МЕЖДУНАРОДНЫЙ СОЮЗ ЭЛЕКТРОСВЯЗИ

МСЭ-Т X.501

СЕКТОР СТАНДАРТИЗАЦИИ
ЭЛЕКТРОСВЯЗИ МСЭ

(08/2005)

СЕРИЯ X: СЕТИ ПЕРЕДАЧИ ДАННЫХ,
ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ И
БЕЗОПАСНОСТЬ

Справочник

Информационные технологии – Взаимосвязь открытых систем – Справочник: Модели

Рекомендация МСЭ-Т X.501

РЕКОМЕНДАЦИИ МСЭ-Т СЕРИИ X

СЕТИ ПЕРЕДАЧИ ДАННЫХ, ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ И БЕЗОПАСНОСТЬ

СЕТИ ПЕРЕДАЧИ ДАННЫХ ОБЩЕГО ПОЛЬЗОВАНИЯ	
Службы и услуги	X.1–X.19
Интерфейсы	X.20–X.49
Передача, сигнализация и коммутация	X.50–X.89
Сетевые аспекты	X.90–X.149
Техническое обслуживание	X.150–X.179
Административные предписания	X.180–X.199
ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ	
Модель и обозначение	X.200–X.209
Определения служб	X.210–X.219
Спецификации протоколов с установлением соединений	X.220–X.229
Спецификации протоколов без установления соединений	X.230–X.239
Проформы PICS	X.240–X.259
Идентификация протоколов	X.260–X.269
Протоколы обеспечения безопасности	X.270–X.279
Управляемые объекты уровня	X.280–X.289
Испытание на соответствие	X.290–X.299
ВЗАИМОДЕЙСТВИЕ МЕЖДУ СЕТЯМИ	
Общие положения	X.300–X.349
Спутниковые системы передачи данных	X.350–X.369
Сети, основанные на протоколе Интернет	X.370–X.379
СИСТЕМЫ ОБРАБОТКИ СООБЩЕНИЙ	X.400–X.499
СПРАВОЧНИК	**X.500–X.599**
ОРГАНИЗАЦИЯ СЕТИ ВОС И СИСТЕМНЫЕ АСПЕКТЫ	
Организация сети	X.600–X.629
Эффективность	X.630–X.639
Качество обслуживания	X.640–X.649
Наименование, адресация и регистрация	X.650–X.679
Абстрактно-синтаксическая нотация 1 (ASN.1)	X.680–X.699
УПРАВЛЕНИЕ В ВОС	
Структура и архитектура управления системами	X.700–X.709
Служба и протокол связи для общего управления	X.710–X.719
Структура управляющей информации	X.720–X.729
Функции общего управления и функции ODMA	X.730–X.799
БЕЗОПАСНОСТЬ	X.800–X.849
ПРИЛОЖЕНИЯ ВОС	
Фиксация, параллельность и восстановление	X.850–X.859
Обработка транзакций	X.860–X.879
Удаленные операции	X.880–X.889
Общие приложения ASN.1	X.890–X.899
ОТКРЫТАЯ РАСПРЕДЕЛЕННАЯ ОБРАБОТКА	X.900–X.999
БЕЗОПАСНОСТЬ ЭЛЕКТРОСВЯЗИ	X.1000–

Для получения более подробной информации просьба обращаться к перечню Рекомендаций МСЭ-Т.

МЕЖДУНАРОДНЫЙ СТАНДАРТ ИСО/МЭК 9594-2
РЕКОМЕНДАЦИЯ МСЭ-Т X.501

Информационные технологии – Взаимосвязь открытых систем – Справочник: Модели

Резюме

В настоящей Рекомендации | Международном стандарте представлен ряд различных моделей Справочника в качестве основы для других Рекомендаций МСЭ-Т серии X.500. Этими моделями являются общая (функциональная) модель, модель административных полномочий, обобщенные информационные модели Справочника, отражающие взгляды пользователя Справочника и административного пользователя на информацию Справочника, обобщенные модели системного агента Справочника (DSA) и информационные модели DSA, а также функциональная основа и модель системы безопасности.

Источник

Рекомендация МСЭ-Т X.501 утверждена 29 августа 2005 года 17-й Исследовательской комиссией МСЭ-Т (2005–2008 гг.) в соответствии с процедурой, изложенной в Рекомендации МСЭ-Т A.8. Идентичный текст также опубликован как стандарт ИСО/МЭК 9594-2.

ПРЕДИСЛОВИЕ

Международный союз электросвязи (МСЭ) является специализированным учреждением Организации Объединенных Наций в области электросвязи. Сектор стандартизации электросвязи МСЭ (МСЭ-Т) – постоянный орган МСЭ. МСЭ-Т отвечает за изучение технических, эксплуатационных и тарифных вопросов и за выпуск Рекомендаций по ним с целью стандартизации электросвязи на всемирной основе.

На Всемирной ассамблее по стандартизации электросвязи (ВАСЭ), которая проводится каждые четыре года, определяются темы для изучения Исследовательскими комиссиями МСЭ-Т, которые, в свою очередь, вырабатывают Рекомендации по этим темам.

Утверждение Рекомендаций МСЭ-Т осуществляется в соответствии с процедурой, изложенной в Резолюции 1 ВАСЭ.

В некоторых областях информационных технологий, которые входят в компетенцию МСЭ-Т, необходимые стандарты разрабатываются на основе сотрудничества с ИСО и МЭК.

ПРИМЕЧАНИЕ

В настоящей Рекомендации термин "администрация" используется для краткости и обозначает как администрацию электросвязи, так и признанную эксплуатационную организацию.

Соблюдение положений данной Рекомендации носит добровольный характер. Однако в Рекомендации могут содержаться определенные обязательные положения (например, для обеспечения возможности взаимодействия или применимости), и соблюдение положений данной Рекомендации достигается в случае выполнения всех этих обязательных положений. Для выражения необходимости выполнения требований используется синтаксис долженствования и соответствующие слова (такие, как "должен" и т. п.), а также их отрицательные эквиваленты. Использование этих слов не предполагает, что соблюдение положений данной Рекомендации является обязательным для какой-либо из сторон.

ПРАВА ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

МСЭ обращает внимание на вероятность того, что практическое применение или реализация этой Рекомендации может включать использование заявленного права интеллектуальной собственности. МСЭ не занимает какую бы то ни было позицию относительно подтверждения, обоснованности или применимости заявленных прав интеллектуальной собственности, независимо от того, отстаиваются ли они членами МСЭ или другими сторонами вне процесса подготовки Рекомендации.

На момент утверждения настоящей Рекомендации МСЭ не получил извещение об интеллектуальной собственности, защищенной патентами, которые могут потребоваться для выполнения этой Рекомендации. Однако те, кто будет применять Рекомендацию, должны иметь в виду, что это может не отражать самую последнюю информацию, и поэтому им настоятельно рекомендуется обращаться к патентной базе данных БСЭ по адресу: http://www.itu.int/ITU-T/ipr/.

СОДЕРЖАНИЕ

Введение

Настоящая Рекомендация | Международный стандарт вместе с другими Рекомендациями | Международными стандартами разработана для упрощения взаимосвязи систем обработки информации в целях обеспечения справочных служб. Совокупность таких систем вместе с хранимой ими справочной информацией можно рассматривать как единое целое, называемое *Справочником*. Хранимая в Справочнике информация, называемая в совокупности информационной базой Справочника (DIB), обычно используется для содействия обеспечению связи между объектами, с объектами или относительно объектов, примерами которых могут служить объекты прикладного уровня, люди, терминалы и списки рассылки.

Справочник играет существенную роль во взаимосвязи открытых систем, назначение которой заключается, при минимуме технических соглашениях, не входящих в сами стандарты взаимосвязи, в обеспечении взаимосвязи систем обработки информации:

- поставляемых разными производителями;
- находящихся под различным управлением;
- различных уровней сложности; и
- разных поколений.

В настоящей Рекомендации | Международном стандарте представлен ряд различных моделей Справочника в качестве основы для других Рекомендаций МСЭ-Т серии X.500 | частей ИСО/МЭК 9594. Этими моделями являются общая (функциональная) модель, модель административных полномочий, обобщенные информационные модели Справочника, отражающие взгляды пользователя Справочника и административного пользователя на информацию Справочника, обобщенные модели системного агента Справочника (DSA) и информационные модели DSA, а также функциональная основа и модель безопасности.

Обобщенные информационные модели Справочника описывают, например, как группируется информация об объектах для создания статей об этих объектах, и как на основе этой информации формируются имена объектов.

Обобщенные модели DSA и информационные модели DSA, а также функциональная основа обеспечивают поддержку распределения Справочника.

Настоящая Рекомендация | Международный стандарт предусматривает специализацию обобщенных информационных моделей Справочника для поддержки административного управления Схемой Справочника.

Настоящая Рекомендация | Международный стандарт обеспечивает фундаментальные основы для определения другими группами по разработке стандартов и отраслевыми форумами отраслевых профилей. Многие из функций, которые определены в этих основах как необязательные, для конкретных условий могут с помощью профилей определяться как обязательные. Настоящее пятое издание не заменяет четвертое издание данной Рекомендации | Международного стандарта, но является результатом его пересмотра и усовершенствования в техническом аспекте. Реализации могут по-прежнему соответствовать четвертому изданию. Вместе с тем, с некоторого момента четвертое издание поддерживаться не будет (т. е. более не будут устраняться сообщаемые дефекты). Рекомендуется обеспечить в возможно краткие сроки соответствие реализаций настоящему пятому изданию.

В данном пятом издании определяются версии 1 и 2 протоколов Справочника.

В первом и втором изданиях определялась только версия 1. Бóльшая часть служб и протоколов, определенных в настоящем издании, построены для работы согласно версии 1. Однако некоторые усовершенствованные службы и протоколы, например подписанные параметры ошибки, не будут функционировать, если не все объекты Справочника, участвующие в конкретной операции, согласованы с версией 2. Независимо от версии, с которой обеспечено согласование, различия между службами и между протоколами, определенными в пятом издании, за исключением специально предназначенных для версии 2, улаживаются при использовании правил расширяемости, определенных в Рекомендации МСЭ-Т X.519 | ИСО/МЭК 9594-5.

Приложение A, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит резюме использования идентификаторов объектов на языке ASN.1 в Рекомендациях МСЭ-Т серии X.500 | частях ИСО/МЭК 9594.

Приложение B, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит модуль ASN.1, включающий все определения, связанные с информационной инфраструктурой.

Приложение C, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит схему административного управления подсхемой на языке ASN.1.

Приложение D, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит модуль ASN.1 для административного управления службой.

Приложение E, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит модуль ASN.1 для базового управления доступом.

Приложение F, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит модуль ASN.1, включающие все определения, связанные с типами операционных атрибутов DSA.

Приложение G, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит модуль ASN.1, включающий все определения, связанные с операциями управления рабочим связыванием.

Приложение H, которое является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит модуль ASN.1, включающий все определения, связанные с усиленной безопасностью.

Приложение I, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит сводку математической терминологии, связанной со структурой деревьев.

Приложение J, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, описывает некоторые критерии, которые могут быть учтены при составлении имен.

Приложение K, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит некоторые примеры различных аспектов Схемы.

Приложение L, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит обзор семантики, связанной с разрешениями базового управления доступом.

Приложение M, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит подробный пример использования базового управления доступом.

Приложение N, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, описывает некоторые комбинации зависящих от DSA статей.

Приложение O, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит основу для моделирования знаний.

Приложение P, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, описывает критерии того, может ли имя быть альтернативным выделенным именем или первичным выделенным именем, может ли оно содержать альтернативные значения и может ли оно включать контекстную информацию.

Приложение Q, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, описывает концепцию подфильтров.

Приложение R, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит рекомендации и примеры именования членов семейства.

Приложение S, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, является введением в концепции и принципы именования.

Приложение T, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит перечень в алфавитном порядке терминов, определенных в настоящей Рекомендации | Международном стандарте.

Приложение U, которое не является неотъемлемой частью настоящей Рекомендации | Международного стандарта, содержит перечень поправок и сообщений о дефектах, которые были включены для составления данного издания Рекомендации | Международного стандарта.

МЕЖДУНАРОДНЫЙ СТАНДАРТ
РЕКОМЕНДАЦИЯ МСЭ-Т

Информационные технологии – Взаимосвязь открытых систем – Справочник: Модели

РАЗДЕЛ 1 – ОБЩИЕ ПОЛОЖЕНИЯ

1 Сфера применения

Определенные в настоящей Рекомендации | Международном стандарте модели составляют концептуальную и терминологическую основу для других Рекомендаций МСЭ-Т серии X.500 | частей ИСО/МЭК 9594, определяющих различные аспекты Справочника.

Функциональные модели и модели административных полномочий определяют возможные способы – функциональные и административные – распределения Справочника. Также представлены обобщенные модели DSA и информационные модели DSA и функциональная основа, необходимые для поддержки распределения Справочника.

Обобщенные модели информации Справочника описывают логическую структуру DIB с позиции пользователя Справочника и административного пользователя. В этих моделях не виден тот факт, что Справочник носит скорее распределенный, чем централизованный, характер.

Настоящая Рекомендация | Международный стандарт предусматривает специализацию обобщенных информационных моделей Справочника для поддержки административного управления схемой Справочника.

Другие Рекомендации МСЭ-Т серии X.500 | части ИСО/МЭК 9594 используют определенные в настоящей Рекомендации | Международном стандарте понятия для определения специализации обобщенных информационных моделей и моделей DSA в целях обеспечения конкретной информации, моделей DSA и функциональных моделей, поддерживающих конкретные возможности Справочника (например, репликацию):

a) служба, обеспечиваемая Справочником, описывается (в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3) на языке понятий информационной инфраструктуры: это позволяет обеспечивать определенную степень независимой предоставляемой службы от физического распределения DIB;

b) распределенное функционирование Справочника определяется (в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4) так, чтобы предоставлять эту службу и, следовательно, сохранять эту логическую информационную структуру в условиях весьма значительной фактической распределенности DIB;

c) описываются (в Рек. МСЭ-Т X.525 | ИСО/МЭК 9594-9) возможности репликации, обеспечиваемые составными частями Справочника для повышения общей производительности Справочника.

Модель обеспечения безопасности образует основу для спецификации механизмов управления доступом. Она предоставляет механизм опознания действующей схемы управления доступом в конкретной части DIT, а также определяет три гибкие специальные схемы управления доступом, применимые для широкого диапазона приложений и способов использования. Модель обеспечения безопасности создает также основу для поддержки конфиденциальности и целостности данных при выполнении операций справочника с использованием таких механизмов, как кодирование и цифровые подписи Это обусловливает использование данной основы для аутентификации, определенной в Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8, а также обобщенных инструментов обеспечения безопасности верхних уровней, определенных в Рек. МСЭ-Т X.830 | ИСО/МЭК 11586-1.

Модели DSA образуют основу для спецификации функционирования компонентов Справочника, а именно:

a) функциональная модель Справочника описывает, каким образом Справочник проявляется как множество, состоящее из одного или более компонентов, где каждый компонент является DSA;

b) модель распределения Справочника описывает принципы, согласно которым статьи DIB и статьи-копии могут распределяться между DSA;

c) информационная модель DSA описывает структуру пользователя Справочника и функциональной информации, содержащейся в DSA;

d) функциональная инфраструктура DSA описывает средства, с помощью которых формируется структура определения специальных форм сотрудничества между DSA, направленного на достижение конкретных целей (например, для теневого копирования).

2 Нормативные справочные документы

Указанные ниже Рекомендации и международные стандарты содержат положения, которые путем ссылки на них в данном тексте составляют положения настоящей Рекомендации. На момент публикации указанные издания были действующими. Все Рекомендации и стандарты могут подвергаться пересмотру; поэтому сторонам соглашений, основанных на данной Рекомендации | Международном стандарте, предлагается изучить возможность применения последнего издания Рекомендаций и стандартов, перечисленных ниже. Члены МЭК и ИСО ведут регистры действующих в настоящее время международных стандартов. Бюро стандартизации электросвязи МСЭ ведет список действующих в настоящее время Рекомендаций МСЭ-Т.

2.1 Идентичные Рекомендации | Международные стандарты

- ITU-T Recommendation X.200 (1994) | ISO/IEC 7498-1:1994, *Information technology – Open Systems Interconnection – Basic Reference Model: The basic model.*
- Рекомендация МСЭ-Т X.500 (2005 г.) | ИСО/МЭК 9594-1:2005, *Информационные технологии – Взаимосвязь открытых систем – Справочник: Обзор понятий, моделей и услуг*.
- ITU-T Recommendation X.509 (2005) | ISO/IEC 9594-8:2005, *Information technology – Open Systems Interconnection – The Directory: Public-key and attribute certificate frameworks.*
- Рекомендация МСЭ-Т X.511 (2005 г.) | ИСО/МЭК 9594-3:2005, *Информационные технологии – Взаимосвязь открытых систем – Справочник: Определение абстрактной службы.*
- Рекомендация МСЭ-Т X.518 (2005 г.) | ИСО/МЭК 9594-4:2005, *Информационные технологии – Взаимосвязь открытых систем – Справочник: Процедуры распределенных операций.*
- ITU-T Recommendation X.519 (2005) | ISO/IEC 9594-5:2005, *Information technology – Open Systems Interconnection – The Directory: Protocol specifications.*
- Рекомендация МСЭ-Т X.520 (2005 г.) | ИСО/МЭК 9594-6:2005, *Информационные технологии – Взаимосвязь открытых систем – Справочник: Избранные типы атрибутов.*
- Рекомендация МСЭ-Т X.521 (2005 г.) | ИСО/МЭК 9594-7:2005, *Информационные технологии – Взаимосвязь открытых систем – Справочник: Избранные объектные классы.*
- Рекомендация МСЭ-Т X.525 (2005 г.) | ИСО/МЭК 9594-9:2005, *Информационные технологии – Взаимосвязь открытых систем – Справочник: Копирование.*
- 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.660 (2004) | ISO/IEC 9834-1:2005, *Information technology – Open Systems Interconnection – Procedures for the operation of OSI Registration Authorities: General procedures and top arcs of the ASN.1 Object Identifier tree.*
- 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.*
- ITU-T Recommendation X.803 (1994) | ISO/IEC 10745:1995, *Information technology – Open Systems Interconnection – Upper layers security model*.
- ITU-T Recommendation X.811 (1995) | ISO/IEC 10181-2:1996, *Information technology – Open Systems Interconnection – Security frameworks for open systems: Authentication framework*.
- ITU-T Recommendation X.812 (1995) | ISO/IEC 10181-3:1996, *Information technology – Open Systems Interconnection – Security frameworks for open systems – Access control framework*.

– ITU-T Recommendation X.813 (1996) | ISO/IEC 10181-4:1997, *Information technology – Open Systems Interconnection – Security frameworks for open systems – Non-repudiation framework*.

2.2 Парные Рекомендации | Международные стандарты, эквивалентные по техническому содержанию

– CCITT Recommendation X.800 (1991), *Security architecture for Open Systems Interconnection for CCITT applications*.

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

2.3 Другие справочные документы

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

3 Определения

Для целей настоящей Рекомендации | Международного стандарта используются следующие определения.

3.1 Определения передачи данных

Следующие термины определены в Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5:

a) *прикладной контекст;*

b) *объект прикладного уровня;*

c) *прикладной процесс.*

3.2 Определения основных терминов по Справочнику

Следующие термины определены в Рек. МСЭ-Т X.500 | ИСО/МЭК 9594-1:

a) *Справочник*;

b) *протокол доступа к Справочнику*;

c) *информационная база Справочника*;

d) *протокол управления рабочим связыванием Справочника*;

e) *системный протокол Справочника*;

f) *пользователь (Справочника)*.

3.3 Определения терминов по распределенным операциям

Следующие термины определены в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4:

a) *точка доступа*;

b) *иерархическое рабочее связывание*;

c) *разрешение имен*;

d) *неспецифическое иерархическое рабочее связывание*;

e) *релевантное иерархическое рабочее связывание*.

3.4 Определения репликации

Следующие термины определены в Рек. МСЭ-Т X.525 | ИСО/МЭК 9594-9:

a) *кэш-копия*;

b) *ссылка на потребителя*;

c) *статья-копия*;

d) *главный DSA*;

e) *первичное теневое копирование*;

f) *дублируемая область*;

g) *репликация*;

h) *вторичное теневое копирование*;

i) *теневой потребитель*;

j) *теневой поставщик*;

k) *теневая зависящая от DSA статья*;

l) *теневое копирование*;

m) *ссылка на поставщика.*

Определения терминов, приводимые в настоящей Рекомендации | Международном стандарте, включены в начало отдельных пунктов, если применимо. Для упрощения поиска в Приложении T представлен индекс этих терминов.

4 Сокращения

Для целей настоящей Рекомендации | Международного стандарта используются следующие сокращения.

ACDF	Функция принятия решения управления доступом
ACI	Информация по управлению доступом
ACIA	Внутренняя область управления доступом
ACSA	Специальная область управления доступом
ADDMD	Домен управления Справочником администрации
ASN.1	Абстрактно-синтаксическая нотация 1
AVA	Утверждение о значении атрибута
BER	Основные правила кодирования (ASN.1)
DACD	Домен управления доступом к Справочнику
DAP	Протокол доступа к Справочнику
DIB	Информационная база Справочника
DISP	Протокол теневого копирования информации Справочника
DIT	Информационное дерево Справочника
DMD	Домен управления Справочником
DMO	Организация, управляющая доменом
DOP	Протокол управления рабочим связыванием Справочника
DSA	Системный агент Справочника
DSE	Зависящая от DSA статья
DSP	Системный протокол Справочника
DUA	Агент пользователя Справочника
HOB	Иерархическое рабочее связывание
LDAP	Упрощенный протокол доступа к Справочнику
NHOB	Неспецифическое иерархическое рабочее связывание
NSSR	Неспецифическая подчиненная ссылка
PRDMD	Частный домен управления Справочником
RDN	Относительное выделенное имя
RHOB	Релевантное иерархическое рабочее связывание (HOB или NHOB, в зависимости от случая)
SDSE	DSE, имеющая теневую копию

5 Соглашения

За небольшими исключениями, эта спецификация Справочника была подготовлена в соответствии с "*Правилами представления общего текста МСЭ-Т | ИСО/МЭК*", ноябрь 2001 г.

Термин "спецификация Справочника" (как и "эта спецификация Справочника") означает Рекомендацию МСЭ-Т X.501 | ИСО/МЭК 9594-2. Термин "спецификация Справочника" должен означать Рекомендации серии X.500 и все части стандарта ИСО/МЭК 9594.

В данной спецификации Справочника используется термин *системы первого издания* для указания на системы, соответствующие первому изданию спецификаций Справочника, т. е. изданию 1988 года Рекомендаций МККТТ серии X.500 и изданию стандарта ИСО/МЭК 9594:1990. В этой спецификации Справочника используется термин *системы второго издания* для указания на системы, соответствующие второму изданию спецификаций Справочника, т. е. изданию 1993 года Рекомендаций МСЭ-Т серии X.500 и изданию стандарта ИСО/МЭК 9594:1995. В этой спецификации Справочника используется термин *системы третьего издания* для указания на системы, соответствующие третьему изданию спецификаций Справочника, т. е. изданию 1997 года Рекомендаций МСЭ-Т серии X.500 и изданию стандарта ИСО/МЭК 9594:1998. В этой спецификации Справочника используется термин *системы четвертого издания* для указания на системы, соответствующие четвертому изданию спецификаций Справочника, т. е. изданиям 2001 года Рекомендаций МСЭ-Т X.500, X.501, X.511, X.518, X.519, X.520, X.521, X.525 и X.530, изданию 2000 года Рекомендаций МСЭ-Т X.509 и частям 1–10 издания стандарта ИСО/МЭК 9594:2001.

В настоящей спецификации Справочника используется термин *системы пятого издания* для ссылки на системы, соответствующие пятому изданию спецификаций Справочника, т. е. изданиям 2005 года Рекомендаций МСЭ-Т X.500, X.501, X.509, X.511, X.518, X.519, X.520, X.521, X.525 и X.530 и частей 1–10 издания стандарта ИСО/МЭК 9594:2005.

В данной спецификации Справочника нотация на языке ASN.1 дается полужирным шрифтом Helvetica. Когда типы и значения ASN.1 приводятся в обычном тексте, они выделяются полужирным шрифтом Helvetica. Названия процедур, упоминаемых при определении семантики обработки, выделяются в тексте полужирным шрифтом Times. Разрешения на управление доступом предоставляются курсивом шрифта Times.

Если элементы в списке пронумерованы (либо против них указаны "–" или буквы), то эти пункты должны рассматриваться как этапы в процедуре.

РАЗДЕЛ 2 – ОБЗОР МОДЕЛЕЙ СПРАВОЧНИКА

6 Модели Справочника

6.1 Определения

Для целей настоящей Рекомендации | Международного стандарта используются следующие определения:

6.1.1 административный орган: Агент какой-либо организации, управляющей доменом, который осуществляет административное управление Справочником в различных аспектах. Термин *административные полномочия* (строчными буквами) означает полномочия, которыми организация, управляющая доменом, наделяет административный орган для осуществления стратегии.

6.1.2 домен управления справочником администрации (ADDMD): DMD, которым управляет какая-либо администрация.

ПРИМЕЧАНИЕ. – Термин *Администрация* означает государственную администрацию электросвязи или иную организацию, предоставляющую услуги электросвязи общего пользования.

6.1.3 административная и операционная информация справочника: Информация, используемая Справочником для административных и операционных целей.

6.1.4 домен DIT: Часть глобального DIT, которая содержится агентами DSA, образующими DMD.

6.1.5 домен управления справочником (DMD): Один или более DSA и ни одного или несколько DUA, управляемых одной организацией.

6.1.6 организация, управляющая доменом: Организация, которая управляет каким-либо DMD (и соответствующим доменом DIT).

6.1.7 пользовательская информация справочника: Информация, представляющая интерес для пользователей и их приложений.

6.1.8 системный агент справочника (DSA): Прикладной процесс OSI, являющийся частью Справочника.

6.1.9 пользователь (справочника): Конечный пользователь Справочника, т. е. объект или физическое лицо, осуществляющее доступ к Справочнику.

6.1.10 агент пользователя справочника (DUA): Прикладной процесс OSI, представляющий пользователя при доступе к Справочнику.

ПРИМЕЧАНИЕ. – DUA может также предлагать диапазон местных средств и услуг в помощь пользователям при составлении запросов и интерпретации ответов.

6.1.11 клиент LDAP: Прикладной процесс, представляющий пользователя при доступе к Справочнику по упрощенному протоколу доступа к Справочнику (LDAP).

6.1.12 запросчик LDAP: DSA, который способен составлять запросы по упрощенному протоколу доступа к Справочнику (LDAP) и способен понимать и обрабатывать ответы по LDAP.

6.1.13 отвечающий LDAP: DSA, который способен понимать запросы, составленные по упрощенному протоколу доступа к Справочнику (LDAP), и отвечать на них по LDAP.

6.1.14 сервер LDAP: Прикладной процесс, являющийся частью Справочника, которые содержит часть DIB и который отвечает на запросы по упрощенному протоколу доступа к Справочнику (LDAP).

6.1.15 частный домен управления справочником (PRDMD): DMD, которым управляет какая-либо организация, не являющаяся администрацией.

6.2 Справочник и пользователи Справочника

Справочник – это хранилище информации. Такое хранилище называют информационной базой Справочника (DIB). содержанием предоставляемых пользователям услуг, связанных со Справочником, являются различные виды доступа к этой информации.

Услуги, предоставляемые Справочником, определяются в Рек. МСЭ-Т X.511 | ИМО/МЭК 9594-3.

Пользователь Справочника (например, физическое лицо или прикладной процесс) получает услуги Справочника путем доступа к Справочнику. Говоря точнее, реально доступ к Справочнику получает *агент пользователя Справочника (DUA)* или *клиент упрощенного протокола доступа к Справочнику (LDAP)*, которые взаимодействуют с ним для получения услуги от имени конкретного пользователя. Справочник предоставляет одну или более *точку доступа*, в которых возможно получение доступа. Это показано на рисунке 1.

DUA объявляется как прикладной процесс. В любой реализации связи каждый DUA представляет только одного пользователя справочника.

Справочник объявляется как один или более прикладных процессов, называемых *системными агентами Справочника (DSA)* и/или *серверами LDAP*, каждый из которых обеспечивает ни одной, одну или более точек доступа. Более подробное описание DSA см. в п. 21.2.

ПРИМЕЧАНИЕ 1. – Некоторые открытые системы могут предоставлять реальным пользователям (прикладным процессам, физическим лицам и т. д.) централизованную функцию DUA по поиску информации. Для Справочника это является прозрачным процессом.

ПРИМЕЧАНИЕ 2. – Функции DUA и какой-либо DSA могут существовать в рамках той же открытой системы, а вопрос о том, являются ли один или более DUA видимыми в среде OSI в качестве объектов прикладного процесса, решается в процессе реализации.

ПРИМЕЧАНИЕ 3. – DUA может обладать определенными свойствами и структурой, выходящими за пределы области применения спецификаций Справочника. Например, DUA, представляющий пользователя справочника – человека, может предлагать диапазон местных средств и услуг в помощь своему пользователю при составлении запросов и интерпретации ответов.

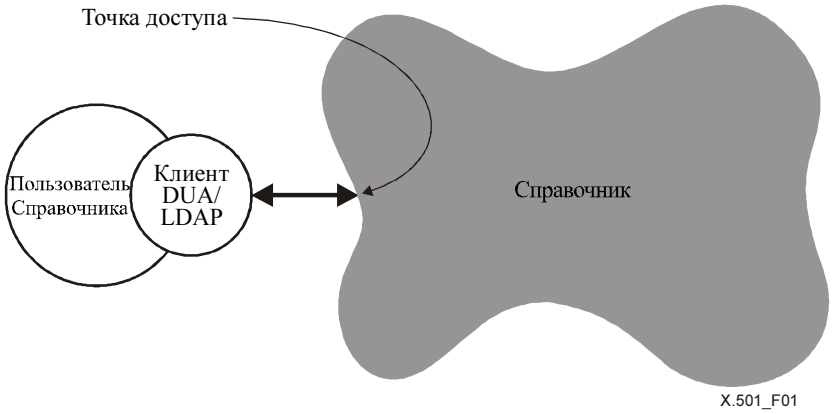


Рисунок 1 – Доступ к Справочнику

6.3 Справочник и информационные модели DSA

6.3.1 Родовые модели

Информация Справочника может быть классифицирована как либо:

- пользовательская информация, помещенная в Справочник самими пользователями или от их имени и управляемая, соответственно, самими пользователями или от их имени; модель такой информации представлена в разделе 3; либо
- административная и операционная информация, содержащаяся в Справочнике для удовлетворения различных административных и функциональных требований. В разделе 5 представлена модель такой информации. Кроме того, в разделе 5 содержится описание взаимосвязи между пользовательской, административной и операционной информационными моделями.

Эти модели, представляющие DIB с разных точек зрения, называются родовыми информационными моделями Справочника.

Информационные модели Справочника описывают, как Справочник представляет информацию в целом. Из модели следует, что состав Справочника образует набор потенциально взаимодействующих DSA. Информационная модель DSA, с другой стороны, относится конкретно к DSA и к информации, которая должна ими храниться, с тем чтобы все DSA, входящие в состав Справочника, могли совместно реализовать Информационную модель Справочника. Информационная модель DSA описывается в пп. 22–23.

Информационная модель DSA является родовой моделью, описывающей информацию, которая хранится DSA, а также взаимосвязь между этой информацией и DIB и DIT.

Часть информации, представляемой информационной моделью DSA, но не вся информация, доступна через абстрактную службу Справочника. Следовательно, административное управление всей информацией, описанной в данных спецификациях Справочника, через абстрактную службу Справочника невозможно. Предполагается, что административное управление информацией DSA изначально будет являться вопросом местной компетенции, но, в конечном счете, для обеспечения доступа ко всей информации, описанной в информационной модели DSA, будет использоваться некая общая служба системного управления.

6.3.2 Специальные информационные модели

После разработки родовых моделей для Справочника в целом и для его компонентов необходимы видовые информационные модели для стандартизации определенных аспектов функционирования Справочника и его компонентов.

Родовые информационные модели Справочника образуют основу для последующих видовых информационных моделей:

- информационная модель управления доступом;
- информационная модель подсхемы;
- информационная модель совокупного атрибута.

В свою очередь, родовая информационная модель DSA образует основу для следующих видовых информационных моделей:

- модель для присущего DSA знания о распространении;
- модель для присущего DSA знания о тиражировании.

6.4 Модель административного органа Справочника

Домен управления Справочником (DMD) – это совокупность одного или более DSA и нуля или более DUA, управляемых одной организацией.

Эта часть глобального DIT, которую содержит (образуемая агентами DSA) DMD, называется *доменом DIT*. Между доменами DMD и доменами DIT существует взаимно-однозначное соответствие. Термин "DMD" используется, когда речь идет об управлении функциональными компонентами Справочника. Термин "домен DIT" используется, когда речь идет об управлении информацией Справочника. Двумя важнейшими принципами, обусловливающими эту терминологию, являются следующие:

- Домен DIT состоит из одного или более независимых поддеревьев DIT (см. п. 11.5). Домен DIT не содержит корня глобального DIT.
- Термин "DMD" может также использоваться в качестве общего термина, если имеются в виду оба аспекта управления.

Организация, которая управляет каким-либо DMD (и соответствующим доменом DIT) называется *организацией, управляющей доменом (DMO)*.

ПРИМЕЧАНИЕ 1. – DMO может быть администрацией (т. е. государственной администрацией электросвязи или иной организацией, предоставляющей услуги электросвязи общего пользования), в каковом случае управляемый DMD называется DMD администрации (ADDMD); в противном случае это частный DMD (PRDMD). Следует отметить, что обеспечение поддержки для частных справочных систем членами МСЭ-Т является вопросом национальных регламентарных норм. Таким образом, описываемые технические возможности могут предоставляться или не предоставляться администрацией, которая предоставляет услуги справочника. Внутреннее функционирование и конфигурация частных DMD не входят в область применения рассматриваемых спецификаций Справочника.

На рисунке 2 показана взаимосвязь между DMO, DMD и доменом DIT.

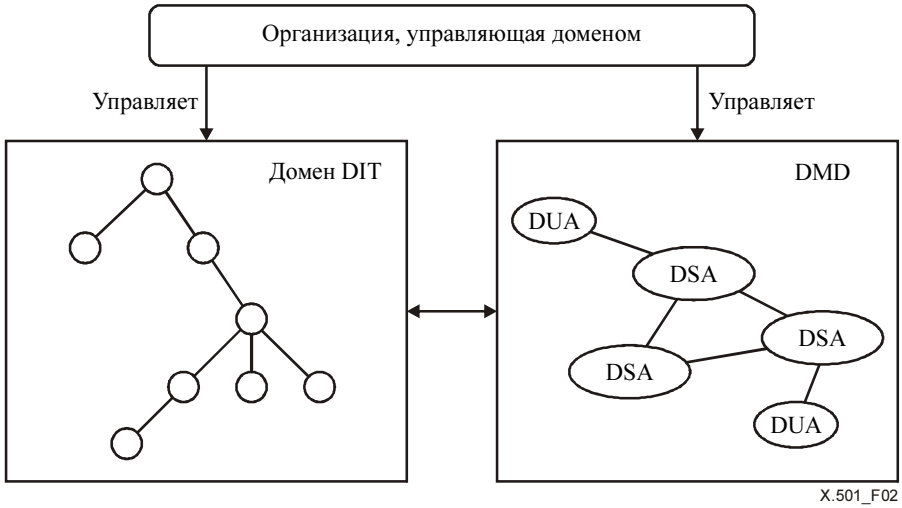


Рисунок 2 – Управление Справочником

Осуществляемое DMO управление DUA предполагает непрерывную ответственность за обслуживание этого DUA, например сопровождение или в некоторых случаях владение, которое выполняет DMO. DMO может принять решение или не принимать решения об использовании этих спецификаций Справочника для руководства любым взаимодействием между агентами DUA и DSA, которые целиком находятся в пределах данного DMD.

Агент какого-либо DMO, осуществляющий административное управление Справочником в различных аспектах, называется *административным органом*. Термин *административные полномочия* (строчными буквами) означает полномочия, которыми DMO наделяет административный орган для осуществления стратегии.

ПРИМЕЧАНИЕ 2. – Модель административного органа Справочника описана в разделе 4.

Для облегчения ссылок, например в правилах поиска, домену DMD может быть присвоен идентификатор объекта (DMD-id).

РАЗДЕЛ 3 – МОДЕЛЬ ПОЛЬЗОВАТЕЛЬСКОЙ ИНФОРМАЦИИ СПРАВОЧНИКА

7 Информационная база Справочника

7.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

7.1.1 статья псевдонима: Статья "псевдонима" класса, содержащая информацию, которая используется для обеспечения альтернативного имени объекта или статьи псевдонима.

7.1.2 порождающий элемент: Статья в корне иерархии членов семейства, которая содержит составную статью.

7.1.3 составная статья: Представление объекта в единицах членов семейства, которые иерархически организованы в одно или более семейство статей.

7.1.4 производная статья: Информация статьи, являющаяся результатом операции поиска, которая содержит значения атрибутов, полученные путем выполнения объединения данных из более чем одной статьи Справочника.

7.1.5 прямой суперкласс: Относительно подкласса – класс объектов, из которого непосредственно выведен данный подкласс.

7.1.6 информационная база Справочника (DIB): Вся совокупность информации, к которой Справочник обеспечивает доступ, включая все элементы информации, которые можно читать и с которыми можно совершать действия, используя операции Справочника.

7.1.7 информационное дерево Справочника (DIT): DIB рассматривается как дерево, вершины которого (не корень) являются статьями Справочника.

ПРИМЕЧАНИЕ. – Термин "DIT" используется вместо термина "DIB" только в контексте древовидной структуры информации.

7.1.8 статья (справочника): Именованный набор информации в пределах DIB. DIB состоит из статей.

7.1.9 семейство: Иерархическая подгруппа статей членов семейства, которые представляют конкретный класс информации в пределах составной статьи. Корень каждого семейства в рамках составной статьи является порождающим элементом, но, не считая коллективного порождающего члена, семейства не имеют общих членов. Семейство отличается от других семейств в пределах составной статьи наличием общего класса (структурного класса объектов) для каждого члена семейства, который является непосредственно подчиненным по отношению к порождающему элементу

7.1.10 член семейства: Член иерархического набора статей, которые образуют составную статью.

7.1.11 непосредственно предшествующая(ий), старшая(ий) (имя существительное): Относительно конкретной статьи или объекта (что именно имеется в виду, должно быть ясно из контекста) – непосредственно предшествующая, старшая статья или непосредственно предшествующий, старший объект.

7.1.12 непосредственно предшествующая, старшая статья: Относительно конкретной статьи – статья, являющаяся начальной вершиной дуги DIT, конечной вершиной которой является эта конкретная статья.

7.1.13 непосредственно предшествующий, старший объект: Относительно конкретного объекта – объект, статья *объекта* которого является непосредственно предшествующей, старшей по отношению к *любым* статьям (объект или псевдоним) второго объекта.

7.1.14 объект (представляющий интерес): Что-либо в некотором "мире", как правило, в мире электросвязи и информатики или в некоторой его части, что может быть опознано (может быть названо) и что представляет достаточный интерес, для того чтобы хранить об этом информацию в DIB.

7.1.15 класс объектов: Индивидуализированное семейство объектов (или потенциальных объектов), обладающих общими определенными характеристиками.

7.1.16 статья объекта: Статья, которая является основным набором информации в DIB об объекте и которая, таким образом, может считаться представляющей этот объект в DIB.

7.1.17 связанные статьи: Группа статей (справочника), каждая из которых может быть определена как хранящая информацию в DIB о конкретном представляющем интерес объекте реального мира. Различные статьи в группе могут содержать информацию различных типов об этом объекте реального мира и могут даже содержать противоречивую информацию.

ПРИМЕЧАНИЕ 1. – Ценность содержащейся в группе связанных статей информации зависит от надежности отождествления каждой статьи с реальным миром.

ПРИМЕЧАНИЕ 2. – Для связанных статей возможно, но не необходимо, существовать в отдельных DIT и иметь идентичные выделенные имена. Аналогичным образом, для несвязанных статей возможно иметь идентичные выделенные имена, однако рекомендуется использовать идентичные выделенные имена только для связанных статей.

7.1.18 подкласс: Относительно одного или более суперклассов – класс объектов, выведенный из одного или более суперклассов. Члены подкласса обладают всеми характеристиками суперкласса, а также дополнительными характеристиками, которыми не обладает ни один из членов этих суперклассов.

7.1.19 подчиненная(ый), младшая(ий): Противоположность предшествующей(му), старшей(му).

7.1.20 суперкласс: Относительно подкласса – прямой суперкласс или суперкласс по отношению к классу объектов, который является прямым суперклассом (рекурсивно).

7.1.21 предшествующая(ий), старшая(ий): (Применимо к статье или объекту) непосредственно предшествующий, старший или предшествующий, старший по отношению к статье/объекту, которые являются непосредственно предшествующими, старшими (рекурсивно).

7.2 Объекты

Задачей Справочника является хранение и обеспечение доступа к информации об *объектах*, *представляющих интерес (объектах)*, которые существуют в некоем "мире". Объектом может быть что-либо опознаваемое (которое может быть названо).

ПРИМЕЧАНИЕ 1. – Под словом "мир", как правило, подразумевается мир электросвязи и информатики или какая-либо его часть.

ПРИМЕЧАНИЕ 2. – Объекты, известные Справочнику, могут не соответствовать напрямую группе "реальных" объектов мира. Например, реально существующая в мире личность в контексте Справочника может определяться как два разных объекта – как лицо, ведущее коммерческую деятельность, и как житель. Отображение в настоящей спецификации не описывается, этот вопрос решается пользователями и поставщиками услуг Справочника в соответствии со своими прикладными программами.

Класс объектов – это индивидуализированное семейство объектов или потенциальных объектов, обладающих общими определенными характеристиками. Каждый объект принадлежит, по крайней мере, одному классу. Класс объектов может быть *подклассом* других классов объектов, в каковом случае члены такого подкласса также считаются членами этих классов – суперклассов. Могут существовать подклассы подклассов и т. д., до произвольной глубины.

7.3 Статьи Справочника

DIB составляют *статьи (Справочника)*. Статья – это именованный набор информации.

Существуют четыре вида статей:

- *Статьи объектов*: Представляют основной набор информации в DIB о конкретном объекте. Для любого конкретного объекта существует только одна статья объекта или составная статья (см. п. 8.10). Статья объекта считается представляющей объект. Статья объекта может быть единой статьей или составной статьей, содержащей множество статей, которые вместе представляют объект.
- *Статьи псевдонима*: Используются для обеспечения альтернативных имен статьи объекта (могут быть порождающим элементом составной статьи, но не дочерними членами семейства).
- *Подстатьи*: Представляют набор информации в DIB, необходимый для выполнения административных и функциональных требований Справочника. Подстатьи рассматриваются в разделе 5.
- *Члены семейства*: Конкретные статьи, являющиеся компонентами составной статьи. Порождающий элемент составной статьи также является членом семейства.

Структура статей справочника с точки зрения пользователя показана на рисунке 3 и описываются в п. 8.2.

Каждая статья содержит индикацию классов объектов и их суперклассов, которым она принадлежит.

Некоторые статьи объектов специально назначены для целей административного управления Справочником. Такие статьи называются административными статьями. Пользователь Справочника, как правило, не знает об этом и просматривает их так же, как и другие статьи объектов.

7.4 Информационное дерево Справочника (DIT)

Для выполнения требований, необходимых для осуществления распределения и управления DIB очень большого объема, а также для обеспечения возможности однозначного именования и быстрого поиска статей, двумерная структура, очевидно, не подходит, Следовательно, может использоваться иерархическая структура, как правило, используемая для объектов (например, лицо работает в департаменте, который принадлежит какой-либо организации, штаб-квартира которой находится в какой-то стране), которая организуется путем построения из статей дерева, называемого *информационным деревом справочника* (*DIT*).

ПРИМЕЧАНИЕ. – Введение в понятия и терминологию древовидной структуры содержатся в Приложении I.

