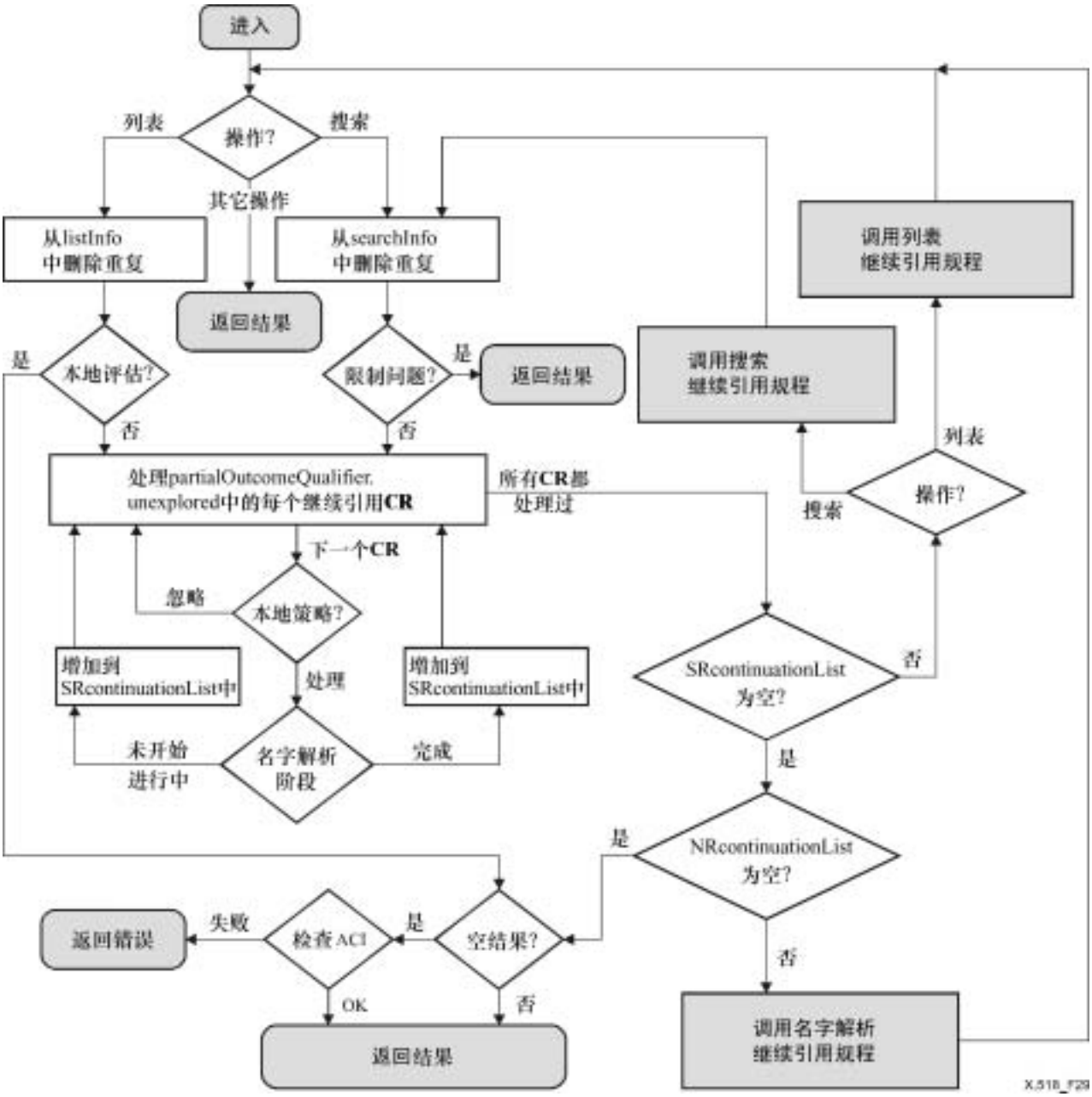


图 29－结果合并规程

22 分布式鉴权规程

本节规定了支持号码簿分布式鉴权服务所必需的规程。这些服务以及相应的规程，被分类为：

— 发起者鉴权，对它的支持或者采用一种未被保护（基于简单身份）的方式，或者采用一种安全（基于数字签名）的方式；以及

— 结果鉴权，采用类似被保护的方式（也是基于数字签名）。

22.1 发起者鉴权

22.1.1 基于身份的鉴权

基于身份的鉴权服务使得 DSA 可以出于实现本地访问控制的目的，对信息的初始请求者进行鉴权。希望使用此服务的 DSA 必须采用下列规程：

— 对于一个要求对 DAP 或 LDAP 请求进行鉴权的 DSA，在一个 DUA 连接（DUA 到 DSA）或 LDAP 客户机连接（LDAP 客户机到 DSA）建立的时刻，该 DSA 通过绑定规程获得请求者的识别名。这些规程的成功结果在任何时候都不会损害后续使用此连接来处理操作时所要求的鉴权级别。

— 与 DUA 或 LDAP 客户机存在连接的 DSA 必须为链接到其他 DSA 的所有子请求，将请求者的识别名插入到 **ChainingArguments** 的发起者字段中。

— 一个 DSA，在接收到一个链接操作时，可能满足此操作，也可能不满足，依赖于访问权限的决定（一个本地定义的机制）。如果输出不满意，则可能返回一个问题为 **insufficientAccessRights** 的 **securityError**。

22.1.2 基于签名的发起者鉴权

基于签名的发起者鉴权服务使得 DSA 可以对一个特定的服务请求的发起者进行鉴权（以一种安全的方式）。DSA 在实现此服务时使用的规程在本节中描述。

基于签名的鉴权服务由一个 DUA 来调用，方法是使用一个可选的被保护请求中的 **PROTECTED** 变量，其中 **DirQOP** 取值为 **signed** 或 **signedAndEncrypted**。

一个 DSA，在从另一个 DSA 处接收到一个签名的请求后，必须在处理此操作之前删除那个 DSA 的签名。假设任何签名合法性验证的结果证明是满足的，则 DSA 将继续执行此操作。如果在处理过程中，DSA 需要执行链接，则每个相关的链接操作的变量集必须按照如下所述来构建：

— DSA 构造一个可能被签名的变量集；该变量集由输入的已签名变量集和一个修改后的 **ChainingArguments** 共同组成。

如果 DSA 能够将信息发布给响应，则基于签名服务请求的发起者鉴权可能被用来决定对此信息的访问权限。

如果一个 DSA 接收到一个关于某信息的未被签名的服务请求，而根据发起者鉴权该信息只能被释放，则必须返回一个问题为 **protectionRequired** 的 **securityError**。

22.2 结果鉴权

本服务的提供使得号码簿操作的请求者（DUA、LDAP 客户机或 DSA）能够验证（使用数字签名机制这样一个安全方式）结果的来源。可以要求使用结果鉴权服务，而不用考虑是否使用了发起者鉴权。

结果鉴权服务的发起是使用包含在号码簿操作变量集内的 **protectionRequest** 组件的签名值来实现的；一个 DSA 如果接收到一个带有此选项的操作时，可能会可选地对任何子请求结果进行签名。在保护请求中的签名选项，其作用是向 DSA 指示了请求者的优先级；实际上，DSA 可能对，也可能不对任意子请求结果进行签名。

在一个 DSA 执行链接的情况下，根据返回到请求者的结果形式，DSA 拥有一系列的选项，这些选项为：

a) 向请求者返回一个复合响应（签名的或未签名的）；

b) 向请求者返回两个或多个未被合并的部分响应的集合（签名的或未签名的）；在此集合中可能有零个或多个成员被签名，也可能有零个或一个未被签名。如果有一个未被签名的部分结果存在，则该成员实际上可能是一个或多个未被签名的部分响应的集合，这些部分响应可能是从其他 DSA 处接收到的，或者是由本 DSA 提供的，或者两种都有可能。

当一个 DSA 对相关条目进行合并时，执行合并的 DSA 可能会对结果进行签名。

第 6 部分 — 知识管理

23 知识管理概述

为了以一种可接受的一致性程度和性能对一个广泛分布的号码簿进行操作，需要有规程对每个 DSA 中拥有的知识进行创建、维护和扩展。下列机制被共同使用来管理 DSA 的知识。

a) *分等级操作绑定和非特定分等级操作绑定* — 这些规程和协议在第 24 节和第 25 节中定义。它们被用来维护下级引用、非特定下级引用、直接上级引用、以及命名上下文的上下文前缀信息等。这些操作绑定是在主 DSA 之间建立的，这些主 DSA 拥有彼此之间分等级相关的命名上下文，即具有直接下级与直接上级的关系。本规程的触发可能是修改条目 RDN、增加条目、或删除条目等操作的一个副作用，那些条目的直接上级不是由拥有该条目的同一个 DSA 所拥有的。

b) *镜像操作绑定* — 这些规程和协议在 ITU-T X.525 建议书 | ISO/IEC 9594-9 中定义。它们被用来以两种方法创建和维护知识引用。第一种是作为建立（或终止）镜像合约的一个副作用，访问点被加入到 **consumerKnowledge** 中或可选的 **secondaryShadow** 操作属性中，或从中删除。因此这些信息可能被上面讨论的规程和协议使用来更新上级主 DSA 中的下级引用，或者下级主 DSA 中的直接上级引用。第二种是 DISP 将主 DSA 中拥有的知识引用传播到镜像消费者 DSA 处。

c) *交叉引用* — 交叉引用的分布是 DSP 的一个特性。它用于创建和维护交叉引用，在第 23.2 节给出了摘要。

注 — 初始化和维护上级引用以及 **myAccessPoint** 的机制，不在本号码簿规范的定义范围之内。

23.1 知识引用的维护

本节描述了 DOP 是如何被用来维护表示知识的 DSA 操作属性的。关于知识属性和用于维护这些知识属性的协议之间的关系，有一个简单示例在附件 E 中描述。

23.1.1 提供者和主DSA对消费者知识的维护

一个消费者引用，通过 **consumerKnowledge** 属性的值来表示，由一个镜像提供者 DSA 所拥有，并与一个命名上下文的上下文前缀相关联；一个提供者引用，通过 **supplierKnowledge** 属性的值来表示，由一个镜像消费者 DSA 所拥有，并与一个命名上下文的上下文前缀相关联。这两个属性都由类型为 **cp** 的 DSE 所拥有。这些属性的每个值都是在镜像操作绑定建立时被创建的，并且在镜像操作绑定修改时更新。

一个提供者 DSA 可能会获取信息来构建 **secondaryShadows** 属性的值，这是在它与某个消费者的 **ShadowingAgreementInfo** 中的可选组件 **secondaryShadows** 取值为 **TRUE** 的情况下。在这种情况下，无论何时，当消费者 DSA 检测出拥有公共可用复制区拷贝的 DSA 集合（包括它的消费者，以及消费者的消费者等，可能携带任意深度的二次镜像）被修改了（通过增加、修改或删除访问点），则它会通过一个 **modifyOperationalBinding** 操作来传达此新信息（一个 **SupplierAndConsumers** 的集合），该操作在 ITU-T X.525 建议书 | ISO/IEC 9594-9 中描述。

一个提供者 DSA 维护其自身的，与上下文前缀相关的 **secondaryShadows** 属性，如下所述：

a) 通过 **modifyOperationalBinding** 操作从一个消费者处接收到的 **SupplierAndConsumers** 集合可能会被用于创建或替代本属性的值。**SupplierAndConsumers** 中的提供者组件表示了一个消费者 DSA（或消费者的消费者等，依赖于二次镜像的深度）的访问点；消费者组件表示消费者的消费者集合（或其消费者，依赖于二次镜像的深度）。

b) 每个在 **modifyOperationalBinding** 操作中提供其提供者的消费者，都包含一个 **SupplierAndConsumers** 集合，具有下列值：它的 **secondaryShadows** 属性的值，以及一个新构造的值。这个新值是这样构造的：使用它自己的访问点 **myAccessPoint**（作为提供者组件），以及包含在 **consumerKnowledge** 属性中的表示拥有公共可用镜像的消费者的访问点值（作为消费者组件）。

本规程的递归使用可以使一个命名上下文的主 DSA 能够知道它的所有的二次镜像消费者 DSA，这些 DSA 拥有从该命名上下文派生的公共可用的复制区。于是，这些信息可用于对下级引用、非特定下级引用和直接上级引用等的维护。

23.1.2 主DSA中下级和直接上级知识的维护

一个下级引用通过 **specificKnowledge** 属性的一个值来表示，该属性由拥有此下级引用的直接上级命名上下文的 DSA 中类型为 **subr** 的 DSE 所拥有；一个直接上级引用通过 **specificKnowledge** 属性的一个值来表示，该属性由拥有此上级引用的直接下级命名上下文的 DSA 中类型为 **immSupr** 的 DSE 所拥有。这些属性的每个值都在相应的上级和下级主 DSA 中建立 HOB 时创建，并在 HOB 修改时更新。

一个下级主 DSA 通过它在 DOP 中传送到上级的 **SubordinateToSuperior** 参数中的 **accessPoints** 组件，向其上级主 DSA 提供信息来构造它的下级引用。包含在 **accessPoints** 中的信息由下级 DSA 所拥有的属性值来决定，如下所述：

a) **myAccessPoint** 属性的值（由根 DSE 拥有）被用来构造 **accessPoints** 中的元素，其中 **category** 的取值为 **master**。

b) **consumerKnowledge** 和 **secondaryShadows** 的值（两个都由下级上下文前缀 DSE 拥有）被用于构造 **accessPoints** 中的附加元素，其中 **category** 的取值为 **shadow**。

一个上级主 DSA 通过它在 DOP 中传送到下级的 **SuperiorToSubordinate** 参数中的 **contextPrefixInfo** 组件，向其下级主 DSA 提供信息来构造它的直接上级引用。该组件的值类型为 **SEQUENCE OF Vertex**，包含了与 DIT 中的根到此下级上下文前缀之间的路径相对应的元素序列。对于这些元素中的其中一个与直接上级命名上下文的上下文前缀相应的元素，可选组件 **accessPoints** 将存在。下级 DSA 将此信息作为类型为 **immSupr** 的 DSE 中的一个 **specificKnowledge** 属性，相应于 **contextPrefixInfo** 中的这个元素。包含在上级 DSA 的 **accessPoints** 中的信息由上级 DSA 所拥有的属性值来决定，如下所述：

a) **myAccessPoint** 属性的值（由根 DSE 拥有）被用来构造 **accessPoints** 中的元素，其中 **category** 的取值为 **master**。

b) **consumerKnowledge** 和 **secondaryShadows** 的值（两个都由上级上下文前缀 DSE 拥有）被用于构造 **accessPoints** 中的附加元素，其中 **category** 的取值为 **shadow**。

注 — 仅有那些与接收到公共可用复制区的消费者 DSA 相应的访问点才应当被上级 DSA 和下级 DSA 根据它们的 **consumerKnowledge** 属性选中，并将其包含到 **accessPoints** 中。构造 **secondaryShadows** 的规程会确保这些访问点可以标识出拥有公共可用复制区的镜像 DSA。

23.1.3 消费者DSA中下级和直接上级知识的维护

一个镜像消费者 DSA 与其提供者之间有合约，可以接收到与某个复制单元相关的直接上级和下级知识，如果该合约有效，则也有合约由它的镜像提供者 DSA 通过 DISP 来维护其直接上级和下级引用。

注 — 对于某些复制规范单元，对于消费者 DSA 来说，为了能够让它的提供者可能向其提供下级知识，可能有必要定义合约来接收 **extendedKnowledge**。

23.2 请求交叉引用

为了改进号码簿系统的性能，可以使用普通的号码簿操作来扩展交叉引用的本地集合。如果一个 DSA 支持 DSP，它可能会请求另一个 DSA（该 DSA 也支持 DSP）返回那些包含了与一个普通号码簿操作的目标对象名字相关的命名上下文的位置信息的知识引用。

如果 **ChainingArguments** 中的 **returnCrossRefs** 组件被设置为 **TRUE**，则 **ChainingResults** 中的 **crossReferences** 组件可能会存在，且包含了一个交叉引用项的集合。

如果一个 DSA 不能够将一个请求链接到下一个 DSA，则一个提名会被返回到发起请求的 DSA。如果 **ChainingArguments** 中的 **returnCrossRefs** 组件为 **TRUE**，则该提名中可能另外还包含此提名所指向的命名上下文的上下文前缀。如果该提名是基于非特定下级引用，则 **contextPrefix** 组件缺失。一个提名所返回的交叉引用是基于产生该提名的 DSA 所拥有的知识的。

在两种情况下（链接结果和提名），某个主管当局，通过他的 DSA，可能会选择忽略那些返回交叉引用的请求。

23.3 知识的不一致性

号码簿必须支持一致性检查机制来保证某种程度的知识一致性。

注 — 在某种环境中，一个知识引用是正确的（在下面描述的情况下不是非法的），但是对于某个 DSA 来说使用它是无效的，因为被引用的 DSA 的 DMD 根本不希望被引用它的 DSA 所接触（例如，一个 DSA 不知何故获得了一个指向被引用的 DSA 的交叉引用），或者不希望它以某种角色被接触（例如，作为某个命名上下文的主 DSA）。

23.3.1 知识不一致性的检测

不一致性的类型及其检测根据知识引用类型的不同而不同。

a) *交叉引用和下级引用* — 如果被引用的 DSA 没有拥有一个命名上下文或者一个复制区，该复制区所来自的命名上下文具有引用中所包含的上下文前缀时，这种类型的引用是不合法的。这种不一致性可以在名字解析过程中被检测出来，方法是通过检查 **ChainingArguments** 中的 **operationProgress** 和 **referenceType** 组件。

b) *非特定下级引用* — 如果被引用的 DSA 没有拥有一个本地命名上下文，该命名上下文的上下文前缀是引用中所包含的上下文前缀减去最后一个 RDN 而形成的时，这种类型的引用是不合法的。一致性检查的应用如上所述。

c) *上级引用* — 一个非法的上级引用是不能够构成到根的一个引用路径的引用。上级引用的维护必须通过外部方式来维护，且不在本号码簿规范的定位范围之内。

   注 — 并不总是能够检测出一个非法的上级引用。

d) *直接上级引用* — 如果被引用的 DSA 没有拥有一个命名上下文或者一个复制区，该复制区所来自的命名上下文具有引用中所包含的上下文前缀时，这种类型的引用是不合法的。另外，仅当 **ChainingArguments** 中的 **operationProgress** 组件取值为 **notStarted** 或 **proceeding** 时，使用这种类型的引用才是合法的。这种不一致性可以在名字解析过程中被检测出来，方法是通过检查 **ChainingArguments** 中的 **operationProgress** 和 **referenceType** 组件。

e) *提供者引用* — 这种类型的引用，标识了某个复制区的提供者，并且可选地标识了该复制区所来自的命名上下文的属主，当被引用的 DSA 不是使用该引用的 DSA 的镜像提供者时（当 **ChainingArguments** 的 **referenceType** 组件取值为 **supplier** 时），或者当被引用的 DSA 不是该命名上下文的属主时（当 **referenceType** 的取值为 **master** 时），这种类型的引用是非法的。这种不一致性可以在操作处理的名字解析阶段和操作评估阶段被检测出来，方法是通过检查 **ChainingArguments** 中的 **referenceType** 组件。

23.3.2 知识不一致性的上报

如果使用链接来执行一个号码簿请求，则所有的知识不一致性都将被拥有非法知识引用的 DSA 检测出来，方法是通过接收到一个问题为 **invalidReference** 的 **serviceError**。

如果一个 DSA 返回了一个基于某个非法知识引用的提名，则请求者如果使用了该提名的话，将被返回一个问题为 **invalidReference** 的 **serviceError**。错误条件是如何被传播到存储此非法引用的 DSA 的，不在本号码簿规范的定义范围之内。

23.3.3 不一致的知识引用的处理

当一个 DSA 在检测到一个非法引用后，它必须尝试去重新建立知识的一致性。例如，这可以通过简单地删除一个非法交叉引用来实现，或者通过将其替换为一个使用 **returnCrossRefs** 机制而获得的正确引用来实现。

实际上，DSA 如何处理非法引用的方法属于本地事务，不在本号码簿规范的定义范围之内。

23.4 知识引用和上下文

知识引用中的名字必须是主识别名，对于任意 RDN 中使用的任意属性，也可能包含可替代的识别值以及在 **valuesWithContext** 中拥有的上下文信息，如同 ITU-T X.501 建议书 | ISO/IEC 9594-2 的 9.3 节中的描述。

依赖于一个知识引用是如何被获得的（尤其是对于一个第三版本之前的 DSA，其是否拥有该引用，或者是否是获取引用的链接中的一部分），可能一个知识引用中不会包含所有可能的可替代识别名。这可能会导致一个声称

的名字不被该知识引用的所有者认为是同一个名字，这在某种情况下，会导致需要在名字解析过程中执行额外的步骤，或者在某种情况下，会导致不一致的结果或名字解析失败。在主识别名已知时，通常使用主识别名，会优化号码簿处理名字中的上下文变量的能力。

24 分等级操作绑定

一个分等级操作绑定被用来表示两个 DSA 之间的关系，这两个 DSA 拥有两个命名上下文，其中一个是另一个的直接下级。在 HOB 的情况下，上级 DSA 拥有一个指向下级 DSA 所拥有的命名上下文的下级引用；下级 DSA 拥有一个指向上级 DSA 所拥有的命名上下文的直接上级引用。操作绑定确保了可以在两个 DSA 之间交换和维护适当的知识信息，因此，两个 DSA 都能够在名字解析和操作评估的处理过程中，按照第 18 节和 19 节的定义来运转。

24.1 操作绑定类型特性

24.1.1 对称与角色

分等级操作绑定类型是一种非对称的操作绑定类型。在这种类型的绑定中有两种角色，分别为：

a) 上级命名上下文的主 DSA 所承担的角色，称为*上级 DSA*（相关的抽象角色为“A”）；以及

b) 下级命名上下文的主 DSA 所承担的角色，称为*下级 DSA*（相关的抽象角色为“B”）。

24.1.2 合约

在建立分等级操作绑定的过程中，所交换的合约信息是 **HierarchicalAgreement** 的一个值。它包括新的上下文前缀的相对识别名（**rdn** 组件）以及该新的命名上下文的直接上级条目的识别名（**immediateSuperior** 组件）。此信息必须被发起该 HOB 的 DSA 所提供。

```
HierarchicalAgreement ::= SEQUENCE {
        rdn                 [0]  RelativeDistinguishedName,
        immediateSuperior   [1]  DistinguishedName }
```

rdn 必须是主 RDN，且 **immediateSuperior** 必须是一个主识别名。上下文信息以及所有的可替代识别值必须被包含在任意 RDN 的 **AttributeTypeAndDistinguishedValue** 的 **valuesWithContext** 组件中，如同 ITU-T X.501 建议书 | ISO/IEC 9594-2 的 9.3 节中的描述。

24.1.3 发起者

24.1.3.1 建立

一个分等级操作绑定的建立可以由任意一种角色来发起。上级 DSA 的发起可以是由于一个增加条目操作而引起的，其中在 **targetSystem** 扩展中指定了下级 DSA，或者是由于管理干预而引起的。下级 DSA 的发起（将一个本地存在的条目或子树与全球 DIT 连接起来）是由于管理干预而引起的。

24.1.3.2 修改

一个分等级操作绑定的修改可以由任意一种角色来发起。由于上级上下文前缀信息的修改，使得上级 DSA 可能会发起修改。这可以是任意修改操作的结果，或者是管理干预的结果。

如果下级命名上下文的上下文前缀条目的 RDN 被修改，则任意一个 DSA 都可能修改此合约。上级 DSA 发起此修改，是由于一个相对识别名被修改为高于此 DIT 而引起的，或者是由于管理干预而引起的。下级 DSA 发起此修改，是由于针对一个上下文前缀的 ModifyDN 而引起的，或者是由于管理干预而引起的。

如果该命名上下文的访问点信息发生了变化，则任何一个 DSA 也都可能修改此 HOB。

24.1.3.3 终止

一个分等级操作绑定的终止可以由任意一种角色来发起。上级 DSA 的发起可以是由于管理干预而引起的。下级 DSA 的发起可以是由于一个删除条目操作而引起的，该操作删除了下级命名上下文的上下文前缀条目，或者是由于管理干预而引起的。

24.1.4 建立参数

一个 HOB 的两种角色，上级 DSA 和下级 DSA，它们的建立参数是不同的。上级 DSA 角色使用的建立参数是 **SuperiorToSubordinate** 的一个值，下级 DSA 角色使用的建立参数是 **SubordinateToSuperior** 的一个值。

24.1.4.1 上级DSA的建立参数

由上级 DSA 所发起的建立参数是 **SuperiorToSubordinate** 的一个值，向下级 DSA 提供了关于新的命名上下文的上下文前缀的上级 DIT 顶点的相关信息（包含了直接上级引用），可选的，还提供了下级上下文前缀条目的用户属性和操作属性，以及新上下文前缀的直接上级条目的用户属性和操作属性的拷贝。

```
SuperiorToSubordinate ::= SEQUENCE {
        contextPrefixInfo          [0]    DITcontext,
        entryInfo                  [1]    SET SIZE (1..MAX) OF Attribute OPTIONAL,
        immediateSuperiorInfo      [2]    SET SIZE (1..MAX) OF Attribute OPTIONAL }
```

Vertex 或 **SubentryInfo** 中的 **rdn** 必须是主 RDN，且上下文信息和所有其他的识别值都必须被包含在 RDN 的 **AttributeTypeAndDistinguishedValue** 组件中，如同在 ITU-T X.501 建议书 | ISO/IEC 9594-2 的 9.3 节中的描述。

24.1.4.1.1 上下文前缀信息

SuperiorToSubordinate 中的 **contextPrefixInfo** 组件是类型为 **DITcontext** 的一个值，这是一个 **Vertex** 值的序列。

```
DITcontext ::= SEQUENCE OF Vertex

Vertex ::= SEQUENCE {
        rdn             [0]  RelativeDistinguishedName,
        admPointInfo    [1]  SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries      [2]  SET SIZE (1..MAX) OF SubentryInfo OPTIONAL,
        accessPoints    [3]  MasterAndShadowAccessPoints OPTIONAL }
```

contextPrefixInfo 组件是 RDN 的序列，这些 RDN 构成了新的上下文前缀的直接上级的识别名，每个 RDN（由 **rdn** 组件给定）可选地还伴随有附加的信息。

一个 **Vertex** 中可选的 **admPointInfo** 组件表示了该 DIT 顶点是一个管理点，并至少提供了它的 **administrativeRole** 操作属性。

与一个管理点相关联的子条目信息由 Vertex 中的 **subentries** 组件提供，该组件是一个或多个 **SubentryInfo** 值的集合。每个 **SubentryInfo** 值由子条目的 RDN（**rdn** 组件）和子条目的属性（**info** 组件）组成。

```
SubentryInfo ::= SEQUENCE {
        rdn  [0]  RelativeDistinguishedName,
        info [1]  SET OF Attribute }
```

一个 **Vertex** 中可选的 **accessPoints** 组件表示了该顶点符合直接上级命名上下文的上下文前缀。上级使用此组件以便向下级提供其直接上级引用所需的信息。

注 — **accessPoints** 中的主访问点与建立和修改操作绑定操作中传递到 **accessPoint** 参数中的主访问点相同。

24.1.4.1.2 条目信息

SuperiorToSubordinate 中可选的 **entryInfo** 组件是建立新的上下文前缀条目内容的一个属性集合。

24.1.4.1.3 直接上级条目信息

SuperiorToSubordinate 中可选的 **immediateSuperiorInfo** 组件是新的上下文前缀的直接上级条目的属性集的一个拷贝，尤其是 **objectClass** 和 **entryACI**。

注 — 本组件可能被下级使用来优化一个列表请求的评估，该列表请求针对某个基对象产生了一个空的 **ListResult**，而此基对象是下级上下文前缀的直接上级（见 19.3.1.2.2 节中第 2 项的注）。