Составные части DIT трактуются следующим образом:

a) вершинами являются статьи. Статьи объектов могут быть содержащими либо не содержащими листьев вершинами. Корень не является статьей по существу, но может, когда это удобно (например, в определениях b) и c), ниже), рассматриваться как нулевая статья объекта (см. d) ниже);

b) дуги определяют отношения между вершинами (и, следовательно, статьями). Дуга между вершиной A и вершиной B означает, что статья в A является *непосредственно предшествующей*, *старшей статьей (непосредственно старшая)* по отношению к статье в B. *Предшествующие*, *старшие статьи* (*предшествующие*, *старшие*) конкретной статьи являются ее непосредственно предшествующими, старшими вместе со своими предшествующими, старшими (рекурсивно). *Подчиненные*, *младшие статьи* (*подчиненные*, *младшие*) конкретной статьи являются ее непосредственно подчиненными, младшими вместе со своими подчиненными младшими (рекурсивно);

c) объект, представленный статьей, по отношению к своим подчиненным, младшим является органом именования (см. пункт 8) или весьма подобен ему;

d) корень представляет высший уровень органа именования для данной DIB.

Отношения предшествующий, старший/подчиненный, младший между объектами могут выводиться из отношений между статьями объекта. Объект является *непосредственно предшествующим*, *старшим объектом (непосредственно предшествующим*, *старшим)* по отношению к другому объекту тогда и только тогда, когда статья объекта первого объекта является непосредственно предшествующей, старшей по отношению к любой статье объекта второго объекта. Термины *непосредственно подчиненный*, *младший объект*, *непосредственно подчиненный*, *младший; предшествующий*, *старший* и *подчиненный*, *младший* (применяемые к объектам) имеют соответствующие аналогичные значения.

Разрешенные отношения предшествующий, старший/подчиненный, младший обусловлены определениями структуры DIT (см. п. 13.7).

Справочник, наряду с информацией, связанной со статьями Справочника, хранит дополнительную информацию, касающуюся наборов статей Справочника. Такими наборами могут быть *поддеревья* (дерева DIT) или *уточнения поддеревьев* (если используется не истинно древовидная структура). См. п. 12.

8 Статьи Справочника

8.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

8.1.1 опорный атрибут: Пользовательский атрибут, имеющий "друзей", как определено в рамках соответствующей подсхемы. Опорный атрибут может использоваться для того, чтобы вызвать включение дружественных атрибутов в совокупность атрибутов, которые должны выбираться, или может рассматриваться для сопоставления в операции поиска, не присутствуя в статье.

8.1.2 атрибут: Информация определенного типа. Статьи состоят из атрибутов.

8.1.3 атрибут пользователя: Атрибут, представляющий информацию пользователя.

8.1.4 иерархия атрибута: Аспект атрибута, который позволяет извлекать тип атрибута пользователя из более общего типа атрибута пользователя. Взаимоотношение определений двух типов атрибутов (которое управляет конкретным поведением атрибутов в соответствии с типами атрибутов) является, таким образом, иерархическим.

8.1.5 подтип атрибута (подтип): Тип атрибута A соотносится с другим типом атрибута B таким образом, что либо A извлечен из B, и в этом случае A является *прямым* подтипом B, либо A извлечен из типа атрибута, который является подтипом B, в каковом случае A является *косвенным* подтипом B.

8.1.6 супертип атрибута (супертип): Тип атрибута B соотносится с другим типом атрибута A таким образом, что либо A извлечен из B, и в этом случае B является *прямым* супертипом A, или A извлечен из типа атрибута, который является подтипом B, в каковом случае B является *косвенным* супертипом A.

8.1.7 тип атрибута: Компонент атрибута, который указывает класс информации, передаваемой атрибутом.

8.1.8 значение атрибута: Конкретный экземпляр класса информации, указанного типом атрибута.

8.1.9 утверждение о значении атрибута: Высказывание, которое может быть истинным, ложным или неопределенным согласно конкретным правилам сопоставления для данного типа, касающимся наличия в статье типа атрибута конкретного типа.

8.1.10 вспомогательный класс объектов: Класс объектов, который является классом описания статей или классов статей и не используется для структурной спецификации DIT.

8.1.11 коллективный атрибут: Атрибут пользователя, значения которого одинаковы для всех членов набора статей.

8.1.12 контекст: Характеристика, которая может быть связана со значением атрибута пользователя для точного указания информации, которая может использоваться для определения применимости этого значения.

8.1.13 утверждение о контексте: Высказывание, которое может быть истинным или ложным относительно типа контекста и конкретных значений контекстов для данного типа, определяющее применимость значения атрибута.

8.1.14 тип контекста: Компонент контекста, который указывает его тип и назначение.

8.1.15 перечень контекстов: Совокупность контекстов, связанных со значением атрибута.

8.1.16 значение контекста: Конкретный экземпляр характеристики, указывающей тип контекста.

8.1.17 производный атрибут: Атрибут, значение или значения которого полностью или частично рассчитываются, а не хранятся в явном виде.

8.1.18 производное значение класса объектов: Значение класса объектов, наличие которого не управляется пользователем, но рассчитывается. Выдаваемые значения класса объектов относятся к категории абстрактных.

8.1.19 прямая ссылка на атрибут: Ссылка (в абстрактной службе Справочника и DSA) на одно или более значений атрибутов с использованием идентификатора их типа атрибутов.

8.1.20 выделенное значение: Значение атрибута в статье, которое может присутствовать в относительно выделенном имени этой статьи.

8.1.21 фиктивный атрибут: Атрибут, который определяется как атрибут пользователя, но который никогда не присутствует в статье. Фиктивным атрибутом может быть только опорный атрибут.

8.1.22 набор статей: Набор статей, принадлежащих явно определенному поддереву или уточнению поддерева DIT.

8.1.23 дружественные атрибуты: Совокупность атрибутов пользователя, связанная административным полномочием с особым атрибутом пользователя (известным как опорный атрибут), для включения в совокупность атрибутов, возвращаемых, если определен опорный атрибут, или используемых потенциально для согласования с предикатом, который включает условие по опорному атрибуту

8.1.24 косвенная ссылка на атрибут: Ссылка (в абстрактной службе Справочника и DSA) на одно или более значений атрибутов с использованием идентификатора супертипа их типа атрибутов.

8.1.25 правило сопоставления: Правило, образующее часть схемы Справочника, которое позволяет осуществлять выбор статей с помощью определенного заявления (утверждение по правилу сопоставления) относительно их значений атрибутов.

8.1.26 утверждение по правилу сопоставления: Высказывание, которое может быть истинным, ложным или неопределенным, относительно наличия в статье значений атрибутов, которые отвечают критериям, определенным правилом сопоставления.

8.1.27 операционный атрибут: Атрибут, представляющий операционную и/или административную информацию.

8.1.28 структурный класс объектов: Класс объектов, используемый для структурной спецификации DIT.

8.1.29 структурный класс объектов статьи: В отношении конкретной статьи для определения применяемых к этой статье правила контента DIT и правила структуры DIT используется *единый* структурный класс объектов. Этот класс объектов указывается операционным атрибутом **structuralObjectClass**. Данный класс объектов является старшим по отношению к подчиненным классам объектов в цепочке суперкласса структурных классов объектов данной статьи.

8.2 Общая структура

Как показано на рисунке 3, статья состоит из совокупности *атрибутов*.

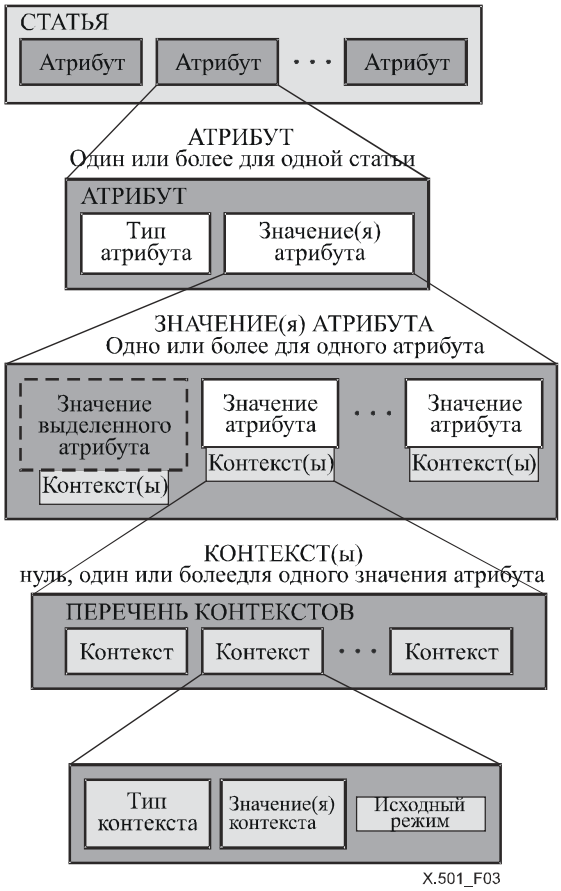


Рисунок 3 – Структура статьи

Каждый атрибут представляет часть информации об объекте или описывает конкретную характеристику объекта, которому соответствует данная статья.

ПРИМЕЧАНИЕ 1. – Примеры атрибутов, которые могут существовать в статье, включают информацию об имени, такую как имя собственное объекта, и адресную информацию, такую как номер телефона.

Атрибут состоит из *типа атрибута*, который определяет класс выдаваемой атрибутом информации, и соответствующих *значений атрибута*, которые являются конкретными экземплярами этого класса, существующими в данной статье. Значением атрибута пользователя может быть нуль, один или более контекстов, связанных с ним в его перечне контекстов. Значения операционных атрибутов не имеют контекстов.

ПРИМЕЧАНИЕ 2. – Типы атрибутов, значения атрибутов и контексты описаны в пп. 8.4, 8.5 и 8.8, соответственно. Операционные атрибуты описаны в пункте 12.

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

Атрибут может быть обозначен как содержащий одно значение или множество значений. Справочник обеспечивает, что содержащий одно значение атрибут имеет только одно единственное значение. Это значение может содержать перечень контекстов для связи характеристик со значением атрибута. Атрибуты, находящиеся в запоминающем устройстве, имеют по меньшей мере одно значение, но иногда могут иметь нуль значений при передачи в запоминающее устройство или из него (например, поскольку значения скрыты процессом управлением доступом).

8.3 Классы объектов

Классы объектов используются в Справочнике для ряда задач, в число которых входят:

- описание и категоризация объектов и статей, которые соответствуют этим объектам;
- в надлежащих случаях управление функционированием Справочника;
- регулирование расположения статей в DIT в соответствии со спецификациями правил структуры DIT;
- регулирование атрибутов, которые содержаться в статьях, в соответствии со спецификациями правил контента DIT;
- определение классов статей, которые должны быть связаны с конкретной стратегии надлежащим административным полномочием.

Некоторые классы объектов стандартизованы на международном уровне. Другие определяются национальными административными органами и/или частными организациями. Это означает, что за определение классов объектов и их однозначное обозначение несут ответственность несколько различных органов. Такое определение осуществляется путем обозначения каждого класса объекта с помощью идентификатора объекта при определении класса объектов. Применяемая для этих целей нотация представлена в п. 13.3.3.

ПРИМЕЧАНИЕ 1. – Административный орган может использовать классы объектов, отличные от классов объектов, определенных и внесенных в спецификации Справочника. Административный орган может сам составлять и вносить классы объектов, например в дополнение к определенным в спецификациях Справочника.

Класс объектов (*подкласс*) может быть производной класса объектов (его прямого *суперкласса*), который сам является производной еще более общего класса объектов. Для структурных классов объектов этот процесс завершается наиболее общим классом **top** (вершина). Упорядоченное множество суперклассов до самого верхнего предшествующего, старшего класса объектов по отношению к данному классу объектов является *цепочкой суперклассов.*

Класс объектов может быть выведен из двух и более прямых суперклассов (суперклассов, не являющихся частью той же цепочки суперклассов). Этот метод выведения производных подклассов называется *множественным наследованием*.

Спецификация класса объектов статьи или членов семейства определяет, обязательным или необязательным является атрибут; такая спецификация также применяется для подклассов этого класса. Можно сказать, что подкласс *наследует* спецификацию обязательных и необязательных атрибутов своего суперкласса. Спецификация подкласса может указывать, что необязательный атрибут данного суперкласса является обязательными в данном подклассе.

Если класс объектов определяет опорный атрибут, имеющий дружественные атрибуты, как необязательный или обязательный, это автоматически вызывает включение дружественных атрибутов в качестве необязательных атрибутов без необходимости включения их в какое-либо определение класса объекта или правило контента.

Класс объектов может определять фиктивный атрибут как обязательный или необязательный атрибут, если фиктивный атрибут является опорным атрибутом. Если класс объектов определяет фиктивный атрибут как атрибут обязательного или необязательного типа, опорный атрибут не присутствует в статье данного класса объектов, но если он определен как обязательный атрибут, в статье будет присутствовать, по крайней мере, один из его дружественных атрибутов. Вместе с тем, если нефиктивный опорный атрибут определен как атрибут обязательного типа, будет присутствовать атрибут типа данного опорного атрибута.

Типы дружественных атрибутов не присутствуют, если они исключены правилами контента.

Существуют три вида классов объектов:

- абстрактные классы объектов;
- структурные классы объектов; и
- дополнительные классы объектов.

ПРИМЕЧАНИЕ 2. – Настоящая спецификация Справочника не ограничивает определение подклассов этими видами (т. е. абстрактным, структурным и дополнительным); однако администраторам следует принять к сведению, что совместимость с серверами LDAP в некоторых ситуациях может быть существенно нарушена, особенно при использовании структурных классов объектов, которые являются подклассами дополнительных классов объектов, и наоборот.

Каждый класс объектов является исключительно классом одного из этих видов и остается классом этого вида в любом месте Справочника. Определение каждого класса объектов указывает, объекты какого вида он включает.

Все статьи являются членами класса объектов **top** и, по крайней мере, одного иного класса объектов.

8.3.1 Абстрактные классы объектов

Абстрактный класс объектов используется в основном для производства классов других объектов при условии наличия общих характеристик классов таких объектов. Любая статья не принадлежит только к абстрактным классам объектов.

top является абстрактным классом объектов, который используется как суперкласс всех структурных классов объектов.

Наряду с использованием этого класса для выведения других классов объектов, значение абстрактного класса объектов может быть производным значением, т. е. его присутствие является рассчитанным или выведенным Справочником. Например, значение родительского класса объектов **parent** для конкретной статьи рассчитывается или выводится по присутствию члена семейства, дополнительного дочернего класса объектов **child,** являющегося непосредственно подчиненным по отношению к данной статье.

8.3.2 Структурные классы объектов

Класс объектов, определенный для использования в структурной спецификации DIT, называется *структурным классом объектов.* Структурные классы объектов используются в определении структуры имен объектов для согласующихся статей.

Статья объекта или псевдонима характеризуется исключительно одной цепочкой суперкласса структурных классов объектов, которая имеет единственный структурный класс объектов в качестве наиболее подчиненного класса объектов. Этот структурный класс объектов называется *структурным классом объектов данной статьи.*

Структурные классы объектов соотносятся со связанными статьями:

- статья, соответствующая структурному классу объектов, представляет объект реального мира, входящий в класс данных объектов;
- правила структуры DIT относятся только к структурным классам объектов; структурный класс объектов любой статьи используется для указания местоположения статьи DIT;
- структурный класс объектов статьи используется, наряду с соответствующим правилом контента DIT, для управления контентом статьи.

Структурный класс объектов статьи не изменяется.

8.3.3 Дополнительные классы объектов

Для определенных приложений, использующих Справочник, зачастую оказывается полезным определить *дополнительный класс объектов*, который может использоваться при построении статей нескольких типов. Например, системы обработки сообщений используют дополнительный класс пользователя MHS (см. Рек. МСЭ-Т X.402 | ИСО/МЭК 10021-2) для определения комплекта обязательных и необязательных атрибутов обработки сообщений для типов статей, структурный класс объектов которых является переменным, например индивидуум, входящий в состав организации или имеющий адрес местожительства.

В некоторых средах существует необходимость иметь возможность добавлять или удалять из перечня атрибутов, разрешенных в статье, которая относится к конкретному, возможно стандартизованному, классу (или классам).

Это требование можно выполнить путем определения и использования дополнительного класса объектов, семантика которого известна и поддерживается в местном сообществе и который при необходимости время от времени меняется.

Это требование можно также выполнить, используя средства определений правила контента DIT для динамического (т. е. без регистрации) добавления или исключения атрибутов из статей в конкретных точках DIT (см. п. 13.3.3).

Дополнительные классы объектов являются средствами описания статей или классов статей.

Таким образом, статья может быть не только членом данного структурного класса объектов, но и – в необязательном порядке – членом одного или более дополнительных классов объектов.

Дополнительные классы объектов статьи могут со временем изменяться.

ПРИМЕЧАНИЕ. – Нерегистрируемый класс объектов, включенный в первое издание настоящих спецификаций Справочника для поддержки требований, рассматриваемых в данном пункте, в этом издании исключен с целью использования правила контента DIT.

8.3.4 Определение класса объектов и первое издание настоящей спецификации Справочника

Классы объектов, определенные с использованием терминологии первого издания данной спецификации Справочника, не классифицируются как структурные, дополнительные или абстрактные.

Классы объектов псевдонимов, определенные с использованием терминологии первого издания данной спецификации Справочника, могут считаться определенными как абстрактные, дополнительные или структурные классы объектов и использоваться в подсхеме соответствующим образом.

8.4 Типы атрибутов

Некоторые типы атрибутов стандартизуются на международном уровне. Другие типы атрибутов определяются национальными административными органами и частными организациями. Это означает, что за определение типов и их однозначное обозначение несут ответственность несколько различных органов. Такое определение осуществляется путем обозначения каждого типа атрибута с помощью идентификатора объекта после определения

типа. Используя нотацию класса информационных объектов **ATTRIBUTE**, как это показано в п. 13.4.8, тип атрибута определяется следующим образом:

AttributeType ::= ATTRIBUTE.&id

Все атрибуты в статье имеют индивидуальные типы атрибутов.

Некоторые атрибуты могут не сохраняться в запоминающем устройстве и не быть доступными в рамках статьи, но они предназначены для включения в команды в целях переноса информации эксплуатационного характера, например диагностической информации, которую удобно передавать в виде атрибутов. Другие атрибуты, называемые *атрибутами управления*, могут, как часть своего определения, указывать конкретную процедуру, которая должна быть выполнена, исходя из содержащейся в атрибуте информации. Атрибут управления может быть определен в команде, помещаемой в статьи, и т. д. Примеры см. в п. 7.5.3 Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6.

Существует ряд типов атрибутов, известных Справочнику, которые он использует в своих целях. К ним относятся:

a) **objectClass** – Атрибут этого типа присутствует в каждой статье и указывает классы и суперклассы, к которым принадлежит данный объект.

b) **aliasedEntryName** – Атрибут этого типа присутствует в каждой статье и содержит имя (см. п. 8.5) статьи, на которую ссылается статья псевдонима.

Данные атрибуты определяются в п. 13.4.8.

Типы атрибутов пользователя, которые присутствуют или могут присутствовать в статье объекта или псевдонима, регулируются правилами, которые применяются к указанным классам объектов, а также правилом контента DIT для данной статьи (см. п. 13.8). Типы атрибутов, которые могут присутствовать в подстатье, регулируются правилами схемы системы.

Некоторые статьи Справочника могут содержать специальные атрибуты, не видимые, как правило, для пользователя Справочника. Эти атрибуты называются операционными атрибутами и используются для удовлетворения различных административных и функциональных требований Справочника. Более подробно операционные атрибуты рассматриваются в разделе 5.

8.5 Значения атрибутов

Определение атрибута также включает описание синтаксиса и, следовательно, типа данных, которому должно соответствовать каждое значение этих атрибутов. Используя нотацию класса информационных объектов **ATTRIBUTE**, как это показано в п. 13.4.8, значение атрибута определяется следующим образом:

AttributeValue ::= ATTRIBUTE.&Type

Значение атрибута может быть обозначено как *выделенное значение*, в каковом случае это значение атрибута может образовывать часть относительного выделенного имени статьи (см. п. 9.3). Возможно наличие множества выделенных значений, различающихся в зависимости от контекста, как это описано в п. 9.3.

Значения, предоставляемые клиентом, фиксируются для хранения в Справочнике. Значения сравнений являются эфемерным и не затрагивает сохраненное значение.

8.6 Иерархия типов атрибутов

При определении типа атрибута в качестве основы определения могут факультативно использоваться характеристики более общего типа атрибутов. Новый тип атрибутов является *прямым подтипом* более общего типа атрибута – *супертипа*, производной которого этот новый атрибут является.

Иерархия атрибутов позволяет осуществлять доступ к DIB с различной степенью детализации. Это достигается благодаря разрешению доступа к компонентам значения атрибутов путем использования либо их конкретного идентификатора типа атрибута (прямая ссылка на атрибут), либо идентификатора более общего типа атрибута (косвенная ссылка).

Семантически связанные атрибуты могут быть выстроены в иерархическую структуру, в которой более конкретные типы располагаются как подчиненные по отношению к более общим. Поиск или считывание атрибутов и их значений облегчается при указании более общего типа атрибута; определенный таким образом пункт фильтра проверяется для более конкретных типов, а также для указанного типа; утверждение о контексте, определенное для более общего типа атрибутов, также применяется к более конкретному типу.

Если в качестве результата поиска выбраны конкретные типы, являющиеся подчиненными, должны возвращаться эти типы, если таковые имеются. Если в качестве результата поиска выбраны более общие типы, должны возвращаться

как общие, так и конкретные типы, если таковые имеются. Значение атрибута всегда должно возвращаться как значение своего собственного типа атрибута.

Для того чтобы статья содержала значение типа атрибута, принадлежащего иерархии атрибутов, этот тип должен быть явным образом включен либо в описание класса объектов, которому принадлежит статья, либо он должен быть разрешен правилом контента DIT, применимым к данной статье.

Все типы атрибутов в иерархии атрибутов для целей административного управления статьей и изменения пользователем содержания статьи трактуются как отдельные и несвязанные типы.

Значение атрибута, хранящееся в статье объекта или псевдонима Справочника, имеет только один тип атрибута. Этот тип указывается при первом добавлении значения в статью.

8.7 Дружественные атрибуты

Дружественные атрибуты являются атрибутами пользователя, определенными административным органом как связанные некоторым фактическим образом с конкретным опорным атрибутом. Если опорный атрибут указан в информации, которая должна быть возвращена операциями чтения Read или поиска Search, эта функция разрешает возврат дружественных атрибутов для данного опорного атрибута согласно элементам управления службой и административного управления (включая управление доступом, правила поиска и т. д.). Аналогичным образом, если опорный атрибут указан в элементе фильтра в рамках предиката поиска, дружественные атрибуты могут использоваться для удовлетворения предиката, если правило сопоставления для "друга" совместимо с предложенным значением.

Если опорный атрибут разрешен в пределах статьи путем включения в обязательные или необязательные перечни значений классов объектов для данной статьи, дружественные атрибуты также разрешены, если только они не исключены правилами контента. Если опорный атрибут не является обязательным атрибутом, он может отсутствовать в статье, даже при наличии дружественных атрибутов.

В пределах подсхемы любой атрибут пользователя может быть обозначен как опорный атрибут.

ПРИМЕЧАНИЕ 1. – В качестве примера опорного атрибута рассмотрим гипотетический атрибут **commsAddr,** который в конкретной подсхеме имеет дружественные атрибуты, являющиеся типами атрибутов адресов связи, например номер телефона, адрес электронной почты, URL и т. д.

Взаимоотношение опорный-дружественный не является ни коммутативным, ни транзитивным:

- если опорный атрибут A имеет "друга" B, из этого не следует, что A является "другом" B;
- если опорный атрибут A имеет "друга" B, а B имеет "друга" C, из этого не следует, что C является "другом" A.

Если атрибут A является "другом" какого-либо опорного атрибута, то все подтипы A являются "друзьями" этого опорного атрибута. Вместе с тем, из этого не следует, что супертипы A также являются "друзьями" этого опорного атрибута.

Обозначение атрибута как "друга" не предоставляет специальных возможностей управления доступом или защиты по правилу поиска, если только он не ассоциируется с принадлежностью класса объектов опорного атрибута (членом которого он становится автоматически).

ПРИМЕЧАНИЕ 2. – В настоящее время правила управления доступом и поиска не используют классы объектов в качестве средства определения множеств атрибутов для предоставления специальных привилегий или защиты.

8.8 Контексты

Информационная модель может быть уточнена путем связи с характеристиками значений атрибутов, называемыми контекстами. Связанное с любыми атрибутом пользователя значение может быть перечнем контекстов, предоставляющих дополнительную информацию, которая может использоваться для определения применимости данного значения атрибута.

ПРИМЕЧАНИЕ 1. – Например, контексты могут использоваться для связи конкретных языка, времени или территории со значением атрибута.

Каждый контекст состоит из поля типа, поля значения, синтаксис которого определяется типом, и флагом возврата в исходный режим **fallback**. Используя нотацию класса информационных объектов **CONTEXT**, как это показано в п. 13.9, контекст определяется следующим образом:

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

Типом контекста **contextType** является **OBJECT IDENTIFIER,** и он описывается с использованием класса информационных объектов **CONTEXT**, который определен в п. 13.9. Он описывает конкретную характеристику, представляемую контекстом.

Значениями контекста **contextValues** является совокупность одного или более значений характеристики, определенной **contextType,** которые связаны с конкретным значением атрибута.

Флаг **fallback** используется для обозначения одного или более значений атрибута для предписанного поведения в соответствии с типом контекста. Кроме конкретных **contextValues** этого связанного с ним типа контекста, значение атрибута, для которого данный флаг имеет значение **TRUE** (истина) для данного **contextType**:

– рассматривается как связанное с любым значением данного **contextType**, с которым не связаны иным образом какие-либо другие значения того же атрибута. Таким образом, утверждение о контексте данного типа контекста, не давшее совпадения ни с одним значением атрибута по правилам сопоставления **contextValues**, даст совпадение с любым значением атрибута, для которого флаг **fallback** имеет значение **TRUE** для данного типа контекста;

ПРИМЕЧАНИЕ 2. – Например, результатом попытки выбора значения атрибута, связанного с конкретным языком, станут значения с флагом **fallback**, установленным в значение **TRUE**, если ни одно из значений атрибута не связано иным образом с выбранным языком.

– рассматривается как значение, которое должно быть зафиксировано в течение выполнения операции, сбрасывающей значения атрибутов для данного типа атрибутов. Команда изменения Modify (сброс значения) удалит все значения для выбранного типа атрибута, имеющие связанный контекст, для которого установлено значение FALSE (ложь) флага **fallback**.

ПРИМЕЧАНИЕ 3. – Операция Modify (сброс значения) рассматривается более подробно в п. 11.3.2 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

Значение атрибута, не имеющее контекста или имеющее перечень контекстов, в который не включен контекст определенного типа, рассматривается как применимое при всех значениях контекста данного конкретного типа.

ПРИМЕЧАНИЕ 4. – Например, в результате выбора, базирующегося на значении контекста "французский" контекста языка, будет отобрано значение атрибута, не имеющее какого-либо контекста языка, конкретно связанного с ним (а также тех значений атрибутов, которые имеют контекст языка "французский", связанный с ним специально).

Все контексты в перечне контекстов значения атрибута имеют определенные типы контекстов.

Контекстная информация, связанная со значениями атрибутов, может считываться с этими значениями атрибутов (например, для проведения различия между этими значениями атрибутов). Пользователь Справочника может также использовать контексты для уточнения выбора и поиска информации в процессе выполнения операций Справочника.

8.9 Правила сопоставления

8.9.1 Обзор

Исключительно важной для Справочника является способность выбора совокупности статей из DIB на основе утверждений о содержащихся в этих статьях значениях.

Правило сопоставления позволяет осуществлять выбор статьей путем выдачи конкретного утверждения, касающегося значений атрибутов этих статей.

Самым простым типом утверждения является утверждение о значении атрибутов. Более сложные утверждения могут составляться с использованием утверждений по правилам сопоставления. Утверждение по правилу сопоставления является высказыванием, которое может быть истинным, ложным или неопределенным в зависимости от наличия в статье значений атрибутов, отвечающих определенным этим правилом сопоставления критериям.

Утверждение о значении атрибута или по правилу сопоставления оценивается на основании правила сопоставления, связанного с этим утверждением.

Правило сопоставления определяется путем описания:

– диапазона синтаксиса атрибута, поддерживаемого данным правилом;
– конкретных типов совпадений, поддерживаемых данным правилом;
– синтаксиса, необходимого для описания утверждения по каждому конкретному типу совпадения;
– правил для выведения значения синтаксиса утверждения из значения синтаксиса атрибута, если это необходимо.

ПРИМЕЧАНИЕ. – Не налагается каких-либо ограничений на правила сопоставления, которые могут определяться для поддержки конкретного приложения. Вместе с тем, правила, определенные для поддержки одного конкретного приложения, могут не поддерживаться полностью агентами DUA и DSA. По возможности вместо описания новых правил, следует использовать правила сопоставления, определенные в Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6.

Иногда между правилом сопоставления и типами поддерживаемых сопоставлений будет существовать взаимно-однозначное соответствие. Например, абстрактная служба Справочника поддерживает правило сопоставления наличия для выявления наличия атрибута в статье.

Иногда между правилом и типами поддерживаемых сопоставлений будет существовать соответствие множество-множество. Например, абстрактная служба Справочника поддерживает правило упорядочения, разрешающее типы сопоставлений "больше либо равно" или "меньше либо равно".

8.9.2 Утверждения о значениях атрибутов

Утверждение о значении атрибута (AVA) является высказыванием, которое может быть истинным, ложным или неопределенным согласно конкретным правилам сопоставления для данного типа, касающимся наличия в статье типа атрибута конкретного типа. Утверждение включает тип атрибута, значение атрибута, о котором составлено это утверждение, и факультативно утверждение о контекстах, связанных с этим значением атрибута:

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

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

Синтаксис компонента **assertion** утверждения AVA определяется правилом сопоставления на равенство, описанным для данного типа атрибута, и может отличаться от синтаксиса самого атрибута.

8.9.2.1 Оценка AVA

Значение оценки AVA является:

a) неопределенным в любом из следующих случаев:

1) тип атрибута неизвестен;

2) тип атрибута не имеет правила сопоставления на равенство;

3) значение не соответствует типу данных, указанному синтаксисом утверждения по правилу сопоставления на равенство данного атрибута;

ПРИМЕЧАНИЕ. – 2) и 3) обычно указывают на то, что AVA неверно; вместе с тем, 1) может возникать как локальная ситуация (например, конфигурация конкретного DSA не поддерживает данный конкретный тип атрибута).

b) истинным, если данная статья содержит атрибут этого типа, и атрибут содержит это значение, и значение содержит контекст, который соответствует **assertedContexts**, как это описано в п. 8.9.2.2;

c) ложным в любом ином случае.

8.9.2.2 Использование assertedContexts или утверждений о контексте по умолчанию

Включение **assertedContexts** (заявленных контекстов) в **AttributeValueAssertion** является необязательным. Если **assertedContexts** определен, тогда оценка **assertion** выполняется только по тем значениям атрибутов, для которых **assertedContexts** являются истинными, как это описано в п. 8.9.2.3.

Если **assertedContexts** не содержатся в **AttributeValueAssertion**, аналогичным образом может применяться утверждение о контексте по умолчанию; т. е. **assertion** оценивается только по тем значениям атрибута, для которых, как это определено в п. 8.9.2.3, утверждение о контексте по умолчанию является истинным. Существуют три возможных источника утверждения о контексте по умолчанию: утверждение определено для операции в целом, имеется в подстатьях в DIT, и содержится локально в конкретном DSA. Они применяются следующим образом:

1) если **assertedContexts** не содержатся в **AttributeValueAssertion**, тогда применяется любое утверждение, которое было задано для операции в целом для данного типа атрибута в качестве части **operationContexts**, как это описано в п. 7.3 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3;

2) если пользователь не задал **assertedContexts** для AVA, и для данного типа атрибута не имеется утверждения о контексте, заданного для операции в целом, тогда применяется утверждение о контексте по умолчанию для данного типа атрибута, содержащееся в подстатьях утверждения о контексте (если таковые имеются), управляющих данной статьей, как это описано в п 14.7.

3) если после шагов 1) и 2) утверждения о контексте не имеется, DSA может применять локально определенное утверждение о контексте по умолчанию для данного типа атрибута. Такое значение по умолчанию обычно отражает локальные параметры, такие как язык или местоположение использования DSA, или текущее время суток, но оно может быть настроено данным DSA иным образом для каждого DUA, которому оно соответствует;

4) если утверждение о контексте не получено ни из одного из этих источников, тогда **assertion** оценивается по всем значениям данного атрибута.

8.9.2.3 Оценка assertedContexts

assertedContexts является истинным, если:

a) **allContexts** определен (это разрешает утверждению о контекста игнорировать любые значения утверждения о контексте по умолчанию, которые в ином случае могли бы быть применимы, если бы **assertedContexts** не был включен в **AttributeValueAssertion**); или

b) каждый **ContextAssertion** в **selectedContexts** является истинным, как это описано в п. 8.9.2.4.

В любом ином случае **assertedContexts** является ложным.

8.9.2.4 Оценка ContextAssertion

ContextAssertion является истинным для конкретного значения атрибута, если:

a) значение атрибута имеет контекст того же типа **contextType** проверки **ContextAssertion** и любое из сохраненных значений **contextValues** этого контекста совпадает с любым из заявленных **contextValues** согласно определению того, как устанавливается совпадение для этого **contextType**; или

b) значение атрибута не содержит контекстов проверенного типа **contextType**; или

c) ни один из других значений атрибутов не удовлетворяет **ContextAssertion** согласно 1) или 2) в п. 8.9.2.2, выше, но значение атрибута содержит контекст заявленного **contextType** с флагом **fallback**, значение которого установлено в **TRUE**.

В любом ином случае **ContextAssertion** является ложным.

8.9.3 Утверждения о типах атрибутов

Утверждение о типах атрибутов является высказыванием, которое может быть истинным, ложным или неопределенным согласно связанным контекстам.

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

8.9.3.1 Оценка утверждения о типах атрибутов

Утверждение о типе атрибута является:

a) неопределенным, если тип атрибута неизвестен или если атрибут не присутствует в данной статье;

b) истинным, если данная статья содержит атрибут этого типа, и атрибут содержит одно или более значений, которые содержат контекст, совпадающий с **assertedContexts**, как описано в п. 8.9.3.2;

c) ложным в любом ином случае.

8.9.3.2 Использование assertedContexts или утверждений о контексте по умолчанию

Включение **assertedContexts** (заявленных контекстов) в **AttributeTypeAssertion** является необязательным. Если **assertedContexts** определен, **assertedContexts** является истинным для по крайней мере одного значения атрибута согласно правилам, определенным в п. 8.9.2.4.

Если **assertedContexts** не содержится в **AttributeTypeAssertion**, аналогичным образом может применяться утверждение о контексте по умолчанию; т. е. утверждение о контексте по умолчанию является истинным для по крайней мере одного значения атрибута согласно правилам, определенным в п. 8.9.2.4. Возможные источники утверждения о контексте по умолчанию описаны в п. 8.9.2.2.

8.9.4 Утверждения по встроенным правилам сопоставления

Для Справочника являются понятными ряд категорий связанных правил сопоставления, семантика которых повсеместно понятна и применима к значениям многих различных типов атрибутов:

– наличие;

– равенство;

– подстроки;

– упорядочение;

– приблизительное сопоставление.

Синтаксис для проверки конкретных типов сопоставлений, связанных с данными категориями правил сопоставления, встроен в абстрактную службу Справочника:

– синтаксис **present** для правила наличия;

- синтаксис **equality** для правил равенства;
- синтаксисы **greaterOrEqual** и **lessOrEqual** для правил упорядочения;
- синтаксисы **initial**, **any** и **final** для правил подстрок;
- синтаксис **approximateMatch** для правил приблизительного сопоставления.

Синтаксис **present** может использоваться для любого атрибута любого типа. Сопоставление на наличие осуществляет проверку на наличие какого-либо значения конкретного типа.

Специальные правила сопоставления равенства, подстрок и упорядочения могут быть связаны с каким-либо типом атрибута, если он определен. Эти специальные правила применяются при выполнении оценок утверждений по правилам равенства, упорядочения и подстрок с использованием встроенного в абстрактную службу синтаксиса. Если специальные правила не предоставлены, составленные утверждения, касающиеся этих атрибутов, являются неопределенными.

Синтаксис **approximateMatch** поддерживает правило приблизительного сопоставления, определение которого относится к компетенции DSA.

8.9.5 Требования к правилам сопоставления

Для обеспечения стабильного и строго определенного функционирования Справочника необходимо наложить некоторые ограничения на правила сопоставления, которые будут использоваться вместе со встроенным в абстрактную службу Справочника синтаксисом.

Для правила сопоставления равенства, в котором синтаксис утверждения отличается от синтаксиса атрибута, к которому применяется это правило сопоставления, применяются правила выведения значения синтаксиса утверждения из значения синтаксиса атрибута.

Правила сопоставления равенства для атрибутов, применяемых для именования, являются транзитивными, коммутативными и имеют синтаксис утверждения, идентичный синтаксису этих атрибутов.

Транзитивность правила сопоставления означает, что если какое-либо значение **a** совпадает со значением **b**, и если это значение **b** совпадает с третьим значением **c**, то согласно данному правилу значение **a** совпадает со значением **c**.

Коммутативность правила сопоставления означает, что если какое-либо значение **a** совпадает со значением **b,** тогда значение **b** совпадает со значением **a**. Атрибут **presentationAddress** является примером атрибута, поддерживающего синтаксис атрибута, правило сопоставления которого не является коммутативным.

По отношению к конкретному типу атрибута правила равенства и упорядочения (если оба имеются) всегда связаны по крайней мере следующим образом: два значения равны при использовании данного отношения равенства, если и только если они равны при использовании данного отношения упорядочения. Кроме того, отношение упорядочения является полностью упорядоченным, т. е. для всех x, y и z, для которых согласно отношению x предшествует y, и y предшествует z, x всегда предшествует z.

ПРИМЕЧАНИЕ. – Эти требования подразумевают, что если определено упорядочение, также определено и равенство.

По отношению к конкретному типу атрибута правила равенства и подстрок (если оба имеются) всегда связаны по крайней мере следующим образом: для всех x и y, которые совпадают согласно данному отношению равенства, и тогда для всех значений z данного отношения подстрок, результат оценки утверждения по значению x равен результату оценки утверждения по значению y. То есть два значения, которые являются неразличимыми при использовании отношения равенства, также являются неразличимыми при использовании отношения подстрок.

8.9.6 Правила сопоставления равенства для идентификатора объекта и выделенного имени

Существует ряд известных Справочнику и используемых им в своих целях правил сопоставления равенства, которые применяются для оценки утверждений о значениях атрибутов. К ним относятся:

- **objectIdentifierMatch**: Это правило используется для сопоставления атрибутов с синтаксисом **ObjectIdentifier**.
- **distinguishedNameMatch**: Это правило используется для сопоставления атрибутов с синтаксисом **DistinguishedName**.

8.10 Наборы статей

8.10.1 Обзор

Набор статей объектов и псевдонимов может иметь общие характеристики (например, некоторые атрибуты, имеющие то же значение для всех статей в наборе), в силу наличия каких-либо общих характеристик или общих взаимоотношений соответствующих объектов. Такая группировка статей называется "набор статей".

Наборы статей могут содержать статьи объекта или псевдонима, которые связаны своей позицией в DIT. Эти наборы определяются как поддеревья или уточнения поддеревьев, как это описано в разделе 5.

Статья может принадлежать нескольким наборам статей при соблюдении административных ограничений, установленных в разделе 5.

8.10.2 Коллективные атрибуты

Если атрибуты пользователя являются общими для статей в наборе статей, они называется *коллективными атрибутами.*

Разрешается также независимая связь того же коллективного атрибута с двумя и более наборами статей. В этих случаях коллективный атрибут статьи имеет множественные значения. Коллективные атрибуты, таким образом, всегда определяются как содержащие множество значений.

Несмотря на то что для пользователей операций запроса Справочника они представляются атрибутами статей, коллективные атрибуты обрабатываются иначе, чем атрибуты статей в информационной модели Справочника. Это различие объявляется пользователям операцией модификации Справочника таким образом, что коллективные атрибуты не подлежат административному управлению (т. е. модификации) через статьи, в которых они присутствуют, но могут управляться через свои связанные подстатьи.

ПРИМЕЧАНИЕ. – Независимые источники этих значений не показываются пользователям операций запроса Справочника.

Для того чтобы коллективный атрибут появился в статье, правилом контента DIT, которое управляет данной статьей, должно быть разрешено наличие этого типа атрибута

Статьи могут намеренно исключать конкретный коллективный атрибут. Это достигается с помощью атрибута **collectiveExclusions**, который описан в п. 12.7 и определен в п. 14.6.

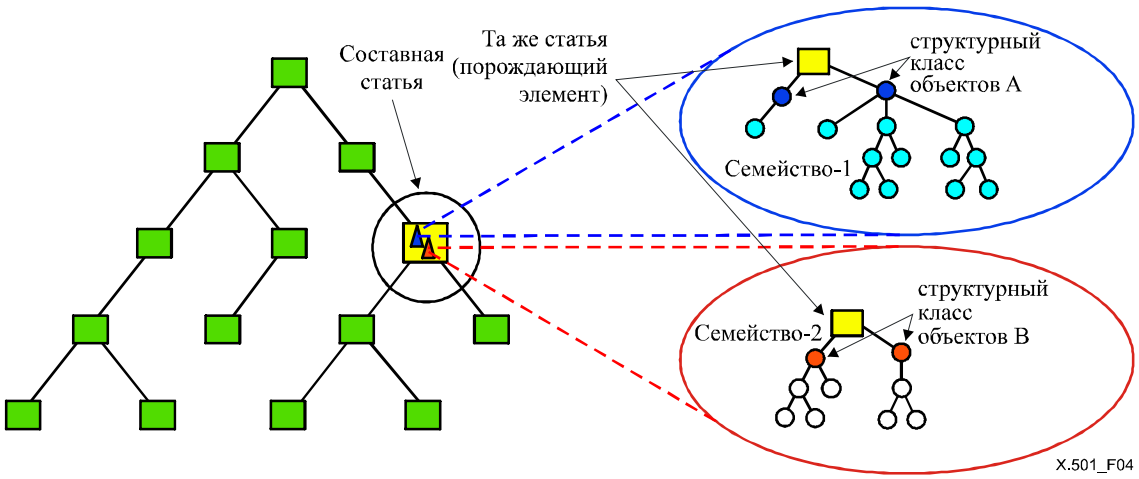
8.11 Составные статьи и семейства статей

Составная статья – это специальная статья, которая содержит статьи членов семейства. Эти члены семейства образуют иерархию и, следовательно, выдают иерархически организованную информацию об объекте, который представлен данной составной статьей. Составная статья представлена в DIT порождающим членом семейства, который находится у корня дерева, содержащего членов семейств.

Члены семейства сами могут быть организованы в одно и более семейств для целей фильтрации и поиска информации. Каждое семейство является поддеревом, отдельные семейства не имеют общих членов семейств, кроме общего корня, который является порождающим элементом. Таким образом, семейство содержит порождающий элемент плюс множество подчиненных членов семейства.

Семейство, кроме порождающего элемента, состоит из всех непосредственно подчиненных членов семейства того же структурного класса объектов. Их подчиненные члены, если таковые имеются, также являются частями того же семейства, независимо от их структурных классов объектов.

Эти понятия представлены графически на рисунке 4.



4 –

Член семейства, являющийся дочерним в пределах дерева семейства, маркируется как относящийся к дополнительному классу объектов **child**. Наличие значения класса объектов **child** для статьи вызывает немедленную маркировку непосредственно предшествующей статьи значением абстрактного класса объектов **parent**. Статья, которая является одновременно **parent** и **child** в пределах дерева семейства, маркируется значениями обоих классов объектов. Порождающий элемент является единственным членом семейства, не относящимся к классу объектов **child**. Построение составных статей выполняется с помощью маркирования статей значениями классов объектов **child**.

Все подчиненные члены по отношению к не являющемуся порождающим элементом члену семейства сами являются членами семейства и маркируются значением класса объектов **child**.

Определение ASN.1 этих классов объектов содержится в п. 13.3.3.

Члены семейства составной статья помещаются в тот же контекст именования, что и порождающий элемент. Не разрешается, чтобы члены семейства были статьями псевдонима. Псевдоним не указывает на дочерний член семейства.

9 Имена

9.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

9.1.1 псевдоним, имя псевдонима: Альтернативное имя объекта при условии использовании статей псевдонима.

9.1.2 разыменование (псевдонима): Процесс преобразования имени псевдонима объекта в выделенное имя.