24.1.4.2 下级DSA的建立参数

下级 DSA 发起的建立参数是 **SubordinateToSuperior** 的一个值，向上级 DSA 提供了与下级命名上下文相关的信息。

```
SubordinateToSuperior ::=    SEQUENCE {
    accessPoints     [0]     MasterAndShadowAccessPoints OPTIONAL,
    alias            [1]     BOOLEAN DEFAULT FALSE,
    entryInfo        [2]     SET SIZE (1..MAX) OF Attribute OPTIONAL,
    subentries       [3]     SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }
```

SubordinateToSuperior 中的 **accessPoints** 组件被下级使用，向上级提供其下级引用所需的信息。

注 1 — **accessPoints** 中的主访问点与建立和修改操作绑定操作中传递到 **accessPoint** 参数中的主访问点相同。

SubordinateToSuperior 中的 **alias** 组件用于向上级表示该下级命名上下文包含一个单独的别名条目。

SubordinateToSuperior 中的 **entryInfo** 组件包含了新的上下文前缀条目的属性集的一个拷贝，尤其是 **objectClass** 和 **entryACI** 属性，而且如果可应用的话，还包含 **administrativeRole** 操作属性。

注 2 — 前两个属性可能被上级使用来优化一个列表请求或一个一级搜索请求的评估，而这些请求的基对象是下级上下文前缀的直接上级条目，同时，最后一个属性被用来避免对一个搜索操作进行不必要的处理，使其进入到某个服务特定的管理区或者从某个服务特定的管理区出来。

SubordinateToSuperior 中的 **subentries** 组件被下级使用来向上级传递包含了预定 ACI 的子条目。

24.1.5 修改参数

对于 HOB 的修改，上级角色的修改参数 **SuperiorToSubordinateModification** 是 **SuperiorToSubordinate**，其中有一个限制为 **entryInfo** 组件可能不存在；下级角色的修改参数是 **SubordinateToSuperior**。

```
SuperiorToSubordinateModification ::= SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})
```

这些参数与相应的建立参数是相同的（除了上面所标注的限制外），并且被用于表示在 HOB 建立之后，建立参数中提供的信息所发生的变化。

如果 **SuperiorToSubordinate**（或者 **SuperiorToSubordinateModification**）、或者 **SubordinateToSuperior** 中的任何组件经历了一次变化（例如 **SuperiorToSubordinate** 中的 **contextPrefixInfo** 组件），则修改参数中的相应组件（例如 **SuperiorToSubordinateModification** 中的 **contextPrefixInfo** 组件）必须在修改操作绑定中被完整提供。

24.1.6 终止参数

当终止一个 HOB 时，两种角色都不提供终止参数。

24.1.7 类型标识

分等级操作绑定由对象标识符来标识，这些对象标识符是在 24.2 节定义 **hierarchicalOperationalBinding OPERATIONAL-BINDING** 信息对象时被分配的。

24.2 操作绑定信息对象类定义

本小节使用 ITU-T X.501 建议书 | ISO/IEC 9594-2 中定义的 **OPERATIONAL-BINDING** 信息对象类模板，定义了分等级操作绑定的类型。

```
hierarchicalOperationalBinding OPERATIONAL-BINDING ::= {
        AGREEMENT       HierarchicalAgreement
        APPLICATION CONTEXTS {
                        {directorySystemAC} }
        ASYMMETRIC
            ROLE-A {            -- 上级 DSA
                ESTABLISHMENT-INITIATOR     TRUE
                ESTABLISHMENT-PARAMETER     SuperiorToSubordinate
                MODIFICATION-INITIATOR      TRUE
                MODIFICATION-PARAMETER      SuperiorToSubordinateModification
```

```
        TERMINATION-INITIATOR           TRUE }
    ROLE-B {            -- 下级 DSA
        ESTABLISHMENT-INITIATOR         TRUE
        ESTABLISHMENT-PARAMETER         SubordinateToSuperior
        MODIFICATION-INITIATOR            TRUE
        MODIFICATION-PARAMETER          SubordinateToSuperior
        TERMINATION-INITIATOR           TRUE }
ID                  id-op-binding-hierarchical }
```

24.3 分等级操作绑定管理的DSA规程

在下面的规程中，由一个 DSA 创建的一个新 DSE 或一个标记（即一个与信息的某些项相关联的状态指示）必须被存储在一个静态存储器中。只有这样做，才能使得遵循下面规程的两个 DSA 在出现通信和终端系统失败时，有可能对 HOB 的参数维护一个一致的理解。

在下面所描述的 **establishment** 和 **modification** 两个规程中，承担了响应者角色的 DSA（即没有发起建立或修改操作）可能会向承担了发起者角色的 DSA 提供信息（例如操作属性），而由于这样或那样的原因，这些信息是不可接受的。在这些情况下，发起者 DSA 可能会终止此操作绑定。

24.3.1 建立规程

24.3.1.1 上级DSA发起的建立

如果一个 DSA 与另一个在 **targetSystem** 扩展中指定的不同 DSA 之间执行了一个增加条目操作，则该 DSA 必须根据下列规程建立一个分等级的操作绑定。出于管理原因，如果一个 DSA 希望与一个下级 DSA 之间建立一个 HOB，且它支持 DOP HOB 协议，则必须遵循下列规程：

1) 上级 DSA 创建一个类型为 **subr** 的新 DSE，其名字为新条目的名字，并且将这个新的 DSE 标记为“*将被增加*”。上级 DSA 产生一个唯一的 **bindingID**，并将其与新的 DSE 存储在一起。

2) 上级 DSA 必须向下级 DSA 发送一个建立操作绑定操作，并包含下列参数：
   a) **bindingType** 被设置为 **hierarchicalOperationalBindingID**；
   b) **SuperiorToSubordinate** 建立参数中，**contextPrefixInfo** 和 **entryInfo** 组件存在；所有其他参数都是可选的；
   c) **HierarchicalAgreement** 中，**immediateSuperior** 组件被设置为新条目的直接上级的识别名，且 **rdn** 组件被设置为新条目的 RDN；以及
   d) 适当的 **bindingID**，**myAccessPoint** 和 **valid** 参数。

3) 如果下级 DSA 接受此操作，则它将创建所需的类型相应为 **glue**、**subentry**、**admPoint**、**rhob** 和 **immSupr** 的 DSE 来表示 **contextPrefixInfo**；或者类型相应为 **cp** 和 **entry** 或 **alias** 的 DSE 来表示新的上下文前缀对象或别名条目；以及类型相应为 **rhob** 和 **entry** 的 DSE 来表示 **immediateSuperiorInfo**。它将 **bindingID** 与新的上下文前缀条目的 DSE 存储在一起，并且向上级 DSA 返回一个 **SubordinateToSuperior** 参数。

   如果下级 DSA 拒绝此操作，则它会返回一个操作绑定错误，并且设置适当的问题值。

   如果命名上下文已经存在，且已经存在的命名上下文与新的上下文的 **bindingID** 值相同，即下级 DSA 已经创建了所请求的命名上下文，在这种情况下，下级 DSA 向上级 DSA 返回一个结果。如果这两个值不相同，则发送一个问题为 **invalidAgreement** 的操作绑定错误；这就意味着上级 DSA 有一个永久的知识不一致，需要管理者的纠正。

4) 如果上级 DSA 接收到一个错误，则它删除类型为 **subr** 的被标记 DSE，并且为增加条目操作返回一个错误。

   如果上级 DSA 接收到一个结果，则它从表示 **subr** 的 DSE 中删除标记，并且为增加条目操作返回一个结果。

   如果有任何失败出现（如通信失败或终端系统失败），上级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的分等级操作绑定建立操作都接收到一个结果或错误为止。

如果建立是由于一个增加条目操作而引起的，且在建立完成之前由请求者中止了该操作（例如通过释放或中止应用连接），则上级 DSA 必须忽略此事件，并且完成此次建立（可能成功，也可能不成功）。在这种情况下，增加条目操作的输出不会通知到用户。

注 1 — 对下级进行标注可以帮助恢复和进行同步控制。另一个用户不能增加一个已经被标注的条目，且在一次失败后，DSA 将为所有被标注的下级重复建立操作绑定。

注 2 — 通过上述规程，知识仅具有临时的不一致性。当下级引用被标注时，上级 DSA 如何处理那些对下级引用进行阅读的不相关操作属于本地事务。

24.3.1.2 下级DSA发起的建立

下级 DSA 可能发起建立一个分等级的操作绑定。这可能是由于某个管理者希望将该 DSA 内拥有的一棵条目子树与全球 DIT 中的某个点相关联起来而引起的。在这种情况下，下级 DSA 必须根据下列规程建立一个 HOB：

1) 下级 DSA 或者具有一个类型为 **cp** 的 DSE 作为一个已经存在的命名上下文的一部分，或者创建一个新的这种 DSE。它将这个新的 DSE 标记为“将被增加”，且产生一个唯一的 **bindingID**，并将其与上下文前缀 DSE 存储在一起。
2) 下级 DSA 向上级 DSA 发起一个建立操作绑定操作，包含下列参数：
   a) **bindingType** 被设置为 **hierarchicalOperationalBindingID**；
   b) 适当的 **SubordinateToSuperior** 建立参数；
   c) **HierarchicalAgreement** 中，**immediateSuperior** 组件被设置为新条目的直接上级的识别名，且 **rdn** 组件被设置为新条目的 RDN；以及
   d) 适当的 **bindingID**，**myAccessPoint** 和 **valid** 参数。

   如果上级 DSA 拒绝此操作，则它返回一个操作绑定错误，并且设置适当的问题值。
3) 上级 DSA 检测出它是新的上下文前缀条目的直接上级的属主，或者返回一个问题为 **roleAssignment** 的 **operationalBindingError**。
4) 上级 DSA 检测被请求的新的上下文前缀的 RDN 是否尚未被使用。如果使用本地拥有的信息没有找到匹配的 RDN，但是直接上级 DSE 的类型为 **nssr**，则遵循 19.1.5 中的规程。如果使用此规程没有发现匹配的 RDN，则上级 DSA 创建一个类型为 **subr** 的 DSE，并且将 **bindingID** 与其存储在一起，然后返回一个结果。

   如果发现一个下级引用与此 RDN 匹配，则比较两个 **bindingID** 的值。如果它们相等，则返回一个结果。且由上级 DSA 所返回的 **SuperiorToSubordinate** 参数中，不能够包含 **entry** 组件。如果两个 **bindingID** 值不相等，则发送一个问题为 **invalidAgreement** 的 **operationalBindingError**；这就意味着上级 DSA 具有一个永久的知识不一致，需要管理者的纠正。

   如果通过部署一个 NSSR，发现一个匹配的 RDN，则发送一个问题为 **invalidAgreement** 的 **operationalBindingError**，这也意味着上级 DSA 具有一个永久的知识不一致，需要管理者的纠正。
5) 如果下级 DSA 接收到一个错误，它将删除新的上下文前缀 DSE 及其标记。如何决定该上下文前缀 DSE 所来源的条目信息的命运，属于本地事务。

   如果下级 DSA 接收到一个结果，它将增加必要的相应类型为 **glue**、**subentry**、**admPoint**、**rhob** 和 **immSupr** 的一个 DSE，来表示 **contextPrefixInfo**；以及相应类型为 **rhob** 和 **entry** 的一个 DSE，来表示 **immediateSuperiorInfo**。上下文前缀 DSE 的标记被删除。

   如果有任何失败出现（如通信失败或终端系统失败），下级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的分等级操作绑定建立操作都接收到一个结果或错误为止。

24.3.2 修改规程

定义了下列规程来修改一个 HOB，该 HOB 是根据 24.3.1 节详述的规程而建立的。

24.3.2.1 上级发起的修改规程

本规程的调用可能是由于修改操作引起的，如 19.1 节中的描述，或者是由于管理干预而引起的（例如传递 HOB 的 **myAccessPoint**、**agreement** 或 **valid** 参数的变化）。另外，如果一个上级 DSA 检测出它提供给下级 DSA

的 **SuperiorToSubordinate** 中的 **contextPrefixInfo** 或 **immediateSuperiorInfo** 组件发生了变化，则它必须使用下列规程将新的信息传播给下级 DSA：

1) 将类型为 **subr** 的 DSE 标记为“将被修改”，并且如果这种修改是由于对下级上下文前缀条目的 RDN 进行修改而引起的，则一个新的 **subr** 类型为的 DSE 被增加，并被标记为“将被增加”。

2) 上级 DSA 根据已经存在的值产生一个新的 **bindingID** 值，方法是增加其 **version** 组件的值。使用此新的 **bindingID**，它向下级 DSA 发送一个修改操作绑定操作，且修改参数为 **SuperiorToSubordinateModification**。

3) 下级 DSA 检查 **bindingID** 中的 **identifier** 组件。如果它与上级之间无此合约，或者如果 **version** 组件小于 HOB 的版本，则它必须返回一个问题为 **invalidAgreement** 的 **operationalBindingError**。

4) 下级 DSA 可能会接受对 HOB 的修改，即修改或重建表示上下文前缀信息的 DSE，更新它的 **bindingID** 中的 **version** 组件，并且返回一个结果。可选的，它也可能会返回一个错误并且终止此合约。

5) 如果上级 DSA 接收到一个结果，则修改完成。如果这种修改是由于对下级上下文前缀条目的 RDN 进行修改而引起的，则类型为 **subr**，并被标注为“将被增加”的新的 DSE，其标注被删除，并且被标注为“将被修改”的老的 DSE 被删除。如果不是，则仅是“将被修改”的标注被简单删除。

   如果上级 DSA 接收到一个错误，则修改失败。“将被修改”的标注被删除。如果这种修改是由于对下级上下文前缀条目的 RDN 进行修改而引起的，则类型为 **subr**，并被标注为“将被增加”的新的 DSE 被删除。如果不是，则所采取的措施不在本号码簿规范的定义范围之内。

   如果有任何失败出现（如通信失败或终端系统失败），上级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的分等级操作绑定修改操作都接收到一个结果或错误为止。如果修改是由于一个修改下级上下文前缀条目的 RDN 的 **ModifyDN** 操作而引起的，且在修改完成之前由请求者中止了该操作（例如通过释放或中止应用连接），则上级 DSA 必须忽略此事件，并且完成此次修改（可能成功，也可能不成功）。在这种情况下，修改 DN 操作的输出不会被通知到用户。

24.3.2.2 下级发起的修改规程

本规程的调用可能是由于管理干预而引起的（例如传递 HOB 的 **myAccessPoint**、**agreement** 或 **valid** 参数的变化）。另外，如果一个下级 DSA 检测出它提供给上级 DSA 的 **SubordinateToSuperior** 的值发生了变化，则它必须使用下列规程将新的信息传播给上级 DSA：

1) 将类型为 **cp** 的 DSE 标注为“将被修改”。

2) 下级 DSA 根据已经存在的值产生一个新的 **bindingID** 值，方法是增加其 **version** 组件的值。使用此新的 **bindingID**，它向上级 DSA 发送一个修改操作绑定操作，且修改参数为 **SubordinateToSuperior**。

3) 上级 DSA 检查 **bindingID** 中的 **identifier** 组件。如果它与下级之间无此合约，或者如果 **version** 组件小于 HOB 的版本，则它必须返回一个问题为 **invalidAgreement** 的 **operationalBindingError**。

4) 上级 DSA 可能会接受对 HOB 的修改，即修改表示下级引用的 DSE，并返回一个结果。可选的，它也可能会返回一个错误并且终止此合约。

   另外，如果这个将被重命名的 DSE（类型为 **subr**）的上级 DSE 的类型为 **nssr**，则 DSA 在响应 HOB 修改请求之前，必须遵循 19.1.5 节中定义的规程（修改操作和 NSSR）以确保条目的新名字是无二义性的。

5) 如果下级 DSA 接收到一个结果，则修改完成，并且删除了标记。如果它接收到一个错误，则所采取的措施不在本号码簿规范的定义范围之内。

   如果有任何失败出现（如通信失败或终端系统失败），下级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的分等级操作绑定修改操作都接收到一个结果或错误为止。

24.3.3 终止规程

定义了下列规程来终止一个 HOB，该 HOB 是根据 24.3.1 节详述的规程而建立的。

24.3.3.1 上级DSA发起的终止

上级 DSA 发起的对分等级操作绑定的终止只能是由于管理干预而引起的。下列规程必须被遵循：

1) 上级 DSA 将表示下级引用的 DSE 标注为“将被删除”，因此在名字解析过程中，该下级引用不能再被使用。
2) 针对分等级操作绑定，上级 DSA 向下级 DSA 发出一个终止操作绑定的操作。**bindingID** 中的 **version** 组件被上级忽略。
3) 当下级 DSA 接收到该终止操作绑定，则它将删除关于此分等级操作绑定的所有信息，并且发送一个结果，除非 **bindingID** 中的 **identifier** 组件未知，在这种情况下，将返回一个问题为 **invalidID** 的 **operationalBindingError**。如何决定与该下级命名上下文相关的条目信息的命运，属于本地事务。
4) 如果上级 DSA 接收到一个结果，或者一个问题为 **invalidID** 的 **operationalBindingError**，则它必须删除标记为“将被删除”的 DSE，该 DSE 表示了与分等级操作绑定相关的下级引用，并且删除与操作绑定相关的所有信息。

   如果有任何失败出现（如通信失败或终端系统失败），上级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该DSA为发起者的每个悬置的分等级操作绑定终止操作都接收到一个结果或错误为止。

24.3.3.2 下级DSA发起的终止

由下级 DSA 发起的终止规程可以是由于一个删除条目操作而引起的，该操作删除了下级命名上下文中的最后一个条目，即上下文前缀条目，或者是由于管理干预而引起的。必须遵循下列规程：

1) 下级 DSA 将命名上下文的上下文前缀 DSE 标记为“将被删除”。
2) 下级 DSA 向上级 DSA 发起一个关于分等级操作绑定的终止操作绑定操作。**bindingID** 中的 **version** 组件被下级 DSA 忽略。
3) 当上级 DSA 接收到该终止操作绑定，则它将删除表示下级引用的 DSE，此下级引用是与分等级操作绑定相关联的，同时删除关于此操作绑定的所有信息，并且发送一个结果，除非 **bindingID** 中的 **identifier** 组件未知，在这种情况下，将返回一个问题为 **invalidID** 的 **operationalBindingError**。
4) 如果下级 DSA 接收到一个结果或者一个问题为 **invalidID** 的 **operationalBindingError**，则它必须删除关于此操作绑定的所有信息。

   注 — 命名上下文的条目信息的命运对下级 DSA 来说是一种本地事务。由于重新命名（即删除）一个命名上下文是不被修改 DN 操作所允许的，则一个管理者可能会终止 HOB，为命名上下文选择另一个上下文前缀，并且将其与 DIT 的另一部分重新连接起来（即建立一个新的 HOB）。

   如果有任何失败出现（如通信失败或终端系统失败），下级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该DSA为发起者的每个悬置的分等级操作绑定终止操作都接收到一个结果或错误为止。

24.4 操作规程

在一个分等级操作绑定的合作状态中可以被执行的操作是在应用上下文 **directorySystemAC** 中定义的那些操作。

在一个分等级操作绑定中包含的 DSA 所必须遵循的规程在第 16 节到 22 节中定义。

24.5 应用上下文的用法

为了能够使用本号码簿标准中的协议和规程来建立、修改或终止一个分等级操作绑定，一个 DSA 必须使用 **operationalBindingManagementAC** 应用上下文。

25 非特定分等级操作绑定

一个非特定分等级操作绑定被用来表示两个 DSA 之间的关系，这两个 DSA 拥有两个命名上下文，其中一个是另一个的直接下级。在 NHOB 的情况下，上级 DSA 拥有一个指向下级 DSA 所拥有的命名上下文的非特定下级引用；下级 DSA 拥有一个指向上级 DSA 所拥有的命名上下文的直接上级引用。操作绑定确保了在两个 DSA 之间可以交换和维护适当的知识信息，因此，两个 DSA 都能够在名字解析和操作评估的处理过程中，按照第 18 节和 19 节的定义来运转。

25.1 操作绑定类型特性

25.1.1 对称和角色

分等级操作绑定类型是一种非对称的操作绑定类型。在这种类型的绑定中有两种角色，分别为：

a) 上级命名上下文的主 DSA 所承担的角色，称为*上级 DSA*（相关的抽象角色为“A”）；以及

b) 下级命名上下文的主 DSA 所承担的角色，称为*下级 DSA*（相关的抽象角色为“B”）。

25.1.2 合约

在建立非特定分等级操作绑定的过程中，所交换的合约信息是 **NonSpecificHierarchicalAgreement** 的一个值。它仅包括新的命名上下文的直接上级条目的识别名（**immediateSuperior** 组件）。该信息必须被发起该 NHOB 的 DSA 所提供。

```
NonSpecificHierarchicalAgreement ::= SEQUENCE {
        immediateSuperior      [1]   DistinguishedName }
```

注 — 下级 DSA 如何判断新的命名上下文的名字是无二义性的，不在本建议书 | 国际标准的定义范围之内。如果名字被有关命名当局正确分配，且没有其他的 DSA 拥有与此名字相同的一个主条目，则该名字是无二义性的。

25.1.3 发起者

25.1.3.1 建立

一个非特定分等级操作绑定的建立仅可以由下级 DSA 角色来发起。下级 DSA 的发起（将一个或多个本地存在的条目或子树与全球 DIT 连接起来）是由于管理干预而引起的。

25.1.3.2 修改

一个非特定分等级操作绑定的修改可以由任意一种角色来发起。由于上级上下文前缀信息的修改，使得上级 DSA 可能会发起修改。这可以是任何修改操作的结果，或者是管理干预的结果。

如果该命名上下文（或者在下级角色的情况下，它的直接下级命名上下文中的一个）的访问点信息发生了变化，则任何一个 DSA 也都可能修改此 NHOB。

25.1.3.3 终止

一个分等级操作绑定的终止可以由任意一种角色来发起。上级 DSA 的发起是由于管理干预而引起的。下级 DSA 的发起可以是由于一个删除条目操作而引起的，该操作删除了合约中 **immediateSuperior** 组件的直接下级所拥有的最后一个上下文前缀条目，或者是由于管理干预而引起的。

25.1.4 建立参数

上级 DSA 发起的建立参数是 **NHOBSuperiorToSubordinate** 的一个值，它与相应的 HOB 建立参数相同，除了 **entryInfo** 组件是缺失的。

```
NHOBSuperiorToSubordinate ::= SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})
```

下级 DSA 发起的建立参数是 **NHOBSubordinateToSuperior** 的一个值，它与相应的 HOB 建立参数是相同的，除了 **alias** 和 **entryInfo** 组件是缺失的。

```
NHOBSubordinateToSuperior ::= SEQUENCE {
        accessPoints      [0]   MasterAndShadowAccessPoints OPTIONAL,
        subentries        [3]   SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }
```

25.1.5 修改参数

这些参数与相应的建立参数是相同的，并且被用于表示在 NHOB 建立之后，建立参数中提供的信息所发生的变化。

如果 **NHOBSuperiorToSubordinate** 或者 **NHOBSubordinateToSuperior** 中的任何组件经历了一次变化（例如 **NHOBSuperiorToSubordinate** 中的 **contextPrefixInfo** 组件），则修改参数中的相应组件（例如 **NHOBSuperiorToSubordinate** 中的 **contextPrefixInfo** 组件）必须在修改操作绑定中被完整提供。

25.1.6 终止参数

当终止一个 NHOB 时，两种角色都不提供终止参数。

25.1.7 类型标识

非特定分等级操作绑定由对象标识符来标识，这些对象标识符是在 25.2 节定义 **nonSpecificHierarchicalOperationalBinding OPERATIONAL-BINDING** 信息对象时被分配的。

25.2 操作绑定信息对象类定义

本小节使用 ITU-T X.501 建议书 | ISO/IEC 9594-2 中定义的 **OPERATIONAL-BINDING** 信息对象类模板，定义了非特定分等级操作绑定的类型。

```
nonSpecificHierarchicalOperationalBinding  OPERATIONAL-BINDING  ::= {
        AGREEMENT       NonSpecificHierarchicalAgreement
        APPLICATION CONTEXTS {
                        { directorySystemAC } }
        ASYMMETRIC
            ROLE-A {            -- 上级 DSA
                ESTABLISHMENT-PARAMETER     NHOBSuperiorToSubordinate
                MODIFICATION-INITIATOR      TRUE
                MODIFICATION-PARAMETER      NHOBSuperiorToSubordinate
                TERMINATION-INITIATOR       TRUE }
            ROLE-B {            -- 下级 DSA
                ESTABLISHMENT-INITIATOR     TRUE
                ESTABLISHMENT-PARAMETER     NHOBSubordinateToSuperior
                MODIFICATION-INITIATOR      TRUE
                MODIFICATION-PARAMETER      NHOBSubordinateToSuperior
                TERMINATION-INITIATOR       TRUE }
        ID                  id-op-binding-non-specific-hierarchical }
```

25.3 非特定分等级操作绑定管理的DSA规程

在下面的规程中，如同 24.3 节中描述的规程那样，由一个 DSA 创建的一个新 DSE 或一个标记必须被存储在一个静态存储器中。

在下面所描述的建立和修改两个规程中，承担了响应者角色的 DSA（即没有发起建立或修改操作）可能会向承担了发起者角色的 DSA 提供信息（例如操作属性），而由于这样或那样的原因，这些信息是不可接受的。在这些情况下，发起者 DSA 可能会终止此操作绑定。

25.3.1 建立规程

仅有下级 DSA 才可能会发起一个分等级操作绑定。这可能是由于一个管理者希望将 DSA 内拥有的一个或多个条目子树与全球 DIT 中的某个点相关联起来而引起的。在这种情况下，下级 DSA 必须根据下列规程建立一个 NHOB：

1) 下级 DSA 或者具有一个类型为 **cp** 的 DSE，该 DSE 是一个已经存在的命名上下文的一部分，或者创建一个新的这种 DSE。它将此 DSE 标记为“*将被增加*”，且产生一个唯一的 **bindingID**，并将其与上下文前缀 DSE 存储在一起。
2) 下级 DSA 向上级 DSA 发送一个建立操作绑定操作，并包含下列参数：
   a) **bindingType** 被设置为 **nonSpecificHierarchicalOperationalBindingID**；
   b) 适当的 **NHOBSubordinateToSuperior** 建立参数；
   c) **NonSpecificHierarchicalAgreement** 中的 **immediateSuperior** 组件被设置为新条目的直接上级的识别名；

d) 适当的 **bindingID**、**myAccessPoint** 和 **valid** 参数。

3) 上级 DSA 检测出它是新的上下文前缀条目的直接上级的属主，或者返回一个问题为 **roleAssignment** 的 **operationalBindingError**。

4) 上级 DSA 将 DSE 类型 **nssr**（以及 **nonSpecificKnowledge** 属性信息）加入到新条目的直接上级的 DSE 中，并将 **bindingID** 与其一起存储起来，然后返回一个结果。

5) 如果下级 DSA 接收到一个错误，它将删除新的上下文前缀 DSE 及其标记。如何决定该上下文前缀 DSE 所来源的条目信息的命运，属于本地事务。

如果下级 DSA 接收到一个结果，它将增加必要的相应类型为 **glue**、**subentry**、**admPoint**、**rhob** 和 **immSupr** 的一个 DSE，来表示 **contextPrefixInfo**；以及相应类型为 **rhob** 和 **entry** 的一个 DSE，来表示 **immediateSuperiorInfo**。上下文前缀 DSE 的标记被删除。

如果有任何失败出现（如通信失败或终端系统失败），下级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的分等级操作绑定建立操作都接收到一个结果或错误为止。

25.3.2 修改规程

如果上级 DSA 检测出在一个非特定分等级操作绑定内，它提供给下级 DSA 的 **NHOBSuperiorToSubordinate** 信息发生了任何变化，则它必须将发生变化的信息传播给下级 DSA。如果 NHOB 是使用 25.3.1 节中的规程而建立的，则它的修改必须根据 24.3.2.1 节中为修改分等级操作绑定而定义的规程（其中，**NHOBSuperiorToSubordinate** 取代了 **SuperiorToSubordinateModification**）。

类似的，如果下级 DSA 检测出它提供给上级 DSA 的 **NHOBSubordinateToSuperior** 信息发生了任何变化，则它必须将此变化传播给上级 DSA。如果 NHOB 是使用 25.3.1 节中的规程而建立的，则它的修改必须根据 24.3.2.2 节中为修改分等级操作绑定而定义的规程（其中，**NHOBSubordinateToSuperior** 取代了 **SubordinateToSuperior**）。