9.1.3 выделенное имя (статьи): Все статьи объекта, статьи псевдонима и подстатьи имеют по крайней мере одно выделенное имя. Если какое-либо RDN статьи или любой предшествующей, старшей статьи включает атрибут, для которого существуют несколько выделенных значений, отличающихся контекстами (как описано в п. 9.3), то статья имеет несколько выделенных имен, отличающихся контекстами. *Первичное выделенное имя* – это такое выделенное имя, в котором все RDN имеют первичное выделенное значение каждого входящего в состав атрибута в качестве первичного значения в конструкции RDN.

9.1.4 имя (в справочнике): Конструкция, которая выделяет конкретный объект из всех прочих объектов. Имя должно быть однозначным (т. е. обозначать только один объект), но, вместе с тем, оно необязательно должно быть уникальным (т. е. единственным именем, которое однозначно обозначает объект).

9.1.5 имя (статьи): Конструкция, которая выделяет конкретную статью из всех прочих статей.

9.1.6 локальное имя члена: Имя для члена семейства, составленное последовательностью RDN от порождающего элемента вниз до этого члена, не включая RDN для порождающего элемента.

9.1.7 орган именования: Орган, ответственный за распределение имен в некотором регионе DIT.

9.1.8 потенциальное имя: Конструкция, которая синтаксически является именем, но (еще) не объявлена как действительное имя.

9.1.9 относительное выделенное имя (RDN): Совокупность одной или более пар типа и значения атрибута, каждая из которых совпадает с отдельным выделенным значением атрибута данной статьи.

9.2 Имена в целом

Имя (в справочнике) это конструкция, которая идентифицирует конкретный объект среди множества всех объектов. Имя должно быть однозначным, т. е. указывать только на один объект. Вместе с тем имя не обязательно должно быть уникальным, т. е. единственным именем, однозначно обозначающим объект. Имя (в справочнике) также идентифицирует статью. Эта статья является либо статьей объекта, которая представляет объект, либо статьей псевдонима, содержащей информацию, которая помогает Справочнику находить статью, представляющую объект.

ПРИМЕЧАНИЕ 1. – Множество имен объекта, таким образом, включает множество имен псевдонима для этого объекта вместе с выделенными именами этого объекта.

Объекту может быть присвоено выделенное имя без представления какой-либо статьей в Справочнике, и это имя, следовательно, является именем, которая имела бы статья этого объекта, если бы она представляла его в Справочнике.

Синтаксически каждое имя объекта или статьи представляет собой упорядоченную последовательность относительно выделенных имен (см. п. 9.3).

```
Name ::=        CHOICE { -- теперь только одна возможность -- rdnSequence  RDNSequence }

RDNSequence  ::=  SEQUENCE OF  RelativeDistinguishedName

DistinguishedName  ::=  RDNSequence
```

ПРИМЕЧАНИЕ 2. – Имена, которые формируются отличными от описанных здесь способами, являются возможным будущим расширением.

Каждая начальная подпоследовательность имени объекта также является именем объекта. Последовательность объектов, идентифицированных таким образом, начиная от корня и заканчивая именуемым объектом, является такой, что каждый ее член является непосредственно предшествующим, старшим по отношению к объекту, следующему за ним в последовательности.

Потенциальное имя – это конструкция, которая синтаксически является именем, но (еще) не объявлена как действительное имя.

9.3 Относительные выделенные имена

Каждый объект и статья имеют, по крайней мере, одно *относительное выделенное имя (RDN)*. RDN статьи объекта или псевдонима состоит из множества пар типа и значения атрибута (с необязательным перечнем контекстов), каждая из которых совпадает, согласно правилу сопоставления равенства и применимому правилу сопоставления контекстов, с отдельным выделенным значением атрибута данной статьи.

Каждый атрибут, входящий в состав RDN, может иметь более одного выделенного значения, отличающегося контекстом, как описано ниже. Это позволяет иметь альтернативные RDN для того же объекта. В пределах множества выделенных значений (отличающихся контекстами) атрибута только одно из них обозначается как первичное выделенное значение. *Первичное относительно выделенное имя* объекта включает множество первичных выделенных значений из множества атрибутов, которые образуют RDN. При переносе в протоколе каждый атрибут в RDN указывает первичное выделенное значение (если оно присутствует) и может факультативно включать контекст для этого значения и дополнительные альтернативные значения атрибута с контекстом. В этом случае каждое значение атрибута со своим контекстом совпадает с отдельным выделенным значением атрибута статьи для данного типа атрибута согласно применимому правилу сопоставления равенства правилам сопоставления контекстов.

ПРИМЕЧАНИЕ 1. – Правило сопоставления равенства может применяться, поскольку для атрибутов именования синтаксис атрибутов и синтаксис утверждения по правилу сопоставления равенства являются одинаковыми. Аналогичным образом, для контекстов, которые могут использоваться для различения выделенных значений в атрибуте именования, синтаксис контекста и синтаксис утверждения о контексте являются одинаковыми.

Имена RDN всех статей с определенным непосредственно предшествующим, старшим элементом являются индивидуальными независимо от любого связанного перечня контекстов. Ответственность за обеспечение этого путем надлежащего присвоения выделенных значений атрибутов лежит на соответствующем органе именования статьи. Распределение имен RDN рассматривается как административное мероприятие, которое может требовать или не требовать проведения переговоров между участвующими организациями или администрациями. В настоящей спецификации Справочника не представлен механизм такого согласования и не делается предположения о порядке его достижения.

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

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

Множество, которое образует имя RDN, содержит исключительно одно значение **AttributeTypeAndDistinguishedValue** для каждого атрибута, который содержит выделенные значения в статье; т. е. данный тип атрибута не может дважды появиться в том же RDN.

Значение атрибута, которое определено для появления в RDN, называется выделенным значением. Могут существовать другие значения того же атрибута, которые не являются выделенными значениями и, следовательно, не могут использоваться в RDN. Атрибут может иметь несколько выделенных значений, только если они различаются по связанным контекстам. Это позволяет объекту иметь альтернативные имена, различаемые по контекстам. Это единственный случай, когда атрибут может иметь более одного выделенного значения. В этом случае выделенные значения имеют перечни контекстов, содержащие тот же (те же) тип(ы) контекстов, значения контекстов которых обеспечивают применимость только одного из выделенных значений при любом конкретном контексте.

RDN для данной статьи формируется с использованием одного выделенного значения от каждого атрибута, имеющего выделенные значения. Простейшим случаем является статья, которая имеет одно выделенное значение; следовательно, она имеет одно RDN, формируемое с использованием этого выделенного значения. Входить в состав RDN могут более одного атрибута в статье. Если каждый входящий в состав атрибут имеет только одно выделенное значение, значит, статья имеет единственное RDN, формируемое с использованием выделенного значения каждого атрибута. Если какой-либо из входящих в состав атрибутов имеет несколько выделенных значений, отличающихся контекстами, тогда статья имеет несколько RDN, каждое из которых формируется с использованием одной из возможных комбинаций, в которых для каждого типа атрибута, формирующего RDN, выбирается одно выделенное значение.

Все RDN для статьи содержат пару тип **type** и значение **value** для каждого данного типа атрибута, формирующего часть RDN. Для указания того, что **value** является первичным выделенным значением этого типа атрибута, используется **primaryDistinguished**. **valuesWithContext** используется для переноса перечня контекстов для выделенного значения атрибута в **value** в случае необходимости такого переноса. Оно также используется для переноса в едином RDN некоторых или всех прочих выделенных значений того же типа атрибута. Каждое значение

distingAttrValue сопровождается своим перечнем контекстов **contextList**. **distingAttrValue** пропускается только в случае выделенного значения, которое содержится в значении **value**; таким образом обеспечивается присутствие перечня контекстов для этого значения в данном RDN.

Одно и только одно из выделенных значений для данного типа атрибута в статье рассматривается в качестве *первичного выделенного значения* для этого типа атрибута. Это значение используется как **value** в **AttributeTypeAndDistinguishedValue** при формировании первичного относительного выделенного имени данного объекта (см. пп. 9.8 и 9.6). *Первичное относительное выделенное имя* находится в RDN, в котором первичные выделенные значения для каждого атрибута в RDN имеются в компонентах **value** каждого **AttributeTypeAndDistinguishedValue** в RDN. Контекст и альтернативные выделенные значения могут присутствовать в компоненте **valuesWithContext** каждого **AttributeTypeAndDistinguishedValue.**

RDN может быть при необходимости изменено путем полной замены всех выделенных значений всех входящих в состав атрибутов.

Члены семейства, как и прочие статьи, имеют имена RDN. RDN может состоять из нескольких пар типа и значения атрибута. Использоваться могут только первичные RDN. *Локальное имя члена* члена семейства является последовательностью RDN от порождающего элемента вниз до этого члена. Локальное имя члена порождающего элемента – это пустая последовательность.

ПРИМЕЧАНИЕ 2. – Имена RDN предназначены для длительного существования, с тем чтобы пользователи Справочника могли сохранять выделенные имена объектов (например, в самом Справочнике), не беспокоясь о том, что они могут устареть. Вследствие этого следует с осторожностью подходить к изменению RDN.

ПРИМЕЧАНИЕ 3. – Изменение RDN не имеющей листьев статьи автоматически изменяет имя подчиненных статей.

ПРИМЕЧАНИЕ 4. – Контекст, в котором применимо формирование части RDN конкретными типом и значением атрибута, является не зависящим от контекстов, связных с другой частью этого RDN или другими RDN в выделенном имени.

ПРИМЕЧАНИЕ 5. – Например, действительное выделенное имя для статьи может быть сформировано сочетанием RDN, обозначенным как вариант Language = French (язык = французский) RDN этой статьи с выделенным именем Language = English (язык = английский) ее предшествующей, старшей статьи.

9.4 Сопоставление имен

В процессе функционирования Справочника часто возникает необходимость определить, совпадают ли два имени. Для этого требуется, чтобы совпадали соответствующие RDN. Здесь описан общий подход к сопоставлению имен; конкретные подходы для практического применения при сопоставлении имен описываются в соответствующих случаях.

Потенциальное RDN считается совпавшим с целевым RDN, если каждое значение **AttributeTypeAndDistinguishedValue** в потенциальном RDN совпадает с **AttributeTypeAndDistinguishedValue** для того же типа атрибута в целевом RDN. Совпадение происходит, если потенциальное **value** или любое **distingAttrValue** потенциального **AttributeTypeAndDistinguishedValue** совпадает либо с целевым **value** либо с любым **distingAttrValue** в целевом **AttributeTypeAndDistinguishedValue**. **primaryDistinguished**, если присутствует либо в потенциальном либо целевом **AttributeTypeAndDistinguishedValue**, для целей сопоставления игнорируется.

ПРИМЕЧАНИЕ 1. – Правило сопоставления равенства может применяться, поскольку для атрибутов именования синтаксис атрибутов и синтаксис утверждения по правилу сопоставления равенства являются одинаковыми.

ПРИМЕЧАНИЕ 2. – Не гарантируется, что все выделенные значения для данного атрибута именования обязательно присутствуют в **AttributeTypeAndDistinguishedValue** для этого типа атрибута в данном RDN. Два RDN для того же объекта могут быть сформированы с использованием разных выделенных значений (отличающихся контекстами) для того же типа атрибута. Если отсутствует наложение множеств выделенных значений для данного атрибута, которые используются каждым из них, то они не совпадают, даже если потенциальное RDN и целевое RDN являются альтернативными RDN для того же объекта. Каким образом это может произойти и какое окажет воздействие., зависит от цели сопоставления имен (например, разрешение имени, управление доступом, фильтрование).

Если в качестве результата вышесказанного значения атрибута совпадения не обнаруживается, значит, эти имена RDN не совпадают. Если значения атрибута совпадения обнаруживаются, то прежде чем рассматривать пары типа и значения атрибута совпадающими, должно быть также обнаружено совпадение между связанными контекстами для этих значений, если таковые существуют. Все контексты из перечня контекстов потенциального значения атрибута рассматриваются в качестве утверждения о контексте относительно перечня контекстов значения целевого атрибута совпадения, и для того чтобы контексты считались совпадающими, утверждение должно быть истинным, как описано в п. 8.9.2.4. При формировании оценок контекстов **fallback** в потенциальных контекстах игнорируется.

ПРИМЕЧАНИЕ 3. – Потенциальные контексты могут использоваться таким образом в качестве утверждений о контексте, поскольку синтаксис утверждения о контексте и синтаксис контекста для типов контекстов, которые могут использоваться с выделенными значениями, являются одинаковыми.

Если **valuesWithContext** не присутствует в потенциальном RDN, то утверждения о контексте предоставляются как часть операции, а также значения по умолчанию, которые установлены применимыми для операции, также являются применимыми, как это описано в п. 8.9.2.2. Исключением из этого является случай сопоставления имен в процессе разрешения имени, выполняемом в процессе какой-либо операции Справочника, в этом случае не применимы никакие утверждения о контексте, если в **valuesWithContext** не содержится ни одного утверждения.

9.5 Имена, возвращаемые в процессе выполнения операций

Многие операции Справочника возвращают имя статьи. Если операция возвращает имя статьи или имена нескольких статей, она должна возвращать первичное выделенное имя для каждой статьи и может кроме того возвращать информацию об альтернативных выделенных именах и информацию о контекстах (см. п. 7.7 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3).

9.6 Имена, хранимые как значения атрибутов или используемые как параметры

Если имя хранится как значение атрибута в пределах какого-либо иного атрибута или проходит в качестве значения атрибута в процессе какого-либо обмена (например, указатель псевдонимов), всегда возникает вопрос, может ли сохраняемое имя быть альтернативным выделенным именем или оно должно быть первичным выделенным именем, может ли оно содержать альтернативные выделенные значения, и может ли оно содержать контекстную информацию. Там где необходимо, в настоящих спецификациях Справочника упоминаются конкретные ограничения.

ПРИМЕЧАНИЕ. – В Приложении O содержатся предложения по совершенствованию функциональной совместимости с системами, соответствующими изданиям до третьего, и обеспечению прогнозируемого поведения при использовании контекстов с именами.

9.7 Выделенные имена

Выделенное имя данного объекта определяется как имя, содержащее последовательность имен RDN статьи, которая представляет объект, и имен RDN всех предшествующих, старших статей (в убывающем порядке). В силу взаимно-однозначного соответствия между объектами и статьями объектов выделенное имя объекта является выделенным именем статьи объекта.

ПРИМЕЧАНИЕ 1. – Предпочтительно, чтобы выделенные имена объектов, с которыми работают люди, были удобными для пользователей.

ПРИМЕЧАНИЕ 2. – В Рек. МСЭ-Т X.650 | ИСО/МЭК 7498-3 определяется понятие примитивного имени. Выделенное имя может использоваться как примитивное имя для объекта, который оно идентифицирует.

ПРИМЕЧАНИЕ 3. – Поскольку задействуются только статья объекта и ее предшествующие, старшие, выделенные имена объектов ни при каких условиях не могут включать в себя статьи псевдонима.

Статьи псевдонимов также имеют выделенные имена; однако эти имена не могут быть выделенными именами объектов. Если возникает необходимость в проведении такого различия, используется полный термин "выделенное имя статьи псевдонима". Выделенное имя статьи псевдонима определяется, аналогично определению выделенного имени статьи объекта, как последовательность имен RDN псевдонима и имен RDN всех предшествующих, старших статей (в убывающем порядке).

Удобным также является определение "выделенного имени" корня, хотя такое имя никогда не может быть выделенным именем объекта. Выделенное имя корня определяется как пустая последовательность.

Если какой-либо атрибут, входящий в состав RDN, в пределах выделенного имени объекта имеет несколько выделенных значений, отличающихся контекстами, значит, этот объект имеет несколько выделенных имен. Каждое однозначно идентифицирует объект. Первичное выделенное имя – это выделенное имя, для которого все RDN являются первичными RDN. При переносе в протоколе первичное выделенное имя формируется с использованием первичного выделенного значения в качестве **value** в **AttributeTypeAndDistinguishedValue** для каждого атрибута в каждом RDN, формирующем это имя. Альтернативные выделенные имена формируются с использованием альтернативных выделенных значений для атрибутов в одном или более RDN. В некоторых случаях использования имени применяется первичное выделенное имя. В других случаях могут использоваться альтернативные выделенные имена. Поскольку **AttributeTypeAndDistinguishedValue** в именах RDN могут включать альтернативные выделенные значения в компоненте **valuesWithContext**, любое выделенное имя может включать альтернативные значения в пределах своих RDN.

ПРИМЕЧАНИЕ 4. – Считается, что выделенное имя включает в себя альтернативные имена, если какое-либо RDN включает несколько выделенных значений для любого входящего в состав атрибута.

Контекстная информация может быть включена вместе с выделенным именем в компонент **valuesWithContext** в пределах любого RDN. При использовании имен в настоящих спецификациях Справочника указывается, является ли имя первичным выделенным именем, может ли имя включать альтернативные значения, и может ли быть включена контекстная информация. При отсутствии явного указания могут использоваться альтернативные выделенные значения, и это имя может включать альтернативные значения и/или контекстную информацию.

ПРИМЕЧАНИЕ 5. – Не является необходимым отображение для конечного пользователя требований, связанных с использованием в протоколе первичного выделенного имени вместо альтернативного выделенного имени.

Пример, иллюстрирующий понятия RDN и выделенного имени, представлен на рисунке 5.

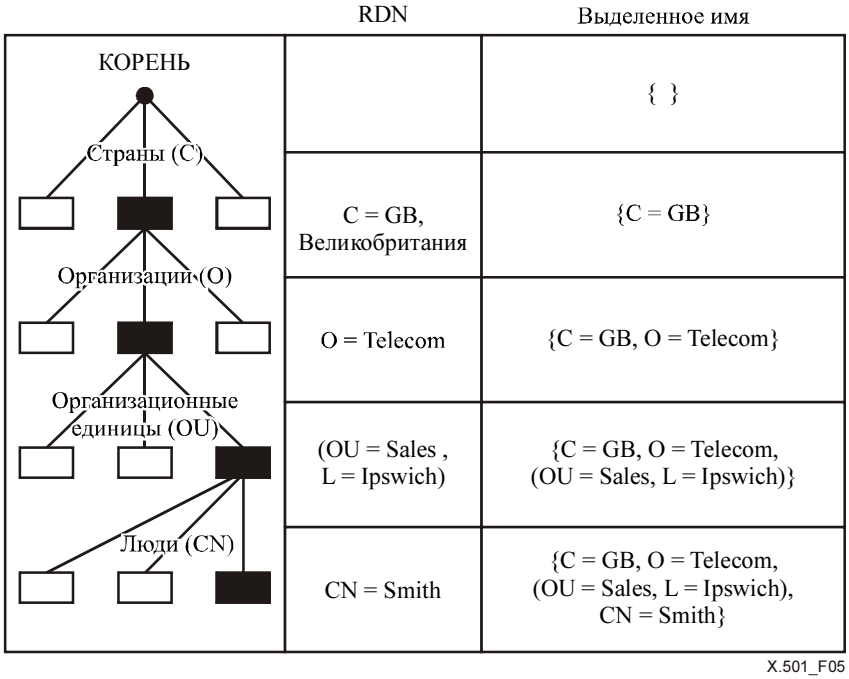


Рисунок 5 – Определение выделенных имен

9.8 Имена псевдонимов

Псевдоним или *имя псевдонима* для объекта является альтернативным именем для объекта или статьи объекта, которое предусмотрено использованием статей псевдонима.

Любая статьи псевдонима содержит в пределах атрибута **aliasedEntryName** имя некоторого объекта. Таким образом, выделенное имя статьи псевдонима также является именем этого объекта.

ПРИМЕЧАНИЕ 1. – Считается, что имя в пределах **aliasedEntryName** указывается псевдонимом. Оно не должно быть выделенным именем какого-либо объекта.

ПРИМЕЧАНИЕ 2. – Значение атрибута **AliasedEntryName** может быть первичным выделенным именем или любым альтернативным выделенным именем, если таковое существует. Согласованность и межсетевое взаимодействие с DSA предварительного третьего издания могут быть нарушены, если не будет использоваться первичное выделенное имя.

Преобразование имени псевдонима в имя объекта называется "разыменованием" (псевдонима) и заключается в семантической замене имен псевдонимов, которые обнаруживаются в пределах потенциального имени, на значение соответствующего атрибута **aliasedEntryName**. Это процесс может потребовать анализа более одной статьи псевдонима.

Любая статья в DIT может иметь нуль или более имен псевдонимов. Отсюда следует, что несколько статей псевдонимов могут указывать на ту же статью. Статья псевдонима может указывать на статью, которая не является статьей – листом дерева, и может указывать на другую статью псевдонима.

Статья псевдонима не должна иметь подчиненных, т. е. статья псевдонима – всегда статья – лист дерева.

Все статьи псевдонимов должны принадлежать классу объектов **alias**, что определено в п. 13.3.3.

Не разрешается, чтобы члены семейства были статьями псевдонимов.

10 Иерархические группы

10.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

10.1.1 иерархический дочерний элемент: Для статьи иерархический дочерний элемент – это статья, по отношению к которой данная статья является иерархическим родительским элементом.

10.1.2 иерархическая группа: Иерархическая группа – это набор статей, включая составные статьи, образующие логическое дерево, которое необязательно имеет отношение к DIT.

10.1.3 иерархический лист дерева: Это статья в пределах иерархической группы, не имеющая иерархического дочернего элемента.

10.1.4 иерархический уровень: Целое число, которое показывает расстояние от какой-либо статьи в пределах иерархической группы до иерархической вершины в форме количества иерархических связей между этой статьей и иерархической вершиной.

10.1.5 иерархическая связь: Это общий термин для указания логической взаимосвязи между двумя статьями, которые имеют логическое взаимоотношение непосредственно родительский элемент/непосредственно дочерний элемент.

10.1.6 иерархический родительский элемент: Для статьи иерархическими родительскими элементами являются непосредственно родительский элемент, его непосредственно родительский элемент, рекурсивно все по ходу вверх, включая иерархическую вершину.

10.1.7 иерархический братский элемент: Для статьи иерархическими братскими элементами являются статьи, имеющие тот же непосредственно иерархический родительский элемент, что и данная статья.

10.1.8 иерархический братский-дочерний элемент: Для статьи его иерархическими братскими-дочерними элементами является полное множество иерархических дочерних элементов, находящихся на всех более низких уровнях, его иерархических братских элементов.

10.1.9 иерархическая вершина: Это статья в пределах иерархической группы, которая является корнем иерархии. Иерархическая вершина не имеет непосредственно иерархического родительского элемента.

10.1.10 непосредственно иерархический дочерний элемент: Для статьи непосредственно иерархический дочерний элемент – это статья, по отношению к которой данная статья является непосредственно иерархическим родительским элементом. Непосредственно иерархический дочерний элемент необязательно должен быть непосредственно подчиненной статьей в пределах DIT.

10.1.11 непосредственно иерархический родительский элемент: Для статьи ее непосредственно иерархический родительский элемент – это статья, которая в пределах данной иерархический группы является ее непосредственно предшествующей, старшей статьей. Непосредственно иерархический родительский элемент необязательно должен быть непосредственно предшествующей, старшей статьей в пределах DIT.

10.2 Иерархическая взаимосвязь

Между статьями Справочника существует иерархическая взаимосвязь, в соответствии с которой они располагаются в DIT. Вместе с тем, статьи могут иметь иерархические взаимосвязи, которые не отражаются в структуре DIT. Например, имеющая динамическую структуру организация может пожелать не отражать свою текущую структуру непосредственно в DIT, поскольку вероятной станет необходимость частых изменений структуры DIT. Таким образом, в Справочнике действует требование обеспечения возможности отображения иерархических взаимосвязей независимо от структуры DIT. Такие взаимосвязи формируют *иерархические группы.* Иерархическая группа образует логическое дерево с вершиной, которая называется *иерархической вершиной.*

Делая ссылки на иерархическую взаимосвязь, с помощью операции поиска Search возможно извлекать информацию не только из данной статьи, но и из других статей в пределах той же иерархической группы.

Составная статья в контексте иерархических групп рассматривается как единая статья. Дочерний член семейства не может самостоятельно быть частью иерархической группы.

ПРИМЕЧАНИЕ. – Назначением иерархических групп является обеспечение возможности моделирования наборов самостоятельных объектов, между которыми существуют логически неформальные взаимоотношения и, в частности, взаимоотношения, которые являются или могут являться временными. Составные статьи, наоборот, моделируют объекты, которые включают в себя подобъекты, которые традиционно рассматриваются как иерархические.

Для описания навигации в пределах иерархической группы удобно определить термины для обозначения взаимосвязей, которые действуют между данной статьей и другими статьями в пределах данной группы. Это сделано в п. 10.1. Некоторые из определенных в этом пункте терминов для прямых взаимосвязей аналогичны определенным для взаимосвязей между статьями в пределах DIT (*непосредственно иерархический дочерний элемент*, *иерархический дочерний элемент*, *непосредственно иерархический родительский элемент* и *иерархический родительский элемент*). Вместе с тем, удобно также определить термины для более отдаленных отношений. В некоторых ситуациях пользователь может пожелать найти информацию для *иерархических братских элементов* и даже для их иерархических дочерних элементов (*иерархических братских-дочерних элементов*).

Одномоментно статья может быть членом единственной иерархической группы.

Статья, которая является частью иерархической группы, содержит операционные атрибуты, как это описано в п. 14.9. Эти операционные атрибуты отражают взаимосвязь с другими статьями в пределах данной группы, включая *иерархический уровень* этой статьи в пределах данной группы. Если частью иерархической группы является составная статья, эти операционные атрибуты содержит порождающий элемент.

Иерархическая группа должна быть полностью за пределами административной области, зависящей от службы (см. п. 16.3), или должна полностью входить в административную область, зависящую от службы. Иерархическая группа должна быть ограничена единственным DSA. Служба Справочника должна обнаруживать и предотвращать попытки нарушения этих правил.

10.3 Последовательное упорядочение иерархической группы

В ряде ситуаций, например при передаче иерархической группы, требуется правило последовательного упорядочения. Последовательное упорядочение иерархической группы осуществляется путем следования по всем цепочкам иерархической группы следующим образом:

a) Статья-вершина является первой статьей в последовательности, за которой следуют остальные статьи в пределах полной цепочки по направлению вниз от вершины до иерархического листа. Выбор цепочки, которая станет первой, является вопросом местной компетенции.

b) Следующей выбирается цепочка, которая до этого не выбиралась и которая имеет максимальное количество статей, общих с предыдущей выбранной цепочкой. Если несколько статей по этому признаку идентичны, выбор является вопросом местной компетенции. В последовательность включаются только те статьи, которые не включались ранее.

c) Процедура, описанная в пункте b), повторяется до включения всех цепочек.

РАЗДЕЛ 4 – АДМИНИСТРАТИВНАЯ МОДЕЛЬ СПРАВОЧНИКА

11 Модель административного органа Справочника

11.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

11.1.1 административная область: Поддерево DIT, рассматриваемое с точки зрения административного управления.

11.1.2 административная статья: Статья, расположенная в административной точке.

11.1.3 административная точка: Корневая вершина административной области.

11.1.4 административный пользователь: Представитель административного органа. Полное определение понятия административного пользователя не входит в область применения настоящей спецификации Справочника.

11.1.5 автономная административная область: Поддерево DIT, административное управление всеми статьями которого осуществляется тем же административным органом. Автономные административные области не перекрываются.

11.1.6 административный орган домена DIT: Административный орган в роли элемента, ответственного за административное управление частью DIT.

11.1.7 стратегия в домене DIT: Общие цели и применимые процедуры для домена DIT.

11.1.8 административный орган DMD: Административный орган в роли элемента, ответственного за административное управление в DMD.

11.1.9 стратегия в DMD: Стратегия, управляющая функционированием агентов DSA в DMD.

11.1.10 стратегия DMO: Стратегия, которую определяет DMO, сформулированная в терминах стратегии в домене DIT и стратегии в DMD.

11.1.11 внутренняя административная область: Определенная административная область, сфера действия которой полностью входит в сферу действия другой определенной административной области того же типа.

11.1.12 стратегия: Формулирование административным органом общих целей и применимых процедур.

11.1.13 атрибут стратегии: Обобщенный термин для любого операционного атрибута Справочника, который выражает стратегию.

11.1.14 объект стратегии: Элемент, на который направлена стратегия.

11.1.15 процедура осуществления стратегии: Правило, которое определяет, как должна рассматриваться совокупность объектов стратегии и какие действия должны быть предприняты в результате этого рассмотрения.

11.1.16 параметр стратегии: Процедура осуществления стратегии описывается определенными *параметрами стратегии*, которые зависят от конфигурации (т. е. выбора), устанавливаемой административным органом.

11.1.17 специальная административная область: Подмножество (в форме поддерева) автономной административной области, определенное для конкретного аспекта административного управления: управление доступом, административное управление подсхемой или набором статей. Получив определение, специальные административные области *определенного вида* разделяют на части автономную административную область.

11.1.18 специальная административная точка: Корневая вершина специальной административной области.

11.2 Обзор

Основополагающей задачей информационной модели Справочника является рассмотрение строго определенных наборов статей таким образом, чтобы к ним можно было бы применять единообразное административное управление как к отдельной единице. В данном пункте разъясняются характер и область действия органов, ответственных за такое административное управление, и методы реализации их полномочий.

Понятие стратегии, определенное в п. 11.3, обеспечивает механизм, с помощью которого административные органы осуществляют управление справочником.

Ряд аспектов административной модели Справочника поддерживаются моделью административной и операционной информации Справочника (см. пункт 12). Это позволяет выполнять моделирование информации, необходимое для регулирования пользовательской информации Справочника и для других целей административного характера.

Другие аспекты административной модели Справочника требуют поддержки для распределения административной и операционной информации среди составных частей Справочника, т. е. DSA. В пп. 22–24 описывается информационная модель для удовлетворения этих требований.

11.3 Стратегия

Стратегия – это формулирование административным органом, действующим в качестве агента DMO, общих целей и применимых процедур. Стратегия определяется в форме правил, выполнение которых должно обеспечиваться (Справочником, в соответствующих случаях), и в форме аспектов, в пределах которых административный пользователь имеет определенную степень свободы действий и конкретные обязательства.

Административный орган формулирует стратегию в DMO в терминах:

- стратегии в домене DIT;
- стратегии в DMD.

Эти стратегии могут быть выражены как атрибуты стратегии. Модель стратегий в DIT определяется в п. 11.6.

ПРИМЕЧАНИЕ. – В п. 14 определяется схема системы, необходимая для поддержки административного управления коллективными атрибутами. В п. 15 определяется основа для поддержки стратегии административного управления подсхемами. В п. 17 определяется основа для поддержки стратегии управления доступом.

Стратегия в DMD относится непосредственно к агентам DSA как компонентам распределенного Справочника. Различные стратегии в DMD описываются в п. 11.7, в котором определяется модель для административного управления DSA.

Наконец, существуют стратегии, которые касаются внешних вопросов (таких как двусторонние соглашения между DMO), и, в силу этого, не описываются более подробно в данном документе.

Объектом стратегии является элемент, на который направлена (например, административная область подсхемы является объектом стратегии) стратегия.

Процедурой осуществления стратегии является правило, которое определяет, как должна рассматриваться совокупность объектов стратегии и какие действия должны быть предприняты (и при каких обстоятельствах) в результате этого рассмотрения. (например, в п. 15 определяются процедуры осуществления стратегии административного управления подсхемой).

Процедура осуществления стратегии описывается определенными *параметрами стратегии*, которые зависят от конфигурации (т. е. выбора), устанавливаемой административным органом.

Для представления параметров стратегии используются операционные атрибуты. Значения такого атрибута составляют выражение части или всего параметра стратегии, который они представляют.

11.4 Специальные административные органы

Административное управление доменом DIT включает выполнение пяти функций, связанных с разными аспектами административного управления:

- административное управление именованием;
- административное управление подсхемой;
- административное управление безопасностью;
- административное управление коллективными атрибутами;
- административное управление службами.

Специальный административный орган – это административный орган в роли элемента, ответственного за один из этих конкретных аспектов стратегии в домене DIT.

Термин *орган именования* (см. п. 9) определяет функцию данного административного органа как связанную с распределением имен и административным управлением структурой этих имен. Функция органа подсхемы заключается реализации этих структур имен в подсхеме.

Термин *орган подсхемы* определяет функцию данного административного органа как связанную с установлением, административным управлением и осуществлением стратегии в отношении подсхемы при управлении именованием и содержанием статей в домене DIT. В п. 15 описывается, как Справочник поддерживает административное управление подсхемой.

Термин *орган безопасности* (см. Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8) определяет функцию данного административного органа как связанную с установлением, административным управлением и выполнением стратегии обеспечения безопасности, которая управляет поведением Справочника в отношении статей в каком-либо домене DIT.

Термин *орган коллективных атрибутов* определяет функцию данного административного органа как связанную с установлением и административным управлением коллективными атрибутами (см. п. 12.7) в домене DIT.

Термин *орган службы* определяет функцию данного административного органа как связанную с установлением и административным управлением ограничениями и регулированием службы.

11.5 Административные области и административные точки

11.5.1 Автономные административные области

Административное управление любой статьей в DIT осуществляется исключительно одним административным органом (который может выполнять различные функции). *Автономная административная область* – это поддерево DIT, административное управление всеми статьями которого осуществляется тем же административным органом.

Домен DIT может быть разбит на одну или более неперекрывающихся автономных административных областей.

Множество, состоящее из одной или более автономных административных областей, в отношении которых DMO имеет административные полномочия, является доменом DIT этой DMO. Это показано на рисунке 6.

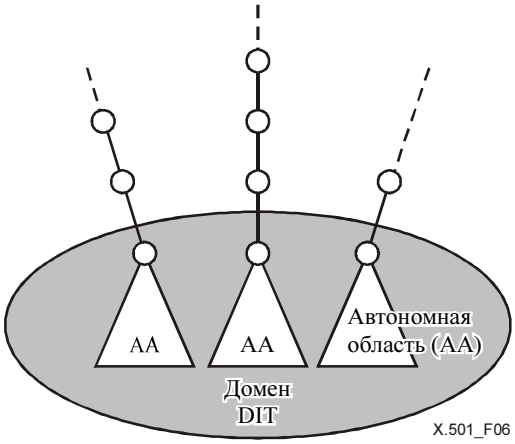


Рисунок 6 – Домен DIT

11.5.2 Специальные административные области

Аналогично тому, как административный орган может выполнять специальную функцию, статьи в административной области могут рассматриваться в терминах специальной административной функции. В таком контексте административная область называется *специальной административной областью*. Существуют пять видов специальной административной области:

- административные области подсхемы;
- административные области управления доступом;
- административные области коллективных атрибутов;
- административные области контекстов по умолчанию;
- административные области служб.

Автономная административная область может рассматриваться как определенная явным образом единичная специальная административная область для каждого конкретного аспекта административного управления. В этом случае существует точное соответствие между каждой специальной административной областью и автономной административной областью.

Альтернативно, для каждого конкретного аспекта административного управления автономная административная область может быть разбита на неперекрывающиеся специальные административные области.

При таком разбиении по конкретному аспекту административного управления каждая статья автономной административной области содержится в одной и только одной специальной административной области этого аспекта.

Специальный административный орган несет ответственность за каждую специальную административную область. Если, в отношении конкретного административного аспекта автономная административная область не разбита на части, специальный административный орган несет ответственность за этот административный аспект по всей автономной административной области.

11.5.3 Внутренние административные области

Для целей управления безопасностью или коллективными атрибутами в пределах административных областей этих видов могут определяться *внутренние (административные) области*:

a) для представления ограниченной формы делегирования; или

b) для удобства в административном или операционном отношении (например, если административная точка поддерева находится не в том DSA, который содержит статьи данного поддерева, это поддерево может быть обозначено как внутренняя область для разрешения административного управления через местный DSA).

Внутренняя административная область может располагаться в рамках другой внутренней административной области.

Внутренние области представляют области ограниченной автономии. Административное управление статьями во внутренней области осуществляется специальными административными органами специальных административных областей, в которых они содержатся, а также административными органами внутренних областей, в которых они содержатся. Эти первые упомянутые административные органы полностью управляют стратегией, регулирующей данные статьи, в то время как последние упомянутые органы имеют (ограниченный) контроль по тем аспектам стратегии, которые им делегированы первыми.

Правила для вложенных внутренних областей, если они разрешены, должны определяться как часть определения конкретного административного аспекта, в пределах которого они находятся.

11.5.4 Административные точки

Определение протяженности автономной административной области является неявно выраженным и содержит определение точки в DIT (корень поддерева автономной административной области), *автономной административной точки*, в которой эта административная область начинается и от которой она распространяется вниз до другой автономной административной точки, в которой начинается другая автономная область.

ПРИМЕЧАНИЕ 1. – Непосредственно подчиненные по отношению к корню DIT являются автономными административными точками.

Если автономная административная область *не разбита* на части для конкретного аспекта административного управления, то административная область для этого аспекта совпадает с автономной административной областью. В этом случае автономная административная точка также является *специальной административной* точкой для этого аспекта административного управления.

Если автономная административная область *разбита* на части для конкретного аспекта административного управления, то определение протяженности каждой специальной административной области содержит обозначение корня поддерева специальной административной области, *специальную административную точку*, в которой эта специальная административная область начинается и от которой она распространяется вниз до другой специальной административной точки (того же административного аспекта), в которой начинается другая специальная административная область.

Специальные административные области всегда ограничены автономной административной областью, частями которой они являются.

Отдельная административная точка может быть корнем какой-либо автономной административной области и может быть корнем одной или более специальных административных областей.

Определение протяженности внутренней административной области (в пределах специальной административной области) содержит обозначение корня поддерева внутренней административной области, *внутреннюю административную точку*. Внутренняя административная область ограничена специальной административной областью, в пределах которой она определена.

Административная точка, соответствующая корню автономной административной области, представляет границу области DIT (и DSA). Это значит, что непосредственно предшествующая, старшая в DIT должна находиться под управлением административного органа другой DMD.

ПРИМЕЧАНИЕ 2. – Это означает, что DMO не может разбивать домен DIT на автономные административные области произвольным образом.

Любая административная точка в информационной модели Справочника представлена статьей, содержащей атрибут **administrativeRole**. Значения этого атрибута указывают тип административной точки. Этот атрибут определяется в п. 14.3.

В пунктах 22–24 описывается, как административные области отображаются в агентах DSA, и информационная модель DSA.

На рисунке 7 показана автономная административная область, которая разбита на две специальные административные области для конкретного аспекта административного управления (например, управление доступом). В одной специальной административной области создана вложенная внутренняя административная область (например, поскольку данное поддерево должно содержаться в другом DSA, чем остальная часть специальной административной области).

На рисунке 7 используются следующие сокращения: AAP (автономная административная точка), SAP (специальная административная точка) и IAP (внутренняя административная точка).

Рисунок 7 – Административные точки и области

11.5.5 Административные статьи

Статья, расположенная в административной точке, является *административной статьей*. Административные статьи могут обладать, в качестве непосредственно подчиненных, специальными статьями, называемыми *подстатьями*. Административная статья и связанные с ней подстатьи используются для управления статьями, заключенными в связанной административной области.

При использовании внутренних административных областей сферы действия этих областей могут перекрываться.

Следовательно, для каждого конкретного аспекта административного полномочия требуется определение метода комбинирования административной информации, в случае если возможно включение статей в более чем одно поддерево или уточнение поддерева, связанные с внутренней областью, определенной для этого аспекта.

ПРИМЕЧАНИЕ. – Для административной точки необязательно представлять каждый конкретный аспект административного органа. Например, может существовать административная точка, подчиненная по отношению к корню автономной административной области, которая используется исключительно для целей управления доступом.

11.6 Стратегия в домене DIT

Стратегия в домене DIT содержит следующие компоненты: объекты стратегии в DIT, процедуры осуществления стратегии в DIT и параметры стратегии в DIT.

Операционный атрибут, который представляет параметр стратегии в DIT, *называется атрибутом стратегии в DIT* (например, административные операционные атрибуты подсхемы, определенные в пункте 14, являются атрибутами стратегии в домене DIT).

Для отдельного DSA возможные значения параметра стратегии могут не соответствовать отдельным, реализуемым образам действия для этого компонента. Это может произойти, например в том случае, если DSA не имеет технической возможности выполнять все аспекты процедуры стратегии (например, реализация конкретной схемы управления доступом). Процедура осуществления стратегии, для того чтобы быть строго определенной, должна учитывать эти обстоятельства как часть своего определения.

Конкретные объекты и атрибуты стратегии в домене DIT определяются в п. 15 в целях поддержки административного управления подсхемой.

11.7 Стратегия в DMD

Стратегия в DMD – это стратегия, связанная с функционированием одного или более DSA в DMD. Стратегия в DMD может применяться либо ко всем DSA в DMD единообразно, либо к подмножеству DSA в DMD, или же она может применяться к одному конкретному DSA.

Одним из видов стратегии в DMD является ограничение или в иное управление абстрактной службой Справочника и DSA, обеспечиваемой одним или более DSA.

Примерами таких ограничений являются:

a) Ограничение базовой службы, предоставляемой пользователям (не административным) Справочника, только операциями опроса в соответствии с Рек. МККТТ F.500.

b) Ограничение службы, предоставляемой пользователям, которые осуществляют непрямой доступ к DSA по цепочке, в том числе проведение различия в зависимости от того, пересекает ли запрос пользователя истинную траекторию.

c) Ограничение запросов, получаемых от пользователей, который осуществляют прямой доступ к DSA, когда требуется формирование цепочки к агентам DSA в DMD, определяемое в зависимости от ограничений того типа, который указан в предыдущей точке.

d) Ограничения на виды операций поиска, которые могут выполнять некоторые пользователи, и на характеристики таких операций поиска (например, стратегия ослабления).

РАЗДЕЛ 5 – МОДЕЛЬ АДМИНИСТРАТИВНОЙ И ОПЕРАЦИОННОЙ ИНФОРМАЦИИ СПРАВОЧНИКА

12 Модель административной и операционной информации Справочника

12.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

12.1.1 база: Корневая вершина поддерева или уточнения поддерева, являющегося результатом оценки спецификации поддерева.

12.1.2 усечение: Совокупность утверждений, связанных с именами подчиненных по отношению к базе.

12.1.3 операционный атрибут справочника: Операционный атрибут, определенный и видимый в модели административной и операционной информации Справочника.

12.1.4 схема системы справочника: Совокупность правил и ограничений, касающихся операционных атрибутов и подстатей.

12.1.5 статья: Статья Справочника или расширенная статья Справочника – в зависимости от контекста (либо пользователи и их приложения или администрация и функционирование Справочника), в котором используется этот термин.

12.1.6 подстатья: Особый вид статьи, известный Справочнику, который используется для содержания информации, связанной с поддеревом или уточнением поддерева.

12.1.7 поддерево: Набор статей объектов и псевдонимов, находящихся на вершинах какого-либо дерева. Приставка "под" специально указывает на то, что база (или корень) вершины данного дерева, как правило, является подчиненной по отношению к корню DIT.

12.1.8 уточнение поддерева: Определенное явным образом подмножество статей в поддереве, в котором статьи не располагаются на вершинах единственного поддерева.

12.1.9 спецификация поддерева: *Явная* спецификация поддерева или уточнения поддерева. Спецификация поддерева состоит из нуля или более баз элементов спецификации, усечения и фильтров спецификации. Определение называется "явным" (в отличие от определения для административной области), поскольку часть подчиненного по отношению к базе DIT, которая включается в данное поддерево или уточнение поддерева, определяется явным образом.

12.2 Обзор

С административной точки зрения пользовательская информация, хранимая в DIB, дополняется административной и операционной информацией, выраженной:

- *операционными атрибутами*, которые представляют информацию, используемую для управления функционированием Справочника (например, информация по управлению доступом) или используемую Справочником для обозначения какого-либо аспекта своего функционирования (например, информация об отметках времени); и
- *подстатьями*, которые связывают значения совокупности атрибутов (например, коллективных атрибутов) со статьями в пределах сферы действия данной подстатьи. Сферой действия подстатьи является поддерево или уточнение поддерева.

Эта информация, графически представленная на рисунке 8, может размещаться в Справочнике административными органами или агентами DSA и используется Справочником в процессе функционирования.

К этой стороне информации Справочника относятся два следующих механизма абстрактной службы Справочника:

- **EntryInformationSelection** был расширен для разрешения выбора операционных атрибутов в статье; и
- была добавлена управляющая функция службы **subentries** для разрешения применения операций составления списков List и поиска Search либо к статьям объектов и псевдонимов, либо к подстатьям.

Доступ к операционной информации, аналогично доступу к пользовательской информации, может быть ограничен механизмом управления доступом.