25.3.3 终止规程

定义了下列规程来终止一个 NHOB，该 NHOB 是根据 25.3.1 节详述的规程而建立的。

25.3.3.1 上级DSA发起的终止

上级 DSA 发起的对分等级操作绑定的终止只能是由于管理干预而引起的。下列规程必须被遵循：

1) 上级 DSA 将 **nonSpecificKnowledge** 属性中对应于下级 DSA 的值标注为“将被删除”，该属性由直接上级条目的 DSE 所拥有。

2) 针对与下级 DSA 之间的 NHOB，上级 DSA 发送一个终止操作绑定操作。**bindingID** 中的 **version** 组件被上级 DSA 忽略。

3) 当下级 DSA 接收到该终止操作绑定的操作时，它将删除关于此 NHOB 的所有信息，并且发送一个结果，除非 **bindingID** 中的 **identifier** 组件未知，在这种情况下，将返回一个问题为 **invalidID** 的 **operationalBindingError**。如何决定与该下级命名上下文相关的条目信息的命运，属于本地事务。

4) 如果上级 DSA 接收到一个结果，或者一个问题为 **invalidID** 的 **operationalBindingError**，则它必须删除标记为“将被删除”的 **nonSpecificKnowledge** 属性的值，该属性表示了与 NHOB 相关的访问点信息，并且删除与操作绑定相关的所有信息。如果这是 **nonSpecificKnowledge** 属性的最后一个值，则它从 DSE 中删除此 **nonSpecificKnowledge** 属性和 DSE 类型 **nssr**。

如果有任何失败出现（如通信失败或终端系统失败），上级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的 NHOB 终止操作都接收到一个结果或错误为止。

25.3.3.2 下级DSA发起的终止

由下级 DSA 发起的终止规程可以是由于一个删除条目操作而引起的，该操作删除了下级命名上下文中的最后一个条目，即由下级 DSA 所拥有的最后一个下级命名上下文的上下文前缀条目，或者是由于管理干预而引起的。必须遵循下列规程：

1) 下级 DSA 将命名上下文的上下文前缀 DSE 标注为“将被删除”。

2) 下级 DSA 向上级 DSA 发起一个关于分等级操作绑定的终止操作绑定操作。**bindingID** 中的 **version** 组件被下级 DSA 忽略。

3) 当上级 DSA 接收到该终止操作绑定的操作，它将删除表示与 NHOB 相关的访问点信息的 **nonSpecificKnowledge** 属性的值，同时删除关于此操作绑定的所有信息，从下级命名上下文的直接上级 DSE 中删除 **nonSpecificKnowledge** 属性和 DSE 类型 **nssr**（如果被删除的值是 **nonSpecificKnowledge** 属性的最后一个值），并且发送一个结果，除非 **bindingID** 中的 **identifier** 组件未知，在这种情况下，将返回一个问题为 **invalidID** 的 **operationalBindingError**。

4) 如果下级 DSA 接收到一个结果或者一个问题为 **invalidID** 的 **operationalBindingError**，则它必须删除关于此操作绑定的所有信息。如何决定与该下级命名上下文相关的条目信息的命运，属于本地事务。

如果有任何失败出现（如通信失败或终端系统失败），下级 DSA 都必须从步骤 2)开始重复执行上述步骤，直到以该 DSA 为发起者的每个悬置的终止 NHOB 操作都接收到一个结果或错误为止。

25.4 操作规程

在一个非特定分等级操作绑定的合作状态中可以被执行的操作是在应用上下文 **directorySystemAC** 中定义的那些操作。

在一个非特定分等级操作绑定中包含的 DSA 所必须遵循的规程在第 16 节到 22 节中定义。

25.5 应用上下文的用法

为了能够使用本号码簿标准中的协议和规程来建立、修改或终止一个非特定分等级操作绑定，一个 DSA 必须使用 **operationalBindingManagementAC** 应用上下文。

附件 A

分布式操作的ASN.1定义

（本附件是本建议书 | 国际标准的组成部分）

本附件包含了本号码簿规范中的所有 ASN.1 类型和值定义，以 ASN.1 模块 **DistributedOperations** 的形式提供。

```
DistributedOperations  {joint-iso-ITU-T ds(5) module(1) distributedOperations(3) 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号码簿规范所包含的其它的ASN.1模块中，供其使用，
-- 也可以被其它应用所使用，这些应用将使用本模块中的定义来访问号码簿服务。
-- 其它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号码簿服务而进行的扩展和修改。


IMPORTS

-- 来自ITU-T X.501建议书 | ISO/IEC 9594-2

        basicAccessControl, commonProtocolSpecification, directoryAbstractService, enhancedSecurity,
        informationFramework,selectedAttributeTypes, serviceAdministration, upperBounds
            FROM UsefulDefinitions {joint-iso-ITU-T ds(5) module(1) usefulDefinitions(0) 5}

        DistinguishedName, Name, RDNSequence
            FROM InformationFramework informationFramework

        MRMapping, SearchRuleId
            FROM ServiceAdministration serviceAdministration

        AuthenticationLevel
            FROM BasicAccessControl basicAccessControl

        OPTIONALLY-PROTECTED{ }
            FROM EnhancedSecurity enhancedSecurity

-- 来自ITU-T X.511建议书 | ISO/IEC 9594-3

        abandon, addEntry, CommonResults, compare, directoryBind, list,
        modifyDN, modifyEntry, read, referral, removeEntry, search, SecurityParameters
            FROM DirectoryAbstractService directoryAbstractService

-- 来自ITU-T X.519建议书 | ISO/IEC 9594-5


        ERROR, id-errcode-dsaReferral, OPERATION
            FROM CommonProtocolSpecification commonProtocolSpecification

-- 来自ITU-T X.520建议书 | ISO/IEC 9594-6

        DirectoryString{}, PresentationAddress, ProtocolInformation, UniqueIdentifier
            FROM SelectedAttributeTypes selectedAttributeTypes

        ub-domainLocalID, ub-labeledURI
            FROM UpperBounds upperBounds;
```

```
-- 派生链接操作所需的参数化类型 --
chained { OPERATION : operation } OPERATION  ::=  {
        ARGUMENT  OPTIONALLY-PROTECTED {
            SET {
                chainedArgument ChainingArguments,
                argument        [0]     operation.&ArgumentType } }
        RESULT    OPTIONALLY-PROTECTED {
            SET {
                chainedResult   ChainingResults,
                result          [0]     operation.&ResultType } }
        ERRORS{ operation.&Errors EXCEPT referral | dsaReferral }
        CODE    operation.&operationCode }
-- 绑定和解绑定操作 --
dSABind             OPERATION  ::=  directoryBind
--dSAUnbind          OPERATION  ::=  directoryUnbind
-- 链接操作 --
chainedRead         OPERATION  ::=  chained { read }
chainedCompare      OPERATION  ::=  chained { compare }
chainedAbandon      OPERATION  ::=  abandon
chainedList         OPERATION  ::=  chained { list }
chainedSearch       OPERATION  ::=  chained { search }
chainedAddEntry     OPERATION  ::=  chained { addEntry }
chainedRemoveEntry  OPERATION  ::=  chained { removeEntry }
chainedModifyEntry  OPERATION  ::=  chained { modifyEntry }
chainedModifyDN     OPERATION  ::=  chained { modifyDN }
-- 错误和参数 --
dsaReferral   ERROR  ::=  {
        PARAMETER OPTIONALLY-PROTECTED {
            SET {
                reference       [0]     ContinuationReference,
                contextPrefix   [1]     DistinguishedName OPTIONAL,
                COMPONENTS OF CommonResults } }
        CODE        id-errcode-dsaReferral }
-- 公共变量和结果 --
ChainingArguments  ::=  SET {
        originator              [0]     DistinguishedName OPTIONAL,
        targetObject            [1]     DistinguishedName OPTIONAL,
        operationProgress       [2]     OperationProgress
                                        DEFAULT { nameResolutionPhase notStarted },
        traceInformation        [3]     TraceInformation,
        aliasDereferenced       [4]     BOOLEAN DEFAULT FALSE,
        aliasedRDNs             [5]     INTEGER OPTIONAL,
                                -- 仅在第一版本系统中存在
        returnCrossRefs         [6]     BOOLEAN DEFAULT FALSE,
        referenceType           [7]     ReferenceType DEFAULT superior,
        info                    [8]     DomainInfo OPTIONAL,
        timeLimit               [9]     Time OPTIONAL,
        securityParameters      [10]    SecurityParameters DEFAULT { },
        entryOnly               [11]    BOOLEAN DEFAULT FALSE,
        uniqueIdentifier        [12]    UniqueIdentifier OPTIONAL,
        authenticationLevel     [13]    AuthenticationLevel OPTIONAL,
        exclusions              [14]    Exclusions OPTIONAL,
        excludeShadows          [15]    BOOLEAN DEFAULT FALSE,
```

```
        nameResolveOnMaster     [16]    BOOLEAN DEFAULT FALSE,
        operationIdentifier     [17]    INTEGER OPTIONAL,
        searchRuleId            [18]    SearchRuleId  OPTIONAL,
        chainedRelaxation       [19]    MRMapping     OPTIONAL,
        relatedEntry            [20]    INTEGER  OPTIONAL,
        dspPaging               [21]    BOOLEAN DEFAULT FALSE,
        nonDapPdu               [22]    ENUMERATED { ldap (0) } OPTIONAL,
        streamedResults         [23]    INTEGER OPTIONAL,
        excludeWriteableCopies  [24]    BOOLEAN DEFAULT FALSE }

Time ::= CHOICE {
        utcTime         UTCTime,
        generalizedTime   GeneralizedTime }

DomainInfo   ::=   ABSTRACT-SYNTAX.&Type

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

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

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

TraceInformation  ::= SEQUENCE OF TraceItem

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

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

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

LabeledURI  ::= DirectoryString{ub-labeledURI}

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

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

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

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

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

ContinuationReference  ::=  SET {
        targetObject                [0]     Name,
        aliasedRDNs                 [1]     INTEGER OPTIONAL, --仅在第一版本系统中存在
        operationProgress           [2]     OperationProgress,
        rdnsResolved                [3]     INTEGER OPTIONAL,
        referenceType               [4]     ReferenceType,
        accessPoints                [5]     SET OF AccessPointInformation,
        entryOnly                   [6]     BOOLEAN DEFAULT FALSE,
        exclusions                  [7]     Exclusions OPTIONAL,
        returnToDUA                 [8]     BOOLEAN DEFAULT FALSE,
        nameResolveOnMaster         [9]     BOOLEAN DEFAULT FALSE }

END  -- DistributedOperations
```

附件 B

分布式名字解析的示例

（本附件不是本建议书 | 国际标准的组成部分）

图 B.1 是一个关于如何使用分布式名字解析来处理不同的号码簿请求的示例。本例的基础是 ITU-T X.501 建议书 | ISO/IEC 9594-2 的附件 O（知识的建模）中所描述的一棵假设的 DIT 以及相应的 DSA 配置，为了便于阅读在这里复制如下。

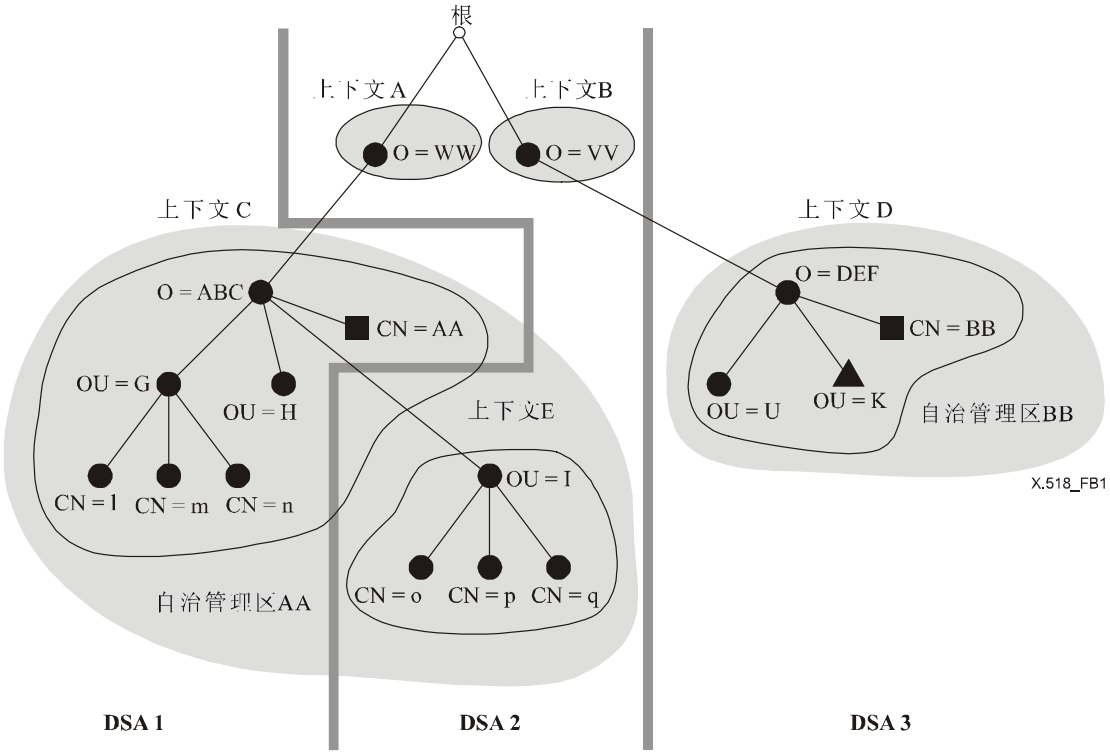


图 B.1－映射到三个DSA上的假设DIT

假设一种链接的传播模式，发给 DSA 1 的下列请求应按如下处理：

1) 一个请求，其中识别名为{C = WW, O = ABC, OU = G, CN = l}
   — 名字解析将对目标名字中的每个 RDN 与 DSA 1 所拥有的 DSE 进行成功地匹配，直到目标 DSE 被定位。

2) 一个请求，其中识别名为{C = WW, O = JPR}
   — DSA 1 中的名字解析规程将匹配到 DSE C = WW，但无法进行后续的匹配。在这点上，DSA 1 潜在地发现两个引用可以帮助它来继续进行：一个是在 DSE C＝WW 中的 **immSupr** 引用；另一个是在根 DSE 中的 **supr** 引用。在这个假设的示例中，两个都指向 DSA 2。因此，该请求被链接至 DSA 2。
   — 在 DSA 2 中，**名字解析**规程将匹配到 DSE C = WW，但无法进行后续的匹配。在这种情况下，由于 DSE C = WW 的类型为 **cp** 和 **entry**，且 DSA 2 是该条目的主 DSA，而在 C＝WW 处没有 **nssr**，因此 DSA 2 可以判断出在本号码簿中没有此名字。一个问题为 **noSuchObject** 的 **nameError** 将被返回。

3) 一个请求，其中识别名为{C = VV, O = DEF, OU = K}

— DSA 1 中的**名字解析**规程不能匹配到任何一个 DSE。仅有的一个可用的引用是根 DSE 中的 **supr** 引用，该引用指向 DSA 2。因此该请求被链接至 DSA 2。

— 在 DSA 2 中，**名字解析**规程将匹配到 DSE C = VV，然后匹配到 DSE O = DEF，然后就无法进行后续的匹配了。由于 DSE O = DEF 被发现其类型为 **subr**，这个特定的指向 DSA 3 的知识引用被使用，并将本请求链接至 DSA 3。

— 在 DSA 3 中，**名字解析**规程将匹配到完整的目标对象名字，并且发现这个被定位的 DSE 类型为 **alias**。假设在这种情况下，别名将被解除引用，使用包含在已匹配 DSE 中的 **aliasedEntryName**，可以构建一个新的名字。于是 DSA 3 将重新进入到**名字解析**规程中继续进行处理。

附件 C

鉴权的分布式使用

（本附件不是本建议书 | 国际标准的组成部分）

C.1 摘要

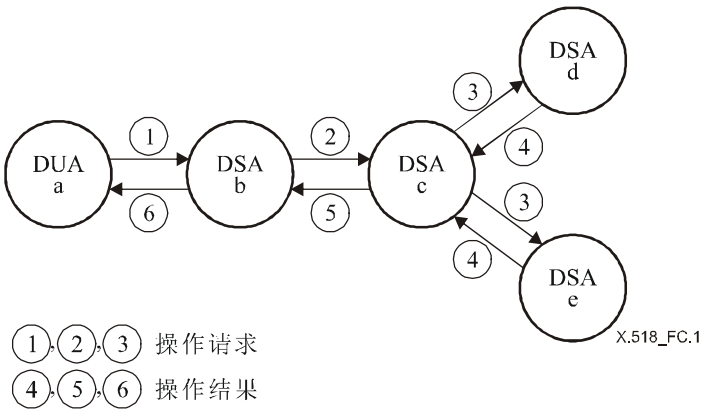
安全模式在 TU-T X.501 建议书 | ISO/IEC 9594-2 的第 17 节定义。下面是该模型的要点摘要：

a) 在 DSP 中支持强鉴权，方法是对请求、结果和错误等进行签名。

b) 在 DSP 中支持对请求、结果和错误等进行加密。

本附件描述了在分布式号码簿中如何实现这些要求。它使用了 ITU-T X.509 建议书 | ISO/IEC 9594-8 中定义的术语和表示法。

C.2 分布式保护模型



图C.1－分布式保护

图 C.1 举例说明了在规定分布式保护过程中所使用的模型。该模型标识了在一般情况下列表操作或搜索操作的信息流序列。该操作被认为是由 DUA 'a'发起的，引用了一个位于执行操作的 DSA 'c'内的目标对象，DSA 'b'、'c'、'd'和'e'也被包含在内。

最初，DUA 'a'与没有拥有目标对象的任何 DSA（DSA 'b'）交互，但该 DSA 能够通过链接，将请求导航到拥有目标对象的一个 DSA（DSA 'c'）。如果所有的 DSA 都可以以提名方式运行，则该模型将被明显地简化，且每个 DSA/DSA 的交互，从保护来看，将等同于 DUA 'a'和 DSA 'b'之间的交互。

C.2.1 保护的质量

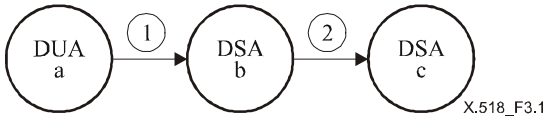
在应用连接的生命周期中使用的保护质量是在号码簿绑定操作过程中建立的。系统策略将对 DUA 和 DSA 必须遵守的保护级别进行评估。DIRQOP 是一个信息对象类，可被用于规范与每个操作（请求、结果或错误）相关联的保护质量。DUA 在 **DirectoryBindArgument** 中传送 **DIRQOP** 信息对象类，而 DSA 在 **DirectoryBindResult** 中接受此保护级别。保护质量可被用于提供下列类型的保护：签名的、加密的、或签名并加密的。

C.3 签名的链接操作

如果支持数字签名的链接操作，则由 DUA 来负责验证 DSA 所返回的列表或搜索结果中的数字签名。如果使用一个分布式环境来产生列表或搜索结果，则要求 DUA 有能力验证来自多个 DSA 的数字签名。将列表和搜索操作的结果相关联起来是 DUA 的责任。DSA 不必代表 DUA 来合并这些结果。在某些情况下，DUA 可能会从多个不同的 DSA 中接收信息，而每个 DSA 支持不同级别的鉴权和数字签名。则由 DUA 来决定当数字签名非法时，是否使用所返回的信息。

C.3.1 链接的签名变量

如果 DUA 对一个 DAP 变量进行了签名，则该签名必须在请求的生命周期内都被维护。当 DSA 执行访问控制验证时，该签名可被 DSA 验证和使用。如果 DSA 决定该请求需要链接到另一个 DSA 来处理时，它应当在必要的链接变量中，包含此 DUA 签名的请求。如果该 DSA 准备支持签名的 DSP 操作（DSA 到 DSA），则 DSA 的许可证将被用于对 DSP 的 **ChainingArguments** 进行签名，且 DUA 的签名应当与原始的 DAP 请求一起被维护。

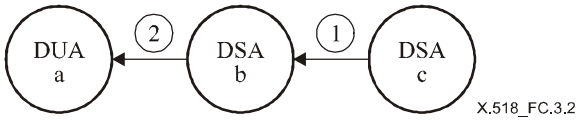


1 DUA 'a'用户对DAP请求签名

2 DUA 'b'对DSP链接变量签名
（由DUA 'a'用户签名的DAP请求）

C.3.2 链接的签名结果

如果 DUA 用户希望从号码簿中接收到签名的结果，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **SIGNED**。远端 DSA 应当有能力被配置为可以发送数字签名后的 **ChainingResults**。可选地，远端 DSA 能够对 DAP 结果和 DSP 的 **ChainingResults** 进行签名，以便支持端到端的签名。DSA 'b'将负责验证远端 DSA 的 DSP 签名，而 DUA 'a'将负责验证 DSA 的 DAP 结果签名。

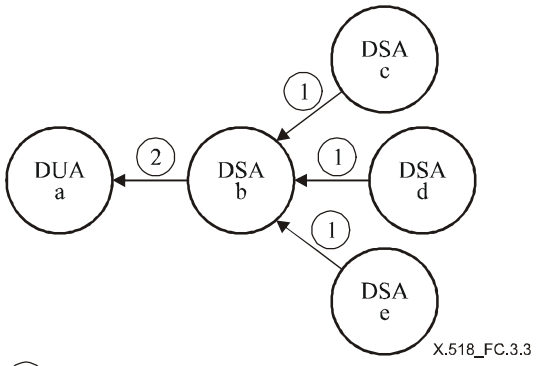


1 DSA 'c'对DSP链接结果和DAP结果签名

2 DSA 'b'返回由DSA 'c'签名的DAP结果

C.3.3 合并签名的列表或搜索结果

如果使用一个分布式环境来产生列表或搜索结果，则要求 DUA 有能力验证来自多个 DSA 的数字签名。将列表和搜索操作的结果合并起来是 DUA 的责任。DSA 不必代表 DUA 来合并这些结果。在某些情况下，DUA 可能会从多个不同的 DSA 中接收信息，而每个 DSA 支持不同级别的鉴权和数字签名。这种时候由 DUA 来决定当数字签名非法时，是否使用所返回的信息。



(1) DSA 'c', 'd', 'e'对DSP链接结果（由DSA 'c', 'd', 'e'签名的DAP结果）签名

(2) DSA 'b'返回由DSA 'c', 'd'和'c'签名的部分DAP结果，DSA 'b'不合并DAP结果

注 — DSA 到 DSA 的 DSP 协议也能够被签名，被加密或被签名并加密。

C.3.4 多链接请求

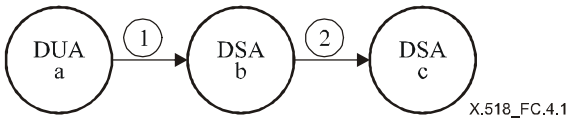
如果 DSA 判断 DAP 请求需要被链接到多个其他 DSA 时，它能够将请求进行多链接，或者以并行方式，或者以顺序方式。有两种分解模式被描述：非特定下级引用（NSSR）分解，或请求分解。在 NSSR 分解中，DSA 向其他指定的 DSA 发送相同的请求。在请求分解中，DSA 向其他的每个 DSA 发送一个部分的（可能是不同的）并发的请求。

C.4 加密的链接操作

如果支持加密，则在号码簿的每个组件之间需要提供等同的保护。要形成一个有关策略等同性的协定，需要使用映射，但此映射不在本号码簿规范的定义范围之内。

C.4.1 对请求的点到点（DUA->DSA 或DSA->DSA）加密

如果一个 DUA 用户想对 DAP 请求进行加密，则加密仅能够基于点到点方式出现。DUA 将为 DSA 'b'加密该 DAP 请求；然而，DUA 用户不知道该请求最终是否会被链接到一个远端 DSA 来处理。DSA 'b'将解密该请求，并且尝试满足该请求。如果 DSA 'b'判断该请求应当被链接到另一个 DSA（DSA 'c'）来处理，则 DSA 'b'为 DSA 'c'加密该链接操作。为 DSP 请求和响应（链接的操作变量和结果）选择点对点保护是由 DSA 'b'和 DSA 'c'之间在 DSP 绑定中建立的 **dirqop** 来指示的。

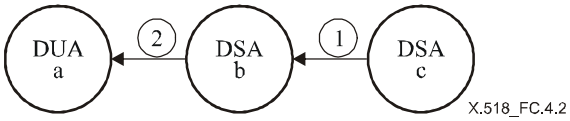


(1) DUA 'a'用户为DSA 'b'加密DAP请求

(2) DSA 'b'对DSP链接操作变量加密

C.4.2 对结果的点到点（DUA<-DSA或DSA<-DSA）加密

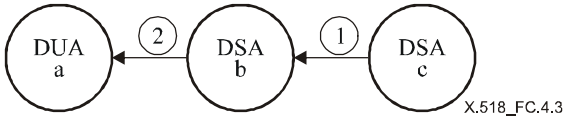
如果 DUA 用户希望从号码簿中接收加密的结果或错误，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **ENCRYPTED**，或者如果该字段不存在，则处于链接操作变量中的 **SecurityParameters.ProtectionRequest** 字段将被设置为可以体现 DAP **BindArgument** 中的 **DIRQOP**。远端 DSA（DSA 'c'）应当有能力被配置为可以发送加密后的链接操作结果。在这个场景中，DSA 'c'系统判断出它能够满足该请求，于是它产生一个 DAP 结果和 DSP 链接操作结果。通过 DSA 'c'为 DSA 'b'加密 DSP 的链接操作结果，则能够实现点到点加密。DSA 'b'能够解密 DSP 链接操作结果，并且为 DUA 'a'用户加密 DAP 结果。这就提供了结果的点到点加密。DUA 'a'将负责对它的本地 DSA（DSA 'b'）的 DAP 结果进行解密。



(1) DSA 'c'对DSP链接操作结果加密
(2) DSA 'b'为DUA 'a'加密DAP结果

C.4.3 对DAP结果的端到端加密和对DSP链接结果的点到点加密

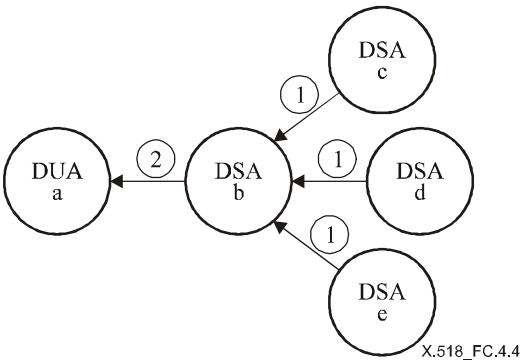
如果 DUA 'a'用户希望从号码簿中接收加密后的结果或错误，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **ENCRYPTED**，或者如果该字段不存在，则处于链接操作变量中的 **SecurityParameters.ProtectionRequest** 字段应当被设置为可以体现 DAP 绑定中的 **DIRQOP**。远端 DSA 'c'应当有能力被配置为可以发送加密后的链接操作结果。在这个场景中，DSA 'c'系统判断出它能够满足该请求，于是它对 DAP 结果（为 DUA 用户）产生一个端到端加密，并对 DSP 链接操作结果产生一个点到点加密。端到端加密可被 DSA 'c'执行，因为它知道谁将是 DUA 'a'用户。通过 DSA 'c'为 DSA1 加密 DSP 链接操作结果，则能够实现对 DSP 链接操作结果的点到点加密。DSA 'b'能够解密 DSP，并且将加密后的 DAP 结果传递到 DUA 'a'用户。DUA 'a'将负责解密它通过 DSA 'b'从 DSA 'c'接收到的 DAP 结果。



(1) DSA 'c'为DSA 'b'加密DSP链接操作结果；这包括来自DSA 'c'的为DUA 'a'加密的DAP结果
(2) DSA 'b'返回DSA 'c'为DUA 'a'加密的DAP结果