Статьи становятся видимыми для пользователей Справочника через абстрактную службу Справочника, но их взаимосвязи с агентами DSA, которые в итоге содержат эти статьи, остаются невидимыми для пользователей. Модель информации DSA, описанная в пп. 22–24, показывает отображение этих статей в информационных хранилищах агентов DSA.

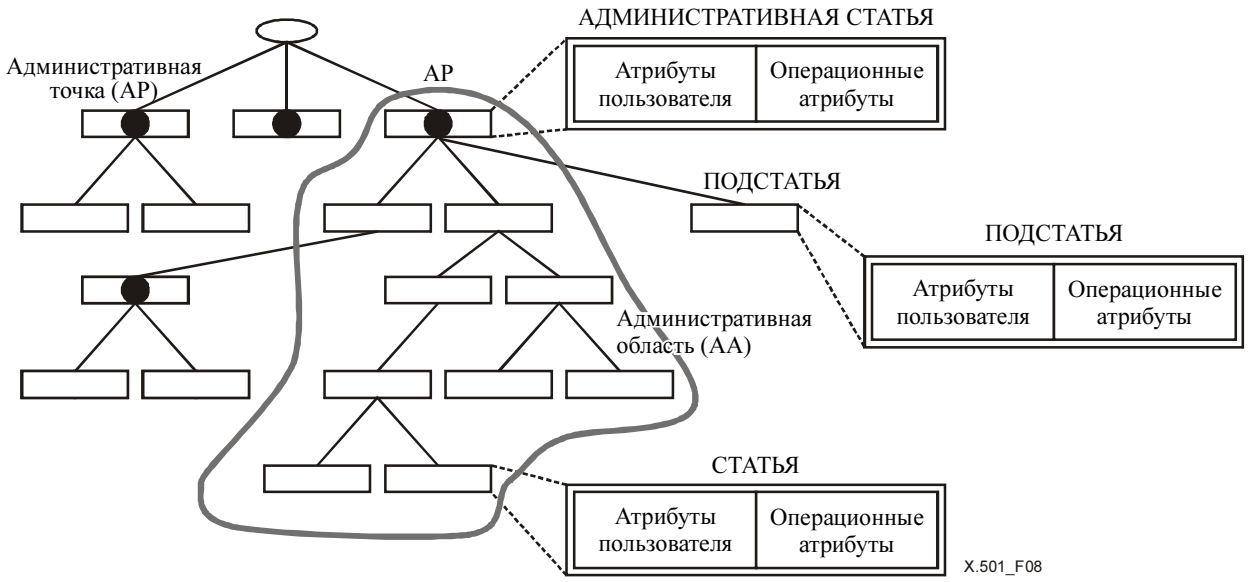


Рисунок 8 – Модель административной и операционной информации Справочника

12.3 Поддеревья

12.3.1 Обзор

Поддерево – это набор статей объектов и псевдонимов, расположенных на вершинах дерева. Поддеревья не содержат подстатей. Приставка "под" в слове поддерево специально указывает на то, что база (или корень) вершины данного дерева, как правило, является подчиненным по отношению к корню DIT.

Поддерево начинается в какой-либо вершине и простирается до некоторой идентифицируемой нижней границы, и может простираться до листьев. Поддерево всегда определяется в контексте, который неявно указывает границы этого поддерева. Например, вершина и нижняя граница поддерева, определяющего копируемую область, указываются контекстом именования. Аналогичным образом, сфера действия поддерева, определяющего специальную административную область, ограничивается контекстом охватывающей автономной административной области.

12.3.2 Спецификация поддерева

Спецификацией поддерева является определение подмножества статей, находящихся ниже конкретной вершины, которая образует базу поддерева или уточнения поддерева.

Вершина и/или нижняя граница поддерева могут указываться неявно, и в этом случае они определяются контекстом, в котором используется это поддерево.

Вершина и/или нижняя граница могут быть определены явно с помощью механизма, описываемого в данном пункте. Этот механизм может использоваться также для спецификации уточнений поддерева, которые не являются истинными древовидными структурами.

ПРИМЕЧАНИЕ. – При рассмотрении таких спецификаций полезно использовать топологическое представление (под)дерева, хотя конкретная спецификация может описывать набор статей, которые *не* располагаются на вершинах одного (под)дерева. При таком расположении статей предпочтительнее использовать термин *уточнение поддерева*.

Спецификация поддерева состоит из трех необязательных элементов спецификации, которые указывают базу поддерева и затем сокращают набор подчиненных статей. Этими тремя элементами спецификации являются:

a) *база* – корневая вершина поддерева или уточнения поддерева, являющегося результатом оценки спецификации поддерева;

b) *усечение* – совокупность утверждений, касающихся имен подчиненных статей; и

c) *фильтр спецификации* – строгое подмножество оценочных возможностей фильтра, применяемое к этим подчиненным.

Спецификация поддерева или уточнения поддерева может быть представлена следующей записью ASN.1:

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

    -- пустая последовательность определяет целую административную область
```

Три компонента этой последовательности соответствуют трем определенным выше элементам спецификации.

Если значение **SubtreeSpecification** определяет набор статей, которые расположены на вершинах единственного поддерева, этот набор называется "поддерево", в противном случае этот набор называется "уточнение поддерева".

Тип **SubtreeSpecification** обеспечивает универсальный механизм для спецификации поддеревьев и уточнений поддеревьев. В зависимости от конкретного случая применения этого механизма определяется конкретная семантика исключительно того, что определяется, и на компоненты **SubtreeSpecification** могут налагаться пределы или ограничения.

Если отсутствуют все компоненты **SubtreeSpecification** (т. е. значение типа **SubtreeSpecification** является пустой последовательностью, **{}**), описанное таким образом поддерево неявно определяется контекстом, в котором используется **SubtreeSpecification**.

Данные термины графически представлены на рисунке 9 для случая, когда поддеревья размещаются в контексте административных областей.

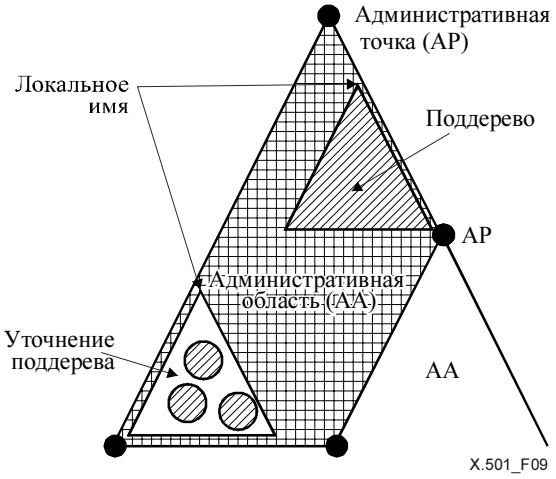


Рисунок 9 – Спецификация поддеревьев и уточнений поддеревьев в среде административных областей

12.3.3 База

Компонент **base** спецификации **SubtreeSpecification** представляет корневую вершину поддерева или уточнения поддерева. Это может быть статья, являющаяся подчиненной по отношению к корневой вершине указанной сферы действия, или это может быть сама корневая вершина указанной сферы действия (вариант по умолчанию).

Относительным именем корневой вершины поддерева по отношению к корневой вершине указанной сферы действия является значение типа **LocalName**:

```
LocalName ::= RDNSequence
```

Следует заметить, что корневая вершина указанной сферы действия и корневая вершина поддерева совпадают, если в **SubtreeSpecification** опущено **LocalName**.

Имена RDN, которые используются для именования корневой вершины поддерева, должны быть первичными RDN.

12.3.4 Спецификация усечения

Компонент спецификации усечения **ChopSpecification** состоит из совокупности утверждений, касающихся имен подчиненных базы. Его составляет значение типа **ChopSpecification**:

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

Этот тип предназначен для того, чтобы разрешить составление спецификации структуры дерева (или его подмножества), начинающейся в базе, двумя способами: специальные исключения и расстояние от базы.

Если какой-либо атрибут в RDN в **chopBefore** или **chopAfter** имеет несколько выделенных значений, отличающихся контекстами, в качестве **value** в данном RDN в **LocalName** должно использоваться первичное выделенное значение.

12.3.4.1 Специальные исключения

Компонент специальных исключений **specificExclusions** имеет две формы – **chopBefore** и **chopAfter**, которые могут использоваться по отдельности или в сочетании.

Компонент **chopBefore** определяет перечень исключений, обозначаемых в форме некоей предельной точки, которая должна быть исключена вместе с ее подчиненными из данного поддерева или уточнения поддерева. Предельными точками являются статьи, обозначаемые **LocalName** относительно базы.

Компонент **chopAfter** определяет перечень исключений, обозначаемых в форме некоей предельной точки, подчиненные которой должны быть исключены из данного поддерева или уточнения поддерева. Предельными точками являются статьи, обозначаемые **LocalName** относительно базы.

12.3.4.2 Компоненты минимум и максимум

Эти компоненты разрешают исключение всех статей, являющихся предшествующими, старшими по отношению к статьям, которые являются дугами RDN **minimum** ниже базы, а также статей, являющихся подчиненными по отношению к статьям, которые являются дугами RDN **maximum** ниже базы. Эти расстояния указываются значениями типа **BaseDistance**:

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

Для цели спецификации усечения составная статья учитывается как единая статья. В составной статье все члены семейства учитываются как имеющие то же расстояние от базы, что и порождающий элемент, поскольку они все являются частью той же логической статьи.

Значение **minimum**, равное нулю (значение по умолчанию), соответствует базе. Отсутствие компонента **maximum** указывает, что на поддерево или уточнение поддерева не налагается каких-либо нижних ограничений.

12.3.5 Фильтр спецификации

Компонент фильтра спецификации **specificationFilter** состоит из строгого подмножества оценочных возможностей фильтра (см. Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3), применяемых к подчиненным базы. Только те статьи, для которых произведенная фильтром оценка является истиной, включаются в результирующее уточнение поддерева. Его составляет значение типа **Refinement**:

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

Результатом оценки **Refinement** является истина TRUE, как если бы он являлся фильтром для утверждения равенства **equality** в отношении значений только атрибутов типа **objectClass**.

Если какой-либо член семейства исключается из поддерева этой спецификацией, исключаются также все его подчиненные члены семейства.

12.4 Операционные атрибуты

Существуют три разновидности операционных атрибутов: операционные атрибуты Справочника, операционные атрибуты, коллективно используемые агентами DSA, и операционные атрибуты, зависящие от DSA.

Операционные атрибуты Справочника входят в информационную модель Справочника и используются для представления управляющей информации (например, информации по управлению доступом) или иной информации, предоставляемой Справочником (например, индикация того, является ли статья статьей-листом или статьей-не листом).

Операционные атрибуты, *коллективно используемые агентами DSA*, входят только в информационную модель DSA и являются абсолютно невидимыми во всех информационных моделях Справочника.

Операционные атрибуты, *зависящие от DSA*, входят только в информационную модель DSA и являются абсолютно невидимыми во всех информационных моделях Справочника.

ПРИМЕЧАНИЕ. – Эти атрибуты описываются в пп. 23–24.

Определение и использование всех операционных атрибутов является предметом спецификации в составе соответствующей спецификации Справочника.

12.5 Статьи

12.5.1 Обзор

С административной точки зрения пользовательская информация, содержащаяся в статье, может быть дополнена административной и операционной информацией, которую представляют операционные атрибуты.

Справочник использует атрибут данного класса объектов и правила контента DIT, применимые к статье, для управления атрибутами пользователя, требуемыми и разрешенными в этой статье. Управление операционными атрибутами статьи осуществляет схема системы Справочника (см. п. 14), применимая к этой статье.

12.5.2 Доступ к операционным атрибутам

Обычно не видимые операционные атрибуты справочника в пределах статей можно сделать видимыми для уполномоченных (например, административных) пользователей абстрактной службы Справочника. Некоторые операционные атрибуты (например, **entryACI** или **modifyTimestamp**) могут быть также доступны обычным пользователям.

12.6 Подстатьи

12.6.1 Обзор

Подстатья – это особый вид статьи, которая является непосредственно подчиненной по отношению к административной точке. Она содержит атрибуты, имеющие отношение к поддереву (или уточнению поддерева), связанному с ее административной точкой. Подстатьи и их административная точка составляют часть того же контекста именования (см. п. 21).

Одна подстатья может обслуживать все или несколько аспектов административных полномочий. Альтернативным образом, конкретный аспект административных полномочий может обрабатываться одной или несколькими из собственных подстатей. Для административных полномочий подсхемы разрешена максимум одна подстатья. Органы управления доступом и коллективных атрибутов могут иметь несколько подстатей.

Подстатья не рассматривается в операциях составления списков и поиска, если только в запрос не включена управляющая функция службы **subentries**.

Подстатья не должна иметь подчиненных.

Структура подстатьи, соответствующей административной точке, показана на рисунке 10.

Рисунок 10 – Структура подстатьи

Подстатью составляют:

- атрибут **commonName**, определенный в Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6, который содержит RDN этой подстатьи;
- атрибут **subtreeSpecification**, определенный в п. 14;
- атрибут **objectClass**, определенный в п.е 13, который указывает назначение(я) данной подстатьи в процессе функционирования Справочника;
- другие атрибуты в зависимости от значений атрибута **objectClass**.

Подстатьи могут также содержать операционные атрибуты с соответствующей семантикой (см. п. 12.6.4).

12.6.2 Атрибут RDN подстатьи

Атрибут **commonName**, используемый как идентификатор подстатьи, служит для распознавания различных подстатей, которые могут быть определены как непосредственно подчиненные специальной административной статье.

ПРИМЕЧАНИЕ. – Значение этого атрибута может выбираться, с тем чтобы служить мнемоническим значением для представителей административного органа.

Атрибут **commonName** для подстатьи не может содержать несколько выделенных значений, отличающихся контекстами; разрешено единственное выделенное значение.

12.6.3 Атрибут спецификации поддерева

Атрибут спецификации поддерева **subtreeSpecification** определяет набор статьей в пределах административной области, к которой относится это поддерево.

12.6.4 Использование атрибута класса объектов

Контент подстатьи регулируется значениями атрибута **objectClass** подстатьи.

Атрибут класса объектов **objectClass** всех подстатей должен содержать значение **subentry**. Класс объектов **subentry** является структурным классом объектов, который определяется в пункте 14, используемым для включения атрибутов **commonName**, **subtreeSpecification** и **objectClass** во все подстатьи.

Для управления остальными атрибутами должны использоваться другие значения атрибута **objectClass**, представляющие дополнительные классы объектов, которые разрешены для этой подстатьи.

Определение семантики одного из этих значений включает обозначение и спецификацию нуля или более типов атрибутов, которые должны или могут присутствовать в данной подстатье, если атрибут **objectClass** принимает это значение. Определение семантики значения атрибута **objectClass** должно включать:

- индикатор того, возможно ли включение статьи в более чем одно поддерево или уточнение поддерева, связанное с конкретным назначением (например, включение может не быть разрешено в случае **subschema**, но может быть разрешено для управления доступом); и в этом случае,

– последствия комбинации атрибутов связанных подстатей, если таковые имеются.

Подстатья конкретного класса объектов может быть подчиненной по отношению к административной статье, только если атрибут **administrativeRole** разрешает, чтобы этот класс подстатьи был подчиненным.

Что касается статей объектов и псевдонимов, хранящаяся в подстатье информация может быть дополнена административной и операционной информацией, которую представляют операционные атрибуты. Например, для подстатьи разрешено содержание ACI статьи только при условии, что эта ACI разрешена значением атрибута **accessControlScheme** соответствующей конкретной точки управления доступом и не противоречит ему. Аналогично подстатья может содержать **modifyTimestamp**.

12.6.5 Другие атрибуты подстатьи

Остальные атрибуты в подстатье зависят от значений атрибута **objectClass**. Например, атрибут подсхемы может быть помещен в подстатью, только если ее атрибут **objectClass** в качестве одного из своих значений имеет **subschema**.

12.7 Информационная модель для коллективных атрибутов

Автономная административная область может быть обозначена как специальная административная область коллективных атрибутов, с тем чтобы применять коллективные атрибуты и осуществлять административное управление ими. Это должно указываться наличием значения **id-ar-collectiveAttributeSpecificArea** в атрибуте **administrativeRole** связанной административной статьи (в дополнение к присутствию значения **autonomousArea** и, возможно, других значений).

Такая автономная административная область может быть разбита на части в целях применения коллективных атрибутов и административного управления ими в конкретных частях. В этом случае административные статьи для каждой из специальных административных областей коллективных атрибутов помечаются наличием значения **id-at-collectiveAttributeSpecificArea** в атрибуте **administrativeRole** этих статей.

Если такая автономная административная область не разбита на части, для коллективных атрибутов существует единая специальная административная область, которая охватывает всю эту автономную административную область.

Кроме того, специальная административная область, определенная для целей административного управления коллективными атрибутами, может быть поделена далее для той же цели на вложенные внутренние области. Атрибут **administrativeRole** административных статей для каждой такой внутренней административной области, должен указывать это посредством наличия значения **id-ar-collectiveAttributeInnerArea**.

Набор статей и связанные с ним коллективные атрибуты представляются в информационной модели Справочника подстатьей, называемой *подстатья коллективных атрибутов*, атрибут **objectClass** которой имеет значение **id-sc-collectiveAttributeSubentry**, как это определено в п. 14. Подстатья этого класса может быть непосредственно подчиненной по отношению к административной статье, атрибут **administrativeRole** которой содержит значение **id-ar-collectiveAttributeSpecificArea** или **id-ar-collectiveAttributeInnerArea**.

Если в пределах данной области коллективных атрибутов существуют различные наборы статей, каждый из них должен иметь свою подстатью.

Сам набор статей определяется значением операционного атрибута **subtreeSpecification** подстатьи. Это значение определяет *сферу действия* подстатьи операционных атрибутов. Атрибуты пользователя этой подстатьи являются коллективными атрибутами данного набора статей.

ПРИМЕЧАНИЕ 1. – В силу того, что уточнение поддерева базируется на классе объектов, связь коллективных атрибутов со статьями объектов может быть установлена таким образом, который естественно расширяет схему для этих статей. Например, статьи **organizationalPerson** какой-либо организации могут быть расширены совокупностью коллективных атрибутов, соответствующих всем лицам, которые являются членами этой организации, путем создания подстатьи, связанное поддерево которой уточнено для включения только статей **organizationalPerson** и содержит совокупность коллективных атрибутов этой организации. Кроме того, потребуется определить правило контента DIT для таких статей, с тем чтобы коллективные атрибуты стали видимыми в этих статьях.

Типы коллективных атрибутов и типы неколлективных атрибутов различаются семантически. Тип атрибута, способный отображать коллективную семантику, в момент его определения должен быть обозначен как тип коллективного атрибута.

ПРИМЕЧАНИЕ 2. – Процедуры слияния, используемые Справочником в случае независимых источников значений типа коллективного атрибута, описаны в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

Коллективные атрибуты могут быть исключены для появления в конкретной статье путем использования атрибута **collectiveExclusions**, определенного в п. 14.

12.8 Информационная модель для контекстов по умолчанию

Автономная административная область может быть обозначена как специальная административная область контекстов по умолчанию для использования контекстов по умолчанию и административного управления ими. Это должно быть указано наличием значения **id-ar-contextDefaultSpecificArea** в атрибуте **administrativeRole** связанной административной статьи (в дополнение к наличию значения **id-ar-autonomousArea** и, возможно, других значений).

Такая автономная административная область может быть разбита на части в целях применения контекстов по умолчанию и административного управления ими в конкретных частях. В этом случае административные статьи для каждой из специальных областей контекстов по умолчанию помечаются наличием значения **id-ar-contextDefaultSpecificArea** в атрибуте **administrativeRole** этих статей.

Если автономная административная область не разбита на части, для контекстов по умолчанию существует единая специальная административная область, которая охватывает всю эту автономную административную область.

Контексты по умолчанию представляются в информационной модели Справочника подстатьей, называемой *подстатья контекстов по умолчанию*, атрибут **objectClass** которой имеет значение **id-sc-contextAssertionSubentry**, как это определено в п. 14.7. Подстатья этого класса может быть непосредственно подчиненной по отношению к административной статье, атрибут **administrativeRole** которой содержит значение **id-ar-contextDefaultSpecificArea**.

Подстатья контекстов по умолчанию определяет множество утверждений о контексте, из которых применимо любое в случае отсутствия применимого к данному типу атрибута утверждения о контексте, указанного пользователем при доступе к части DIT, определенной операционным атрибутом **subtreeSpecification** этой подстатьи. Применение утверждений о контексте по умолчанию описывается в пп. 8.9.2.2 и 7.6.1 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

РАЗДЕЛ 6 – СХЕМА СПРАВОЧНИКА

13 Схема Справочника

13.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

13.1.1 синтаксис атрибута: Для представления значений атрибута используется тип данных ASN.1.

13.1.2 схема справочника: Совокупность правил и ограничений, касающихся структуры DIT, контента DIT, использования контента DIT, классов объектов и типов атрибутов, синтаксиса и правил сопоставления, которые описывают DIB. Схема Справочника объявляется как множество неперекрывающихся подсхем, каждая из которых управляет статьями автономной административной области (или конкретной частью ее подсхемы). Схема Справочника имеет отношение только к пользовательской информации Справочника.

13.1.3 подсхема (справочника): Совокупность правил и ограничений, касающихся структуры DIT, контента DIT, использования контента DIT, классов объектов и типов атрибутов, синтаксиса и правил сопоставления, которые описывают статьи DIB в пределах какой-либо автономной административной области (или конкретной части ее подсхемы).

13.1.4 правило контента DIT: Правило, управляющее содержимым статей конкретного структурного класса объектов. Оно описывает дополнительные классы объектов и дополнительные типы атрибутов, наличие которых в статьях указанного структурного класса объектов разрешено или исключается.

13.1.5 использование контекста DIT: Правило, управляющее типами контекстов, которые могут быть связаны со значениями атрибутов конкретных типов атрибутов. Оно описывает разрешенные и обязательные типы контекстов для данного типа атрибута.

13.1.6 правило структуры DIT: Правило, управляющее структурой DIT путем описания разрешенной взаимосвязи предшествующей, старшей статьи с подчиненной статьей. Правило структуры устанавливает связь формы имени и, следовательно, структурного класса объектов, с правилами структуры старшего. Это разрешает существование в DIT идентифицируемых по форме имени статей структурного класса объектов в качестве подчиненных по отношению к статьям, управляемым указанными правилами структуры старшего.

13.1.7 правило, управляющее структурой (статьи): Относительно конкретной статьи – *единое* правило структуры DIT, которое применяется к статье. Это правило указывается операционным атрибутом **governingStructureRule**.

13.1.8 форма имени: Форма имени определяет допустимое RDN для статей конкретного структурного класса объектов. Форма имени указывает именованный класс объектов и один или более тип атрибутов, которые должны использоваться для именования (т. е. для RDN). Формы имени являются элементарными порциями спецификации, используемой в определении правил структуры DIT.

ПРИМЕЧАНИЕ. – Формы имени регистрируются и имеют глобальную сферу действия. Правила структуры DIT не регистрируются и имеют сферу действия административной области, с которой они связаны.

13.1.9 правило структуры предшествующего, старшего: Относительно конкретной статьи – правило структуры DIT, управляющее предшествующим, старшим данной статьи.

13.2 Обзор

Схема Справочника является совокупностью определений и ограничений, касающихся структуры DIT, возможных способов именования статей, информации, которая может содержаться в статье, атрибутов, используемых для представления этой информации, и их организации в иерархии для упрощения поиска и извлечения информации, а также способов сопоставления значений атрибутов со значением атрибута и утверждений по правилам сопоставления.

ПРИМЕЧАНИЕ 1. – Эта схема позволяет системе Справочника, например:

- предотвращать создание подчиненных статей неверного класса объектов (например, страна как подчиненный элемент по отношению к лицу);
- предотвращать добавление к статье типов атрибутов, не соответствующих классу объектов (например, серийный номер к статье, относящейся к индивидууму);
- предотвращать добавление значения атрибута, синтаксис которого не соответствует определенному для этого типа атрибута (например, печатаемая строка к битовой строке).

Формально схему Справочника образуют:

a) описания *форм имен*, которые определяют элементарные отношения именования для структурных классов объектов;

b) описания *правил структуры DIT*, определяющие имена, которые могут иметь статьи, и способы, с помощью которых статьи могут быть связаны друг с другом в DIT;

c) описания *правил контента DIT*, которые расширяют спецификацию разрешенных атрибутов для включения статей, не являющихся статьями структурных классов объектов этих статей;

d) описания *класса объектов*, определяющие базовое множество обязательных и необязательных атрибутов, которые должны и могут присутствовать, соответственно, в статье данного класса, и указывающие вид определяемого класса объектов (см. п. 7.3);

e) описания *типов атрибутов*, определяющие идентификатор объекта, по которому известен атрибут, его синтаксис, связанные правила сопоставления, является ли этот атрибут операционным атрибутом, и если является, то его тип, является ли этот атрибут коллективным атрибутом, разрешено ли ему иметь несколько значений, и является ли он производной другого типа атрибута;

f) описания *правила сопоставления*, определяющие правила сопоставления;

g) описания *использования контекста DIT*, управляющие типами контекстов, которые могут быть связаны со значениями атрибутов любого конкретного типа атрибута.

На рисунке 11 показаны взаимоотношения между определениями схемы и подсхемы, с одной стороны, и DIT, статьями справочника, атрибутами и значениями атрибутов, с другой стороны.

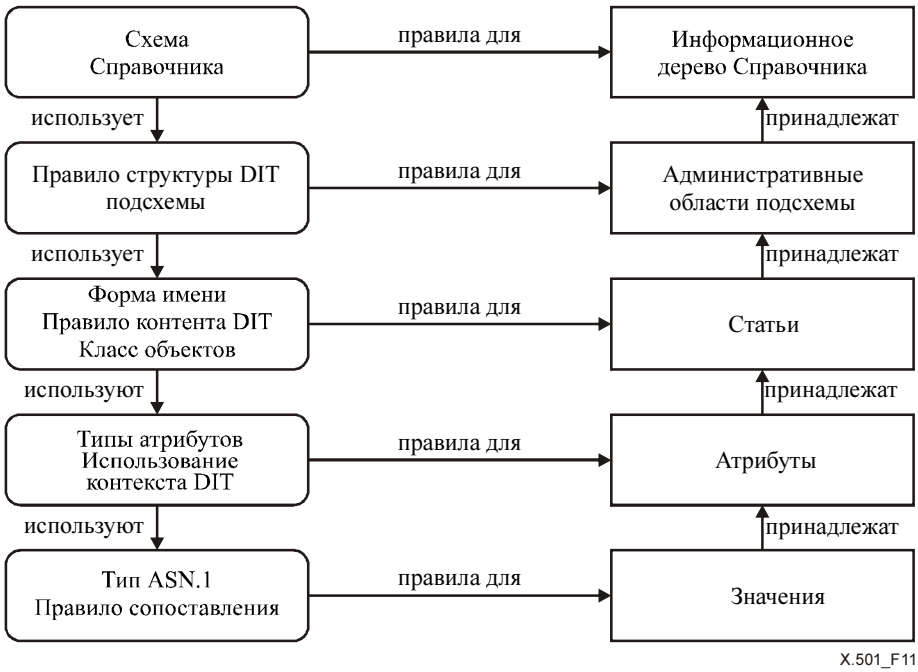


Рисунок 11 – Общее представление схемы Справочника

Рисунок 11 поясняется следующим образом:

– пункты, перечисленные по вертикали слева, представляют элементы схемы;

– пункты, перечисленные по вертикали справа, представляют примеры соответствующих позиций схемы, реализованные в соответствии с правилами, определенными этими позициями схемы;

– взаимоотношения между позициями схемы показаны с помощью отношения "использует";

– взаимоотношения между примерами различных аспектов схемы показаны с помощью отношения "принадлежат".

Схема Справочника является распределенной, как и сама DIB. Она объявляется как множество непересекающихся подсхем, каждая из которых управляет статьями автономной административной области (или конкретной частью ее подсхемы). Административный орган подсхемы устанавливает правила и ограничения, составляющие подсхему.

Административный орган подсхемы может выбрать использование отдельных элементов схемы Справочника, имеющих глобальную сферу действия, которые определяются в настоящих спецификациях Справочника: формы имени, классы объектов и атрибуты (типы и правила сопоставления). Он может также определять альтернативы этим элементам, в большей степени соответствующие его собственной среде, или выбрать некий промежуточный подход, используя и стандартизованные, и собственные элементы схемы.

Административный орган подсхемы определяет те элементы схемы, сфера действия которых ограничена подсхемой: правила структуры DIT, правила контента DIT и использование контента DIT. Кроме того, административный орган подсхемы может также определять, какие правила сопоставления применимы к каким типам атрибутов.

Схема Справочника имеет отношение только к пользовательской информации Справочника. Хотя для спецификации операционной информации, в обозначении, определяемом в настоящем пункте, предоставляется определенная поддержка, управлением административной и операционной информацией Справочника занимается *схема системы Справочника*.

ПРИМЕЧАНИЕ 2. – Схема системы Справочника описывается в п. 14.

13.3 Определение класса объектов

Определение класса объектов включает:

a) указание, по отношению к каким классам данный класс объектов должен являться подклассом;

b) указание, какой класс вид класса объектов определяется;

c) перечисление типов *обязательных* атрибутов, которые статья этого класса объектов должна содержать в дополнение к типам обязательных атрибутов всех своих суперклассов;

d) перечисление типов *необязательных* атрибутов, которые статья этого класса объектов может содержать в дополнение к необязательным атрибутам всех своих суперклассов;

e) присвоение данному классу объектов идентификатора объекта.

ПРИМЕЧАНИЕ. – Коллективные атрибуты не должны присутствовать в типах атрибутов определения класса объектов.

13.3.1 Создание подклассов

В отношении создания подклассов действуют ограничения, а именно:

– только абстрактные классы объектов должны быть суперклассами других абстрактных классов объектов.

Существует один специальный класс объектов, все структурные классы объектов которого являются подклассами. Этот класс объектов называется **top**. **top** – это абстрактный класс объектов.

13.3.2 Атрибут класса объектов

Все статьи должны содержать атрибут типа **objectClass** для указания классов и суперклассов объектов, которым принадлежат статьи. Определение этого атрибута приводится в п. 13.4.8. Этот атрибут является атрибутом, содержащим множество значений.

Должно существовать одно значение атрибута **objectClass** для структурного класса объектов статьи и по одному значению для каждого из его суперклассов. **top** может быть опущен.

Структурный класс объектов статьи не должен меняться. Начальные значения атрибута **objectClass** даются пользователем при создании данной статьи.

При использовании дополнительных классов объектов статья может содержать значения атрибута **objectClass** для дополнительных классов объектов и их суперклассов, разрешенные правилом контента DIT. Если имеется значение для дополнительного класса объектов, то должны также существовать значения для всех суперклассов этого дополнительного класса объектов.

Если атрибут **objectClass** содержит значение идентификатора объекта для дополнительного класса объектов, то данная статья должна содержать обязательные атрибуты, указанные этим классом объектов.

ПРИМЕЧАНИЕ 1. – Требование об обязательном наличии атрибута **objectClass** attribute в каждой статье отражено в определении **top**.

ПРИМЕЧАНИЕ 2. – Поскольку класс объектов рассматривается как принадлежащий всем его суперклассам, все члены цепи суперклассов до **top** представляются каким-либо значением в атрибуте **objectClass** (и любое значение в этой цепи может быть сопоставлено с помощью фильтра).

ПРИМЕЧАНИЕ 3. – На изменения атрибута **objectClass** могут быть наложены ограничения, связанные с управлением доступом.

В соответствии с применимыми правилами контента DIT Справочник обеспечивает соответствие определенного класса объекта для каждой статьи в DIB. Должна пресекаться любая попытка изменения статьи, которая может нарушить определение класса объектов этой статьи, что не разрешено явным образом правилом контента DIT статьи.

ПРИМЕЧАНИЕ 4. – В частности Справочник, как правило, не допускает:

a) добавление к статье данного класса объектов типов атрибутов, отсутствующих в определении структурного класса объектов и не разрешенных правилом контента DIT этой статьи;

b) создание статьи с одним или более отсутствующими типами атрибутов, которые обязательны для класса объектов этой статьи;

c) удаление обязательного типа атрибута для данного класса объектов данной статьи.

13.3.3 Спецификация классов объектов

Классы объектов могут быть определены как значения класса информационных объектов **OBJECT-CLASS**:

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

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

Для класса объектов, который определяется с использованием данного класса информационных объектов:

a) **&Superclasses** – множество классов объектов, которые являются его прямыми суперклассами;

b) **&kind** – вид;

c) **&MandatoryAttributes** – множество атрибутов, которые должны содержаться в статьях данного класса;

d) **&OptionalAttributes** – множество атрибутов, которые могут содержаться в статьях данного класса объектов, за исключением случая, когда атрибут присутствует в обязательном и необязательном множествах; такой атрибут должен рассматриваться как обязательный;

e) **&id** – присвоенный классу идентификатор объекта.

Ниже определяются ранее упоминавшиеся классы объектов (**top** и **alias**):

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

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

ПРИМЕЧАНИЕ 1. – Класс объектов **alias** не описывает соответствующие типы атрибутов для RDN статьи псевдонима. Административные органы могут определять подклассы класса **alias**, которые описывают пригодные типы атрибутов для имен RDN статей псевдонимов.

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

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

Ни один из классов объектов **parent** и **child** не должен комбинироваться с классом объектов **alias** для формирования статьи псевдонима.

Класс объектов **parent** выводится по присутствию непосредственно подчиненного члена семейства, отмеченного наличием значения класса объектов **child**. Он может не быть объектом непосредственного административного управления. Значение класса объектов **child** может добавляться или удаляться, только если результат согласуется с архитектурой составных статей (например, подчиненные члены семейства всегда должны иметь класс объектов **child)**.

ПРИМЕЧАНИЕ 2. – Классы объектов **parent** и **child** не определяют какие-либо типы атрибутов, подходящие для имен RDN членов семейства. Это должно выполняться стандартным способом через соответствующие структурные классы объектов и формы имен для этих статей.

13.4 Определение типов атрибутов

Определение типа атрибута включает:

a) необязательное указание, что данный тип атрибута является подтипом ранее определенного типа атрибута, его прямого супертипа;

b) описание синтаксиса атрибута для данного типа атрибута;

c) необязательное указание правила (правил) сопоставления равенства, упорядочения и/или подстрок для данного типа атрибута;

d) указания, должен ли атрибут данного типа иметь только одно значение или он может иметь более одного значения;

e) указания, операционным или пользовательским является данный тип атрибута;

f) необязательное указание, что тип атрибута пользователя является коллективным;

g) необязательное указание, что операционный атрибут не подлежит изменению пользователем;

h) для операционных атрибутов, указание приложения;

i) присвоение идентификатора объекта данному типу атрибута.

Любой атрибут пользователя может определяться административным органом как опорный атрибут, имеющий дружественные атрибуты. Следовательно, определение типа атрибута не указывает "друзей" опорного атрибута. Это может меняться в разных подсхемах.

13.4.1 Операционные атрибуты

Ряд операционных атрибутов находятся под непосредственным управлением пользователей. В остальных случаях значения операционных атрибутов контролируются Справочником. В последнем случае определение операционного атрибута должно указывать, что пользователям не разрешено каким-либо образом изменять значения этого атрибута.

Спецификация типа операционного атрибута должна указывать его приложение, которое должно быть одним из следующих:

– операционный атрибут Справочника (например, атрибуты управления доступом);

– операционный атрибут, коллективно используемый агентами DSA (например, атрибут главной точки доступа);

– коллективный атрибут, зависящий от DSA (например, атрибут статуса копии).

13.4.2 Иерархии атрибутов

Иерархия атрибутов должна содержать либо атрибуты пользователя, либо операционные атрибуты, но не оба типа одновременно. Отсюда следует, что атрибут пользователя не является производным операционного атрибута и что операционный атрибут не является производным атрибута пользователя.

Операционный атрибут, который является подтипом другого операционного атрибута, должен иметь то же приложение, что и его супертип.

Если тип атрибуты не является подтипом другого типа атрибута, синтаксис и правила сопоставления (если применимы) атрибута должны быть описаны в определении этого типа атрибута. Описание синтаксиса атрибута должно выполняться путем непосредственного определения типа данных ASN.1.

Если тип атрибута является подтипом указанного типа, в определении не требуется описывать синтаксис атрибута, в каковом случае синтаксис атрибута является синтаксисом его прямого супертипа. Если синтаксис атрибута указан, а атрибут имеет прямой супертип, указанный синтаксис должен быть совместим с синтаксисом супертипа, т. е. все возможные значения, удовлетворяющие этому синтаксису атрибута, должны удовлетворять синтаксису супертипа.

Если тип атрибута является подтипом другого типа атрибута, правила сопоставления, применимые к этому супертипу, применимы и к подтипу, если только в описание этого подтипа не внесены уточнения или изменения. Правило сопоставления, определенное для супертипа, не может быть удалено при определении подтипа.

13.4.3 Дружественные атрибуты

Перечень дружественных атрибутов опорного атрибута должен содержать только атрибуты пользователя. Это взаимоотношение не накладывает каких-либо ограничений на семантику, синтаксис или иные характеристики дружественного атрибута.

ПРИМЕЧАНИЕ. – Опорный атрибут может быть определен как фиктивный атрибут.

13.4.4 Коллективные атрибуты

Операционный атрибут не должен определяться как коллективный.

Атрибута пользователя может определяться как коллективный. Это служит признаком того, что в наборе статей будут присутствовать те же значения атрибута, в зависимости от использования атрибута **collectiveExclusions**.

Коллективные атрибуты должны содержать множество значений.

13.4.5 Производные атрибуты

Производным атрибутом является атрибут, который содержит информацию, соответствующую синтаксису информации атрибутов, но значения которого рассчитываются, а не сохраняются в DIB.

В целях включения информации о семействе для использования в службе Справочника вводится производный атрибут **family-information**, как это описано в п. 7.7.1 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

Агенты DSA также могут использовать технологию производных атрибутов для получения других атрибутов. Например, все операционные атрибуты, которые включают значение **AccessPoint** конкретного DSA, могут (и, возможно, должны) выводить это значение из единого источника информации, который может соответствующим образом быть управляемым в административном отношении.

13.4.6 Синтаксис атрибута

Если для данного типа атрибута определено правило сопоставления равенства, Справочник должен гарантировать использование верного синтаксиса атрибута для всех значений данного типа атрибута.

13.4.7 Правила сопоставления

Правила сопоставления равенства, упорядочения и подстрок могут быть указаны в определении данного типа атрибута. То же правило сопоставления может использоваться для одного и более этих типов сопоставлений, если семантика данного правила предусматривает более одного типа этих разных сопоставлений.

ПРИМЕЧАНИЕ 1. – Данный факт должен быть отражен в определении указанного правила сопоставления.

Если правил сопоставления равенства не указано, Справочник:

a) интерпретирует значения этого атрибута как имеющие тип **ANY**, т. е. Справочник может не проверять соответствие данных значений типу данных или любому иному правилу, указанному для атрибута;

b) не разрешает использование этого атрибута для именования;

c) не допускает добавления или удаления отдельных значений атрибутов, содержащих множество значений;

d) не выполняет сравнения значений этого атрибута;

e) не предпринимает попыток оценить результат **AVA** с использованием значений такого типа атрибута.

Если какое-либо правило сопоставления равенства указано, Справочник:

a) интерпретирует значения этого атрибута как имеющие тип, определенный в поле **&Type** определения атрибута (или тип атрибута, производным которого является данный атрибут);

b) использует указанное правило сопоставления равенства для целей оценки утверждений о значениях атрибута, касающихся данного атрибута;

c) проводит сопоставление представленного значения только соответствующего типа данных, который описан в определении типа атрибута.

ПРИМЕЧАНИЕ 2. – Этот подпункт в равной степени применим к атрибуту, правило сопоставления равенства которого использует синтаксис утверждения, отличный от синтаксиса данного типа атрибута.

Если не указано ни одного правила сопоставления упорядочения, Справочник интерпретирует все утверждения сопоставления упорядочения, используя синтаксис, предоставляемый абстрактной службой Справочника, как неопределенные.

Если не указано ни одного правила сопоставления подстрок, Справочник интерпретирует все утверждения сопоставления подстрок, используя синтаксис, предоставляемый абстрактной службой Справочника, как неопределенные.

Тип атрибута должен описывать только те правила сопоставления, определения которых применяются к синтаксису атрибута данного атрибута.

13.4.8 Определение атрибута

Атрибуты могут быть определены как значения класса информационных объектов **ATTRIBUTE**:

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

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

Для атрибута, который определяется с использованием данного класса информационных объектов:

a) **&derivation** – атрибут, если существует, подтипом которого он является;

b) **&Type** – его синтаксис атрибута. Это должен быть тип ASN.1, но не тип, содержащий какое-либо **EmbeddedPDV**;

c) **&equality-match** – правило сопоставления равенства (если указано);

d) **&ordering-match** – правило сопоставления упорядочения (если указано);

e) **&substrings-match** – правило сопоставления подстрок (если указано);

f) **&single-valued** – является TRUE, если это единственное значение, и FALSE в любом ином случае;

g) **&collective** – является TRUE, если это коллективный атрибут, и FALSE в любом ином случае;

h) **&dummy** – является **TRUE**, если это фиктивный атрибут, и **FALSE** в любом ином случае;

i) **&no-user-modification** – является TRUE, если это операционный атрибут, который не может быть изменен пользователем;

j) **&usage** показывает операционное использование атрибута. **userApplications** означает атрибут пользователя, **directoryOperation**, **distributedOperation** и **dSAOperation** означают справочник, распределенный атрибут или операционный атрибута DSA, соответственно;

k) **&id** – присвоенный ему идентификатор объекта.

Типы атрибутов, определенные в первом издании настоящей спецификации Справочника, известные Справочнику и используемые им для своих целей, определяются следующим образом:

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

aliasedEntryName ATTRIBUTE ::= {
        WITH SYNTAX                     DistinguishedName
```

```
    EQUALITY MATCHING RULE      distinguishedNameMatch
    SINGLE VALUE                TRUE
    ID                          id-at-aliasedEntryName }
```

ПРИМЕЧАНИЕ. – Сами правила сопоставления, которые указывается в этих определениях, определяются в п. 13.5.2.

Атрибуты **objectClass** и **aliasedEntryName** определяются как атрибуты пользователя, даже если они используются для операций Справочника и семантически должны определяться как операционные. Это объясняется тем, что данные атрибуты были определены как атрибуты пользователя до введения понятия операционного атрибута и должны оставаться атрибутами пользователя в целях содействия межсетевому взаимодействию между системами, реализующими разные издания настоящей спецификации Справочника.

13.5 Определение правила сопоставления

13.5.1 Обзор

Определение правила сопоставления включает:

a) необязательное определение родительских правил сопоставления, из которых может быть выведено данное правило сопоставления;

b) определение синтаксиса утверждения правила сопоставления;

c) описание различных типов сопоставлений, поддерживаемых данным правилом;

d) определение соответствующих правил для оценки представленного утверждения относительно целевых значений атрибута, хранимых в DIB;

e) присвоение идентификатора объекта данному правилу сопоставления.

Правило сопоставления используется для оценки утверждений о значениях атрибута атрибутов, указывающих это правило в качестве своего правила сопоставления равенства. Синтаксис, используемый в утверждении о значениях атрибута (т. е. компонент **assertion** утверждения о значении атрибута), является синтаксисом утверждения данного правила сопоставления.

Правило сопоставления может применяться ко многим различным типам атрибутов, имеющим разный синтаксис атрибута.

Определение правила сопоставления должно включать спецификацию синтаксиса утверждения правила сопоставления и способа использования значений этого синтаксиса для проведения сопоставления. Это не требует полной спецификации синтаксиса атрибута, к которому может применяться это правило сопоставления. Определение правила сопоставления для использования с атрибутами, имеющими разный синтаксис ASN.1, должно описывать порядок определения совпадений.

Применимость определенных правил сопоставления к содержащимся в спецификации подсхемы атрибутам (в добавление к правилам сопоставления, используемым в определении этих типов атрибутов) указывается через операционный атрибут спецификации подсхемы **matchingRuleUse**, который определяется в п. 15.7.7.