C.4.4 合并列表/搜索结果（与DSA 1的重新加密合并）

如果 DUA 'a'用户希望从号码簿中接收加密的列表或搜索结果或错误，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **ENCRYPTED**，或者如果该字段不存在，则处于链接操作变量中的 **SecurityParameters.ProtectionRequest** 字段应当被设置为可以体现 DAP 绑定中的 **DIRQOP**。本地 DSA（DSA 'b'）可能选择将此列表/搜索请求多链接到多个其他 DSA 处（或者是并行的，或者是顺序的）。远端 DSA（DSA 'c'、'd'和'e'）应当有能力被配置为可以发送加密后的链接列表/搜索结果。在本模型中，每个远端 DSA（'c'、'd'和'e'）都满足请求，并产生 DAP 结果和加密的 DSP 链接操作结果。由远端 DSA（'c'、'd'和'e'）产生的链接操作结果被传递到 DSA 'b'。DSA 'b'接收到每个链接操作结果，解密结果并将这些结果整理或合并为一个公共结果。DSA 'b'于是便加密此新的公共列表/搜索结果，并将其发送到 DUA 'a'用户。点到点加密的完成是通过远端 DSA 为 DSA 'b'加密 DSP 链接操作结果，然后 DSA 'b'为 DUA 'a'用户加密 DAP 结果而实现的。DUA 将负责解密一个合并后的 DAP 结果。

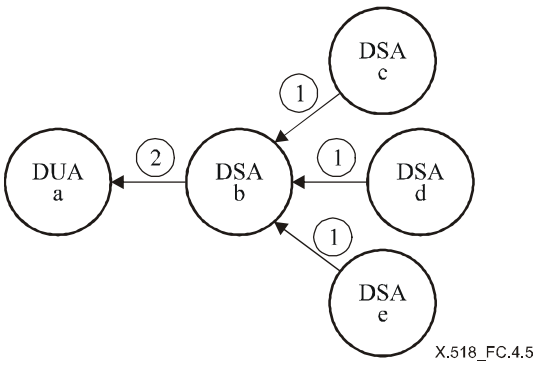


(1) DSA 'c'，'d'，'e'加密DSP链接操作结果（包括DAP结果）
(2) DSA 'b'解密来自DSA 'c'，DSA 'd'和DSA 'e'的DSP链接操作结果，然后合并DAP结果，并为DUA 'a'重新加密DAP结果

C.4.5 不允许合并列表/搜索结果

（提供端到端的 DAP 列表/搜索结果加密的 DSA 'b'不执行合并）

如果 DUA 用户希望从号码簿中接收加密的列表或搜索结果或错误，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **ENCRYPTED**，或者如果该字段不存在，则处于链接操作变量中的 **SecurityParameters.ProtectionRequest** 字段应当被设置为可以体现 DAP 绑定中的 **DIRQOP**。本地 DSA 可能选择将此列表/搜索请求多链接到多个其他的 DSA 处（或者是并行的，或者是顺序的）。远端 DSA（DSA 'c'、'd'和'e'）应当有能力被配置为可以发送加密后的链接列表/搜索结果。在本场景中，每个远端 DSA（'c'、'd'和'e'）都满足请求，并产生加密的 DAP 结果（为 DUA 'a'用户）和加密的 DSP 链接操作结果（为 DSA 'b'）。由远端 DSA（'c'、'd'和'e'）产生的链接操作结果被传递到 DSA 'b'。DSA 'b'将列表/搜索结果（由'c'、'd'和'e'加密）不做任何改变地转送并发送到 DUA 'a'。端到端加密的完成是通过远端 DSA 为 DUA 'a'用户加密 DAP 列表/搜索结果而实现的，而点到点加密的完成是通过远端 DSA 为 DSA 'b'加密 DSP 链接操作结果而实现的。DUA 'a'将负责解密每一个返回的 DAP 列表/搜索结果。

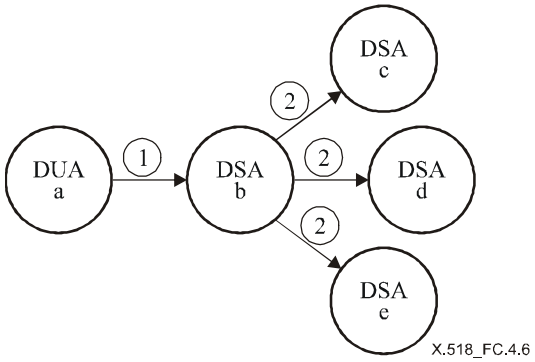


① DSA 'c'，'d'，'e'为DSA 'b'加密DSP链接操作结果；这包括那些已经为DUA 'a'用户加密的内容

② DSA 'b'解密来自DSA 'c'，DSA 'd'和DSA 'e'的DSP链接操作结果，然后不执行解密或合并便将DAP结果（该结果由'c'，'d'和'e'为DUA 'a'加密）传递到DUA 'a'

C.4.6 使用一个加密密钥（净密钥）多链接一个DAP请求

如果 DUA 'a'用户希望从号码簿中接收加密的结果或错误，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **ENCRYPTED**，或者如果该字段不存在，则链接操作变量中的 **SecurityParameters.ProtectionRequest** 字段应当被设置为可以体现 DAP 绑定中的 **DIRQOP**。本地 DSA 可能选择将此列表/搜索请求多链接到多个其他的 DSA 处（或者是并行的，或者是顺序的）。本地 DSA（DSA 'b'）可能被配置为支持一个加密密钥或净密钥。一个净密钥是一个对称的加密密钥，由链接中的所有 DSA 共享。通过使用一个净密钥，DSA 'b'仅需要对链接请求加密一次。每个远端 DSA 都知道此净密钥，并能够使用此净密钥来解密 DSP 链接操作变量。在本场景中，点到点加密的实现是通过 DUA 用户为 DSA 'b'加密 DAP 请求来实现的，而 DSA 'b'能够使用一个净密钥来实现到远端 DSA 的点到点加密。



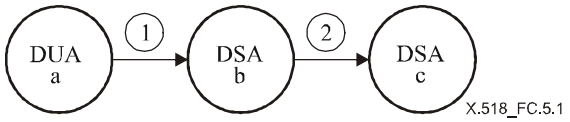
(1) DUA 'a'为DSA 'b'加密一个DAP变量

(2) DSA 'b'解密此请求，并尝试满足此请求；如果DSA 'b'不能满足请求，它使用一个“净密钥”来加密DSA链接操作请求（包括DAP请求）。链接请求被发送到DSA 'c'，'d'和'e'

C.5 签名并加密的分布式操作

C.5.1 端到端签名，与点到点加密

如果一个 DUA 'a'用户希望对 DAP 请求进行签名并加密，则签名可以被端到端提供，而加密仅能够基于点到点方式出现。DUA 'a'能够为 DSA 'b'签名并加密 DAP 请求；然而，DUA 'a'用户不知道该请求最终是否会被链接到一个远端 DSA（DSA 'c'）来处理。DSA 'b'将解密该请求并验证签名。然后它将尝试满足该请求。如果 DSA 'b'判断该请求应当被链接到另一个 DSA（DSA 'c'）来处理，则 DSA 'b'为 DSA 'c'加密该 DSP **ChainingArguments**。原始签名的 DAP 请求将保持，并与加密的 DSP **ChainingArguments** 一起传递。

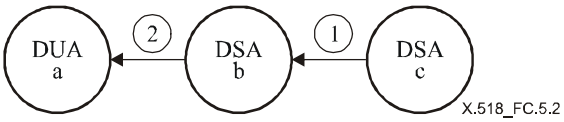


(1) DUA 'a'用户为DSA 'b'签名并加密DAP请求

(2) DSA 'b'解密此DAP请求并验证签名；在尝试本地满足此请求后，DSA 'b'判断该请求需要被链接到DSA 'c'。DSA 'b'发送原始签名的DAP请求（由DUA 'a'用户签名），并为DSA 'c'产生和加密DSP链接变量

C.5.2 对DAP结果的端到端签名并加密，对DSP的点到点签名并加密

如果 DUA 'a'用户希望从号码簿中接收签名并加密的结果，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **SIGNED-AND-ENCRYPTED**，或者如果该字段不存在，则 **ChainingArguments** 中的 **SecurityParameters.ProtectionRequest** 字段应当被设置为可以体现 DAP 绑定中的 **DIRQOP**。远端 DSA 应当有能力被配置为可以发送签名并加密后的链接操作。在本模型中，DSA 'c'系统能够满足请求，并且对 DAP 结果产生并执行一个端到端加密（为 DUA 'a'用户），同时对 DSP **ChainingResults** 产生并执行一个点到点加密。端到端的签名并加密能够由 DSA 'c'执行，因为它知道谁将是 DUA 'a'用户。对 DSP **ChainingResults** 进行点到点的签名并加密可以通过 DSA 'c'为 DSA 'b'签名并加密 DSP **ChainingResults** 来实现。DSA 'b'能够解密并验证 DSA 'c'对签名的 DSP **ChainingResults** 的签名，并且将签名并加密后的 DAP 结果传递到 DUA 'a'用户。DUA 'a'将负责对通过 DSA 'b'接收到的来自 DSA 'c'的 DAP 结果进行解密并对签名进行验证。

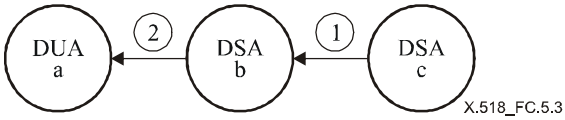


(1) DSA 'c'为DSA 'b'签名并加密DSP链接结果；这包括为DUA 'a' 用户签名并加密的DAP结果

(2) DSA 'b'解密来自DSA 'c'的DSP链接结果，并且为DUA 'a'转发签名并加密后的DAP结果

C.5.3 对DAP的端到端签名，对DSP和DAP结果的点到点加密

如果 DUA 'a'用户希望从号码簿中接收签名并加密后的结果，则 **SecurityParameters.ProtectionRequest** 字段应当被设置为 **SIGNED-AND-ENCRYPTED**，或者如果该字段不存在，则 **ChainingArguments** 中的 **SecurityParameters.ProtectionRequest** 字段应当被设置为可以体现 DAP 绑定中的 **DIRQOP**。远端 DSA（DSA 'c'）应当有能力被配置为可以发送签名并加密后的链接操作结果。在此模型中，DSA 'c'系统能够满足请求，则它会产生一个签名后的 DAP 结果，并且为了 DSA 'b'，对 DAP 结果和 DSP 的 **ChainingResults** 进行签名并加密。DSA 'b' 能够解密并验证 DSA 'c'对 DSP **ChainingResults** 的签名，并且为 DUA 'a'用户重新加密已签名（由 DSA 'c'签名）后的 DAP 结果。 DUA 'a'将负责对从 DSA 'b'接收到的 DAP 结果进行解密，并且对通过 DSA 'b'接收到的来自 DSA 'c' 的 DAP 结果的签名进行验证。



(1) DSA 'c'为DSA 'b'签名并加密DSP链接结果；这包括DAP结果

(2) DSA 'b'解密来自DSA 'c'的DSP链接结果（以及在DSP链接结果中接收到的DAP结果），并且将签名后的DAP结果转发到DUA 'a'

附件 D

分等级和非特定分等级操作绑定类型的规范

（本附件是本建议书 | 国际标准的组成部分）

本附件包含了本号码簿规范中引入的 ASN.1 信息对象类的定义，以 ASN.1 模块 **HierarchicalOperationalBindings** 的形式提供。

```
HierarchicalOperationalBindings
        {joint-iso-ITU-T ds(5) module(1) hierarchicalOperationalBindings(20) 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号码簿规范所包含的其它的ASN.1模块中，供其使用，
-- 也可以被其它应用所使用，这些应用将使用本模块中的定义来访问号码簿服务。
-- 其它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号码簿服务而进行的扩展和修改。

IMPORTS

-- 来自ITU-T X.501建议书 | ISO/IEC 9594-2

        directoryOperationalBindingTypes, directoryOSIProtocols, distributedOperations,
        informationFramework, opBindingManagement
            FROM UsefulDefinitions {joint-iso-ITU-T ds(5) module(1) usefulDefinitions(0) 5}

        Attribute, DistinguishedName, RelativeDistinguishedName
            FROM InformationFramework informationFramework

        OPERATIONAL-BINDING
            FROM OperationalBindingManagement opBindingManagement

-- 来自ITU-T X.518建议书 | ISO/IEC 9594-4

        MasterAndShadowAccessPoints
            FROM DistributedOperations distributedOperations

-- 来自ITU-T X.519建议书 | ISO/IEC 9594-5

        directorySystemAC
            FROM DirectoryOSIProtocols directoryOSIProtocols

        id-op-binding-hierarchical, id-op-binding-non-specific-hierarchical
            FROM DirectoryOperationalBindingTypes directoryOperationalBindingTypes                    ;

-- 类型 --

HierarchicalAgreement  ::=  SEQUENCE {
        rdn                     [0]   RelativeDistinguishedName,
        immediateSuperior       [1]   DistinguishedName }

SuperiorToSubordinate  ::=  SEQUENCE {
        contextPrefixInfo           [0]   DITcontext,
        entryInfo                   [1]   SET SIZE (1..MAX) OF Attribute OPTIONAL,
        immediateSuperiorInfo       [2]   SET SIZE (1..MAX) OF Attribute OPTIONAL }

DITcontext  ::=  SEQUENCE OF Vertex

Vertex  ::=  SEQUENCE {
```

```
        rdn             [0]  RelativeDistinguishedName,
        admPointInfo    [1]  SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries      [2]  SET SIZE (1..MAX) OF SubentryInfo OPTIONAL,
        accessPoints    [3]  MasterAndShadowAccessPoints OPTIONAL }

SubentryInfo ::= SEQUENCE {
        rdn  [0]  RelativeDistinguishedName,
        info [1]  SET OF Attribute }

SubordinateToSuperior ::= SEQUENCE {
        accessPoints [0]  MasterAndShadowAccessPoints OPTIONAL,
        alias        [1]  BOOLEAN DEFAULT FALSE,
        entryInfo    [2]  SET SIZE (1..MAX) OF Attribute OPTIONAL,
        subentries   [3]  SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }

SuperiorToSubordinateModification ::= SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})

NonSpecificHierarchicalAgreement ::= SEQUENCE {
        immediateSuperior     [1]  DistinguishedName }

NHOBSuperiorToSubordinate ::= SuperiorToSubordinate (
        WITH COMPONENTS { ..., entryInfo ABSENT})

NHOBSubordinateToSuperior ::= SEQUENCE {
        accessPoints [0]  MasterAndShadowAccessPoints OPTIONAL,
        subentries   [3]  SET SIZE (1..MAX) OF SubentryInfo OPTIONAL }

-- 操作绑定信息对象 --

hierarchicalOperationalBinding OPERATIONAL-BINDING ::= {
        AGREEMENT       HierarchicalAgreement
        APPLICATION CONTEXTS {
                        {directorySystemAC} }
        ASYMMETRIC
            ROLE-A {            -- 上级DSA
                ESTABLISHMENT-INITIATOR      TRUE
                ESTABLISHMENT-PARAMETER      SuperiorToSubordinate
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       SuperiorToSubordinateModification
                TERMINATION-INITIATOR        TRUE }
            ROLE-B {            -- 下级DSA
                ESTABLISHMENT-INITIATOR      TRUE
                ESTABLISHMENT-PARAMETER      SubordinateToSuperior
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       SubordinateToSuperior
                TERMINATION-INITIATOR        TRUE }
        ID                  id-op-binding-hierarchical }

nonSpecificHierarchicalOperationalBinding OPERATIONAL-BINDING ::= {
        AGREEMENT       NonSpecificHierarchicalAgreement
        APPLICATION CONTEXTS {
                        { directorySystemAC } }
        ASYMMETRIC
            ROLE-A {            -- 上级 DSA
                ESTABLISHMENT-PARAMETER NHOBSuperiorToSubordinate
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       NHOBSuperiorToSubordinate
                TERMINATION-INITIATOR        TRUE }
            ROLE-B {            -- 下级 DSA
                ESTABLISHMENT-INITIATOR      TRUE
                ESTABLISHMENT-PARAMETER NHOBSubordinateToSuperior
                MODIFICATION-INITIATOR       TRUE
                MODIFICATION-PARAMETER       NHOBSubordinateToSuperior
                TERMINATION-INITIATOR        TRUE }
        ID                  id-op-binding-non-specific-hierarchical }

END -- HierarchicalOperationalBindings
```

附件 E

知识维护示例

（本附件不是本建议书 | 国际标准的组成部分）

本附件以一个简单的示例举例说明了第 23 节定义的知识维护。在图 E.1 中，使用了下列符号来描述五个 DSA 的 DSA 信息树。

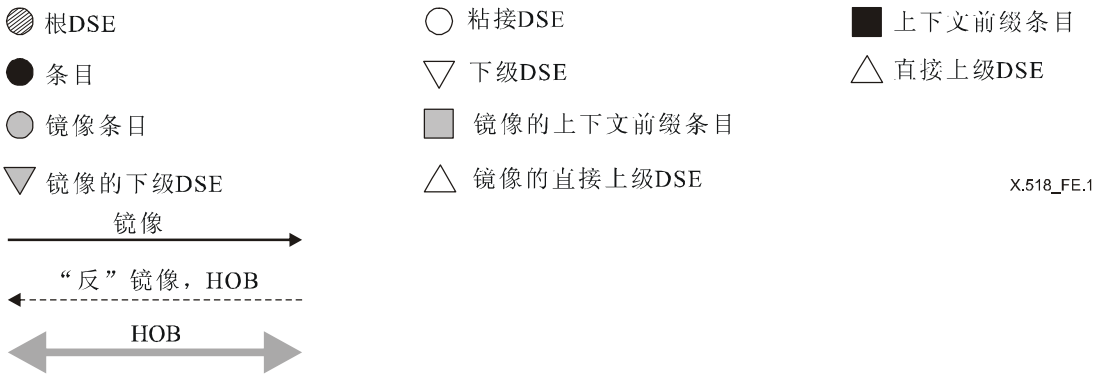


图 E.1－用于描述DSA信息树的符号

在图 E.2 中，DSA 1 是命名上下文{A}的属主，由两个条目{A}和{A, B}组成。DSA 1 拥有命名上下文{A, B, C}的一个下级引用，该引用通过一个与 DSA 3 之间的 HOB 来维护。DSA 1 是 DSA 2 的一个镜像提供者，向其提供命名上下文{A}的用户信息的拷贝，以及指向命名上下文{A, B, C}的下级引用的拷贝，该引用标识了 DSA 3，DSA 4 和 DSA5 的访问点，其中前者是下级命名上下文的属主。

DSA 3 是命名上下文{A, B, C}的属主。除了拥有该命名上下文的单独条目{A, B, C}外，DSA 3 还拥有命名上下文{A}的一个直接上级引用，该引用通过一个与 DSA 1 的 HOB 来维护。DSA 3 是 DSA 4 的一个镜像提供者，向其提供命名上下文{A, B, C}的用户信息的拷贝，以及指向命名上下文{A}的直接上级引用的拷贝，该引用标识了 DSA 1 和 DSA 2 的访问点，其中前者是上级命名上下文的属主。DSA 4 是 DSA 5 的一个（二次）镜像提供者，向其提供从 DSA 3 中接收到的信息的拷贝。

图 E.2 举例说明了用于表示和维护知识的 DSA 操作属性。

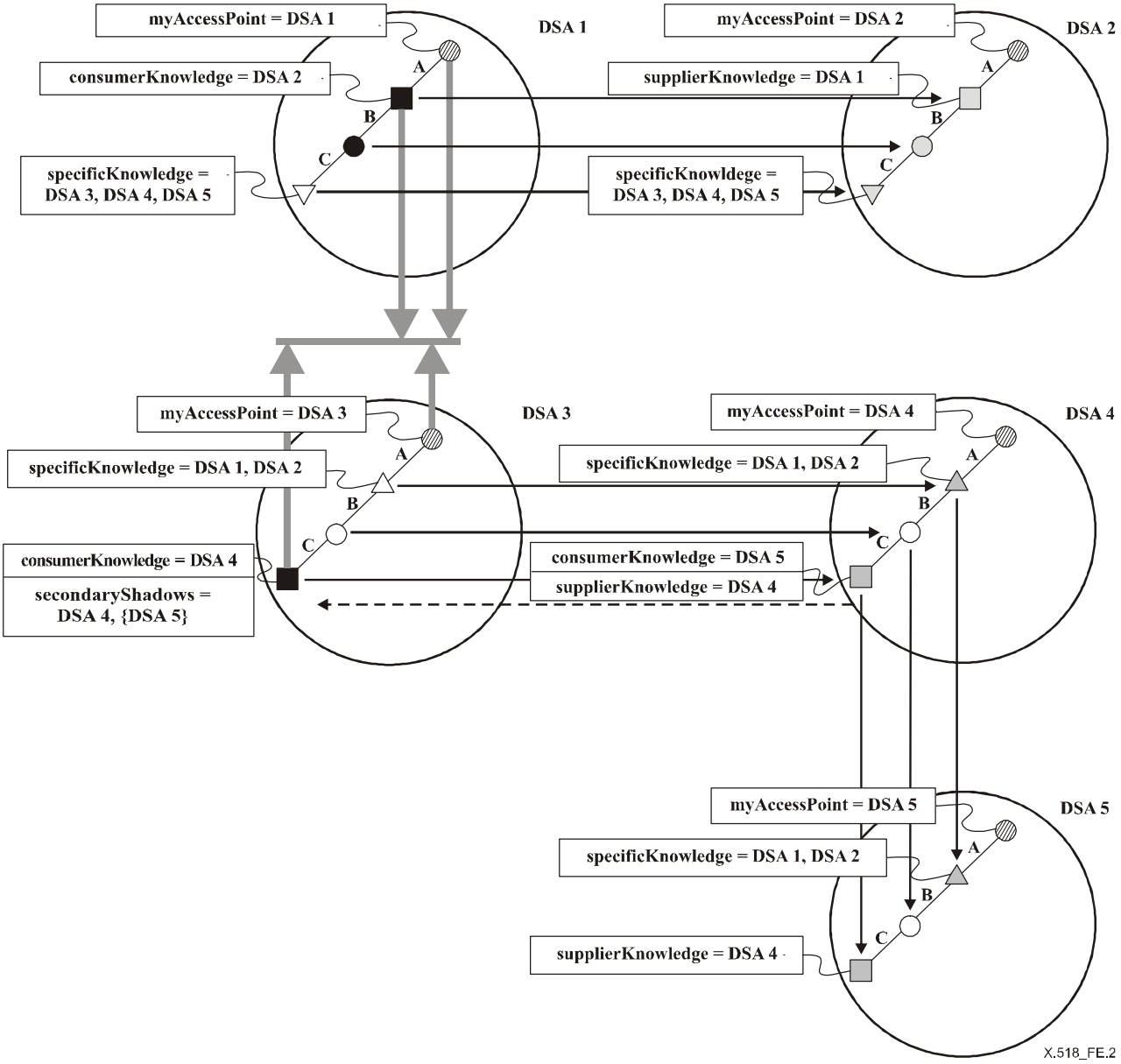


图 E.2－知识维护示例

DSA 1 使用它的 **myAccessPoint** 属性的值（与其根 DSE 相关联）与它的 **consumerKnowledge** 属性的公共可用值（与上下文前缀{A}相关联）来构造 **MasterAndShadowAccessPoints** 类型的一个值，用于与 DSA 3 的 HOB 交互。DSA 3 随之使用它的 **myAccessPoint** 属性的值（与其根 DSE 相关联）与它的 **consumerKnowledge** 属性和 **secondaryShadows** 属性的公共可用值（两个值都与上下文前缀{A, B, C}相关联）来构造 **MasterAndShadowAccessPoints** 类型的一个值，用于与 DSA 1 的 HOB 交互。两个 DSA，共同使用 DOP，来维护 DSA 1 所拥有的一个下级引用，以及 DSA 3 所拥有的一个直接上级引用。DSA 1 的下级引用，由一个与处于{A, B, C}中的 DSE 相关联的 **specificKnowledge** 属性来表示，是基于它从 DSA 3 中接收到的 **MasterAndShadowAccessPoints** 值构造的；DSA 3 的直接上级引用，由一个与处于{A}中的 DSE 相关联的 **specificKnowledge** 属性来表示，类似的，是基于它从 DSA 1 中接收到的 **MasterAndShadowAccessPoints** 值构造的。

DSA 1 和 DSA 2 在镜像操作绑定交互中使用它们的 **myAccessPoint** 值来维护 DSA 1 中的 **consumerKnowledge** 的值（该值标识了 DSA 2 的访问点）以及 DSA 2 中的 **supplierKnowledge** 的值（该值标识了 DSA 1 的访问点），两个属性都与上下文前缀{A}相关联。两个 DSA，共同使用 DOP，维护 DSA 1 所拥有的消费者引用，以及 DSA 2 所拥有的提供者引用。

DSA 2 在与 DSA 1 的 DISP 交互中，从 DSA 1 处接收到一个与上下文前缀{A, B, C}相关联的 **specificKnowledge** 属性的拷贝。这种交互用于维护 DSA 2 中的指向上下文前缀{A, B, C}的下级引用。

DSA 3 和 DSA 4（以及类似的 DSA 4 和 DSA 5）以一种同 DSA 1 和 DSA 2 之间的交互类似的方式，分别维护消费者引用和提供者引用。

DSA 4 在与 DSA 3 的 DISP 交互中，从 DSA 3 处接收到一个与上下文前缀{A4}相关联的 **specificKnowledge** 属性的拷贝。这种交互用于维护 DSA 4 中的指向上下文前缀{A}的直接上级引用。

DSA 4 向 DSA 3 传送在它的 **myAccessPoint** 和 **consumerKnowledge** 属性中所发生的任何变化（以及 **secondaryShadows** 属性，在本例中该属性值为空），方法是使用 DOP 的修改操作绑定操作。DSA 4 向 DSA 3 提供了 **SupplierAndConsumers** 的一个值，其中仅包含了 **consumerKnowledge** 属性的一些值，那些值标识了具有公共可用镜像的 DSA 的访问点；DSA 4 所提供的 **secondaryShadows** 属性的值，如果有的话，将被设计为都是公共可用的。（在本例中，DSA 5 被假设拥有处于{A, B, C}中的命名上下文的一个公共可用的拷贝。）DSA 3 使用该信息来维护它的 **secondaryShadows** 属性中的与上下文前缀{A, B, C}相关联的一个值。正如上述描述的那样，这个属性被用于与 DSA 1 的 DOP 交互，来维护 DSA 1 中的指向上下文前缀{A, B, C}的下级引用。

DSA 5 使用与 DSA 4 的 DISP 交互来维护它的指向上下文前缀{A}的直接上级引用，采用一种同 DSA 3 和 DSA 4 之间的交互相类似的方式。

附件 F

修正案和勘误表

（本附件不是本建议书 | 国际标准的组成部分）

本号码簿规范的本版本包括如下对前一版本的修正案草案内容，该草案经 ISO/IEC 投票批准：

— 为支持分页结果对 DSP 进行扩展的修正案 1。

— X.500 和 LDAP 间最大化融合的修正案 3。

本号码簿规范的本版本包括技术上的勘误，纠正了下述缺陷报告：307。

ITU-T 系列建议书

A系列	ITU-T工作的组织
D系列	一般资费原则
E系列	综合网络运行、电话业务、业务运行和人为因素
F系列	非话电信业务
G系列	传输系统和媒质、数字系统和网络
H系列	视听及多媒体系统
I系列	综合业务数字网
J系列	有线网络和电视、声音节目及其它多媒体信号的传输
K系列	干扰的防护
L系列	电缆和外部设备其它组件的结构、安装和保护
M系列	电信管理，包括TMN和网络维护
N系列	维护：国际声音节目和电视传输电路
O系列	测量设备的技术规范
P系列	电话传输质量、电话设施及本地线路网络
Q系列	交换和信令
R系列	电报传输
S系列	电报业务终端设备
T系列	远程信息处理业务的终端设备
U系列	电报交换
V系列	电话网上的数据通信
X系列	**数据网、开放系统通信和安全性**
Y系列	全球信息基础设施、互联网协议问题和下一代网络
Z系列	用于电信系统的语言和一般软件问题

瑞士印刷
2006年，日内瓦