13.5.2 Определение правила сопоставления

Правила сопоставления могут быть определены как значения класса информационных объектов **MATCHING-RULE**:

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

Для правила сопоставления, которое определяется с использованием данного класса информационных объектов:

a) поле **&ParentMatchingRules** используется, если в определяемом правиле сопоставления сочетаются характеристики двух или более других правил сопоставления. Это имеет вид множества из двух или более идентификаторов объектов для правил сопоставления, которые вносят базовые характеристики определяемого правила сопоставления (например, алгоритм сопоставления); для базового правила сопоставления поле пропускается.

b) **&AssertionType** – синтаксис для утверждения при использовании данного правила сопоставления; если оно опущено, синтаксис утверждения идентичен синтаксису атрибута, к которому применяется правило, если правило сопоставления не определяет иное. Если синтаксис существует, он может определять ограничения для родительского(их) правила(правил) сопоставления, если таковое(ые) имеются, но в этом случае он должен быть совместим с синтаксисом для родительского(их) правила(правил) сопоставления (т. е. значение, соответствующее **&AssertionType**, должно соответствовать **&AssertionType** для родительского(их) правила(правил) сопоставления).

c) **&uniqueMatchIndicator** – тип атрибута уведомления. Если присутствует, требуется однозначное совпадение. Для правила сопоставления на основе отображения (см. п. 13.6), это означает, что отображение по таблице сопоставления должно дать однозначный результат. Если обнаруживаются несколько совпадений с таблицей сопоставления, запрос поиска отклоняется с сообщением об ошибке службы **serviceError** с указанием, что причиной является неоднозначность **ambiguousKeyAttributes**. Кроме того, атрибут уведомления типа, описанный этим полем, помещается в **CommonResults** возвращаемой ошибки.

ПРИМЕЧАНИЕ 1. – Такая ситуация возможна в случае географического сопоставления, когда, например, утверждение может определять "Newton" как пункт в Соединенном Королевстве; существуют много городов с таким названием, которые должны различаться спецификатором (например, "Newton, Cambs").

d) **&id** – присвоенный ему идентификатор объекта.

Если для **ParentMatchingRules** используются два или более правил сопоставления, результатом является комбинированное правило сопоставления, которое для значений, совместимых с **AssertionType**, возвращает результат согласно следующему правилу:

a) если результатом применения любого родительского правила сопоставления является TRUE, комбинированное правило сопоставления возвращает TRUE;

b) в любом ином случае, если результатом применения любого родительского правила сопоставления является FALSE, комбинированное правило сопоставления возвращает FALSE; или

c) в любом ином случае, результат применения правила сопоставления является неопределенным.

В нижеследующей таблице показаны правила комбинирования двух правил сопоставления A и B; размерность этой таблицы, в принципе, может быть увеличена при тех же схемах результатов для охвата случая наличия трех и более родительских правил сопоставления:

		Правило A		
		ИСТИННЫЙ	ЛОЖНЫЙ	НЕОПРЕДЕЛЕННЫЙ
Правило B	ИСТИННЫЙ	ИСТИННЫЙ	ИСТИННЫЙ	ИСТИННЫЙ
	ЛОЖНЫЙ	ИСТИННЫЙ	ЛОЖНЫЙ	ЛОЖНЫЙ
	НЕОПРЕДЕЛЕННЫЙ	ИСТИННЫЙ	ЛОЖНЫЙ	НЕОПРЕДЕЛЕННЫЙ

Комбинируя правила сопоставления, как указано выше, возможно получить действительное совпадение в тех случаях, когда провести сопоставление не удалось бы.

ПРИМЕЧАНИЕ 2. – Особым случаем использования родительского правила сопоставления является комбинирование произвольного правила сопоставления со специальным правилом сопоставления **ignoreIfAbsentMatch**. Последнее вызывает возвращение пунктом фильтра значения TRUE, если атрибут отсутствует; если атрибут присутствует, применяются обычные правила. Это позволяет фильтру поиска анализировать статьи в отсутствии ряда атрибутов, описанных в фильтре поиска. См. п. 7.7.1 Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6.

Правило сопоставления **objectIdentifierMatch** описывается следующим образом:

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

Представленное значение идентификатора типа объекта совпадает с целевым значением идентификатора типа объекта, исключительно если они оба имеют то же количество составляющих компонентов и каждый составляющий компонент первого идентичен соответствующему компоненту второго. Это правило сопоставления является свойственным определению идентификатора типа объекта ASN.1. **objectIdentifierMatch** – правило сопоставления равенства.

distinguishedNameMatch определяется следующим образом

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

Представленное значение выделенного имени совпадает с целевым значением выделенного имени, исключительно если выполняются следующие положения:

a) то же количество RDN в каждом;

b) соответствующие RDN имеют то же число **AttributeTypeAndValue**;

c) соответствующие **AttributeTypeAndValue** (т. е. в соответствующих RDN и с идентичными типами атрибутов) имеют значения атрибутов, которые совпадают, как описано в п. 9.4.

distinguishedNameMatch – правило совпадения равенства.

13.6 Ослабления и усиления

Ослабление и *усиление* являются функциями, которые систематическим образом изменяют сопоставление одного или более пунктов фильтра. Если выполняется ослабление, изменение сопоставления осуществляется таким образом, чтобы повысить вероятность получения большего количества совпадающих статей. Ослабление применяется, если количество совпадающих статей меньше определенного минимума. Усиление выполняется аналогичным образом, если количество совпадающих статей превышает определенный максимум. Существуют два режима ослабления/усиления:

a) правило сопоставления, применимое к конкретному типу атрибута, может изменяться поэтапным замещением правила сопоставления, до тех пор пока не будет достигнут требуемый результат или не будут исчерпаны все возможности, как это описано в п. 13.6.1; и

b) ослабление/усиление может быть применено как часть *совпадения на основе отображения*, как это описано в п. 13.6.2.

13.6.1 Замещение правила сопоставления

Замещение правила сопоставления может контролироваться управляющим правилом поиска в пределах зависящей от службы административной области (см. п. 16.10.7). Оно также может контролироваться пользователем в запросе поиска **search** (см. п. 10.2.1 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3). В обоих случаях конструкция **RelaxationPolicy**, как она определена в п. 16.10, контролирует замещение.

Ослабление/усиление путем замещения правила сопоставления изменяет действие фильтра путем систематического замещения ранее применимых правил сопоставления на правила сопоставления, обеспечивающие более мягкое (или жесткое) сопоставление. Ослабленный или усиленный замещением правила сопоставления процесс поиска полностью повторно оценивается по тому же множеству статей в пределах данной области действия поиска. Повторная оценка может продолжаться, до тех пор пока не будут исчерпаны все варианты ослабления или до получения удовлетворительного (меньше или равно **maximum** или больше **minimum**, согласно контролирующим элементам **RelaxationPolicy**) результата.

В результате фильтр для каждой переоценки остается прежним, а отдельные правила сопоставления, используемые для оценки результата действия фильтра, по необходимости замещаются (см. рисунок 12). Ослабление может оцениваться либо от DSA к DSA без применения координированного ослабления между DSA, либо для определения ослабления, которое необходимо применять, может альтернативно использоваться компонент **chainedRelaxation** выражения **ChainingArguments**.

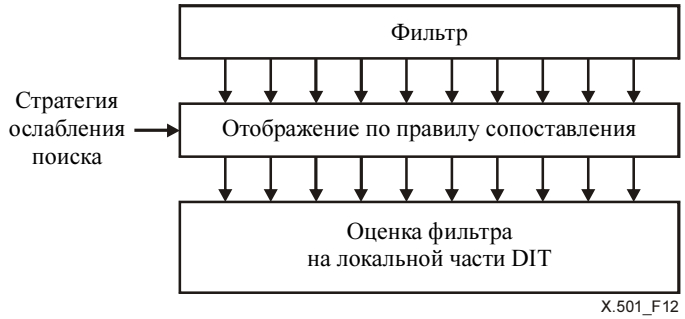


Рисунок 12 – Замещение правила сопоставления

В случае необходимости применения стратегии ослабления DSA перед началом локального поиска выполняет *базовое замещение* для каждого типа атрибута, для которого определено базовое замещение, как указано в стратегии ослабления.

ПРИМЕЧАНИЕ 1. – Одним из полезных применений базового замещения является, например, его применение для типа атрибута **localityName** для замещения правила сопоставления **caseIgnoreSubstringMatch** правилом сопоставления **generalWordMatch** в ситуациях, когда это правило замещения пригодно в большей степени и предполагается, что пользователь соответствующим образом формулирует пункт фильтра **substrings** filter.

Если в результате поиска найдено слишком малое, для данного конкретного DSA, количество статей, применяется первая стратегия ослабления; если в результате опять найдено слишком мало статей, применяется вторая стратегия ослабления и т. д.

Аналогичным образом, если в результате поиска найдено слишком большое количество статей, таким же порядком применяется первая стратегия усиления. Не существует перехода с усиления на ослабление и обратно.

Ослабление, применяемое одним множеством **MRSubstitution** для конкретного атрибута, применяется, до тех пор пока не будет отменено другим **MRMapping**. Отмена может быть сделана явным образом путем описания правила сопоставления или неявным образом путем опущения идентификатора **oldMatchingRule**.

Если ослабление оценки обусловлено слишком малочисленными результатами предыдущей оценки, а после ослабленной оценки получены слишком многочисленные результаты, должны быть возвращены некоторые или все результаты ослабленной оценки. Если усиление оценки обусловлено слишком многочисленными результатами предыдущей оценки, а после усиления оценки получены слишком малочисленные результаты, должны быть возвращены некоторые или все результаты предыдущей оценки. В обоих случаях процесс ослабления или усиления останавливается.

Применимая стратегия ослабления применяется и к **filter**, и к **extendedFilter**, в зависимости от случая.

ПРИМЕЧАНИЕ 2. – Поскольку ослабление разрешает пункту фильтра производить менее или более жесткие оценки для *обычного* фильтра, сокращается необходимость в дополнительных фильтрах при проведении более сложных операций фильтрации.

DSA может передать в результат поиска **search** атрибут уведомления о предлагаемом ослаблении **proposedRelaxation** (см. п. 5.12.15 Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6) в пределах подкомпонента **notification** квалификатора **PartialOutcomeQualifier**. Содержащаяся в нем информация может затем быть использована в подкомпоненте запроса поиска **search** в качестве определенной пользователем стратегии ослабления.

Как крайний случай ослабления, стратегия может обусловливать, что конкретный пункт фильтра выдает оценку TRUE (или FALSE, если это неинвертированный пункт фильтра) в соответствии с правилом сопоставления **nullMatch**.

В пределах специальной служебной административной области проверка по правилам поиска осуществляется после выполнения возможных базовых замещений, как предписано правилом поиска, по которому оценивается результат запроса поиска **search**. Управляющее правило поиска выбирается перед любым последующим замещением правила сопоставления, включая возможные базовые замещения, определенные в запросе **search**.

13.6.2 Сопоставление на основе отображении

Сопоставление на основе отображения актуально для операции поиска, если пользовательское определение реального мира может по некоторым аспектам отличаться от идеализированной модели, часто предлагаемой Справочником. Например, пользовательские варианты написания наименований населенных пунктов или взаимосвязи между населенными пунктами могут сильно отличаться от того, как эти населенные пункты представлены в Справочнике. Для сокращения такого разрыва и повышения коэффициента успешности поисков важно иметь соответствие между пользовательским понятием некоторых объектов реального мира, включая их взаимоотношения, и моделью этих объектов в Справочнике. Это отображение должно также предусматривать "нечеткое" сопоставление, т. е. разрешать значениям некоторых атрибутов отображать больше, чем предусмотрено их точным определением.

ПРИМЕЧАНИЕ 1. – Например, пользователь может определить наименование населенного пункта в фильтре, но искомый объект может находиться вблизи границы в соседней местности.

Сопоставление на основе отображения применимо к географическим аспектам поиска по Белым страницам, аспектам коммерческих категорий поиска по Желтым страницам и т. д.

Процесс сопоставления на основе отображения использует некоторую промежуточную таблицу – *таблицу отображения* – для контроля соответствия. Вместе с тем, базовый принцип метода является общим и показан графически на рисунке 13.

Рисунок 13 – Сопоставление на основе отображения

При применении этого способа пункты фильтра для обозначенных типов атрибутов (*отображаемые пункты фильтра)* проходят через процесс отображения с использованием промежуточной таблицы и некоторого алгоритма отображения. Это отображение приводит к появлению некоторых новых пунктов фильтра, именуемых *отображенными пунктами фильтра*, которые являются заменой отображаемых пунктов фильтра. В исключительных случаях отображение не выполняется и возвращается информация относительно точного характера исключения.

Количество отображенных пунктов фильтра необязательно должно быть тем же, что и количество отображаемых пунктов фильтра и в принципе должно отличаться.

Пункт фильтра типа **extensibleMatch** с отсутствующей спецификацией типа **type** не может быть поддающимся отображению пунктом фильтра.

Сопоставление на базе отображения может быть локальным по отношению к DSA. Если оценка поиска является распределенной, другие DSA, участвующие в этапе оценки поиска, могут применять собственные сопоставления на основе отображения. Вместе с тем, используемое отображение может передаваться другим DSA в компоненте **chainedRelaxation** выражения **ChainedArguments**.

ПРИМЕЧАНИЕ 2. – Для обеспечения в интересах пользователей бесперебойной службы администраторы потенциально участвующих в распределенной оценке поиска DSA должны предусматривать согласование своих таблиц и функций отображения.

На рисунке 14 графически представлен принцип, лежащий в основе функции отображения, определяющей соответствие между реальным миром и моделью этого мира в Справочнике. Аспекты реального мира, влияющие на то, как пользователь формулирует запрос поиска, составляют модель реального мира. Точная модель реального мира должна базироваться на опыте и, вероятно, требует периодического обновления, следуя практике работы пользователей с функцией поиска.

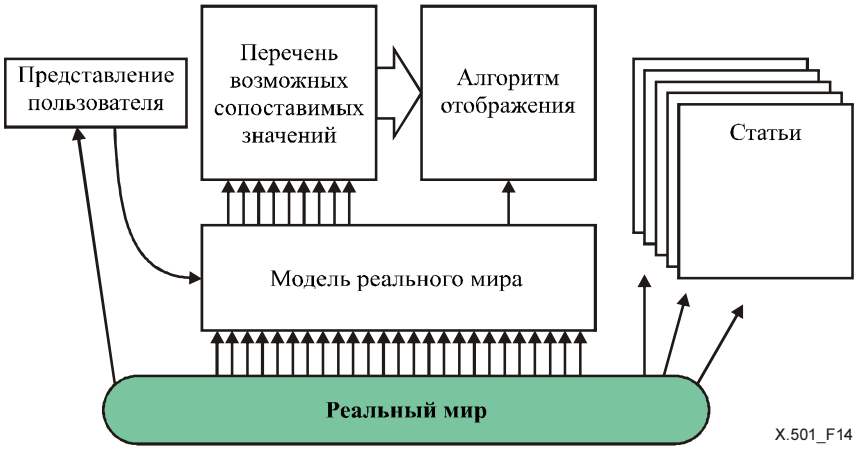


Рисунок 14 – Вывод информации

Эта модель реального мира может задействовать только типы атрибутов, включенные пользователем в запрос поиска, и, вероятно, актуальным является только один тип атрибута. Например, рассматривая модель реального мира в аспекте населенных пунктов, актуальными для анализа будут только типы атрибутов, связанные с населенными пунктами. Пункты фильтра, не ссылающиеся на такие типы атрибутов, не отображаются, но сохраняются и используются вместе с отображенными пунктами фильтра для совпавшей статьи.

Модель реального мира используется для создания таблицы *сопоставимых значений*, т. е. множества значений, которые теоретически должны сопоставляться с отображаемыми пунктами фильтра. Порядок создания такой таблицы отображения сопоставимых значений является вопросом местной компетенции. Результатом сопоставления с этой таблицей отображения может стать нуль или более сопоставлений. Каждое совпадение дает один или более отображенных пунктов фильтра. Алгоритм отображения определяет, как отображенные пункты фильтра применяются к статьям. Способ реализации этого является вопросом местной компетенции. Он может базироваться на значениях традиционных атрибутов в статьях или на "спрятанных" в статьях значениях, которые не имеют смысла за пределами Справочника, например числовых идентификаторах.

Способ применения отображения и обработки результирующих отображенных пунктов фильтра традиционно описывается путем ссылок на подфильтры, определенные в п. 16.5 и подробнее описанные в Приложении Q. Понятие подфильтров используется только здесь в качестве инструмента описания. В реализации может использоваться любой иной алгоритм, дающий тот же результат.

Оценка каждого подфильтра определяется по таблице отображения, а результирующие отображенные пункты фильтра комбинируются с неотображенными пунктами фильтра таким способом, который определен детальным алгоритмом отображения. Совпавшие в результате статьи являются объединением статей, совпавших по каждому из подфильтров.

ПРИМЕЧАНИЕ 3. – Во многих случаях отображаемые пункты фильтра буду заменяться логическим ИЛИ отображенных пунктов фильтров.

В принципе существуют два разных режима отображения. Каждый отображаемый пункт фильтра может отображаться поочередно, или же могут использоваться множественные *комбинируемые* отражаемые пункты фильтров для получения однократного совпадения по таблице отображения. Множественные пункты фильтра применимы к однократному совпадению на основе отображения, исключительно если они являются *комбинируемыми* пунктами фильтра, т. е. содержатся в качестве элементов в пределах одного подфильтра.

ПРИМЕЧАНИЕ 4. – Например, два отдельных географических названия, которые в подфильтре связаны операцией И, могут использоваться для описания единичного географического местоположения пригодного размера, в то время как использование одного географического названия привело бы к описанию неоднозначного географического местоположения или местоположения избыточного размера.

Сопоставление пункта фильтра с таблицей отображения выполняется с использованием правила сопоставления, подразумеваемого или описанного данным пунктом фильтра, по возможности после проведения базового замещения правила сопоставления, которое определено либо в управляющем правиле поиска (если таковое существует), либо в запросе поиска **search**.

ПРИМЕЧАНИЕ 5. – Это может включать использование такого комплексного правила сопоставления, как правило **generalWordMatch**, определенное в Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6, которым разрешены чередование слов, усечение слов, приблизительное совпадение слов и т. д.

ПРИМЕЧАНИЕ 6. – В настоящих спецификациях Справочника не описывается, каким образом в реализации комбинируются соответствующие правила сопоставления для комбинированного сопоставления. Предполагается, что реализация может ограничивать поддерживаемые комбинации пунктов фильтра и правил сопоставления.

Если сопоставление, предпринятое пунктом фильтра или комбинируемыми пунктами фильтра по таблице отображения, не привело к совпадению ни по одному подфильтру, т. е. получен результат FALSE или неопределенный результат, это приведет к нулевому количеству отображенных пунктов фильтра. Если отображаемые пункты фильтра существуют в каждом подфильтре, поиск не даст результата. В этом случае пользователю будет возвращено сообщение об ошибке.

В некоторые случаях, например при географическом зональном сопоставлении, требуется, чтобы сопоставление по таблице отображения давало единственный, однозначный результат. Если подфильтр выдает в таблице отображения совпадение более одной статьи или если разные подфильтры в таблице отображения выдают совпадение разных статей, операция поиска может возвращать слишком много лишних статей. Вместо этого пользователю выводится сообщение, позволяющее инициировать новый и более корректный запрос поиска **search**.

ПРИМЕЧАНИЕ 7. – В более простом случае отображаемые элементы фильтра просто проверяются по таблице отображения. Если такое сопоставление успешно, отображаемые пункты фильтра используются без изменений.

Отображение может иметь динамический характер в том смысле, что оно может регулироваться (ослабляться), если поиск выдает нулевой результат или слишком малое количество совпавших статей. Подробное описание того, как выполняется такое ослабление, выходит за пределы области действия настоящих спецификаций Справочника. Оно определяется местными требованиями. Ослабление может осуществляться поэтапно с постепенным увеличением количества искомых статей. Ослабление должно выполняться таким образом, чтобы после каждого следующего выполненного шага возвращались все статьи, найденные на предыдущих шагах, а также новые статьи, которые могут быть найдены.

Ослабление осуществляется поэтапно путем определения различных уровней ослабления. Нулевой уровень соответствует отсутствию ослабления. Уровень один соответствует первому уровню ослабления и т. д. На рисунке 15 в абстрактной форме показан этот механизм поэтапного ослабления. Что именно означают различные уровни ослабления, в настоящих спецификациях Справочника не определяется. Уровень ослабления может контролироваться конструкцией **RelaxationPolicy**, которая может быть передана в правиле поиска, в запросе **search** или и в том и в другом. Это позволяет синхронизировать между собой ослабление сопоставления на основе отображения и ослабление методом замещений правила сопоставления, поскольку оба способа могут быть определены по каждому этапу ослабления, как это определяется стратегией ослабления **RelaxationPolicy**.

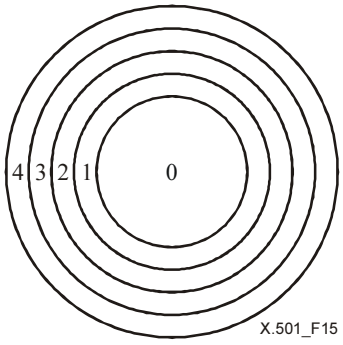


Рисунок 15 – Ослабление поиска

Управление поиском **extendedArea** осуществляет целочисленная величина, которая обеспечивает альтернативный способ управления уровнем ослабления для алгоритма сопоставления на основе отображения. Вопрос о том, может ли сопоставление на основе отображения находиться под управлением этого контроля поиска, является частью настройки алгоритма этого сопоставления.

Если в запросе поиска **request** присутствует управление поиском **extendedArea** и его использование разрешено для алгоритма на основе отображения, в **RelaxationPolicy** игнорируется спецификация любого уровня, независимо от того, включена она в **search** или в управляющее правило поиска.

Опция управления поиском **includeAllAreas** описывает режим ослабления, если он контролируется управлением поиска **extendedArea**. Если эта опция установлена, ослабление выполняется, как описано выше, т. е. при высших уровнях ослабления (*включающее ослабление)* возвращается потенциально большее количество статей. Если эта опция не установлена, пользователь заинтересован только в результате, соответствующем пошаговому ослаблению (*исключающее ослабление)*. Последнее может быть интересным, если пользователь выполняет поэтапное ослабление и интересуется не получением статей, соответствующих предыдущим результатам поиска, а только дополнительными статьями, найденными в результате самого последнего шага ослабления.

ПРИМЕЧАНИЕ 8. – Не гарантируется (особенно при применении комплексного фильтра), что пользователь не получит ряда статей, уже полученных ранее, или что будут возвращены все статьи, которые могут вызывать интерес. Например, поиск французских ресторанов в Уинкфилде может оказаться неудачным; ослабление для поиска всех ресторанов в зоне Уинкфилда, но исключение Уинкфилда приведет к тому, что из результатов поиска будет исключен White Hart Inn – ресторан в Уинкфилде со смешанной кухней.

Некоторые алгоритмы сопоставления на основе отображения могут не поддерживать исключающее ослабление или могут быть настроены не поддерживать этот алгоритм. В этом случае опция управления поиском **includeAllAreas** для этой функции отображения должна игнорироваться, а возможное ослабление должно выполняться как включающее ослабление.

В некоторых условиях может также быть желательным определить отрицательный уровень ослабления, который соответствует *усилению* сопоставления. В этом случае опция управления поиском **includeAllAreas** не имеет значения и в случае наличия игнорируется. Усиление не может быть актуальным для всех типов сопоставления на основе отображения.

DSA может поддерживать одновременно несколько функций отображения, т. е. содержать многочисленные таблицы отображения с соответствующими алгоритмами отображения. Причинами существования многочисленных функций отображения могут быть следующие:

a) Функция отображения, которую следует реализовать, зависит от типа приложения. Конкретным важным приложением сопоставления на основе отображения является географическое зональное сопоставление (см. п. 7.8 Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6). К другим примерам относятся сопоставление на основе отображения для поиска в Желтых страницах, библиографические изыскания и т. д.

b) В пределах конкретного приложения подробное описание порядка выполнения отображения может меняться в зависимости от конкретных условий. Например, отображение для географического зонального сопоставления может зависеть от географической области (например, которая отображается **baseObject** поиска) или от типа предпринимаемого пользователем поиска, т. е. на основе информации, содержащейся в фильтре поиска. Другой пример, отображение может зависеть от используемого в запросе языка.

Если одновременно применимы многие функции отображения и выполнение одной из них приведет к возникновению условия исключения, о котором должно быть сообщено пользователю, не требуется проверки существования многочисленных исключений (но она может это делать).

Спецификация сопоставления на основе отображения (см. далее) определяет, будет ли для данной функции отображения применимо управление поиском **extendedArea**. Если для той же операции поиска активны несколько функций отображения и некоторые из них могут управляться управлением поиска **extendedArea**, все они выполняют одновременное ослабление или усиление согласно управлению поиском **extendedArea**, а также, если применимо, согласно опции управления поиском **includeAllAreas**.

ПРИМЕЧАНИЕ 9. – Приведенный ранее пример показывает, что использование **includeAllAreas** с более чем одним сопоставлением на основе отображения может привести к возникновению проблем.

Если управление поиском **extendedArea** определяет уровень ослабления или усиления, не поддерживаемый DSA для ряда функций отображения, затрагиваемых этим управлением поиском, то этот DSA должен выполнить отображение по принципу оптимального сценария. Если управления поиском **extendedArea** определяет уровень ослабления или усиления, не поддерживаемый данным DSA для какой-либо функции отображения, затрагиваемой этим управлением поиском, в параметре уведомления **notification** выражения **CommonResults** должен быть возвращен атрибут уведомления **searchServiceProblem** со значением **id-pr-unavailableRelaxationLevel**.

ПРИМЕЧАНИЕ 10. – Если оценка операции поиска является распределенной по многочисленным DSA, такие DSA, если они не подверглись взаимной координации, могут использовать разные функции отображения, дающие противоречивые результаты.

Несмотря на то что детали сопоставления на основе отображения относятся к вопросу местной компетенции, возможно описать общие характеристики сопоставления на основе отображения путем определения специального типа правил сопоставления, называемых *правила сопоставления на основе отображения*. Такое правило сопоставления определяется как экземпляр класса информационных объектов **MATCHING-RULE**. Вместе с тем, оно отличается от традиционных правил сопоставления тем, что не описывает сопоставления в традиционном смысле и, следовательно, не описывает синтаксис для сопоставления. Однако в качестве части своего определения правило содержит спецификацию своего назначения, порядка применения и обработки условий исключения. Конкретное поведение правила сопоставления на основе отображения может частично быть описано экземпляром класса информационных объектов ASN.1, выведенного из нижнего родового (параметризованного) класса информационных объектов **MAPPING-BASED-MATCHING**. Этот класс информационных объектов предназначен исключительно для описания тех аспектов, которые теоретически являются настраиваемыми. Спецификация Справочника не налагает требований относительно порядка и места хранения экземпляра такого класса информационных объектов, а лишь в отношении обеспечения доступности каким-либо образом этого класса для данного DSA.

```
MAPPING-BASED-MATCHING
        { SelectedBy, BOOLEAN:combinable, MappingResult, OBJECT IDENTIFIER:matchingRule } ::=

CLASS  {
        &selectBy               SelectedBy              OPTIONAL,
        &ApplicableTo           ATTRIBUTE,
        &subtypesIncluded       BOOLEAN                 DEFAULT  TRUE,
        &combinable             BOOLEAN                 (combinable),
        &mappingResults         MappingResult           OPTIONAL,
        &userControl            BOOLEAN                 DEFAULT  FALSE,
        &exclusive              BOOLEAN                 DEFAULT  TRUE,
        &matching-rule          MATCHING-RULE.&id       (matchingRule),
        &id                     OBJECT IDENTIFIER       UNIQUE }
WITH SYNTAX  {
        [ SELECT BY             &selectBy ]
        APPLICABLE TO           &ApplicableTo
        [ SUBTYPES INCLUDED     &subtypesIncluded ]
        COMBINABLE              &combinable
        [ MAPPING RESULTS       &mappingResults ]
        [ USER CONTROL          &userControl ]
        [ EXCLUSIVE             &exclusive ]
        MATCHING RULE           &matching-rule
        ID                      &id }
```

Поля класса информационных объектов **MAPPING-BASED-MATCHING** описываются следующим образом:

a) Поле **&selectBy** является фиктивной ссылкой на описание порядка выбора экземпляра специализации класса информационных объектов для сопоставления на основе отображения. Специализированный класс информационных объектов описывает, если применимо, тип ASN.1, определяемый вместе с текстовым описанием порядка выполнения выбора. Этот компонент должен игнорироваться, если пользователь вставляет в запрос поиска **search** непустой компонент отображения **mapping** конструкции **RelaxationPolicy**.

ПРИМЕЧАНИЕ 11. – В принципе тем же запросом поиска **search** могут быть выбраны несколько экземпляров возможно различных производных классов информационных объектов.

b) Поле **&ApplicableTo** описывает, какие пункты фильтра должны рассматриваться как отображаемые пункты фильтра, путем определения типов атрибутов для таких пунктов фильтра. Все пункты фильтра для любого типа атрибута, перечисленные этим компонентом, подлежат сопоставлению на основе отображения. Этот компонент должен присутствовать всегда. Все перечисленные этим компонентом типы атрибутов необязательно должны присутствовать в данном фильтре. Это значение определяется экземпляром информационного объекта специализации в данном классе информационных объектов.

c) Поле **&subtypesIncluded** является значением булевого типа, которое определяет, может ли экземпляр производного класса информационных объектов в дополнение к определенным типам атрибутов принимать подтипы атрибутов **&ApplicableTo**. В отсутствие значения подтипы разрешены, при условии что они не отключены другими механизмами. Это значение определяется экземпляром информационного объекта производного класса информационных объектов.

d) Поле **&combinable** является значением булевого типа, которое, если равно **TRUE**, разрешает сопоставление на основе отображения с целью использования множественных комбинируемых типов фильтра для достижения совпадения по таблице отображения. **combinable** – это фиктивная ссылка на значение данного компонента, которое должно быть определено специализацией данного класса информационных объектов.

e) Поле **&mappingResults** является фиктивной ссылкой на описание порядка сообщения об условиях исключения. Производный класса информационных объектов определяет тип ASN.1 для сообщения об актуальных условиях исключения.

f) Поле **&userControl** является значением булевого типа, которое определяет, могут ли экземпляр производного класса информационных объектов и его связанное правило сопоставления на основе отображения контролироваться управлением поиском **extendedArea**.

ПРИМЕЧАНИЕ 12. – Если одновременно применяются несколько сопоставлений на основе отображения, может оказаться уместным разрешить только одному из них использовать управление поиском **extendedArea**.

g) Поле **&exclusive** является значением булевого типа, которое определяет, разрешает ли экземпляр производного класса информационных объектов и его связанного правило сопоставления на основе отображения выполнение исключительного ослабления. Это значение, если присутствует, определяется экземпляром информационного объекта производного класса информационных объектов. Если значение равно **FALSE** или если соответствующий DSA не поддерживает исключительного сопоставления для данного сопоставления на основе отображения, это конкретное отображение функционирует, как если бы была установлена опция управления поиском **includeAllAreas**.

ПРИМЕЧАНИЕ 13. – Если одновременно применяются несколько сопоставлений на основе отображения, может оказаться уместным разрешить только одному из них исключительное ослабление.

h) Поле **&matching-rule** является значением типа идентификатора объекта, указывающего правило сопоставления на основе отображения, для которого этот экземпляр предоставляет дополнительную спецификацию и которое используется для сопоставления на основе отображения. Фиктивная ссылка **matchingRule** на значение этого компонента должна определяться специализацией данного класса информационных объектов. Указанное правило сопоставления должно использоваться для конкретного сопоставления на основе отображения.

i) Поле **&id** является идентификатором объекта, распределенного конкретному сопоставлению на основе отображения.

13.7 Определение структуры DIT

13.7.1 Обзор

Основополагающим аспектом схемы Справочника является спецификация места в DIT, куда может быть помещена статья конкретного класса, и порядка ее именования, учитывая:

– иерархические взаимоотношения статей в DIT (правила структуры DIT);

– атрибут или атрибуты, используемые для формирования RDN данной статьи (формы имен).

13.7.2 Определение формы имени

Определение формы имени включает:

a) описание класса именованных объектов;

b) указание обязательных атрибутов, которые должны использоваться для имен RDN статей данного класса объектов, в котором применяется эта форма имени;

c) указание необязательных атрибутов, если имеются, которые могут использоваться для имен RDN статей данного класса объектов, в котором применяется эта форма имени;

d) присвоение идентификатора объекта для формы имени.

Если для данной статьи данного структурного класса объектов требуются разные наборы атрибутов именования, форма имени должна определяться для каждого отдельного набора атрибутов, которые должны использоваться для именования.

В формах имени используются только структурные классы объектов.

Для того, чтобы статьи конкретного структурного класса объектов существовали в части DIB, в применимой части схемы должна содержаться, по крайней мере, одна форма имени для данного класса объектов. Схема содержит дополнительные формы имени по необходимости.

Атрибут RDN (или атрибуты) необязательно должен выбираться из перечня разрешенных атрибутов данного структурного класса объектов, как указано в определении его структурных классов объектов или объектов псевдонимов.

ПРИМЕЧАНИЕ. – Управление атрибутами именования осуществляют правила контента DIT и использование контента DIT таким же образом, что и другими атрибутами.

Форма именования – это лишь элементарный элемент полной спецификации, необходимой для придания DIT такой формы, которую требуют административные органы и органы именования. Остальные аспекты спецификации структуры DIT рассматриваются в п. 13.7.5.

13.7.3 Спецификация формы имени

Формы имени могут быть определены как значения класса информационных объектов **NAME-FORM**:

```
NAME-FORM ::= CLASS {
        &namedObjectClass       OBJECT-CLASS,
        &MandatoryAttributes    ATTRIBUTE,
        &OptionalAttributes     ATTRIBUTE OPTIONAL,
        &id                     OBJECT IDENTIFIER UNIQUE }

WITH SYNTAX {
        NAMES                   &namedObjectClass
        WITH ATTRIBUTES         &MandatoryAttributes
        [ AND OPTIONALLY        &OptionalAttributes ]
        ID                      &id }
```

Для формы имени, которая определяется с использованием данного класса информационных объектов:

a) **&namedObjectClass** – структурный класс объектов, который она называет;

b) **&MandatoryAttributes** – набор атрибутов, которые должны быть представлены в RDN статьи, которой она управляет;

c) **&OptionalAttributes** – набор атрибутов, которые могут быть представлены в RDN статьи, которой она управляет;

d) **&id** – идентификатор объекта, который ей присвоен.

Все типы атрибутов в обязательном и необязательном перечнях должны быть разными.

13.7.4 Структурный класс объектов статьи

Некоторые спецификации подсхемы должны включать формы имени для не более чем одного структурного класса объектов на цепочку суперкласса структурных классов объектов, представленную в этой подсхеме.

Некоторые спецификации подсхемы могут включать формы имени для более чем одного структурного класса объектов на цепочку суперкласса структурных классов объектов, представленную в этой подсхеме

В обоих случаях, в отношении конкретной статьи, только старший подчиненный класс в цепочке структурных суперклассов, представленной в атрибуте **objectClass** этой статьи, определяет правило контента DIT и правило структуры DIT, применяемые к данной статье. Этот класс называется структурным классом объектов статьи и указывается операционным атрибутом **structuralObjectClass**.

13.7.5 Определение правило структуры DIT

Правило структуры DIT является спецификацией, предоставляемой административным органом подсхемы, которую Справочник использует для управления размещением и именованием статей в рамках области действия подсхемы. Все статьи объекта и псевдонима управляются единым правилом структуры DIT. Управляющая поддеревом DIT подсхема содержит, как правило, несколько правил структуры DIT, разрешающих несколько типов статей в пределах этого поддерева.

Определение правила структуры DIT включает:

a) целочисленный идентификатор, который является уникальным в пределах области действия данной подсхемы;

b) указание формы имени для статей, управляемых данным правилом структуры DIT;

c) совокупность разрешенных предшествующих правил структуры, при необходимости.

Набор правил структуры DIT для подсхемы определяет формы выделенных имен для статей, управляемых данной подсхемой.

Правило структуры DIT разрешает статьям в данной подсхеме присоединяться к конкретной форме имени. Форма последнего компонента RDN **DistinguishedName** статьи определяется формой имени правила структуры DIT, управляющего этой статьей.

Компонент **namedObjectClass** формы имени (класс объектов формы имени) соответствует структурному классу объектов этой статьи.

Правило структуры DIT должно разрешать только статьи, принадлежащие структурному классу объектов, определенному его связанной формой имени. Оно не разрешает статьи, принадлежащие любому подклассу этого структурного класса объектов.

По отношению к конкретной статье. правило структуры DIT, управляющее этой статьей, называется *управляющим правилом структуры* статьи. Это правило может быть определено путем анализа атрибута **governingStructureRule** статьи.

По отношению к конкретной статье. правило структуры DIT, управляющее старшей статьей, называется *старшее правило структуры*.

Статья может существовать в DIT только как подчиненная относительно другой статьи (старшей), если правило структуры DIT существует в управляющей подсхеме, которая:

– указывает форму имени для структурного класса объектов этой статьи; и

– либо включает старшее правило структуры статьи как возможное старшее правило структуры, *либо* не определяет старшее правило структуры, в каковом случае эта статья должна быть административной точкой подсхемы.

Если статья, которая сама является административной точкой подсхемы, не включается в подстатью своей подсхемы для целей административного управления подсхемой, то для управления этой статьей используется подсхема из административной области непосредственно предшествующей подсхемы.

Статьи, которые являются статьями административных точек, но не имеют подстатьи подсхемы (например, статьи вновь созданных административных точек), не имеют управляющего правила структуры. Справочник не должен разрешать создание подчиненных статей, младших по отношению к таким статьям, до тех пор пока не будет введена подстатья подсхемы.

Если статья преобразуется в административную точку новой подсхемы, управляющее правило структуры всех статей в новой административной области подсхемы автоматически изменяется на то которое несет с собой новая подсхема.

13.7.6 Спецификация правила структуры DIT

Абстрактный синтаксис правила структуры DIT описывается следующим модулем ASN.1:

```
DITStructureRule ::= SEQUENCE {
    ruleIdentifier          RuleIdentifier ,
                                -- должен быть уникальным в пределах области действия подсхемы
    nameForm                NAME-FORM.&id,
    superiorStructureRules  SET SIZE (1..MAX) OF RuleIdentifier OPTIONAL }

RuleIdentifier ::= INTEGER
```

Соответствие между частями описания, перечисленными в п. 13.7.5, и различными компонентами модуля ASN.1, определенными выше, следующее:

a) компонент **ruleIdentifier** однозначно указывает правило структуры DIT в пределах подсхемы;

b) компонент **nameForm** правила структуры DIT определяет форму имени для статей, управляемых этим правилом структуры DIT;

c) компонент **superiorStructureRules** указывает разрешенные старшие правила структуры для статей, управляемых этим правилом. Если этот компонент опущен, то это правило структуры DIT применяется к административной точке подсхемы.

Для облегчения документирования правил структуры DIT определяется класс информационных объектов **STRUCTURE-RULE**:

```
STRUCTURE-RULE ::= CLASS {
    &nameForm               NAME-FORM,
    &SuperiorStructureRules STRUCTURE-RULE OPTIONAL,
    &id                     RuleIdentifier }

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

13.8 Определение правила контента DIT

13.8.1 Обзор

Правило контента DIT определяет разрешенный контент статей конкретного структурного класса объектов путем определения необязательного множества дополнительных классов объектов, обязательных, необязательных и предотвращенных атрибутов. Коллективные атрибуты включаются в правила контента DIT, если они должны быть разрешены в статье.

Определение правила контента DIT включает:

a) указание структурного класса объектов, к которому оно применяется;

b) факультативно, указание дополнительных классов объектов, разрешенных для статей, которые находятся под управлением этого правила;

c) факультативно, указание, в добавление к запрашиваемым структурным и дополнительным классам объектов, обязательных атрибутов, которые необходимы для статей, управляемых этим правилом контента DIT;

d) факультативно, указание необязательных атрибутов в дополнение к требуемым структурным и дополнительным классам объектов, которые разрешены для управляемых правилом контента DIT статей;

e) факультативно, указание *необязательного(ых)* атрибута(ов) из структурных и дополнительных классов объектов статьи, появление которых в статьях, управляемых этим правилом, исключается.

Для любой действительной спецификации подсхемы максимально существует одно правило контента DIT для каждого структурного класса объектов.

Все статьи в DIT управляется не более чем одним правилом контента DIT. Это правило может быть определено путем анализа значения атрибута **structuralObjectClass** статьи.

Если для структурного класса объектов не представлено ни одного правила контента DIT, тогда статьи этого класса содержат только атрибуты, разрешенные определением этого структурного класса объектов.

Правила контента DIT суперклассов структурного класса объектов для статьи к этой статье не применяются.

Из того, что правило контента DIT связано со структурным классом объектов, следует, что все статьи того же структурного класса объектов имеют то же правило контента DIT независимо от правила структуры DIT, которое управляет их местоположением в DIT.

Управляемая правилом контента DIT статья может быть связана, в дополнение к структурному классу объектов правила структуры DIT, с подмножеством дополнительных классов объектов, указанным этим правилом контента DIT. Эта связь отражается в атрибуте **objectClass** статьи.

Контент статьи должен быть согласован с классами объектов, указанными его атрибутом **objectClass**, следующим образом:

– обязательные атрибуты классов объектов, указанных атрибутом **objectClass**, должны *всегда* присутствовать в этой статье;

– необязательные атрибуты (не указанные как дополнительные необязательные или обязательные в правиле контента DIT) дополнительных классов объектов, определенные правилом контента DIT, могут присутствовать, только если эти дополнительные классы объектов указаны атрибутом **objectClass**.

Обязательные атрибуты, связанные со структурными или указанными дополнительными классами объектов, не должны исключаться в правиле контента DIT.

13.8.2 Спецификация правила контента DIT

Абстрактный синтаксис правила контента DIT описывается следующим модулем ASN.1:

```
DITContentRule ::= SEQUENCE {
    structuralObjectClass           OBJECT-CLASS.&id,
    auxiliaries                     SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
    mandatory               [1]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
    optional                [2]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
    precluded               [3]     SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL }
```

Соответствие между частями определения, перечисленными в п. 13.8.1, и различными компонентами модуля ASN.1, определенными выше, следующее:

a) компонент **structuralObjectClass** указывает структурный класс объектов, к которому применяется данное правило контента DIT;

b) компонент **auxiliaries** указывает дополнительные классы объектов, разрешенные для статьи, к которой применяется данное правило контента DIT;

c) компонент **mandatory** определяет типы атрибутов пользователя, которые статья – объект данного правила контента DIT должна содержать наряду с теми атрибутами, которые она должна содержать согласно своим структурным и дополнительным классам объектов;

d) компоненты **optional** определяют типы атрибутов пользователя которые статья – объект данного правила контента DIT может содержать наряду с теми атрибутами, которые она может содержать согласно своим структурным и дополнительным классам объектов;

e) компонент **precluded** определяет подмножество типов необязательных атрибутов пользователя структурных и дополнительных классов объектов, исключаемых из статьи, к которой применяется данное правило контента DIT.

ПРИМЕЧАНИЕ. – Правила контента для непосредственно определенных атрибутов (например, в перечнях обязательных, необязательных и исключаемых атрибутов) применяют правила только к атрибутам, которые определены ими, и не применяют к атрибутам подтипов и дружественным атрибутам.

Для облегчения документирования правил контента DIT определяется класс информационных объектов **CONTENT-RULE**:

```
CONTENT-RULE ::= CLASS {
    &structuralClass        OBJECT-CLASS.&id     UNIQUE,
    &Auxiliaries            OBJECT-CLASS         OPTIONAL,
    &Mandatory              ATTRIBUTE            OPTIONAL,
    &Optional               ATTRIBUTE            OPTIONAL,
    &Precluded              ATTRIBUTE            OPTIONAL }
```

```
WITH SYNTAX {
        STRUCTURAL OBJECT-CLASS   &structuralClass
        [ AUXILIARY OBJECT-CLASSES &Auxiliaries ]
        [ MUST CONTAIN             &Mandatory ]
        [ MAY CONTAIN              &Optional ]
        [ MUST-NOT CONTAIN         &Precluded ] }
```

13.9 Определение типа контекста

Определение типа контекста включает:

a) описание синтаксиса контекста;

b) описание синтаксиса утверждения о контексте;

c) факультативное описание значения по умолчанию контекста;

d) определение семантики контекста;

e) описание определения совпадений;

f) описание поведения в отсутствие значения контекста; и

g) присвоение идентификатора типу контекста.

13.9.1 Сопоставление значений контекста

Представленное утверждение о контексте совпадает с хранимым значением контекста того же типа контекста в соответствии с описанием сопоставления, которое является частью определения контекста.

13.9.2 Определение контекста

Контексты определяются с использованием класса информационных объектов **CONTEXT**:

```
CONTEXT ::= CLASS {
        &Type,
        &DefaultValue           OPTIONAL,
        &Assertion              OPTIONAL,
        &absentMatch            BOOLEAN DEFAULT TRUE,
        &id                     OBJECT IDENTIFIER UNIQUE }

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

DEFAULT-VALUE нейтрализует эффект 1) **ABSENT-MATCH**. Эффект 2) **ABSENT-MATCH** может допускаться для любого контекста, определенного со значением **DEFAULT-VALUE**, и в этом случае полем **ABSENT-MATCH** можно пренебречь.

Если **&defaultValue** определено, то поведение запросов на изменение статьи путем добавления значений с контекстами будет соответствовать следующим спецификациям предварительной и дополнительной обработки.

ПРИМЕЧАНИЕ. – DSA не обязан реализовывать точную последовательность шагов, описанных ниже, при условии, что конечный результат обеспечивает то же внешне наблюдаемое поведение.

Предварительная обработка

Для всех запросов изменения **EntryModification** путем добавления значений с контекстами, удаления значений с контекстами или удаления всех значений с контекстами. Для всех типов контекстов, применимых к данному типу атрибута, если тип контекста определен со значением **&defaultValue**:

1) если тип контекста не перечислен в явном виде в запросе, добавить к запросу тип контекста со значением **&defaultValue**;

2) для всех сохраняемых значений атрибутов данного типа атрибута, если значение атрибута не имеет типа контекста, добавить к значению атрибута тип контекста со значением **&defaultValue**.

Нормальная обработка

Дополнительная обработка

Для каждого запроса изменения **EntryModification** путем добавления значений с контекстами, удаления значений с контекстами или удаления всех значений с контекстами. Для всех типов контекстов, применимых к данному типу атрибута, если тип контекста определен со значением **&defaultValue**, тогда для всех сохраняемых значений атрибутов данного типа атрибута,

3) если значение атрибута не имеет типа контекста, удалить значение атрибута;

4) если значение атрибута не имеет типа контекста, а единственным значением контекста этого типа контекста является **&defaultValue**, удалить контекст (но не значение этого атрибута).

Если **&Assertion** опущен, синтаксис утверждения о контексте является тем же, что и **&Type**.

Описание **&absentMatch** как **FALSE** в определении контекста приводит к двум следующим результатам:

a) Значение атрибута, которое не имеет контекста описанного типа контекста, интерпретируется как не имеющее значений этого типа контекста. Это значит, что если значение атрибута не содержит контекстов типа контекста, к которому относится утверждение, **contextType**, то **ContextAssertion** выдает оценку FALSE.

b) Компонент значений контекста такого типа контекста интерпретируется как установленный в значение FALSE независимо от его реальных установок.

Если контекст определен, спецификация должна включать описание семантики и того, как определяется совпадение.

В Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6 содержатся избранные определения контекстов.

13.10 Определение использования контекста DIT

13.10.1 Обзор

Использование контекста DIT является спецификацией, выдаваемой административным органом подсхемы для определения допустимых типов контекста, которые могут сохраняться с атрибутом, и обязательных типов контекстов, которые должны сохраняться с атрибутом.

Определение использования контекста DIT включает:

a) указание типа атрибута, к которому оно применяется;

b) факультативно, указание обязательных типов контекста, которые должны быть связаны со значениями данного типа атрибута, независимо от места хранения этого атрибута;

c) факультативно, указание необязательных типов контекст, которые могут быть связаны со значениями данного типа атрибута, независимо от места хранения этого атрибута.

Если определение использования контекста DIT отсутствует для данного типа атрибута, значения атрибутов этого типа не должны содержать перечней контекстов. Для данной административной области подсхемы может существовать только одно использование контекста DIT для данного типа атрибута. Использование контекста DIT может определяться для применения ко всем типам атрибутов, и в этом случае оно должно быть единственным использованием контекста DIT в данной подсхеме.

13.10.2 Спецификация использования контекста DIT

Абстрактный синтаксис использования контекста DIT описывается следующим модулем ASN.1:

```
DITContextUse ::= SEQUENCE {
    attributeType               ATTRIBUTE.&id,
    mandatoryContexts[1]        SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL,
    optionalContexts    [2]     SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL }
```

Соответствие между частями определения, перечисленными в п. 13.10.1, и различными компонентами модуля ASN.1, определенными выше, следующее:

a) компонент **attributeType** указывает тип атрибута, к которому применяется использование контекста DIT, или любой тип атрибута (**id-oa-allAttributeTypes**);

b) компонент **mandatoryContexts** определяет типы контекста, который должен быть связан со значением атрибута данного типа независимо от места хранения этого атрибута. Если данный компонент опущен, значения атрибута могут существовать без перечней контекстов;

c) компонент **optionalContexts** определяет типы контекста, который может быть связан со значением атрибута данного типа независимо от места хранения этого атрибута. Если этот компонент опущен, но присутствует компонент mandatoryContexts, все значения атрибутов должны присутствовать с обязательными типами контекста и не с какими иными. Если этот компонент опущен и опущен также компонент mandatoryContexts, это эквивалентно отсутствию использования контекста DIT для данного

типа атрибута; т. е. значения атрибутов данного типа атрибута не должны быть связаны с перечнями контекстов.

Для облегчения документирования правил использования DIT определяется класс информационных объектов **DIT-CONTEXT-USE-RULE**:

```
DIT-CONTEXT-USE-RULE ::= CLASS {
        &attributeType              ATTRIBUTE.&id    UNIQUE,
        &Mandatory                  CONTEXT          OPTIONAL,
        &Optional                   CONTEXT          OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE              &attributeType
        [ MANDATORY CONTEXTS        &Mandatory ]
        [ OPTIONAL CONTEXTS         &Optional ] }
```

13.11 Определение "друзей"

Определение множества "друзей" включает:

a) определение опорного атрибута, который имеет это множества "друзей";

b) определение набора атрибутов, которые являются "друзьями" опорного.

Для облегчения документирования правил использования DIT определяется класс информационных объектов **FRIENDS**:

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

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

Любой данный атрибут может иметь только одно множество "друзей" в любой подсхеме.

Пример:

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

14 Схема системы Справочника

14.1 Обзор

Схемой системы Справочника является совокупность определений и ограничений, касающихся информации, которая должна быть известна самому Справочнику для обеспечения корректного функционирования. Эта информация определяется в терминах подстатей и операционных атрибутов.

ПРИМЕЧАНИЕ. – Схема системы позволяет системе справочника, например:

– предотвращать связь подстатей неверного типа с административными статьями (например, создание подстатьи подсхемы, подчиненной по отношению к административной статье, которая определена исключительно как административная статья безопасности);

– предотвращать добавление несоответствующих операционных атрибутов к статье или подстатье (например, операционный атрибут подсхемые к статье, относящейся к индивидууму).

Формально схему Справочника образует совокупность:

a) описаний классов объектов, определяющих атрибуты, которые должны или могут присутствовать в данном классе;

b) описаний типов операционных атрибутов, определяющих характеристики операционных атрибутов, которые известны и используются Справочником.

Полное описание операционного атрибута включает спецификацию порядка использования Справочником этого атрибута и (в соответствующих случаях) порядка предоставления или управления этим атрибутом в процессе своей работы.

Схема системы Справочника является распределенной, также как сама DIB. Каждый административный орган устанавливает ту часть схемы системы, которая будет использоваться для тех частей DIB, административное руководство которыми осуществляет этот орган.

Схема системы Справочника, определяемая в настоящей спецификации Справочника, является неотъемлемой частью самой системы Справочника. Для каждого принимающего участие в системе справочника DSA требуется полное знание о схеме системы, установленной его административным органом. Схема системы для административной области может определяться административным органом, с использованием нотации, содержащейся в данном пункте.

Схема системы Справочника не регулируется правилами структуры или контента DIT. При определении элемента схемы системы выдается спецификация порядка его использования и мест его нахождения в DIT.

В последующих пунктах описываются некоторые аспекты схемы системы справочника.

Схема системы справочника, необходимая для поддержки распределения справочника, определяется в пп. 25–28.

14.2 Схема системы, поддерживающая модель административной и операционной информации

Несмотря на то что **subentry** и **subentryNameForm** описываются с использованием нотации п. 13, подстатьи не регулируются правилами структуры DIT или контента DIT.

14.2.1 Класс объектов подстатьи

Класс объектов подстатьи **subentry** является структурным классом объектов и определяется следующим образом:

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

14.2.2 Форма имени подстатьи

Форма и имени подстатьи **subentryNameForm** разрешает именование статей класса **subentry** с использованием атрибута **commonName**:

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

Для подстатей не используются никакие иные формы имени.

14.2.3 Операционный атрибут спецификации поддерева

Операционный атрибут **subtreeSpecification**, семантика которого определяется в п. 12, определяется следующим образом:

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

Этот атрибут представлен во всех подстатьях; каждое значение определяет совокупность статей (в терминах части административной области возможно с уточнением путем выбора с помощью фильтра класса объектов), которые могут быть субъектами стратегий, определенных этой подстатьей.

ПРИМЕЧАНИЕ. – Это позволяет применять единую комплексную стратегию (например, правило поиска) в отношении многочисленных комбинаций классов объектов, в непересекающихся районах административной области, определив эту стратегию в одной подстатье.

14.3 Схема системы, поддерживающая административную модель

Административная модель, определенная в п. 11, требует, чтобы административные статьи содержали атрибут **administrativeRole** для указания того, что связанная административная область выполняет одну или более административных функций.

Операционный атрибут административной функции **administrativeRole** определяется следующим образом:

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

Настоящая спецификация Справочника определяет следующие значения этого атрибута:

id-ar-autonomousArea
id-ar-accessControlSpecificArea
id-ar-accessControlInnerArea
id-ar-subschemaAdminSpecificArea
id-ar-collectiveAttributeSpecificArea
id-ar-collectiveAttributeInnerArea
id-ar-contextDefaultSpecificArea
id-ar-serviceSpecificArea

Семантика этих значений определяется в п. 12.

Операционный атрибут административной функции **administrativeRole** используется также для управления подстатьями, которым разрешено быть подчиненными по отношению к административной статье. Подстатья не относящаяся к классу, который разрешен атрибутом **administrativeRole**, не может быть подчиненной по отношению к административной статье.

14.4 Схема системы, поддерживающая общие административные и операционные требования

В последующих пунктах описываются операционные атрибуты подсхемы, которые не являются атрибутами в обычном смысле (т. е. не содержатся в пределах статьи), но могут рассматриваться как "виртуальные" атрибуты, представляющие информацию, которая может быть выведена (например, из существующих операционных атрибутов, их значений и другой информации). такие виртуальные атрибуты действительны для всех статей в пределах административной области. Результатом этого является представление операционных атрибутов этой подсхемы во всех статьях.

14.4.1 Метки времени

createTimestamp указывает время создания статьи:

```
createTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- согласно п. 42.3 b) или c) Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-createTimestamp }
```

modifyTimestamp указывает время изменения статьи:

```
modifyTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- согласно п. 42.3 b) или c) Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
        ORDERING MATCHING RULE          generalizedTimeOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-modifyTimestamp }
```

subschemaTimestamp указывает время создания или последнего по времени изменения подстатьи подсхемы для данной статьи (см. п. 15.3). Атрибут существует в каждой статье:

```
subschemaTimestamp  ATTRIBUTE  ::= {
        WITH SYNTAX                     GeneralizedTime
                        -- согласно п. 42.3 b) или c) Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1
        EQUALITY MATCHING RULE          generalizedTimeMatch
```

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

Правила сопоставления **generalizedTimeMatch** и **generalizedTimeOrderingMatch** определяются в Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6.

14.4.2 Операционные атрибуты модификатора статьи

Операционный атрибут **creatorsName** указывает выделенное имя пользователя Справочника, который создал статью:

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

Операционный атрибут **modifiersName** указывает выделенное имя пользователя Справочника, который осуществил последнее по времени изменение статьи:

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

Эти операционные атрибуты должны использовать первичное выделенное имя.

14.4.3 Операционные атрибуты идентификации подстатьи

Операционный атрибут **subschemaSubentryList** указывает подстатью подсхемы, которая управляет статьей. Атрибут существует в каждой статье:

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

Операционный атрибут **accessControlSubentryList** указывает все подстатьи управления доступом, которые затрагивают данную статью. Атрибут существует в каждой статье.

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

Операционный атрибут **collectiveAttributeSubentryList** указывает все подстатьи коллективных атрибутов, которые затрагивают данную статью. Атрибут существует в каждой статье:

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

Операционный атрибут **contextDefaultSubentryList** указывает все подстатьи контекстов по умолчанию, которые затрагивают данную статью. Атрибут существует в каждой статье:

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

Операционный атрибут **serviceAdminSubentryList** указывает все подстатьи администрации служб, если таковые существуют, которые затрагивают данную статью. Атрибут существует в каждой статье, затрагиваемой такой подстатьей:

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

14.4.4 Операционный атрибут наличия подчиненной статьи

Операционный атрибут **hasSubordinates** указывает, существует ли ниже содержащей данный атрибут статьи какие-либо подчиненные статьи, обладающие данным атрибутом. Значение **TRUE** показывает, что подчиненные могут существовать. Значение **FALSE** показывает, что подчиненных не существует. Если этот атрибут отсутствует, информация о существовании подчиненных статей не предоставляется. Этот атрибут, как правило, обнаруживает существование подчиненных, даже если непосредственно подчиненные скрыты опциями управления доступом – в целях предотвращения обнаружения существования подчиненных этот операционный атрибут сам должен находиться под защитой функций управления доступом.

ПРИМЕЧАНИЕ. – Значение **TRUE** может возвращаться, когда подчиненных не существует, если все возможные подчиненные доступны только по не зависящей от подчиненных ссылке (см. Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4) или если единственными подчиненными являются подстатьи или дочерние члены семейства.

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

14.5 Схема системы, поддерживающая управление доступом

Если подстатья содержит предписывающую информацию управления доступом, то ее атрибут **objectClass** должен содержать значение **accessControlSubentry**:

```
accessControlSubentry  OBJECT-CLASS  ::= {
        KIND            auxiliary

        ID              id-sc-accessControlSubentry }
```

Подстатья этого объекта должна содержать исключительно один предписывающий атрибут ACI такого типа, который согласуется со значением атрибута **accessControlScheme** соответствующей конкретной точки управления доступом.

14.6 Схема системы, поддерживающая модель коллективных атрибутов

Подстатьи, поддерживающие специальные или внутренние административные области коллективных атрибутов, определяются следующим образом:

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

Подстатья этого класса объектов должна содержать по крайней мере один коллективный атрибут.

Коллективные атрибуты, содержащиеся в пределах подстатьи данного класса объектов, концептуально доступны для запросов и фильтрации в каждой статье в пределах области действия атрибута **subtreeSpecification** данной подстатьи, но административное управление ими осуществляется через эту подстатью.

Операционный атрибут **collectiveExclusions** позволяет исключать из статьи конкретные коллективные атрибуты:

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

Этот атрибут для всех статей является необязательным.

Значение **id-oa-excludeAllCollectiveAttributes** идентификатора **OBJECT IDENTIFIER** может использоваться, в форме присутствия в качестве значения атрибута **collectiveExclusions**, для исключения всех коллективных атрибутов из статьи.

14.7 Схема системы, поддерживающая значения утверждения о контексте по умолчанию

Подстатьи, предоставляющие значения по умолчанию для утверждений о контексте, определяются следующим образом:

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

Подстатья данного класса объектов должна содержать атрибут **contextAssertionDefaults**:

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

Всякий раз, когда оценивается контекст, а утверждения о контексте пользователем не предоставлено, Справочник предоставляет значения утверждения о контексте по умолчанию, равные значениям этого атрибута в подстатье утверждения о контексте, управляющей статьей, к которой осуществляется доступ, согласно описанию в п. 8.9.2.2.

ПРИМЕЧАНИЕ. – **TypeAndContextAssertion** определяется в п. 7.6 (а его оценка определяется в п. 7.6.3) Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

14.8 Схема системы, поддерживающая модель административного управления службы

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

Подстатья данного класса объектов должна содержать операционный атрибут **searchRules**:

```
searchRules ATTRIBUTE ::= {
        WITH SYNTAX                     SearchRuleDescription
        EQUALITY MATCHING RULE          integerFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-oa-searchRules }

SearchRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF       SearchRule,
        name                [28]    SET SIZE (1 .. MAX) OF DirectoryString { ub-search } OPTIONAL,
        description         [29]    DirectoryString { ub-search } OPTIONAL }
```

Значение операционного атрибута **searchRules** является либо правилом поиска, содержащим действительные ограничения поиска, либо фиктивным правилом поиска, которое не определяет каких-либо ограничений поиска. Такое фиктивное правило поиска определяется наличием **id** нуля и отсутствием компонента **serviceType** (или любого иного компонента правила **SearchRule**, отличного от **id** и **dmdId**). **dmdId** – это идентификатор для управления DMD (см. п. 6.4).

14.9 Схема системы, поддерживающая иерархические группы

```
hierarchyLevel ATTRIBUTE  ::= {
        WITH SYNTAX                 HierarchyLevel
        EQUALITY MATCHING RULE      integerMatch
        ORDERING MATCHING RULE      integerOrderingMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-oa-hierarchyLevel }

HierarchyLevel  ::=  INTEGER

hierarchyBelow ATTRIBUTE  ::= {
        WITH SYNTAX                 HierarchyBelow
        EQUALITY MATCHING RULE      booleanMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-oa-hierarchyBelow }

HierarchyBelow  ::=  BOOLEAN

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

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

Операционный атрибут **hierarchyLevel** должен быть представлен во всех статьях, которые являются членами иерархической группы. Справочник создает и сопровождает этот атрибут. Справочник удаляет этот атрибут, если статья перестает быть членом иерархической группы. Для иерархической вершины данный атрибут принимает значение нуль. Атрибут не присутствует в дочернем члене семейства.

Операционный атрибут **hierarchyBelow** указывает, имеет ли данная статья каких-либо иерархических дочерних членов. Значение **TRUE** показывает, что иерархические дочерние члены существуют. Значение **FALSE** или отсутствие этого типа атрибута показывает, что иерархических дочерних членов не существует. Справочник создает и сопровождает этот атрибут. Справочник удаляет этот атрибут, если статья перестает быть членом иерархической группы.

Атрибут **hierarchyParent** должен быть представлен в операции добавления статьи (Add Entry) или изменения статьи (Modify Entry), когда новая статья или существующая статья становится иерархическим дочерним членом. Значением атрибута должно быть выделенное имя непосредственно иерархического родителя. Если непосредственно иерархическим родителем является составная статья, значением должно быть выделенное имя порождающего элемента. В ином случае Справочник должен вернуть сообщение об ошибке обновления (Update Error) с указанием причины **parentNotAncestor**. Этот атрибут не должен присутствовать в дочернем члене семейства, в статье, которая не входит в иерархическую группу, и в статье, являющейся иерархической вершиной.

Атрибут **hierarchyTop** указывает на статью вершины данной иерархической группы. Этот атрибут предоставляется и сопровождается Справочником. Значением атрибута должно быть выделенное имя статьи вершины. Если статьей вершины является составная статья, значением должно быть выделенное имя порождающего элемента. Этот атрибут не должен присутствовать в дочернем члене семейства, в статье, которая не входит в иерархическую группу, и в статье, являющейся иерархической вершиной.

ПРИМЕЧАНИЕ. – Этот атрибут обеспечивает однозначное обозначение иерархической группы, которой принадлежит данная статья.

Если статья в пределах иерархической группы удаляется операцией удаления статьи (Remove Entry), из этой иерархической группы удаляются все ее иерархические потомки.

14.10 Сопровождение схемы системы

Ответственность за поддержание согласования подстатей и операционных атрибутов со схемой системы лежит на агентах DSA. Не должно возникать несогласованности между различными аспектами схемы системы и между схемой системы и подстатьями и операционными атрибутами.

Справочник выполняет процедуры добавления и изменения статей всякий раз, когда к DIT добавляется новая подстатья или изменяется существующая подстатья. Справочник должен определять, не нарушит ли предложенная операция схему системы, и если нарушит, изменение выполняться не должно.

В частности, Справочник обеспечивает, чтобы добавляемые к DIT подстатьи были согласованы со значениями атрибута **administrativeRole**, чтобы атрибуты в пределах подстатьи были согласованы со значениями атрибута **objectClass** этой подстатьи.

Значение атрибута **administrativeRole** может быть изменено для разрешения того, чтобы классы подстатей, которые еще не представлены, могли быть подчиненными по отношению к административной статье. Значение атрибута **administrativeRole** не должно изменяться так, чтобы вызвать рассогласование существующих подстатей.

Справочник также обеспечивает корректность значений операционных атрибутов, предоставляемых Справочником.

14.11 Схема системы для подчиненных первого уровня

Справочник обеспечивает выполнение следующих правил и ограничений, применяемых к статьям, которые созданы как непосредственно подчиненные по отношению к корню DIT:

- все такие статьи должны быть созданы как статьи административной точки;
- атрибуты класса объекта и атрибуты именования таких статей должны соответствовать определенным в Рек. МСЭ-Т X.660 | ИСО/МЭК 9834-1.

15 Административное управление схемой Справочника

15.1 Обзор

Общее административное управление схемой справочника глобального DIT реализуется через независимое административное управление подсхемами автономных административных областей доменов DIT, которые образуют глобальное DIT.

Координация административного управления схемой справочника на границах между доменами DIT является предметом двустороннего соглашения между DMO и не входит в область действия настоящей спецификации Справочника.

Административные возможности подсхемы, определенные в настоящем пункте для целей управления доменом DIT, включают:

a) создание, удаление и изменение подстатей подсхемы;

b) поддержку механизма публикаций, с тем чтобы разрешить агентам DSA включать информацию схемы в рабочие обмены по обязательному протоколу, а агентам DUA – извлекать информацию подсхемы через DAP;

c) управление подсхемой для целей обеспечения выполнения любых операций изменения в соответствии с применимой спецификацией подсхемы.

15.2 Объекты стратегии

Объект стратегии подсхемы может быть одним из следующих:

- административная область подсхемы;
- статья объекта или псевдонима в пределах административной области подсхемы;
- атрибут пользователя такой статьи объекта или псевдонима.

Автономная административная область может быть обозначена как специальная административная область подсхемы в целях административного управления этой подсхемой. На это должно указывать наличие значения **id-oa-subschemaAdminSpecificArea** в атрибуте **administrativeRole** связанной административной статьи (в дополнение к наличию значения **id-oa-autonomousArea** и, возможно, других значений).

Такая автономная область может быть разбита на части для использования подсхемы конкретных частей и административного управления ею. В этом случае административные статьи для каждой из специальных административных областей подсхемы указываются наличием значения **id-oa-subschemaAdminSpecificArea** в атрибутах **administrativeRole** этих статей.

15.3 Параметры стратегии

Параметры стратегии подсхемы используются для формулирования стратегии административного органа подсхемы. Этими параметрами и используемыми для их представления операционными атрибутами являются следующие:

- параметр *структура DIT*: используется для определения структуры административной области подсхемы и хранения информации об устаревших правилах структуры DIT, которые могли обозначить некоторые статьи в качестве своего управляющего правила структуры DIT. Этот параметр представляется операционными атрибутами **dITStructureRules** и **nameForms**;
- параметр *контент DIT*: используется для определения типа контента статей объекта и псевдонима, которые содержатся в пределах административной области подсхемы, и для хранения информации об устаревших правилах контента DIT, которые Справочник мог использовать при определении контента некоторых старей. Этот параметр представляется операционными атрибутами **dITContentRules**, **objectClasses, attributeTypes**, **contextTypes**, **friends** и **dITContextUse**;
- параметр *возможность сопоставления*: используется для определения возможностей сопоставления, поддерживаемых правилами сопоставления, которые применяются к типам атрибутов, определенным в административной области подсхемы. Этот параметр представляется операционными атрибутами **matchingRules** и **matchingRuleUse**.

Для административного управления подсхемой в административной области этой подсхемы органом подсхемы используется единая подстатья подсхемы. С этой целью подстатья подсхемы содержит операционные атрибуты, представляющие параметры стратегии, которые применяются для выражения стратегии подсхемы. Атрибут **subtreeSpecification** подстатьи подсхемы должен описывать всю административную область подсхемы, т. е. он должен быть пустой последовательностью.

Подстатья подсхемы описывается следующим образом:

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

Операционные атрибуты подстатьи подсхемы определяются в п. 15.7.

15.4 Процедуры осуществления стратегии

С административным управлением подсхемой связаны две процедуры стратегии:

- процедура изменения подсхемы;
- процедура изменения статьи.

15.5 Процедуры изменения подсхемы

Орган подсхемы может осуществлять административное управление ею динамическим образом, включая внесение в подсхему ограничивающих изменений. Это может быть выполнено путем изменения значений операционных атрибутов подсхемы с помощью операций изменения Справочника, фактически меняя таким образом подсхему, действующую в административной области этой подсхемы. Орган подсхемы может также создавать новые области подсхемы или удалять существующие области подсхемы путем создания или удаления подстатей подсхемы, соответственно.

До того как орган подсхемы расширит структуру DIT или правила контента DIT путем добавления нового правила или же путем добавления дополнительного класса объектов, или обязательного или необязательного атрибута к существующему правилу, ссылочная информация схемы должна быть описана в соответствующем атрибуте в подстатье подсхемы. Формы имени, классы объектов, типы атрибутов и правила сопоставления, на которые делаются ссылки (прямо или косвенно) через атрибут **dITStructureRule**, **dITContentRule** или **matchingRuleUse**, не должны удаляться из подстатьи подсхемы.

Определения информационных объектов, таких как классы объектов, типы атрибутов, правила сопоставления и формы имени, которые были зарегистрированы (т. е. им было присвоено имя идентификатора объекта типа) являются статическими и не могут быть изменены. Изменения семантики таких информационных объектов требуют присвоения новых идентификаторов объекта.

Правила структуры DIT и контента DIT могут быть активными или устаревшими. Для управления DIT используются только активные правила. Идентификация и сохранность устаревших правил является административным механизмом, который разрешает размещение (и, возможно, восстановление) статей, добавленных по старым правилам, которые с тех пор изменились.

Этот механизм устаревания должен использоваться, когда внесенные ограничивающие изменения в структуру DIT или правила контента DIT создают несогласованность в DIB; в любом ином случае соответствующее активное правило может быть изменено напрямую. Справочник допускает удаление устаревших правил в любой момент времени.

ПРИМЕЧАНИЕ. – Механизм устаревания, предусмотренный в операционных атрибутах подсхемы, обеспечивает возможность идентифицирования и восстановления всех статей с устаревшей схемой до удаления операционного атрибута устаревшей подсхемы.

Ответственность за поддержание согласованности статей и активной подсхемы посредством абстрактной службы Справочника или иными локальными средствами лежит на административном органе подсхемы. Это может выполняться так, как удобно для административного органа подсхемы. Не существует определения того, когда должны быть выполнены такие корректировки несогласованных статей. Вместе с тем, удаление устаревших правил до размещения и восстановления несогласованных статей значительно усложнит эту задачу

15.6 Процедуры добавления и изменения статей

Справочник выполняет процедуры добавления и изменения статей всякий раз, когда в DIT добавляется новая статья или изменяется существующая статья. Справочник должен определять, не нарушит ли предлагаемая операция стратегию подсхемы.

В частности, Справочник должен обеспечивать согласованность добавленных в DIT статей с соответствующими активными правилами структуры DIT и контента DIT.

Справочник должен разрешать запрос статей, которые не согласуются со своими активными правилами.

Справочник обеспечивает выполнение активных правил при получении запроса на изменение DIB. В случае несогласованности статьи со своим активным правилом запрос на изменение этой статьи разрешается, если его выполнение устранит существующую несогласованность или не внесет дополнительной несогласованности. Запрос, вносящий дополнительную несогласованность, не выполняется.

Для любой действительной статьи в действительной административной области подсхемы может существовать только один старший подчиненный структурный класс объектов в цепочке суперкласса структурных классов объектов. При добавлении статьи в DIT Справочник определяет этот старший подчиненный структурный класс объектов по предоставленным значениям атрибута **objectClass** и создает постоянную его связь с этой статьей через атрибут данной статьи **structuralObjectClass**.

При создании статьи значения атрибута **objectClass** предоставляются таким образом, чтобы контент статьи был согласованным с правилом контента DIT, управляющим этой статьей. В частности, если значение атрибута **objectClass** указывает конкретный класс объектов, имеющий суперклассы, отличные от **top**, должны предоставляться также значения для всех этих суперклассов. В ином случае операция Справочника, создающая статью, не должна выполняться.

Пользователи Справочника могут впоследствии добавлять или удалять значения атрибута **objectClass** для дополнительного класса объектов статьи. При внесении изменений в значения атрибута **objectClass** контент статьи должен оставаться согласованным с управляющим этой статьей правилом контента DIT. В частности, если значение атрибута **objectClass** указывает, что добавлен или удален конкретный класс объектов, имеющий суперклассы, отличные от **top**, то должны быть также добавлены или удалены значения для всех этих суперклассов, за исключением случаев, когда такие суперклассы также представлены в цепочках суперклассов, связанных с другими значениями, которые, соответственно, не добавляются или не удаляются.

15.7 Атрибуты стратегии подсхемы

Следующие подклассы определяют операционные атрибуты стратегии подсхемы. Эти атрибуты:

- представлены в подстатье подсхемы. Административное управление значениями этих атрибутов осуществляется через операции изменения Справочника с использованием выделенного имени подстатьи подсхемы;
- доступны для запроса во всех статьях, управляемых этой подсхемой.

Параметризованный тип ASN.1 **DirectoryString { ub-schema }**, используемый в следующих определениях, описан в Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6.

Правила сопоставления равенства **integerFirstComponentMatch** и **objectIdentifierFirstComponentMatch** также определяются в Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6.

Для целей управления факультативно разрешены ряд воспринимаемых человеком компонентов **name** и компонент **description** в качестве компонентов некоторых операционных атрибутов стратегии подсхемы, которые определяются в последующих пунктах.

Ряд операционных атрибутов стратегии подсхемы, определяемых в последующих пунктах, содержат компонент устаревания **obsolete**. Этот компонент используется для указания того, активным или устаревшим в административной области подсхемы является данное определение.

15.7.1 Операционный атрибут правил структуры DIT

Операционный атрибут **dITStructureRules** определяет правила структуры DIT, действующие в пределах подсхемы:

```
dITStructureRules ATTRIBUTE  ::= {
        WITH SYNTAX                         DITStructureRuleDescription
        EQUALITY MATCHING RULE              integerFirstComponentMatch
        USAGE                               directoryOperation
        ID                                  id-soa-dITStructureRule }

DITStructureRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF  DITStructureRule,
        name                [1]     SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description                 DirectoryString { ub-schema } OPTIONAL,
        obsolete                    BOOLEAN DEFAULT FALSE }
```

Операционный атрибут **dITStructureRules** содержит множественные значения; каждое значение определяет одно правило структуры DIT.

Компоненты атрибута **dITStructureRule** имеют ту же семантику, что и соответствующее определение ASN.1 в п. 13.7.6.

15.7.2 Операционный атрибут правил контента DIT

Операционный атрибут **dITContentRules** определяет правила контента DIT, действующие в пределах подсхемы. Каждое значение данного операционного атрибута помечено идентификатором объекта структурного класса объектов, к которому он относится:

```
dITContentRules ATTRIBUTE  ::= {
        WITH SYNTAX                         DITContentRuleDescription
        EQUALITY MATCHING RULE              objectIdentifierFirstComponentMatch
        USAGE                               directoryOperation
        ID                                  id-soa-dITContentRules }

DITContentRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF           DITContentRule,
        name                    [4]     SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description                     DirectoryString { ub-schema }       OPTIONAL,
        obsolete                        BOOLEAN DEFAULT FALSE }
```

Операционный атрибут **dITContentRules** содержит множественные значения; каждое значение определяет одно правило контента DIT.

Компоненты атрибута **dITContentRule** имеют ту же семантику, что и соответствующее определение ASN.1 в п. 13.8.2.

15.7.3 Операционный атрибут правил сопоставления

Операционный атрибут **matchingRules** определяет правила сопоставления, действующие в пределах подсхемы:

```
matchingRules  ATTRIBUTE  ::= {
        WITH SYNTAX                         MatchingRuleDescription
        EQUALITY MATCHING RULE              objectIdentifierFirstComponentMatch
        USAGE                               directoryOperation
        ID                                  id-soa-matchingRules }

MatchingRuleDescription  ::=  SEQUENCE {
        identifier              MATCHING-RULE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
```

```
        description         DirectoryString { ub-schema }       OPTIONAL,
        obsolete            BOOLEAN                             DEFAULT FALSE,
        information   [0]   DirectoryString { ub-schema }       OPTIONAL }
                            -- описывает синтаксис ASN.1
```

Компонентом **identifier** значения атрибута **matchingRules** является идентификатор объекта, определяющий правило сопоставления.

Компонент **description** содержит описание на естественном языке алгоритмов, связанных с этим правилом.

Компонент **information** содержит определение на языке ASN.1 синтаксиса утверждения данного правила.

Такое определение на языке ASN.1 должно даваться как необязательное порождение импорта ASN.1, за которым следует необязательное порождение присвоения ASN.1 и, далее, порождение типа ASN.1. Все имена записей, определенные в модулях Справочника, являются неявно импортируемыми и не требуют явного импорта. Все имена записей, импортируемые или определенные через функцию присвоения, являются локальными по отношению к определению этого синтаксиса. Если запись ASN.1 включает определенное пользователем ограничение и не является одним из типов ASN.1, определенных в модулях Справочника, то последний UserDefinedConstraintParameter этого ограничения должен быть реальным параметром, управляющим типом которого является **SyntaxConstraint** и значением которого является идентификатор объекта, присвоенный данному ограничению.

```
SyntaxConstraint ::= OBJECT IDENTIFIER
```

ПРИМЕЧАНИЕ 1. – Порождения операций импорта, присваивания и типа языка ASN.1 определяются в Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1. **UserDefinedConstraintParameter** определяется в Рек. МСЭ-Т X.682 | ИСО/МЭК 8824-3.

ПРИМЕЧАНИЕ 2. – Типичным определением на языке ASN.1 является просто имя типа.

Операционный атрибут **matchingRules** содержит множественные значения; каждое значение определяет одно правило сопоставления.

15.7.4 Операционный атрибут типов атрибута

Операционный атрибут **attributeTypes** определяет типы атрибутов, используемые в пределах подсхемы:

```
attributeTypes ATTRIBUTE ::= {
        WITH SYNTAX                 AttributeTypeDescription
        EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
        USAGE                       directoryOperation
        ID                          id-soa-attributeTypes }

AttributeTypeDescription  ::=  SEQUENCE  {
        identifier          ATTRIBUTE.&id,
        name                SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description         DirectoryString { ub-schema }       OPTIONAL,
        obsolete            BOOLEAN                             DEFAULT FALSE,
        information   [0]   AttributeTypeInformation }
```

Компонент **identifier** значения атрибута **attributeTypes** является идентификатором типа объекта, указывающим тип атрибута.

Операционный атрибут **attributeTypes** содержит множественные значения; каждое значение определяет один тип атрибута:

```
AttributeTypeInformation  ::=  SEQUENCE  {
        derivation          [0]     ATTRIBUTE.&id OPTIONAL,
        equalityMatch       [1]     MATCHING-RULE.&id           OPTIONAL,
        orderingMatch       [2]     MATCHING-RULE.&id           OPTIONAL,
        substringsMatch     [3]     MATCHING-RULE.&id           OPTIONAL,
        attributeSyntax     [4]     DirectoryString { ub-schema }   OPTIONAL,
        multi-valued        [5]     BOOLEAN                     DEFAULT TRUE,
        collective          [6]     BOOLEAN                     DEFAULT FALSE,
        userModifiable      [7]     BOOLEAN                     DEFAULT TRUE,
        application         AttributeUsage                      DEFAULT  userApplications }
```

Компоненты **derivation**, **equalityMatch**, **attributeSyntax**, **multi-valued**, **collective** и **application** имеют ту же семантику, что и эквивалентные порции нотации, введенной соответствующим классом информационных объектов.

Компонент **attributeSyntax** содержит текстовую строку, дающую определение ASN.1 синтаксиса атрибута. Такое определение ASN.1 должно даваться как описанное для компонента **information** операционного атрибута правил сопоставления.

15.7.5 Операционный атрибут классов объектов

Операционный атрибут **objectClasses** определяет классы объектов, используемые в пределах подсхемы.

```
objectClasses ATTRIBUTE  ::= {
        WITH SYNTAX                     ObjectClassDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-objectClasses }

ObjectClassDescription  ::=  SEQUENCE  {
        identifier              OBJECT-CLASS.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN                                                 DEFAULT FALSE,
        information   [0]       ObjectClassInformation }
```

Компонент **identifier** значения атрибута **objectClasses** является идентификатором объекта, указывающим класс объектов.

Операционный атрибут **objectClasses** содержит множественные значения; каждое значение определяет один класс объектов:

```
ObjectClassInformation  ::=  SEQUENCE  {
        subclassOf              SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
        kind                    ObjectClassKind                         DEFAULT structural,
        mandatories [3]         SET SIZE (1..MAX) OF ATTRIBUTE.&id      OPTIONAL,
        optionals     [4]       SET SIZE (1..MAX) OF ATTRIBUTE.&id      OPTIONAL }
```

Компоненты **subclassOf**, **kind**, **mandatories** и **optionals** имеют ту же семантику, что и соответствующие порции нотации, введенной соответствующим классом информационных объектов.

15.7.6 Операционный атрибут форм имени

Операционный атрибут **nameForms** определяет формы имени, используемые в пределах подсхемы.

```
nameForms ATTRIBUTE  ::= {
        WITH SYNTAX                     NameFormDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-nameForms }

NameFormDescription  ::=  SEQUENCE  {
        identifier              NAME-FORM.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description             DirectoryString { ub-schema } OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE,
        information   [0]       NameFormInformation }
```

Компонент **identifier** значения атрибута **nameForms** является идентификатором объекта, указывающим класс объектов.

Операционный атрибут **nameForms** содержит множественные значения; каждое значение определяет одну форму имени:

```
NameFormInformation  ::=  SEQUENCE  {
        subordinate             OBJECT-CLASS.&id,
        namingMandatories       SET OF ATTRIBUTE.&id,
        namingOptionals         SET SIZE (1..MAX) OF ATTRIBUTE.&id OPTIONAL }
```

Компоненты **subordinate**, **mandatoryNamingAttributes** и **optionalNamingAttributes** имеют ту же семантику, что и соответствующие порции нотации, введенной соответствующим классом информационных объектов.

15.7.7 Операционный атрибут использования правила сопоставления

Операционный атрибут **matchingRuleUse** используется для указания типов атрибута, к которым в подсхеме применяется правило сопоставления:

```
matchingRuleUse ATTRIBUTE  ::= {
        WITH SYNTAX                     MatchingRuleUseDescription
```

```
        EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
        USAGE                       directoryOperation
        ID                          id-soa-matchingRuleUse }

MatchingRuleUseDescription ::= SEQUENCE {
        identifier          MATCHING-RULE.&id,
        name                SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description         DirectoryString { ub-schema }                           OPTIONAL,
        obsolete            BOOLEAN                                                 DEFAULT FALSE,
        information   [0]   SET OF ATTRIBUTE.&id }
```

Компонент **identifier** значения атрибута **matchingRulesUse** является идентификатором объекта, указывающим правило сопоставления.

Компонент **information** значения определяет совокупность типов атрибутов, к которым применяется это правило сопоставления.

15.7.8 Операционный атрибут структурного класса объектов

Все статьи в DIT имеют операционный атрибут **structuralObjectClass**, который указывает структурный класс объектов этой статьи:

```
structuralObjectClass ATTRIBUTE ::= {
        WITH SYNTAX                 OBJECT IDENTIFIER
        EQUALITY MATCHING RULE      objectIdentifierMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-soa-structuralObjectClass }
```

15.7.9 Операционный атрибут управляющего правила структуры

Все статьи в DIT, за исключением не имеющих подстатьи подсхемы статей административных точек, имеют операционный атрибут **governingStructureRule**, который указывает управляющее правило структуры данной статьи:

```
governingStructureRule ATTRIBUTE ::= {
        WITH SYNTAX                 INTEGER
        EQUALITY MATCHING RULE      integerMatch
        SINGLE VALUE                TRUE
        NO USER MODIFICATION        TRUE
        USAGE                       directoryOperation
        ID                          id-soa-governingStructureRule }
```

15.7.10 Операционный атрибут типов контекста

Операционный атрибут **contextTypes** определяет типы контекста, используемые в пределах подсхемы.

```
contextTypes ATTRIBUTE ::= {
        WITH SYNTAX                 ContextDescription
        EQUALITY MATCHING RULE      objectIdentifierFirstComponentMatch
        USAGE                       directoryOperation
        ID                          id-soa-contextTypes }

ContextDescription ::= SEQUENCE {
        identifier          CONTEXT.&id,
        name                SET SIZE (1..MAX) OF DirectoryString {ub-schema}        OPTIONAL,
        description         DirectoryString { ub-schema }                           OPTIONAL,
        obsolete            BOOLEAN                                                 DEFAULT FALSE,
        information   [0]   ContextInformation }
```

Компонент **identifier** значения операционного атрибута **contextTypes** является идентификатором объекта, определяющим тип контекста.

Операционный атрибут **contextTypes** содержит множественные значения; каждое значение определяет один тип контекста:

```
ContextInformation ::= SEQUENCE {
        syntax              DirectoryString { ub-schema } ,
        assertionSyntax     DirectoryString { ub-schema } OPTIONAL }
```

Компоненты **syntax** и **assertionSyntax** имеют ту же семантику, что и соответствующие порции нотации, введенной соответствующим классом информационных объектов.

Каждый из компонентов **syntax** и **assertionSyntax** содержит текстовую строку, содержащую определение на языке ASN.1 синтаксиса контекста и синтаксиса утверждения о контексте, соответственно. Такое определение на языке ASN.1 должно даваться как необязательное порождение импорта ASN.1, за которым следует необязательное порождение присвоения ASN.1 и, далее, порождение типа ASN.1. Все имена записей, определенные в модулях Справочника, являются неявно импортируемыми и не требуют явного импорта. Все имена записей, импортируемые или определенные через функцию присвоения, являются локальными по отношению к определению этого синтаксиса. Если запись ASN.1 включает определенное пользователем ограничение и не является одним из типов ASN.1, определенных в модулях Справочника, то последний UserDefinedConstraintParameter этого ограничения должен быть реальным параметром, управляющим типом которого является **SyntaxConstraint** и значением которого является идентификатор объекта, присвоенный данному ограничению.

ПРИМЕЧАНИЕ 1. – Порождения операций импорта, присвоения и записи языка ASN.1 определяются в Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1. **UserDefinedConstraintParameter** определяется в Рек. МСЭ-Т X.682 | ИСО/МЭК 8824-3. **SyntaxConstraint** определяется в п. 15.7.3.

ПРИМЕЧАНИЕ 2. – Типичным определением на языке ASN.1 является просто имя записи.

15.7.11 Операционный атрибут использования контекста DIT

Операционный атрибут **dITContextUse** используется для указания контекстов, которые должны или могут использоваться в атрибуте:

```
dITContextUse ATTRIBUTE  ::= {
        WITH SYNTAX                     DITContextUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContextUse }

DITContextUseDescription  ::=  SEQUENCE  {
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }   OPTIONAL,
        obsolete                BOOLEAN                         DEFAULT FALSE,
        information   [0]       DITContextUseInformation }
```

Компонент **identifier** значения операционного атрибута **dITContextUse** является идентификатором объекта типа атрибута, к которому он применяется. Значение **id-oa-allAttributeTypes** указывает, что он применяется ко всем типам атрибута.

Компонент **information** значения указывает обязательные и необязательные типы контекста, связанные с типом атрибута, который определен компонентом **identifier**:

```
DITContextUseInformation  ::=  SEQUENCE  {
        mandatoryContexts[1]    SET SIZE (1..MAX) OF CONTEXT.&id  OPTIONAL,
        optionalContexts    [2] SET SIZE (1..MAX) OF CONTEXT.&id  OPTIONAL }
```

15.7.12 Операционный атрибут "друзей"

Операционный атрибут **friends** используется для указания множеств типов атрибута, которые являются "друзьями" в пределах подсхемы:

```
friends ATTRIBUTE  ::= {
        WITH SYNTAX                     FriendsDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-friends }
FriendsDescription  ::=  SEQUENCE  {
        anchor                  ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE,
        friends                 [0]     SET OF ATTRIBUTE.&id }
```

Компонент **anchor** значения атрибута **friends** – это идентификатор объекта типа атрибута, который является опорным для множества "друзей". Компонент **friends** значения атрибута **friends** – это множество идентификаторов объекта типов атрибута, которые являются "друзьями" типа опорного атрибута.

РАЗДЕЛ 7 – АДМИНИСТРАТИВНОЕ УПРАВЛЕНИЕ СЛУЖБОЙ СПРАВОЧНИКА

16 Модель административного управления службой

В этом пункте описывается модель, представляющая, как административный орган может управлять, ограничивать и регулировать службу согласно либо спецификации пользователя в запросе поиска, чтения или изменения статьи, либо тому, какая информация должна быть возвращена.

16.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

16.1.1 эффективно присутствующий тип атрибута: Тип атрибута, который присутствует, по крайней мере, в одном неинвертированном пункте фильтра каждого подфильтра фильтра поиска и который выполняет требования, определенные для этого типа атрибута в соответствующем правиле поиска. Определения инвертированного и неинвертированного пунктов фильтра см. в п. 7.8.1 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

16.1.2 управляющее правило поиска: Правило поиска, которому соответствует конкретная операция и которое выбрано для управления этой операцией.

16.1.3 именованная служба: Набор типов служб, которые вместе обеспечивают полную службу, например службу Белых страниц.

16.1.4 профиль атрибута запроса: Спецификация того, что требуется для пункта фильтра для соответствующего типа атрибута, с тем чтобы он был эффективно присутствующим.

16.1.5 тип атрибута запроса: Тип атрибута, который согласно спецификации правила поиска может быть представлен в фильтре операции поиска.

16.1.6 правило поиска: Подробная спецификация ограничений/аспектов улучшения, которая выдана для данного типа службы, главным образом предназначенная для данного класса пользователей и рассчитанная на конкретную группу пользователей.

16.1.7 тип службы: Глобально уникальное определение возможностей службы для конкретной цели в пределах строго определенной области действия, например возможности поиска для конкретного типа статей в пределах области DIT. Для всех пользователей могут быть доступны не все аспекты типа службы.

16.1.8 подфильтр: Булев компонент фильтра, который содержит только логические И неинвертированных пунктов фильтра и инвертированных пунктов фильтра, т. е. неформально это может быть выражено как НЕ (пункт фильтра). Любой фильтр может быть описан в канонической форме, содержащей логические ИЛИ подфильтров, как это рассматривается в Приложении Q.

16.1.9 класс пользователей: Идентифицируемая совокупность пользователей, которые в силу своих функций, занимаемой должности в организации и т. д. могут обращаться к определенным аспектам типов службы в пределах именованной службы. Различные группы пользователей, указываемые по своим именам в пределах класса пользователей, могут замечать различия в предоставляемой службе. Группы пользователей могут охватывать классы пользователей.

16.2 Модель тип службы/класс пользователей

Абстрактная служба Справочника, определенная в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3, – это представление всех возможностей служб, предусмотренных спецификациями Справочника. *Тип службы* является подмножеством этой службы для выполнения конкретной функции, например поиска определенного типа объекта в пределах данной области действия.

Именованная служба – это подборка типов служб для конкретных целей, например для предоставления службы Белых страниц, определенного типа службы Желтых страниц и т. д.

Тип службы реализуется в первую очередь через операцию поиска, но также через другие операции, которые могут описывать выбор информации статьи, т. е. операции чтения и изменения статьи. Для целей административного управления службой запрос чтения **read** или изменения статьи **modifyEntry** в некоторых аспектах считается эквивалентным запросу поиска **search** с компонентом **subset**, идентичным компоненту **baseObject**, и компонентом **filter**, идентичным **and : {}**. Административное управление службой не затрагивает того, какая информация может быть изменена операцией изменения статьи. Это регулируется исключительно управлением доступом.

Идентификатор объекта указывает тип службы, обеспечивая таким образом его глобальное уникальное опознавание. Разные *классы пользователей*, в зависимости от своих функций, занимаемых должностей в организации и т. д., могут несколько по-разному воспринимать тип службы. Класс пользователя обозначается целочисленной величиной, единственное предъявляемое требование к которой заключается в уникальности в DMD. Разные DMD могут присваивать разные идентификаторы тому, что может рассматриваться как один и тот же класс пользователей. Вместе с тем, предполагается, что административные органы, сотрудничающие в целях предоставления общей именованной службы в нескольких DMD, будут координировать идентификаторы групп пользователей. Даже в рамках конкретного класса пользователей могут быть различия в службе, предоставляемой пользователям этого класса. Такие различия базируются на выделенных именах пользователей. Например, пользователи конкретного класса пользователей в одной стране могут несколько иначе видеть тип службы по сравнению с пользователями того же класса пользователей в другой стране, например в силу местных законов о неприкосновенности частной жизни.

Определение типа службы для группы пользователей выражается *правилом поиска*, определяющего в деталях порядок выполнения этой операции.

Тип службы и класс пользователей, для которого она предназначена в первую очередь, указываются в правиле поиска.

Группа пользователей может охватывать несколько классов пользователей. Пользователь в пределах класса пользователей может также применять правила поиска, которые в первую очередь предназначены для других классов пользователей, например, пользователи в обладающем большими возможностями классе пользователей могут также получать разрешения, предназначенные для классов пользователей, которым в целом предоставляются более узкие возможности службы.

Группа пользователей не определяется напрямую правилом поиска, но определяется косвенно наличием разрешения на вызов данного правила поиска. Группа пользователей может вызывать любое правило поиска, в отношении которого она имеет разрешение на вызов. Если конкретный пользователь имеет разрешение на вызов нескольких правил поиска для того же типа службы, но для разных групп пользователей, процедуры, определенные в настоящих спецификациях Справочника, выберут, при прочих равных, правило поиска со старшим идентификатором группы пользователей. Это дает административному органу возможность управлять этим выбором посредством надлежащего присвоения идентификаторов классов пользователей.

16.3 Административные области, зависящие от службы

Автономная административная область может быть обозначена как *административная область*, *зависящая от службы*, с тем чтобы использовать правила поиска и осуществлять административное управление ими. Это должно быть указано наличием значения **id-ar-serviceSpecificArea** в атрибуте **administrativeRole** связной административной статьи атрибута (в добавление к наличию значения **id-ar-autonomousArea**, и, возможно, других значений).

Такая автономная административная область может быть разбита на части, для того чтобы использовать правила поиска и осуществлять административное управление ими в конкретных частях. В этом случае административные статьи для каждой из административных областей, зависящих от службы, указываются наличием значения **id-ar-serviceSpecificArea** в атрибутах **administrativeRole** этих статей. Стратегия службы для старших административных областей, зависящих от службы, не является соответствующей подчиненной по отношению к такой административной статье.

Если такая автономная административная область не разбита на части, для охватывающих всю автономную административную область правил поиска существует единая административная область. зависящая от службы,

Одно или более правило поиска представляются в информационной модели Справочника подстатьей, называемой *подстатья службы*, атрибут **objectClass** которой содержит значение **id-sc-serviceAdminSubentry**, как это определено в п. 14.8. Подстатья этого класса должна быть непосредственно предшествующей по отношению к административной статье, атрибут **administrativeRole** которой содержит значение **id-ar-serviceSpecificArea**.

Этап оценки операции в пределах зависящей от службы административной области наряду с прочим определяется тем, какой для этой операции используется базовый объект, возможно, после разыменования псевдонима. Таким образом, правила поиска привязаны к статьям. После определения базового объекта для операции пригодными для управления поиском являются правила поиска, привязанные к данной статье. Связи между правилами поиска в пределах подстатьи и статьями в пределах зависящей от службы административной области устанавливаются операционным атрибутом **subtreeSpecification** этой подстатьи. Статьи, указанные значениями операционного атрибута **subtreeSpecification**, являются, таким образом, привязанными к правилам поиска, размещенным в той же подстатье.

Конкретная статья может быть связана с правилами поиска из нескольких подстатей; они могут иметь те же или разные спецификации поддерева. И наоборот, на разные части административной области может указывать одна подстатья, используя множественные значения спецификации поддерева.

Аргументы операции могут проверяться на соответствие правилу поиска с использованием алгоритма, называемого *функция проверки поиска*.

Рисунок 16 – Функция проверки поиска

Операция является действительной и разрешена к выполнению, если и только если функция проверки поиска выдает результата TRUE для, по крайней мере, одного их доступных правил поиска, связанных с базовым объектом для этой операции. Для того чтобы правило поиска было доступно для операции, запросчик должен иметь разрешения на *запрос* к значению атрибута, содержащему правило поиска. Если существует только одно правило поиска, которому соответствует операция, это правило поиска называется *управляющим правилом поиска* для этой операции, т. е. это правило поиска, которое используется в процессе дальнейшего выполнения операции. Если имеются несколько таких правил поиска, одно из них выбирается на местном уровне управляющим правилом поиска. Процедура выбора управляющего правила поиска содержится в п. 19.3.2.2.1 Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4. Таким образом, управляющее правило поиска постоянно связано с операцией, в соответствии с результатами ее оценки, в пределах зависящей от службы административной области. Это также верно для случая, когда часть операции выполняется другими DSA, содержащими части данной административной области, зависящей от службы.

Административные органы определяют:

- собирать ли несколько правил поиска, требующих разные разрешения на запрос, в единую подстатью (т. е. требовать управления доступом вниз до уровня значения атрибута, если эти разрешения на запрос меняются от значения к значению); или
- собирать ли правила поиска, имеющие те же разрешения на управление доступом, в отдельные подстатьи, так чтобы разрешения на управление доступом могли предоставляться на основании разрешений к полному атрибуту; тогда разные подстатьи могут содержать разные разрешения на управление доступом.

Если для операции, определяющей базовый объект в пределах зависящей от службы административной области, отсутствует доступное правило поиска или если функция проверки поиска возвращает результат FALSE для всех доступных правил поиска, операция отклоняется с сообщением об ошибке.

Если административная область, зависящая от службы, не имеет подстатей, то с этой областью не связаны какие-либо ограничения службы.

Возможно наличие пользователей, к которым не должны применяться ограничения службы, например администраторов, и могут существовать статьи, для которых не требуется ограничения, если они являются базовыми статьями, например нижние статьи в DIT. Следовательно, административное полномочие может включать специальные правила поиска – *пустые правила поиска*.

Иерархическая группа в пределах зависящей от службы административной области должна полностью входить в эту область.

Область действия операции поиска не может пересекать границу зависящей от службы административной области. В Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4 описываются процедуры, не разрешающие операции поиска, начатой в пределах конкретной зависящей от службы административной области, выходить за пределы этой области, даже если в ходе оценки поиска происходит разыменование псевдонимов. Аналогичным образом, поиск, начатый за пределами зависящей от службы административной области, не может распространяться в эту область.

16.4 Введение в правила поиска

Правила поиска являются выражением стратегии, с одной стороны, ограничивающей и корректирующей операции, которые могут выполняться в каком-либо регионе DIT, и, с другой стороны, помогающей в выполнении путем управления процессом выполнения операции. Правило поиска имеет следующие основные характеристики:

- оно выдает требования, которые обязана выполнять операция, если эта операция должна быть выполнена, базируясь на этом правиле поиска;
- оно определяет настройку запроса операции;
- оно выдает детальное описание оценки операции, например путем определения стратегии ослабления, если в результате операции поиска найдено слишком много или слишком мало статей; и
- оно предоставляет спецификации выбора информации статьи.

Когда начинается обработка операции, базовая статья этой операции соответствует одной или более подстатей службы, значения спецификации поддерева которых включают эту базовую статью. Тем самым потенциально количество пригодных правил поиска определено. Детали операции оцениваются на соответствие этим пригодным правилам поиска. Операция может быть выполнена, только если найдено совместимое правило поиска.

16.5 Подфильтры

Если правило поиска назначено управлять операцией поиска, оно может определять совокупность типов атрибутов, которые могут присутствовать в фильтре запроса поиска **search**. Эти типы атрибутов называются *типы атрибутов запроса* для данного правила поиска. В этом фильтре не могут присутствовать какие-либо иные типы атрибутов в любой форме – инвертированной или неинвертированной. В данном подпункте более подробно разъясняется смысл присутствия типа атрибута в фильтре поиска. Правило поиска определяет также требования к действительным комбинациям типов атрибутов запроса. Это может быть требование наличия определенного атрибута; наличия по

крайней мере одного из двух типов атрибутов; требование, не разрешающее один тип атрибута в отсутствие другого, и т. д. Для дальнейшего разъяснения порядка составления комбинаций полезно ввести понятие *подфильтров.*

Согласно исчислению высказываний любой фильтр может в общем случае быть записан в виде последовательности подфильтров, разделенных операторами ИЛИ. Это можно представить следующим образом:

$$f = f_1 + f_2 + ... + f_r,$$

где каждый подфильтр f_i является последовательностью пунктов фильтра или инвертированных пунктов фильтров, разделенных операторами И, что можно представить следующим образом:

$$f_i = f_{i1} f_{i2} ... f_{is},$$

где f_{ij} – либо пункт фильтра, либо его логическое отрицание.

Понятие подфильтра более подробно описывается в Приложении Q.

Для того чтобы фильтр соответствовал правилу поиска, все подфильтры должны соответствовать этому правилу поиска.

Для того чтобы пункт фильтра эффективно представлял тип атрибута в подфильтре, необходимо соответствие требованиям *профиля атрибута запроса* для данного типа атрибута. Профили атрибутов запроса являются частью спецификации правила поиска. Если по крайней мере один пункт фильтра для типа атрибута в каждом подфильтре соответствует профилю атрибута запроса, тип атрибута считается *эффективно присутствующим типом атрибута*.

16.6 Требования фильтра

Для того чтобы тип атрибута эффективно присутствовал в фильтре, тип атрибута или, если в профиле атрибута запроса установлена опция **includeSubtypes** профиля атрибута запроса, один из его подтипов должен быть представлен по крайней мере в одном неинвертированном пункте фильтра каждого подфильтра. Такой неинвертированный пункт фильтра должен соответствовать следующим требованиям:

– он должен быть неинвертированным пунктом фильтра, не относящимся к любому из перечисленных ниже типов:

greaterOrEqual;

lessOrEqual;

present или **contextPresence**, если это явным образом не разрешено профилем атрибута запроса;

– он должен соответствовать спецификации профиля атрибута запроса для данного типа атрибута;

– если это пункт фильтра **extensibleMatch**, тип атрибута должен быть определен в компоненте **type** выражения **MatchingRuleAssertion**.

ПРИМЕЧАНИЕ. – Если это последнее ограничение не введено, данный пункт фильтра может неявным образом включать неопределенное количество типов атрибутов в фильтр запроса и, следовательно, нарушать процедуру проверки поиска.

Если тип атрибута представлен в фильтре, он должен присутствовать эффективно.

Разрешается наличие пунктов фильтра **extensibleMatch** в отсутствие компонента **type** в фильтре. Их наличие не затрагивает оценку поиска на соответствие правилам поиска. Вместе с тем, такой пункт фильтра должен быть применим только к атрибутам, типом которых является тип атрибута запроса, т. е. представленным в управляющем правиле поиска профилем атрибута запроса (см. п. 16.10.2).

16.7 Выбор информации атрибута на основе правил поиска

За пределами административной области, зависящей от службы, возвращаемая информация атрибута выбирается компонентом **selection** запроса операции, возможно, измененного компонентом **operationContext** в **CommonArguments**, и любыми значениями по умолчанию контекста, установленными либо в пределах специальной административной области контекста по умолчанию, либо местными значениями контекста по умолчанию. Для операции поиска выбор информации может также быть изменен компонентом **matchedValuesOnly** в **SearchArgument**. Однако в случае управления операцией управляющим правилом поиска это правило поиска может определять, какая информация должна возвращаться. Если это так, возвращаемая информация атрибута пользователя должна быть пересечением того, что определяет управляющее правило поиска, и того, что возвращалось бы в отсутствие управляющего правила поиска. Если выбор информации статьи в компоненте **selection** указывает выбор операционных атрибутов, то же правило должно применяться для операционных атрибутов. Если выбор информации статьи не определяет возвращение информации операционных атрибутов, возвращаемая информация операционных атрибутов должна определяться исключительно управляющим правилом поиска.

Управляющее правило поиска может определять, какая информация атрибута должна возвращаться, абсолютно независимо от того, какие типы атрибутов могут быть определены в фильтре **search**.

Если информация, которая должна возвращаться, базируется на иерархических группах, выбор информации атрибутов из таких статей выполняется по вышеизложенному принципу, за исключением случаев, когда не действуют спецификации **matchedValuesOnly**.

ПРИМЕЧАНИЕ. – Выбор членов семейства не управляется вышеизложенным принципом (см. п. 16.10.6).

16.8 Аспекты управления доступом правил поиска

Правила поиска обеспечивают некоторые дополнительные возможности управления доступом, кроме описанных в пункте 18. В рамках ориентированного на службу подхода необходимо применять ограничения на то, как может быть сформулирована операция, и на то, какая информация может возвращаться. Это должно базироваться не только на идентификации пользователя, но также на типе службы и классе пользователей, предоставляя таким образом административным органам возможность настройки службы исходя из качества информации, соображений начисления платы и т. д.

Возможности управления доступом, как они определены в п. 18, используются для гарантирования того, что запрашивать правила поиска могут только соответствующие группы пользователей. Эти возможности могут также служить для защиты информации, к которой не должны иметь доступ конкретные группы пользователей.

DSA, который кэширует информацию, исходящую из зависящей от службы административной области, может не иметь правил поиска для контроля ограничений на эту информацию. В отношении управления доступом (см. п. 18.8.2) администратор по безопасности должен быть осведомлен, что DSA с возможностями кэширования может подвергать серьезному риску в отношении безопасности другие DSA.

16.9 Аспекты контекста правил поиска

В силу того, что утверждения о контексте могут быть частью пункта фильтра для операции поиска, в спецификации правил поиска необходимо учитывать контексты. Включение контекстов в правило поиска открывает новые возможности контекстной функции, которые могут упростить требования к реализации DUA и DSA.

Базовая функция контекста разрешает пользователю определять контексты для фильтра поиска и для информации статьи; и она разрешает административным органам устанавливать значения контекстов по умолчанию в пределах специальной административной области значений контекстов по умолчанию. Эти значения по умолчанию применяются без исключения ко всем пользователям и ко всем типам служб. Вместе с тем, функция контекста, как предоставляемая правилами поиска, разрешает пользователю определять минимум контекстной информации и разрешает административным органам составлять индивидуальные спецификации контекстов для каждого правила поиска. Кроме того, возможно, как указано в п. 16.8, обеспечивать управление доступом как функцию с помощью составления надлежащим образом спецификаций контекстов правил поиска. Использование спецификаций контекста в правилах поиска может сделать определение специальных административных областей значений контекстов по умолчанию излишним.

16.10 Спецификация правила поиска

Тип данных ASN.1 **SearchRule** дает синтаксис правила поиска.

```
SearchRule ::= SEQUENCE {
    COMPONENTS OF           SearchRuleId,
    serviceType         [1] OBJECT IDENTIFIER                          OPTIONAL,
    userClass           [2] INTEGER                                    OPTIONAL,
    inputAttributeTypes [3] SEQUENCE SIZE (0..MAX) OF RequestAttribute OPTIONAL,
    attributeCombination[4] AttributeCombination                       DEFAULT and : { },
    outputAttributeTypes[5] SEQUENCE SIZE (1..MAX) OF ResultAttribute  OPTIONAL,
    defaultControls     [6] ControlOptions                             OPTIONAL,
    mandatoryControls   [7] ControlOptions                             OPTIONAL,
    searchRuleControls  [8] ControlOptions                             OPTIONAL,
    familyGrouping      [9] FamilyGrouping                             OPTIONAL,
    familyReturn        [10] FamilyReturn                              OPTIONAL,
    relaxation          [11] RelaxationPolicy                          OPTIONAL,
    additionalControl   [12] SEQUENCE SIZE (1..MAX) OF AttributeType   OPTIONAL,
    allowedSubset       [13] AllowedSubset                             DEFAULT '111'B,
    imposedSubset       [14] ImposedSubset                             OPTIONAL,
    entryLimit          [15] EntryLimit                                OPTIONAL }

SearchRuleId ::= SEQUENCE {
    id                      INTEGER,
    dmdId               [0] OBJECT IDENTIFIER }

AllowedSubset ::= BIT STRING { baseObject (0), oneLevel (1), wholeSubtree (2) }

ImposedSubset ::= ENUMERATED { baseObject (0), oneLevel (1), wholeSubtree (2) }
```

```
RequestAttribute ::= SEQUENCE {
        attributeType                   ATTRIBUTE.&id ({ SupportedAttributes }),
        includeSubtypes         [0]     BOOLEAN                                         DEFAULT FALSE,
        selectedValues          [1]     SEQUENCE SIZE (0..MAX) OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType })     OPTIONAL,
        defaultValues           [2]     SEQUENCE SIZE (0..MAX) OF SEQUENCE {
                entryType                       OBJECT-CLASS.&id  OPTIONAL,
                values                          SEQUENCE OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType }) }   OPTIONAL,
        contexts                [3]     SEQUENCE SIZE (0..MAX) OF ContextProfile        OPTIONAL,
        contextCombination      [4]     ContextCombination                              DEFAULT and : { },
        matchingUse             [5]     SEQUENCE SIZE (1..MAX) OF MatchingUse           OPTIONAL }

ContextProfile ::= SEQUENCE {
        contextType         CONTEXT.&id({SupportedContexts}),
        contextValue        SEQUENCE SIZE (1..MAX) OF CONTEXT.&Assertion
                                ({SupportedContexts}{@contextType})  OPTIONAL }

ContextCombination ::= CHOICE {
        context                 [0]     CONTEXT.&id({SupportedContexts}),
        and                     [1]     SEQUENCE OF ContextCombination,
        or                      [2]     SEQUENCE OF ContextCombination,
        not                     [3]     ContextCombination }

MatchingUse ::= SEQUENCE {
        restrictionType         MATCHING-RESTRICTION.&id ({SupportedMatchingRestrictions}),
        restrictionValue        MATCHING-RESTRICTION.&Restriction
                                        ({SupportedMatchingRestrictions}{@restrictionType}) }

-- Определение следующей совокупности информационных объектов передается, возможно, в стандартизированные
-- профили или в свидетельства о соответствии протокольной реализации. Эта совокупность необходима для описания
-- табличного ограничения, накладываемого на компоненты SupportedMatchingRestrictions.

SupportedMatchingRestrictions  MATCHING-RESTRICTION  ::= { ... }

AttributeCombination ::= CHOICE {
        attribute               [0]     AttributeType,
        and                     [1]     SEQUENCE OF AttributeCombination,
        or                      [2]     SEQUENCE OF AttributeCombination,
        not                     [3]     AttributeCombination }

ResultAttribute ::= SEQUENCE {
        attributeType           ATTRIBUTE.&id ({ SupportedAttributes }),
        outputValues            CHOICE {
                selectedValues          SEQUENCE OF ATTRIBUTE.&Type
                                                ({ SupportedAttributes }{ @attributeType }),
                matchedValuesOnly       NULL }  OPTIONAL,
        contexts                [0]  SEQUENCE SIZE (1..MAX) OF ContextProfile  OPTIONAL }

ControlOptions ::= SEQUENCE {
        serviceControls         [0]     ServiceControlOptions       DEFAULT { },
        searchOptions           [1]     SearchControlOptions        DEFAULT { searchAliases },
        hierarchyOptions        [2]     HierarchySelections         OPTIONAL }

EntryLimit ::= SEQUENCE {
        default             INTEGER,
        max                 INTEGER }

RelaxationPolicy ::= SEQUENCE {
        basic                   [0]     MRMapping DEFAULT { },
        tightenings             [1]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        relaxations             [2]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        maximum                 [3]     INTEGER OPTIONAL,       -- обязательный, если присутствует tightenings
        minimum                 [4]     INTEGER DEFAULT 1 }
```

```
MRMapping ::= SEQUENCE {
    mapping              [0]  SEQUENCE SIZE (1..MAX) OF Mapping              OPTIONAL,
    substitution         [1]  SEQUENCE SIZE (1..MAX) OF MRSubstitution       OPTIONAL }

Mapping ::= SEQUENCE {
    mappingFunction          OBJECT IDENTIFIER (CONSTRAINED BY { -- shall be an
                         -- идентификатор объекта алгоритма сопоставления на основе отображения -- } ),
    level                    INTEGER  DEFAULT 0 }

MRSubstitution ::= SEQUENCE {
    attribute                AttributeType,
    oldMatchingRule      [0] MATCHING-RULE.&id  OPTIONAL,
    newMatchingRule      [1] MATCHING-RULE.&id  OPTIONAL }
```

16.10.1 Компоненты идентификации правила поиска

Компонент **id** позволяет осуществлять однозначную идентификацию правил поиска в пределах DMD. Нулевое значение зарезервировано для пустого правила поиска. Назначение пустого правила поиска описывается в п. 16.3.

Компонент **dmdId** дает однозначную идентификацию DMD, которая установила это правило поиска. Этот компонент вместе с компонентом **id** обеспечивает возможность однозначной глобальной идентификации правила поиска.

ПРИМЕЧАНИЕ. – Стратегия в отношении этой однозначности не водит в область действия настоящей спецификации.

Компонент **id** (с нулевым значением) и компоненты **dmdId** являются единственными компонентами, актуальными для пустого правила поиска.

Компонент **serviceType** является идентификатором объекта, который указывает тип службы, поддерживаемый данным правилом поиска. Этот компонент должен присутствовать всегда, кроме случая пустого правила поиска.

Компонент **userClass** указывает класс пользователей, для которых в первую очередь предназначено это правило поиска. Для данного типа службы может существовать несколько правил поиска, определяющих один класс пользователей. Этот компонент должен присутствовать всегда, кроме случая пустого правила поиска.

16.10.2 Профили атрибутов запроса

Компонент **inputAttributeTypes** должен определять профили атрибутов запроса для всех типов атрибутов, которые должны или могут быть представлены в фильтре **search**. Если фильтр **search** включает пункт фильтра для типа атрибута, не представленного профилем атрибута запроса, проверка поиска на соответствие этому правилу даст отрицательный результат. Тип данных **RequestAttribute** определяет требование к пункту фильтра об эффективном представлении данного типа атрибута, определенного в этом пункте фильтра. Если этот компонент отсутствует, правило поиска не налагает каких-либо ограничений на наличие типов атрибутов, т. е. с этим компонентом согласуется любая операция. Если этот компонент присутствует, но является пустым, с этим компонентом согласуются только запрос чтения **read**, запрос изменения **modifyEntry** или запрос поиска **search** с фильтром по умолчанию (**and : { }**).

Следующие подкомпоненты актуальны для всех типов операций, управляемых правилами поиска:

a) Подкомпонент **attributeType** определяет тип атрибута, к которому применяется эта спецификация. Это единственный обязательный подкомпонент. Для данного типа атрибута в пределах правила поиска может существовать только одна спецификация **RequestAttribute**. Если это единственный подкомпонент, за возможным исключением подкомпонента **includeSubtypes**, на пункты фильтра поиска не налагается каких-либо ограничений в отношении данного типа атрибута, кроме случая когда такие пункты фильтра находятся в фильтре, и по крайней мере один из них должен быть неинвертированным.

b) Подкомпонент **includeSubtypes** определяет, что данный профиль атрибута запроса может быть реализован пунктом фильтра для подтипа данного типа атрибута.

Следующие подкомпоненты актуальны только для операции поиска:

c) Подкомпонент **selectedValues** предоставляет совокупность значений атрибутов типа, переданного в **attributeType**. Если этот тип атрибута представлен в фильтре, для данного типа атрибута должен существовать по крайней мере один неинвертированный пункт фильтра, который совпадает по крайней мере с одним значением этого подкомпонента. В любом ином случае этот тип атрибута эффективно не присутствует в данном фильтре.

   Если этот подкомпонент отсутствует, вышеуказанное совпадение оценивается значением TRUE.

   Если дано пустое множество значений атрибутов, этот тип атрибута может быть эффективно представлен только в:

   – пункте фильтра **present**, если подкомпонент **Contexts** не присутствует; или

   – пункте фильтра **contextPresent**, если подкомпонент **Contexts** присутствует.

d) Подкомпонент **defaultValues** не затрагивает оценки запроса поиска **search** на соответствие правилу поиска, но управляет операцией поиска, если правило поиска выбрано управляющим правилом поиска. Этот компонент предоставляет совокупность значений атрибута типа, переданного в **attributeType**. Если в пределах фильтра определяется пункт фильтра, использующий этот тип атрибута, но в статье (или в группировке семейства) не существует атрибута данного типа, то этот пункт фильтра оценивается как TRUE (или как FALSE, если инвертированный), если он совпадает с одним из значений в данном подкомпоненте. Если этот подкомпонент отсутствует, значений по умолчанию не существует.

Если этот компонент присутствует, но пуст, это указывает на то, что данный компонент принимает все возможные значения, т. е. пункт фильтра для данного типа атрибута всегда оценивается как TRUE (или FALSE, если инвертированный), если этот тип атрибута отсутствует в статье.

ПРИМЕЧАНИЕ 1. – Это отражает ситуацию, когда пункт фильтра должен игнорироваться, если атрибут, на тип которого дана ссылка, отсутствует.

Если статья содержит атрибут данного типа, выполняется нормальное сопоставление на соответствие этому атрибуту.

e) Подкомпонент **contexts** определяет типы контекстов, представление которых разрешено в пункте фильтра для данного типа атрибута. Конкретный тип контекста не должен быть представлен в этом подкомпоненте более одного раза.

- Если этот подкомпонент отсутствует, в пункте фильтра для этого типа атрибута может быть представлена любая контекстная информация.
- Если этот подкомпонент присутствует, в пункте фильтра для этого типа атрибута могут быть представлены контексты только определенных в этом подкомпоненте типов. Если подкомпонент является пустой последовательностью, в пункте фильтра для этого типа атрибута не может быть представлено какой-либо контекстной информации.
- Если определяется только тип контекста, в утверждении о контексте может присутствовать любое значение контекста этого типа.
- Если в этом подкомпоненте содержатся значения контекста данного типа контекста, в соответствующем утверждении о контексте в пункте фильтра могут присутствовать только эти значения.

Если спецификация контекста в этом пункте фильтра не согласуется с вышесказанным, фильтр не согласуется с профилем атрибута запроса для данного типа атрибута.

f) Подкомпонент **contextCombination** определяет действительную в пределах данного профиля атрибута запроса комбинацию типов контекстов, перечисленных в подкомпоненте **contexts** в пределах профиля атрибута запроса. Если этот подкомпонент отсутствует, на комбинацию контекстов этих типов ограничения не налагаются. Если присутствует недействительная комбинация типов контекстов, пункт фильтра не согласуется с профилем атрибута запроса для этого типа атрибута. Этот компонент может указывать, что определенные типы контекстов должны безусловно присутствовать.

g) Подкомпонент **matchingUse** используется для определения возможных ограничений использования применимого правила сопоставления, например минимальные длины для сопоставления подстрок. Применимое правило сопоставления это правило, которое действительно предполагается использовать до какого бы то ни было ослабления, но после возможного базового замещения. Детальная информация об этих ограничениях и порядок их оценки описываются как часть спецификации ограничения. Если этот подкомпонент описывает ограничение сопоставления, определенное для правила сопоставления, которое должно использоваться, проверяется, не нарушается ли это ограничение сопоставления или должны ли применяться неподдерживаемые аспекты этого правила сопоставления. Если это так, тогда:

- если опция управления поиском **performExactly** не установлена, для реализации может использоваться локальное правило другого способа применения правила сопоставления;

  ПРИМЕЧАНИЕ 2. – Такое локальное правило требует обеспечения возможности индивидуальной настройки для рассматриваемого правила сопоставления.

- если опция управления поиском **performExactly** установлена или невозможно применить локальное правило, запрос поиска **search** не согласуется с данным правилом поиска.

16.10.3 Комбинации атрибутов

Компонент **attributeCombination** определяет действительную комбинацию типов атрибутов запроса, перечисленных в компоненте **inputAttributeTypes**. Если этот компонент отсутствует или имеет значение по умолчанию (**and : { }**), комбинация типов атрибутов запроса не ограничивается и все релевантные типы операций согласуются с этим компонентом. Если присутствует недействительная комбинация типов атрибутов запроса, проверка поиска на соответствие правилу поиска завершается неудачей. Этот компонент может указывать, что определенные типы атрибутов должны безусловно эффективно присутствовать в фильтре. Этот компонент должен отсутствовать, если **inputAttributeTypes** отсутствует или пуст. Если этот компонент присутствует и имеет значение не по умолчанию, потенциально согласованной с этим компонентом может быть только операция поиска с фильтром значений не по умолчанию.

16.10.4 Атрибуты, содержащиеся в результате

Компонент **outputAttributeTypes** определяет, какие типы атрибутов (или их подтипы, если опция управления службой **noSubtypeSelection** не установлена) будут потенциально содержаться в результате в зависимости от управления доступом (см. п. 16.7). Если совпавшая статья или составная статья не содержит ни одного из определенных в этом компоненте атрибутов, она не включается в результат. Аналогичное правило применяется к отдельным членам семейства, маркированным как результат сопоставления, или через операции, определенные атрибутами управления в компоненте **additionalControl**. Если такой член семейства не содержит ни одного из определенных этим компонентом атрибутов, это соответствует тому, что данный член семейства и все его подчиненные элементы являются явным образом немаркированными. Тип данных **ResultAttribute** определяет детали порядка представления этого типа атрибута в результате. Этот компонент не затрагивает проверку поиска. Если он отсутствует, правило поиска не затрагивает выбор информации статьи, за исключением случаев, которые могут быть определены компонентами **familyReturn** и **additionalControl**. Этот компонент состоит из следующих подкомпонентов:

a) Подкомпонент **attributeType** определяет тип атрибута, к которому применяется данная спецификация. Это единственный обязательный подкомпонент. Для данного типа атрибута в пределах правила поиска может существовать только одна спецификация **ResultAttribute**.

b) Подкомпонент **outputValues** определяет, какие значения атрибутов пригодны для возвращения. Множество значений может быть далее ограничено подкомпонентом контекста; выбором информации статьи, определяемым запросчиком; управлением доступа и т. д. Если этот подкомпонент отсутствует, пригодными являются все значения атрибутов. Выбор **selectedValues** предоставляет множество значений атрибутов типа, определенного в **attributeType**. Значениями атрибутов, пригодными для возвращения, являются только перечисленные значения. Выбор **matchedValuesOnly** определяет, что пригодными для возвращения являются только те значения атрибута, которые являются значениями атрибута, способствующего получению фильтром значения TRUE через пункты фильтров, отличные от присутствующих (определение понятия "способствующий" см. в п. 10.2.2 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3).

c) подкомпонент **context** содержит совокупность профилей контекста, которые определяют, какая информация значений атрибута возвращается для этого типа атрибута.

   – Если этот подкомпонент отсутствует, правило поиска не налагает каких бы то ни было ограничений на то, какие значения атрибута могут возвращаться исходя из контекстов.

   – Если в этот подкомпонент тип контекста не включен, ни с одним из возвращаемых значений атрибута данного типа атрибута контекстная информация данного типа не возвращается.

   – Если профиль контекста на содержит тип данных **contextValue**, с каждым значением атрибута возвращаются все значения контекста данного типа контекста.

   – Если один или более профилей контекста содержат тип данных **contextValue**, каждый из этих профилей контекста интерпретируется как утверждение **ContextAssertion**, которое должно применяться в отношении значений атрибутов, как это определено в п. 8.9.2.4. Возвращаются только те значения атрибутов, для которых эта оценка дала результат TRUE для всех таких типов контекстов. Если в результате этого не выбрано ни одного возвращаемого значения для этого типа атрибута, атрибут в результат не включается. Аналогичным образом, если в результате выбора информации для статьи не остается, эта статья не возвращается.

   – Если все возвращенные значения атрибутов для этого типа атрибута имеют ту же пару {тип контекста, значение контекста}, которая должна быть возвращена, это значение контекста удаляется из всех значений атрибутов. Если в результате этого остается контекст без какого-либо значения контекста, он полностью удаляется.

     ПРИМЕЧАНИЕ. – Это позволяет настраивать службу таким образом, чтобы имеющий несложное оборудование пользователь в большинстве случаев мог получать информацию без применения контекстов.

16.10.5 Функции управления службой и поиском

Компонент **defaultControls**, если присутствует, используется для определения установки битов, не установленных явным образом для операции в **ServiceControlOptions** в пределах функций управления службой аргумента операции и, если операция является операцией поиска, в **SearchControlOptions** и **HierarchySelections**. Если какая-либо конкретная опция отсутствует, используется элемент **defaultControls** если таковой присутствует.

Если в **defaultControls** полностью отсутствует подкомпонент **hierarchyOptions**, или если **defaultControls** отсутствует, иерархический выбор применяться не должен. Если компонент **hierarchySelection** присутствует в аргументе **search** и определяет что-либо отличное от **self**, проверка поиска на соответствие этим правила поиска завершается неудачей. Соответствующие элементы в **mandatoryControls** и **searchRuleControls** должны быть опущены.

Если компонент **defaultControls** полностью отсутствует, он должен рассматриваться как принимающий стандартное значение по умолчанию **{ serviceControls { }, searchOptions {searchAliases} }**.

Компонент **mandatoryControls** путем установки конкретных битов определяет опции двоичной строки, которые должны присутствовать, как определено в **defaultControls**; если какой-либо бит, определенный компонентом

mandatoryControls, отличается в переданной пользователем опции от **defaultControls**, проверка поиска на соответствие этому правилу поиска завершается неудачей. Биты, не определенные компонентом **mandatoryControls**, принимаются равными нулю. Если операцией является чтение или изменение статьи, рассматривается только подкомпонент **serviceControls**.

Компонент **searchRuleControls** путем установки конкретных битов определяет опции двоичной строки, которые должны быть взяты из **defaultControls**, а не из опций, переданных пользователем. Биты, не определенные компонентом **searchRuleControls**, принимаются равными нулю. Если операцией является чтение или изменение статьи, рассматривается только подкомпонент **serviceControls**.

ПРИМЕЧАНИЕ. – Если пользователь в операции поиска передает $U_{\text{от 0 до p}}$, а битами по умолчанию являются $D_{\text{от 0 до N}}$, результатом применения компонента **defaultControls** является двоичная строка $C_{\text{0 до N}}$, где биты от 0 до p берутся из U, а оставшиеся – из D. Проверка поиска на соответствие правилу поиска завершается неудачей, если двоичная строка C&M не равна D&M, где C значит $C_{\text{от 0 до N}}$, '&' представляет операцию "побитовое И", а $M_{\text{от 0 до N}}$ является двоичной строкой, определенной компонентом **mandatoryControls**. В любом ином случае используемым значением опций является (C&~S | D&S), где S – двоичная строка, определенная компонентом **searchRuleControls**, ~S – ее побитовое НЕ, а '|' представляет операцию "побитовое ИЛИ". Результатом этой последней обработки является удаление битов, указанных компонентом **searchRuleControls**, и замена их значениями битов по умолчанию. Компонент **familyGrouping** определяет спецификацию группировки семейства, которая, если присутствует, имеет преимущество перед (т. е. заменяет) **familyGrouping** в **CommonArguments** аргумента поиска.

16.10.6 Спецификации семейств

Компонент **familyGrouping** определяет выбор группировки семейства, которая, если присутствует, имеет преимущество перед (т. е. заменяет) **familyGrouping** в **CommonArguments** аргумента поиска **search**.

Компонент **familyReturn** определяет выбор возвращаемого члена семейства. Компонент корректирует спецификацию, переданную компонентом **familyReturn** в **EntryInformationSelection** (или его значением по умолчанию) аргумента поиска **search**. Спецификация правила поиска имеет преимущество относительно спецификации в компоненте **memberSelect**, спецификация аргумента поиска **search** имеет преимущество относительно компонента **familySelect**, т. е. если компонент **familySelect** присутствует в аргументе поиска **search**, возможный компонент **familySelect** в правиле поиска должен игнорироваться.

16.10.7 Управление ослаблением

Компонент **relaxation** определяет стратегию ослабления с использованием конструкции **RelaxationPolicy**. Та же конструкция может быть включена в запрос поиска **search** в компоненте ослабления **relaxation**. Связанная с этой конструкцией процедура описывается в настоящем документе как для случая, когда она включена в правило поиска, так и для случая, когда она включена в запрос поиска **search**. Для случая, когда **RelaxationPolicy** включена и в правило поиска, и в запрос поиска **search**, в п. 10.2.2 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3 даются дополнительные спецификации.

Конструкция **RelaxationPolicy** включает следующие подкомпоненты:

a) Подкомпонент **basic**, если присутствует, определяет **MRMapping**, т. е. совокупность замещений правила сопоставления и/или функций сопоставления на основе отображения, которые применяются к фильтру **search** для первой оценки (т. е. без ослабления и усиления). Это дает возможность выбора более подходящего по сравнению с первоначальным совпадения. Пропуск этого пункта или передача его с пустой совокупностью приводит к тому, что для первой оценки используются все нормальные правила сопоставления без применения сопоставления на основе отображения.

b) Подкомпонент **tightenings**, если присутствует, содержит последовательность замещений и отображений, каждая из которых определяется элементом **MRMapping**, которые должны применяться в данном порядке, по одному, если количество совпавших статей слишком велико (превышает **maximum**).

c) Подкомпонент **relaxations**, если присутствует, содержит последовательность замещений и отображений, каждая из которых определяется элементом **MRMapping**, которые должны применяться в данном порядке, по одному, если количество совпавших статей слишком мало (меньше **minimum**).

d) Подкомпонент **maximum** должен присутствовать всегда, когда присутствует подкомпонент **tightenings**, и определять количество статей, при превышении которого должно применяться усиление.

e) Подкомпонент **minimum** определяет количество статей, при достижении которого (или если оно не достигается) должно применяться ослабление; при его отсутствии значение по умолчанию равно нулю.

ПРИМЕЧАНИЕ 1.– Ослабление/усиление не затрагиваются опцией управления поиском **performExactly**.

Замещения правила сопоставления и отображения определяются элементами **MRMapping**, каждый из них состоит из последовательности элементов отображения **Mapping** и элементов замещения **MRSubstitution**. Порядок следования этих элементов значения не имеет.

Элемент отображения **Mapping** включает следующие компоненты:

a) Компонент **mappingFunction** указывает функцию сопоставления на основе отображения со связанной таблицей отображения, которая должна применяться.

b) Компонент **level** указывает уровень ослабления (или усиления при отрицательном значении), который должен применяться для сопоставления на основе отображения. Этот компонент должен игнорироваться, если **&userControl** установлено для сопоставления на основе отображения, а в запрос **search** включен параметр управления поиском **extendedArea**, в каковом случае применяется значение, определенное в **extendedArea**.

ПРИМЕЧАНИЕ 2. – Для базового замещения и отображения значение уровня во многих случаях должно быть нулевым.

Элемент **MRSubstitution** включает следующие компоненты:

a) **attribute** описывает атрибут, к которому должно применяться замещение.

b) **oldMatchingRule** – правило сопоставления, которое должно замещаться. Если оно отсутствует, этот компонент применяется к предыдущему применимому правилу сопоставления указанного типа для этого атрибута, если таковое существует. Для базового замещения или, если базовое замещение не выполняется, для первого замещения ослабления/усиления применимым сопоставлением является сопоставление, которое использовалось бы в ином случае. Для последующих замещений применимым правилом сопоставления является правило, внесенное предыдущим замещением. Если в этом подкомпоненте определено правило сопоставления, которое не является предыдущим применимым правилом сопоставления, замещение не выполняется.

ПРИМЕЧАНИЕ 3. – Например, если пункт фильтра имеет тип **equality** и, следовательно, выбирает правило сопоставления на равенство, а данный подкомпонент указывает правило сопоставления подстрок, замещение не выполняется.

c) **newMatchingRule** – идентификатор объекта для замещающего правила сопоставления, которое должно использоваться вместо старого правила сопоставления. Если оно отсутствует, значением оценки всех соответствующих пунктов фильтров является TRUE для неинвертированных пунктов и FALSE для инвертированных пунктов (т. е. в соответствии с **id-mr-nullMatch**).

Следующее применяется только к замещению правила сопоставления, определенного в запросе **search**. Если указано правило сопоставления, для которого определено ограничение сопоставления для данного типа атрибута (см. п. 16.10.2, g)), каковое ограничение ведет к рассогласованию запроса **search** с управляющим правилом поиска, или если указано неподдерживаемое или несовместимое правило сопоставления, замещение отвергается, и для этого типа атрибута какое-либо замещение более не выполняется.

ПРИМЕЧАНИЕ 4. – Предполагается, что DSA не будет разрешать наличие недействительных замещений в правиле поиска.

Один атрибут может иметь в своем составе множественные элементы **MRSubstitution** в **MRMapping,** при условии что комбинация атрибута и **oldMatchingRule** (если присутствует) является уникальной. Если **oldMatchingRule** отсутствует в одном **MRSubstitution**, но представлен в другом **MRSubstitution**, последний имеет преимущество при отображении правила сопоставления, определенного компонентом **oldMatchingRule**.

16.10.8 Компонент дополнительного управления

Компонент **additionalControl** позволяет адаптировать к конкретной среде воздействие управляющего правила поиска, для чего необходимо дополнительное управление операцией поиска. Он определяет один или более типов атрибутов управления. Семантика, синтаксис и размещение такого типа атрибута управления, на который ссылается данный компонент, описываются как часть определения этого атрибута управления. Такая спецификация может не охватываться настоящими спецификациями Справочника. Описанный атрибут управления включает часть процедур своего описания, которые должны выполняться на основании информации, предоставленной атрибутом управления.

Этот компонент не затрагивает функцию проверки поиска.

Атрибут управления может размещаться таким образом, что будет затрагивать несколько статей, например в административной точке, зависящей от службы, или в подстатье административного управления службой. Он также может размещаться в отдельных статьях. Если атрибут управления размещается в отдельных статьях, он может затрагивать только выбор информации статьи для этих статей. Результатом действия атрибута управления может быть *явное снятие выделения* с определенных статей или членов семейства, что не допускает их включения в результате поиска.

ПРИМЕЧАНИЕ 1. – При размещении атрибута управления в зависящей от службы административной точке атрибут управления может затрагивать метод выполнения сопоставления. Например, тип атрибута, определенный в пункте фильтра, может быть отображен в совокупность типов атрибутов или дополнен ею ("дружественные" типы атрибутов), по которой может быть выполнено сопоставление некоторым определенным образом, например для получения того же результата путем порождения подтипов атрибута. Аналогичным образом, атрибут управления может корректировать возвращаемую информацию статьи.

ПРИМЕЧАНИЕ 2. – При размещении атрибута управления в данной статье становится возможным учитывать отдельные требования, например выполнить требования по защите персональных данных.

Если составные статьи маркируются или маркировка с них снимается в результате обработки одного или более атрибутов управления, это должно происходить до применения спецификации возвращения семейства (как определено компонентом **familyReturn** в **EntryInformationSelection** или как переопределено компонентом **familyReturn** правила поиска). Если снятие явным образом маркировки приводит к тому, что не возвращается ни один из членов составной статьи, эта составная статья полностью удаляется из результата.

16.10.9 Прочие компоненты

Компонент **allowedSubset** определяет действительные выборы спецификации **subset** запроса поиска. Этот компонент правила поиска игнорируется, если присутствует компонент **imposedSubset** правила поиска и в запросе

search не установлен **useSubset** управления поиском. По умолчанию возможен любой выбор **subset**. Если параметр **subset** запроса **search** не определяет значения, совместимого с этим компонентом правила поиска, проверка поиска на соответствие этому правилу оканчивается неудачей. Для операций чтения и изменения статьи, с тем чтобы обеспечить согласование с данным компонентом, должно быть включено значение **baseObject**.

Компонент **imposedSubset** определяет параметр **subset**, который замещает спецификацию **subset** в запросе **search**. Если этот компонент не присутствует или если в запросе **search** установлена функция управления поиском **useSubset**, замещение не выполняется и применяется ограничение, сформулированное в **allowedSubset**. Это компонент должен игнорироваться в процессе оценки запроса **read** или **modifyEntry** на соответствие правилу поиска.

Компонент **entryLimit** содержит два подкомпонента. Подкомпонент **default** указывает ограничение размера, которое должно налагаться Справочником, если не установлена опция управления службой **sizeLimit**. Подкомпонент **max** указывает максимальное разрешенное значение для опции управления службой **sizeLimit**. Если это значение превышается, фактическое значение **sizeLimit** уменьшается до этого значения **max**. Этот компонент должен игнорироваться в процессе оценки запроса **read** или **modifyEntry** на соответствие правилу поиска.

16.10.10 Классы информационных объектов ASN.1

Для упрощения документирования правил поиска определяются классы информационных объектов **SEARCH-RULE**, **REQUEST-ATTRIBUTE** и **RESULT-ATTRIBUTE**:

```
SEARCH-RULE ::= CLASS {
        &dmdId                  OBJECT IDENTIFIER,
        &serviceType            OBJECT IDENTIFIER                       OPTIONAL,
        &userClass              INTEGER                                 OPTIONAL,
        &InputAttributeTypes    REQUEST-ATTRIBUTE                       OPTIONAL,
        &combination            AttributeCombination                    OPTIONAL,
        &OutputAttributeTypes   RESULT-ATTRIBUTE                        OPTIONAL,
        &defaultControls        ControlOptions                          OPTIONAL,
        &mandatoryControls      ControlOptions                          OPTIONAL,
        &searchRuleControls     ControlOptions                          OPTIONAL,
        &familyGrouping         FamilyGrouping                          OPTIONAL,
        &familyReturn           FamilyReturn                            OPTIONAL,
        &additionalControl      AttributeType                           OPTIONAL,
        &relaxation             RelaxationPolicy                        OPTIONAL,
        &allowedSubset          AllowedSubset                           DEFAULT '111'B,
        &imposedSubset          ImposedSubset                           OPTIONAL,
        &entryLimit             EntryLimit                              OPTIONAL,
        &id                     INTEGER UNIQUE }
WITH SYNTAX {
        DMD ID                  &dmdId
        [ SERVICE-TYPE          &serviceType ]
        [ USER-CLASS            &userClass ]
        [ INPUT ATTRIBUTES      &InputAttributeTypes ]
        [ COMBINATION           &combination ]
        [ OUTPUT ATTRIBUTES     &OutputAttributeTypes ]
        [ DEFAULT CONTROL       &defaultControls ]
        [ MANDATORY CONTROL     &mandatoryControls ]
        [ SEARCH-RULE CONTROL   &searchRuleControls ]
        [ FAMILY-GROUPING       &familyGrouping ]
        [ FAMILY-RETURN         &familyReturn ]
        [ ADDITIONAL CONTROL    &additionalControl ]
        [ RELAXATION            &relaxation ]
        [ ALLOWED SUBSET        &allowedSubset ]
        [ IMPOSED SUBSET        &imposedSubset ]
        [ ENTRY LIMIT           &entryLimit ]
        ID                      &id }

REQUEST-ATTRIBUTE ::= CLASS {
        &attributeType          ATTRIBUTE.&id,
        &SelectedValues         ATTRIBUTE.&Type                         OPTIONAL,
        &DefaultValues          SEQUENCE {
                                    entryType    OBJECT-CLASS.&id       OPTIONAL,
                                    valuesSEQUENCE OF ATTRIBUTE.&Type }  OPTIONAL,
        &contexts               SEQUENCE OF ContextProfile              OPTIONAL,
        &contextCombination     ContextCombination                      OPTIONAL,
        &MatchingUse            MatchingUse                             OPTIONAL,
        &includeSubtypes        BOOLEAN                                 DEFAULT  FALSE
        }
```

```
WITH SYNTAX  {
        ATTRIBUTE TYPE                  &attributeType
        [ SELECTED VALUES               &SelectedValues ]
        [ DEFAULT VALUES                &DefaultValues ]
        [ CONTEXTS                      &contexts ]
        [ CONTEXT COMBINATION           &contextCombination ]
        [ MATCHING USE                  &MatchingUse ]
        [ INCLUDE SUBTYPES              &includeSubtypes ] }

RESULT-ATTRIBUTE  ::=  CLASS {
        &attributeType                  ATTRIBUTE.&id,
        &outputValues                   CHOICE {
                selectedValues          SEQUENCE OF ATTRIBUTE.&Type,
                matchedValuesOnly       NULL }                          OPTIONAL,
        &contexts                       ContextProfile                  OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE          &attributeType
        [ OUTPUT VALUES         &outputValues ]
        [ CONTEXTS              &contexts ] }
```

16.11 Определение ограничения сопоставления

Административный орган может пожелать наложить ограничения на порядок применения правила сопоставления. Например, ограничение на правило сопоставления подстрок может указывать минимальные значения длины подстрок, предоставляемых в пункте фильтра поиска. Такие ограничения имеют постоянный характер и в отличие от ослабления поиска не содержат динамических характеристик.

В пределах зависящей от службы административной области наложение ограничений может выполнять соответствующая структура правил поиска, и это единственное место, в котором возможно введение ограничений на сопоставление.

Ограничения сопоставления могут определяться как значения класса информационных объектов **MATCHING-RESTRICTION**:

```
MATCHING-RESTRICTION ::= CLASS {
        &Restriction,
        &Rules              MATCHING-RULE.&id,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
        RESTRICTION         &Restriction
        RULES               &Rules

        ID                  &id }
```

Для ограничения правила сопоставления, определенного с использованием данного класса информационных объектов:

a) **&Restriction** – синтаксис для ограничения составления, которое должно применяться;

b) **&Rules** – совокупность правил сопоставления, к которым может применяться данное ограничение. Ограничения могут определяться только для базового правила сопоставления, т. е. для правил сопоставления, не являющихся родительскими в своем определении;

c) **&id** – присвоенный данному правилу идентификатор объекта.

Для любого одного правила сопоставления могут определяться несколько ограничений, но в конкретной ситуации применяться может только одно.

16.12 Функция проверки поиска

Функция проверки поиска – это абстрактная функция, которая используется для определения совместимости запроса поиска с конкретным правилом поиска. Функция проверки поиска выдает значение TRUE, если запрос поиска согласуется с правилом поиска. В любом ином случае функция выдает значение FALSE. Для обеспечения совместимости запроса поиска с правилом поиска:

– в фильтре поиска, инвертированном или неинвертированном, не должны присутствовать в какой бы то ни было форме отличные от представленных компонентом **inputAttributeTypes**;

– если тип атрибута присутствует в фильтре, он должен также присутствовать эффективно;

  ПРИМЕЧАНИЕ. – Это означает, что любой тип атрибута не должен представляться только инвертированными пунктами фильтра.

– должно выполняться условие фактического присутствия атрибутов запроса, как определено компонентом **attributeCombination** правила поиска;

– если существуют профили атрибутов запроса, включающие подкомпонент **selectedValues**, соответствующие атрибуты должны быть представлены только неинвертированными пунктами фильтра;

– спецификация **subset** в аргументе запроса должна быть согласованной со спецификацией **subset** правила поиска;

– обязательное управление, определенное компонентом **mandatoryControls,** должно точно соответствовать определению **defaultControls** для данного правила поиска.

Для типа атрибута, представленного одним или более пунктами фильтра в подфильтре, для обеспечения фактического присутствия в этом подфильтре по крайней мере один из пунктов фильтра должен соответствовать спецификации **RequestAttribute** для данного типа атрибута, т. е.:

– тип пунктов фильтра должен таким, как определено в п. 16.6;

– если подкомпонент **selectedValues** присутствует в профиле атрибутов запроса и не является пустым, пункт фильтра должен совпадать с этим подкомпонентом;

– спецификация контекста в пункте фильтра должна быть согласованной со спецификациями контекста в профиле атрибута запроса;

– спецификация правила сопоставления в пункте фильтра должна быть согласованной со спецификациями правила сопоставления в профиле атрибутов запроса; и

– должны выполняться все ограничения сопоставления.

Подробно процедура проверки поиска определяется в п. 13 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

РАЗДЕЛ 8 – БЕЗОПАСНОСТЬ

17 Модель обеспечения безопасности

17.1 Определения

В настоящей спецификации Справочника используются следующие термины, определенные в Рек. МККТТ X.800 | ИСО/МЭК 7498-2:

- управление доступом;
- аутентификация;
- стратегия обеспечения безопасности;
- конфиденциальность;
- целостность.

В настоящей спецификации Справочника определяются следующие термины:

17.1.1 схема управления доступом: Средства, с помощью которых можно осуществлять управление доступом к информации Справочника и потенциально доступ к самим правам доступа.

17.1.2 защищенный объект: Элемент информации Справочника, управление доступом к которому может осуществляться отдельно. Защищенными объектами Справочника являются статьи, атрибуты, значения и имена атрибутов.

17.2 Стратегии обеспечения безопасности

Справочник существует в среде, где различные административные органы управляют доступом к своей части DIB. Такой доступ выполняется согласно определенной административно управляемой стратегии обеспечения безопасности (см. Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8).

Двумя аспектами или компонентами стратегии обеспечения безопасности, влияющими на доступ к Справочнику, являются процедуры аутентификации и схема управления доступом.

ПРИМЕЧАНИЕ. – В п. 18 описываются две схемы управления доступом, известные как базовое управление доступом и упрощенное управление доступом, а в п. 19 определяется управление доступом на основе правил. Эти схемы могут использоваться в сочетании с местным административным управлением, однако в силу того, что местная административная стратегия не имеет стандартизованного представления, относящиеся к ней данные не могут передаваться в скрытой информации.

17.2.1 Процедуры и механизмы аутентификации

Процедуры и механизмы аутентификации в контексте Справочника включают методы проверки и при необходимости распространения:

- идентификационной информации агентов DSA и пользователей Справочника;
- идентификационной информации источника информации, полученной в точке доступа.

ПРИМЕЧАНИЕ 1. – Административный орган может предусматривать различные условия для аутентификации административных пользователей по сравнению с условиями для идентификации неадминистративных пользователей.

Общепримененимые процедуры аутентификации определяются в Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8 и могут использоваться в сочетании со схемами управления доступом, описанными в настоящей спецификации Справочника, для обеспечения применения стратегии обеспечения безопасности.

ПРИМЕЧАНИЕ 2. – В последующих изданиях спецификаций Справочника могут определяться иные схемы управления доступом.

ПРИМЕЧАНИЕ 3. – Местная административная стратегия может вводить положение об игнорировании аутентификации, выполняемой в определенных иных агентах DSA (например, DSA в других DMD).

В целом должна существовать некая функция отображения аутентифицированной идентификационной информации (например, идентификация человека-пользователя, аутентифицированная путем обмена аутентификацией) в идентификационной информацией по управлению доступом (например, выделенное имя статьи вместе с необязательным уникальным идентификатором, представляющим этого пользователя). Конкретная стратегия безопасности может считать аутентифицированную идентификационную информацию и идентификационную информацию управления доступом одинаковыми.

Для обозначения имен в идентификационной информации по управлению доступом должны использоваться первичные выделенные имена. Аналогичным образом, если управление доступом использует имена в спецификации своих разрешений и запретов, это должны быть первичные выделенные имена.

17.2.2 Схема управления доступом

Описание схемы управления доступом в контексте Справочника включает методы:

- определения информации по управлению доступом (ACI);
- обеспечения осуществления прав, определенных информацией по управлению доступом;
- поддержания информации по управлению доступом.

Обеспечение соблюдения прав доступа применяется к управлению доступом в части:

- информации Справочника, касающейся имен;
- пользовательской информации Справочника;
- операционной информации Справочника, включая информацию по управлению доступом.

Для реализации своей стратегии обеспечения безопасности административные органы могут использовать любую стандартизированную схему управления доступом целиком или ее части или могут без ограничений определять по своему усмотрению собственные схемы.

Вместе с тем, административные органы могут вводить положения для защиты некоторой или всей операционной информации Справочника. Административные органы не обязаны предоставлять обычным пользователям средства обнаружения положений для защиты операционной информации.

ПРИМЕЧАНИЕ 1. – В рамках стратегии административного управления может быть предоставлен или запрещен доступ к конкретным атрибутам (например, операционным атрибутам) независимо от опций управления доступом, которые могли бы применяться в любом ином случае.

Справочник обеспечивает средства для действующей схемы управления доступом в конкретной части DIB, которая должна быть указана с помощью операционного атрибута **accessControlScheme.** Область действия такой схемы определяется специальной областью управления доступом (ACSA), которая является специальной административной областью, относящейся к компетенции соответствующего органа безопасности. Этот атрибут помещается в административную статью для соответствующей административной точки. Только административным статьям для специальных точек управления доступом разрешено содержать атрибут **accessControlScheme**.

ПРИМЕЧАНИЕ 2. – Если этот операционный атрибут не включен по отношению к доступу к данной статье, поведение DSA должно соответствовать DSA в первом издании (т. е. определение механизма управления доступом и его воздействия на операции, результаты и ошибки относится к местной компетенции).

```
accessControlScheme  ATTRIBUTE  ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        USAGE                           directoryOperation
        ID                              id-aca-accessControlScheme }
```

Содержание ACI в любой подстатье или статье в пределах ACSA допустимо, исключительно если такая ACI допустима и согласуется со значением атрибута **accessControlScheme** соответствующей ACSA.

17.3 Защита операций Справочника

Существуют две доступные для операций Справочника формы защиты: конфиденциальность и целостность.

Конфиденциальность возможна только по принципу точка-точка путем применения защиты транспортного уровня TLS, которая может быть вызвана для протоколов IDM Справочника и для LDAP. TLS для протоколов OSI Справочника не доступна. Следует заметить, что защита точка-точка может оказаться неадекватной в распределенной среде; вместе с тем, сквозная конфиденциальность обеспечивается только путем защиты самих атрибутов.

Целостность обеспечивается двумя способами. Целостность точка-точка может обеспечиваться для протоколов IDM Справочника и для LDAP путем применения TLS. Сквозная целостность может быть обеспечена путем подписания и необязательного сцепления подписанных отличных от LDAP операций Справочника с использованием опции необязательной защиты OPTIONALLY-PROTECTED, описанной ниже. Протокольные блоки PDU, содержащие операции Справочника, не защищаются, но защищаются аргументы, результаты и ошибки. Не существует механизма обеспечения защищенной постоянной регистрации таких событий, как операции DAP. Операции LDAP посредством настоящей спецификации Справочника не защищаются.

ПРИМЕЧАНИЕ. – В экспериментальном запросе для комментариев IETF RFC 2649 "Управление и схема LDAP для хранения подписей операций", предлагается механизм для подписания блоков PDU, содержащих операции LDAP, и для обеспечения постоянной защищенной регистрации этих операций.

OPTIONALLY-PROTECTED – параметризованный тип данных, где параметром является тип данных, значения которого могут, в зависимости от опции генератора, сопровождаться своими цифровыми подписями. Это возможность описывается следующим образом:

```
OPTIONALLY-PROTECTED { Type } ::= CHOICE {
        unsigned            Type,
        signed              SIGNED {Type} }
```

Если защищаемым типом данных является тип данных последовательности без маркировки, вместо **OPTIONALLY-PROTECTED** используется **OPTIONALLY-PROTECTED-SEQ**.

```
OPTIONALLY-PROTECTED-SEQ { Type } ::= CHOICE {
        unsigned            Type,
        signed  [0]         SIGNED { Type } }
```

Параметризованный тип данных **SIGNED**, который описывает подписанную форму информации, определяется в Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8.

18 Базовое управление доступом

18.1 Сфера действия и применение

В данном пункте описывается одна (из многих возможных) конкретная схема управления доступом для Справочника. Определенная в настоящем документе схема управления доступом определяется операционным атрибутом **accessControlScheme** путем присвоения ему значения **basic-access-control**. В п. 17.2.2 поясняется, какие статья содержат операционный атрибут **accessControlScheme**.

ПРИМЕЧАНИЕ. – Схема управления доступом, называемая "Упрощенное управление доступом", описывается в п. 18.9. Она определяется как подмножество схемы базового управления доступом. Если используется упрощенное управление доступом, операционный атрибут **accessControlScheme** должен иметь значение **simplified-access-control.** Дополнительные схемы управления доступом, называемые "Управление доступом на основе правил" описываются в п. 19.

Описываемая ниже схема предоставляет только средства управления доступом к информации Справочника в пределах DIB (возможно включая информацию о структуре дерева и управлении доступом). Схема не касается управления доступом для целей связи с объектом прикладного уровня DSA. Управление доступом к информации означает предотвращение несанкционированного обнаружения, раскрытия или изменения этой информации.

18.2 Модель базового управления доступом

Модель базового управления доступом для Справочника определяет для каждой операции Справочника одну или более точек, в которых принимаются решения по управлению доступом. Все решения по управлению доступом включают:

- элемент информации Справочника, к которому осуществляется доступ, называемый *защищенный объект*;
- пользователя, который запрашивает информацию, называемого *запросчик*;
- конкретное право, необходимое для выполнения части операции, называемое *разрешение*;
- один или более операционных атрибутов, которые вместе содержат стратегию безопасности, управляющую доступом к этому пункту, называемые *пункты ACI*.

Таким образом, базовая модель управления доступом определяет:

- защищенные объекты;
- классы пользователей;
- категории разрешений, необходимые для выполнения каждой операции Справочника;
- область применения и синтаксис пунктов ACI;
- базовый алгоритм, называемый функцией принятия решения по управлению доступом (ACDF), который используется для принятия решения о наличии у конкретного запросчика конкретного разрешения на основании применимых пунктов ACI.

18.2.1 Защищенные объекты

Защищенным объектом является элемент информации Справочника, управление доступом к которому может осуществляться отдельно. Защищенными объектами Справочника являются статьи, атрибуты, значения и имена атрибутов. Для удобства описания стратегий управления доступом базовое управление доступом предоставляет средства для идентификации наборов связанных объектов, таких как атрибуты в статье или все значения атрибутов данного атрибута, и для описания общей защиты для них.

18.2.2 Разрешения на управление доступом и их область действия

Управление доступом осуществляется путем предоставления разрешений или отказа в их предоставлении. Категории разрешений описаны в пп. 18.2.3 и 18.2.4.

Областью действия функций управления доступом может быть единственная статья или набор статей, которые являются логически связанными, находясь в пределах области действия подстатьи для конкретной административной точки.

Категории разрешений в общем случае независимы. В силу того, что статьи Справочника имеют относительное местоположение в пределах DIT, доступ к пользовательской и операционной информации всегда предполагает некоторую форму доступа к информации, касающейся DIT. Таким образом, существуют две основные формы решений по управлению доступом, связанные с любой операцией Справочника: доступ к статьям как к именованным объектам (называемый *доступ к статье*) и доступ к атрибутам, содержащим пользовательскую и операционную информацию (называемый *доступ к атрибуту*). Для большого числа операций Справочника требуются обе формы разрешений. Кроме того, если применимо, отдельные разрешения управляют типом возвращаемого имени или ошибки. Ниже приведены ряд важных аспектов категорий разрешений, форм доступа и принятия решений по управлению доступом:

a) Для выполнения операций Справочника со статьями целиком (например, чтение статьи или добавление статьи), как правило, необходимо получение разрешения относительно атрибутов и значений, содержащихся в пределах этой статьи. Исключениями являются разрешения, контролирующие переименование и удаление статей: ни в одном из этих случаев разрешения относительно атрибутов или значений атрибутов не учитываются.

b) Для выполнения операций Справочника, которые требуют доступа к атрибутам или значениям атрибутов, необходимо получить разрешение на доступ к статье в отношении статьи или статей, которые содержат эти атрибуты или значения.

   ПРИМЕЧАНИЕ 1. – Удаление статьи или атрибута не требует доступа к контенту этой статьи или этого атрибута.

c) Решение о предоставлении или непредставлении разрешения на доступ к статье строго определяется местоположением этой статьи в DIT в отношении выделенного имени и не зависит от методов ее обнаружения Справочником.

d) В основе базового управления доступом лежит тот принцип, что доступ может быть разрешен, только при наличии в информации по управлению доступом, которую Справочник использует для принятия решения по управлению доступом, явным образом предоставленного разрешения. Предоставление одной формы разрешений (например, доступ к статье) ни в коем случае не означает автоматического или неявного предоставления другой формы разрешения (например, доступ к атрибуту). Для административного управления целенаправленными стратегиями управления доступом к Справочнику необходимо, следовательно, устанавливать явным образом стратегию доступа для обеих форм доступа.

   ПРИМЕЧАНИЕ 2. – Определенные комбинации разрешений и отказов являются нелогичными, однако обеспечение отсутствия таких комбинаций скорее является обязанностью пользователей, чем Справочника.

   ПРИМЕЧАНИЕ 3. – В соответствии с вышеуказанным основополагающим правилом предоставление разрешений или отказ в них не означают автоматического получения управления доступом к связанному атрибуту. Кроме того, для доступа к значению(ям) атрибута в процессе операции запроса к Справочнику пользователь должен получить доступ как к типу атрибута, так и к его значению(ям).

e) Единственным в данной модели решением по умолчанию в отношении доступа является отказ в доступе в отсутствие существующей в явном виде информации по управлению доступом, которая предоставляет доступ.

f) Определенный в информации по управлению доступом отказ всегда имеет более высокий приоритет перед разрешением при прочих равных условиях.

g) Конкретный DSA может не иметь информации по управлению доступом, управляющей данными Справочника, которые он кэширует. Администраторы по безопасности должны быть осведомлены, что DSA с возможностями кэширования может подвергать серьезному риску в отношении безопасности другие DSA тем, что может открыть информацию неправомочным пользователям.

h) Для целей опроса коллективные атрибуты, которые связаны со статьей, защищаются точно так же, как если бы это были атрибуты, составляющие часть этой статьи.

   ПРИМЕЧАНИЕ 4. – Для целей изменения коллективные атрибуты связываются с содержащей их подстатьей, а не со статьями в пределах области действия этой подстатьи. Связанные с операцией изменения функции управления доступом, таким образом, нерелевантны по отношению к коллективным атрибутам, за исключением случая, когда они применяются к коллективному атрибуту. а его значения находятся в пределах этой подстатьи.

18.2.3 Категории разрешений для доступа к статье

Категориями разрешений, используемыми для управления доступом к статье, являются *Read* (чтение), *Browse* (просмотр), *Add* (добавить), *Remove* (удалить), *Modify* (изменить), *Rename* (переименовать), *DiscloseOnError* (раскрыть в уведомлении об ошибке), *Export* (экспорт), а также *Import* (импорт) и *ReturnDN* (вернуть DN). Их применение более подробно описывается в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3. В Приложении L содержится обзор значения этих разрешений в общих ситуациях. В данном подпункте эти категории представлены кратким описанием назначения каждой из них. Реальное воздействие конкретного выданного разрешения на решения по управлению доступом, однако, определяется полным контекстом ACDF и точками принятия решений по управлению доступом для каждой операции Справочника:

a) *Read*, если предоставлено, разрешает доступ для чтения операциям Справочника, которые конкретно именуют статью (т. е. в отличие от операций составления списка и поиска), и делает видимой информацию, содержащуюся в статье, к которой применяется это разрешение.

b) *Browse*, если предоставлено, разрешает доступ к статьям с помощью операций Справочника, которые не задают в явном виде имя статьи.

c) *Add*, если предоставлено, разрешает создание статьи в DIT согласно функциям управления в отношении всех атрибутов и значений атрибутов, которые должны быть помещены в новую статью в момент ее создания.

ПРИМЕЧАНИЕ 1. – Для того чтобы добавить статью, должно быть также выдано разрешение на добавление по крайней мере обязательных атрибутов и их значений.

ПРИМЕЧАНИЕ 2. – Не существует специального "разрешения на добавление подчиненного". Разрешение на добавление статьи контролируется с помощью операционных атрибутов **prescriptiveACI**, как описано в п. 18.3.

d) *Remove*, если предоставлено, разрешает удаление статьи из DIT, независимо от функций управления в отношении атрибутов и значений атрибутов в пределах статьи.

e) *Modify*, если предоставлено, разрешает изменение информации, содержащейся в пределах статьи.

ПРИМЕЧАНИЕ 3. – Для изменения информации, содержащейся в пределах статьи, кроме значений атрибутов выделенного имени также должны быть получены соответствующие разрешения для изменения атрибута и его значения.

f) Получение разрешения *Rename* необходимо для переименования статьи с присвоением ей нового RDN с учетом обусловливаемых этим изменений выделенных имен подчиненных статей, если таковые существуют; если имя старшей статьи не меняется, этого разрешения достаточно.

ПРИМЕЧАНИЕ 4. – Для переименования любой статьи не существует предварительных разрешений в отношении содержащихся атрибутов или значений, включая атрибуты этого RDN; это справедливо, даже если операция вызывает добавление или удаление новых значений атрибутов вследствие изменений RDN.

g) *DiscloseOnError*, если предоставлено, разрешает открыть имя статьи в случае, если результатом является ошибка (или если результат пустой).

h) *Export*, если предоставлено, разрешает экспорт статьи и ее подчиненных статей (если таковые существуют); т. е. удаление статьи из текущего местоположения и помещение в новое местоположение при условии предоставления соответствующих разрешений в точке назначения. Если последнее RDN меняется, также требуется разрешение *Rename* в текущем местоположении.

ПРИМЕЧАНИЕ 5. – Для экспорта статьи или ее подчиненных не существует предварительных разрешений в отношении содержащихся атрибутов или значений, включая атрибуты этого RDN; это справедливо, даже если операция вызывает добавление или удаление новых значений атрибутов вследствие изменений RDN.

i) *Import*, если предоставлено, разрешает импорт статьи и ее подчиненных, если таковые существуют; т. е. удаление статьи из какого-либо иного местоположения и помещение ее в местоположение, к которому применяется это разрешение (при условии наличия соответствующих разрешений в исходном местоположении).

ПРИМЕЧАНИЕ 6. – Для импорта статьи или ее подчиненных не существует предварительных разрешений в отношении содержащихся атрибутов или значений, включая атрибуты этого RDN; это справедливо, даже если операция вызывает добавление или удаление новых значений атрибутов вследствие изменений RDN.

j) *ReturnDN*, если предоставлено, разрешает раскрыть выделенное имя статьи в содержании результата операции.

18.2.4 Категории разрешений доступа к атрибуту и значению атрибута

Категориями разрешений, используемыми для управления доступом к атрибуту и к значению атрибута, являются *Compare* (сравнить), *Read* (читать), *FilterMatch* (совпадение с фильтром), *Add* (добавить), *Remove* (удалить) и *DiscloseOnError* (раскрыть в уведомлении об ошибке). Их применение более подробно описывается в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3. В Приложении L содержится обзор значения этих разрешений в общих ситуациях. В данном подпункте эти категории представлены кратким описанием назначения каждой из них. Реальное воздействие конкретного выданного разрешения на решения по управлению доступом, однако, определяется полным контекстом ACDF и точками принятия решений по управлению доступом для каждой операции Справочника.

a) *Compare*, если предоставлено, разрешает использовать атрибуты и их значения в операции сравнения.

b) *Read*, если предоставлено, разрешает возвращать атрибуты и значения как информацию статьи в операциях чтения и поиска.

c) *FilterMatch*, если предоставлено, разрешает оценку фильтра по критерию поиска.

d) *Add*, если предоставлено, разрешает добавление атрибута при условии возможности добавления всех определенных значений атрибута. Если предоставлено в отношении значения атрибута, разрешает добавление значения к существующему атрибуту.

e) *Remove*, если предоставлено для атрибута, разрешает удаление атрибута целиком со всеми его значениями. Если предоставлено для значения атрибута, разрешает удаление значения из существующего атрибута.

f) *DiscloseOnError*, если предоставлено для атрибута, разрешает открыть факт наличия атрибута в сообщении об ошибке атрибута или нарушении безопасности. Если предоставлено для значения атрибута, разрешает открыть факт наличия значения атрибута в сообщении об ошибке атрибута или нарушении безопасности.

g) *Invoke* (вызов), если предоставлено, объект (всегда операционный атрибут или значение операционного атрибута), к которому применяется это разрешение, может быть вызван агентом DSA от имени аутентифицированного пользователя. Выполняемая вызовом функция зависит от атрибута. Пользователю не требуется каких-либо иных разрешений для операционного атрибута или в отношении содержащей его статьи/подстатьи.

18.3 Административные области управления доступом

DIT разбивается на поддеревья, которые называются "автономные административные области", находящиеся под управлением административного органа одной организации, управляющей доменом. Они могут далее разбиваться на поддеревья, называемые "специальные административные области" в соответствии с конкретными аспектами административного управления; иначе вся автономная административная область может содержать единственную специальную административную область. Все такие специальные административные области управляются соответствующими специальными административными органами. Конкретная административная область может использоваться несколькими административными органами на коллективной основе. См. п. 11.

18.3.1 Области управления доступом и домены управления доступом к Справочнику

В случае управления доступом специальный административный орган является органом по безопасности, а специальная административная область называется "специальная область управления доступом" (ACSA). Корень ACSA называется "специальная точка управления доступом". Каждая специальная точка управления доступом представлена в DIT административной статьей, которая в качестве значения своего операционного атрибута **administrativeRole** содержит **access-control-specific-area**; эта статья также имеет (потенциально) одну или более подстатей, которые содержат информацию по управлению доступом. Аналогичным образом, каждая внутренняя точка управления доступом представлена в DIT административной статьей, которая в качестве значения своего операционного атрибута **administrativeRole** содержит **access-control-specific-area**; эта точка также имеет (потенциально) одну или более подстатей, которые содержат информацию по управлению доступом. Каждая такая административная статья, включающая предписывающую информацию ACI подстатьи, имеет значением своего операционного атрибута **accessControlScheme** значение **basic-access-control**, **simplified-access-control** или иное соответствующее значение. Каждая подстатья, принадлежащая специальной точке управления доступом и включающая информацию по управлению доступом, в качестве значения своего атрибута класса объектов имеет **accessControlSubentry**. Административная статья и ее подстатьи могут содержать операционные атрибуты (такие как информация по управлению доступом), которые относятся, соответственно, к административной точке (и, возможно, ее подстатьям) и к наборам статей (в пределах административной области), определенным ее подстатьей **subtreeSpecification**.

Атрибут **accessControlScheme** должен присутствовать, если и только если содержащая административная статья является специальной статьей управления доступом. Административная статья не может быть одновременно специальной статьей управления доступом и внутренней статьей управления доступом; следовательно, соответствующие значения ни при каких условиях не могут существовать в атрибуте **administrativeRole** одновременно.

Область действия подстатьи, которая содержит информацию по управлению доступом, что определяется ее **subtreeSpecification** (который может включать уточнения поддерева), называется доменом управления доступом к Справочнику (DACD).

ПРИМЕЧАНИЕ. – DACD может содержать нуль статей и может охватывать статьи, которые еще не добавлены к DIT.

Орган безопасности может разрешить деление специальной области управления доступом на поддеревья, которые называются внутренними (административными) областями. Каждая такая внутренняя область называется "внутренняя область управления доступом" (ACIA) и имеет значение **access-control-inner-area** в качестве значения операционного атрибута **administrativeRole**. Все подстатьи соответствующей административной точки, содержащей предписывающую информацию ACI, имеют, как и в предыдущем случае, значение **accessControlSubentry** в атрибуте класса объектов.

Область действия (**subtreeSpecification**), определенная в пределах ACIA, также является DACD и содержит статьи внутри связанной внутренней области управления доступом.

Области ACIA обеспечивают в пределах ACSA некоторую возможность делегирования полномочий по управлению доступом. Орган для ACSA сохраняет полномочия в пределах ACIA, так как ACI в подстатьях административной точки ACSA применяется так же, как применяется ACI в подстатьях соответствующих ACIA (в п. 18.6 поясняется, как ACSA управляет полномочиями).

Итак, при оценке функций управления тип схемы управления доступом (например, базовое управление доступом) указывается значением атрибута **accessControlScheme** соответствующей специальной статьи управления доступом; функция каждой актуальной административной статьи в пределах ACSA указывается значениями атрибута **administrativeRole**; наличие предписывающего управления доступом в конкретной подстатье указывается значением **accessControlSubentry** в ее атрибуте класса объектов.

Подстатьи, как и прочие статьи, могут содержать атрибут **entryACI** для защиты собственного контента.

18.3.2 Связь элементов управления с административными областями

Доступ к той или иной данной статье управляется (потенциально) всей совокупностью старших административных точек управления доступом (как внутренних, так и специальных) до первой не являющейся внутренней административной точки управления доступом или до первой встретившейся при движении вверх по DIT в

направлении от статьи к корню автономной административной точки, и включая эту точку. Специальные точки управления доступом, старшие по отношению к данной административной точке управления доступом, не затрагивают управления доступом к данной статье.

ПРИМЕЧАНИЕ 1. – Для целей данного описания автономная административная точка рассматривается как неявная специальная точка управления доступом, даже если она не связана с каким-либо предписывающим элементом управления.

Связь между элементами управления доступом и административными областями характеризуется рядом особенностей:

a) Элементы управления доступом для информации Справочника могут применяться только к выбранным статьям или могут иметь область действия, простирающуюся через части DIB, которые логически связаны общей стратегией обеспечения безопасности и общим администрированием управления доступом.

b) Управление доступом может применяться к статьям в пределах областей ACSA или в пределах областей ACIA путем помещения атрибута **prescriptiveACI** (см. п. 18.5) в пределах одной или нескольких подстатей соответствующей административной статьи управления доступом с областью действия, определенной соответствующим **subtreeSpecification**.

   ПРИМЕЧАНИЕ 2. – Атрибуты **prescriptiveACI** не являются коллективными атрибутами. Существует ряд значительных отличий между атрибутами **prescriptiveACI** и коллективными атрибутами:

   – несмотря на то что атрибут **prescriptiveACI** может влиять на решения по управлению доступом для всех статей в пределах области действия содержащей его подстатьи, атрибут **prescriptiveACI** не рассматривается как предоставляющий доступную информацию для любой такой статьи или как являющийся в каком бы то ни было смысле частью такой статьи;

   – атрибуты **prescriptiveACI** связаны с аспектами управления доступом административного управления и связаны со специальными и внутренними точками управления доступом, но не с административными точками набора статей;

   – назначением атрибута **prescriptiveACI** является выражение стратегии, которая действует в рамках указанной совокупности статей, а назначением коллективных атрибутов является обеспечение информации, которая связывает доступную для пользователей совокупность атрибутов в пределах указанной совокупности статей;

   – атрибуты **prescriptiveACI** представляют информацию стратегии, которая в общем случае не является широко доступной для обычных пользователей. Административные пользователи, которым необходим доступ к информации атрибутов **prescriptiveACI**, могут получить к ним доступ как к операционным атрибутам в пределах подстатей.

c) Операционный атрибут **prescriptiveACI** содержит **ACIItems** (см. п. 18.4.1), общие для всех статей в пределах области действия подстатьи, т. е. DACD, в котором существует **prescriptiveACI**. DACD содержит, как правило, статьи внутри связанной специальной области управления доступом (но может вообще не содержать статей).

d) Хотя отдельные параметры **ACIItems** могут определять атрибуты или значения как защищенные объекты, **ACIItems** логически связаны со статьями. Конкретная совокупность **ACIItems**, связанных со статьей и с контентом этой статьи, является комбинацией:

   – **ACIItems**, которые применяются к этой конкретной статье, определенных как значения операционного атрибута **entryACI**, если он присутствует (см. п. 18.5.2);

   – **ACIItems** из операционных атрибутов **prescriptiveACI**, применимых к этой статье в силу своего размещения в подстатьях административных статей, область действия которых включает эту конкретную статью (см. п. 18.5.1).

e) Каждая статья (управляемая **entryACI** и/или **prescriptiveACI**) обязательно находится в пределах одной и только одной ACSA. Каждая такая статья может также находиться в пределах одной или более ACIA, вложенных в ACSA, которая содержит эту статью. Атрибуты **prescriptiveACI**, которые потенциально воздействуют на результат решения по управлению доступом для данной статьи, располагаются в пределах подстатей (административной статьи) для этой ACSA и для каждой ACIA, которая содержит данную статью. Прочие подстатьи не могут воздействовать на решения по управлению доступом, касающиеся этой статьи.

f) Если статья находится в пределах области действия более чем одного DACD, полная совокупность **ACIItems**, которые могут потенциально воздействовать на решения по управлению доступом, касающиеся этой статьи, включает все атрибуты элементов **prescriptiveACI** этих DACD наряду со всеми атрибутами **entryACI** в самой статье. На рисунке 17 приведен пример. Исполнительным управлением доступом в статье E1 является комбинация **prescriptiveACI** для DACD1, DACD2, DACD3 и **entryACI** (если существует) в статье E1. Исполнительным управлением доступом в статье E2 является комбинация **prescriptiveACI** для DACD1 и DACD3 и **entryACI** (если существует) в статье E2.

   ПРИМЕЧАНИЕ 3. – Защита информации по управлению доступом описывается в п. 18.6.

g) Атрибут **subtreeSpecification** во всех подстатьях определяет набор статей в пределах административной области. Учитывая, что **subtreeSpecification** может определять уточнение поддерева, домены DACD могут произвольным образом перекрываться при пересечении их соответствующих административных областей. Для удобства на рисунке 17 не показаны административные точки, подстатьи или административные области, однако можно считать, что на рисунке представлены три DACD в пределах той же ACSA, все DACD которой соответствуют единственной подстатье административной точки для этой ACSA (и отсутствуют области ACIA). Альтернативным образом, рисунок 17 можно рассматривать как представляющий единственную

ACSA, содержащую единственную ACIA, где DACD1 конгруэнтен ACSA, а DACD3 конгруэнтен ACIA (DACD1 и DACD2 соответствовали бы подстатьям административной точки ACSA, а DACD3 соответствовал бы подстатье административной точки ACIA). Административная область конгруэнтна DACD, если набор статей в DACD тот же. что и набор статей в неявным образом определенном поддереве, соответствующем этой административной области. В Приложении M приведены значения, определяющие взаимоотношения между административными статьями, административными областями, подстатьями и доменами DACD.

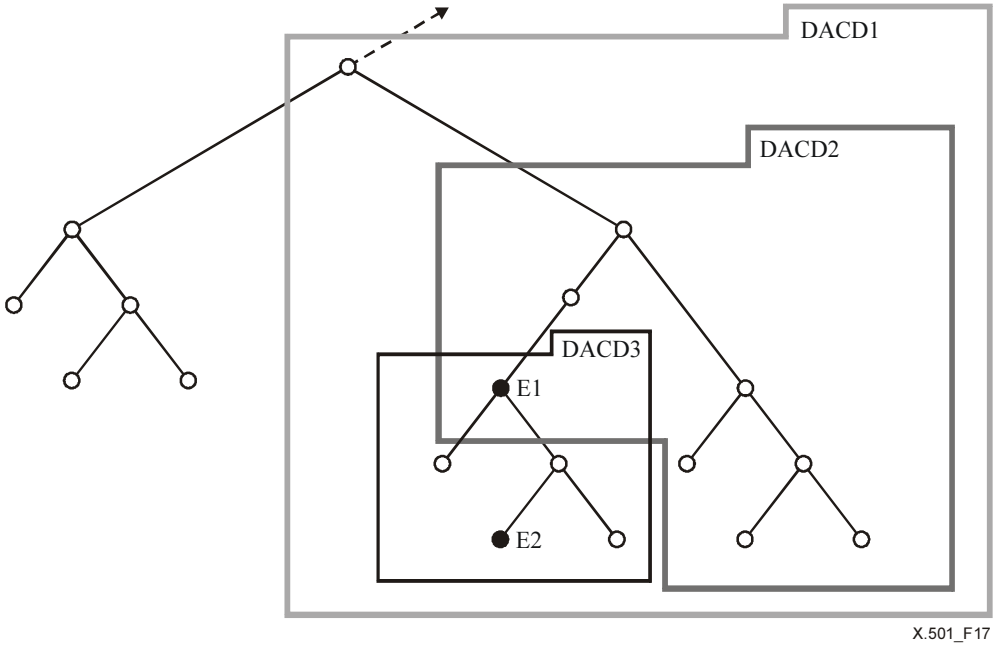


Рисунок 17 – Исполнительное управление доступом с использованием DACD

18.4 Представление информации по управлению доступом

18.4.1 Язык ASN.1 для информации по управлению доступом

Информация по управлению доступом представляется как совокупность **ACIItem**, где каждый **ACIItem** предоставляет разрешение или отказывает в разрешении в отношении некоторых определенных пользователей и защищенных объектов.

На языке ASN.1 информация описывается следующим образом:

```
ACIItem ::= SEQUENCE {
        identificationTag               DirectoryString { ub-tag },
        precedence                      Precedence,
        authenticationLevel             AuthenticationLevel,
        itemOrUserFirst                 CHOICE {
                itemFirst           [0] SEQUENCE {
                        protectedItems      ProtectedItems,
                        itemPermissions     SET OF ItemPermission },
                userFirst           [1] SEQUENCE {
                        userClasses         UserClasses,
                        userPermissions     SET OF UserPermission } } }

Precedence ::= INTEGER (0..255)

ProtectedItems ::= SEQUENCE {
        entry                           [0] NULL                                    OPTIONAL,
        allUserAttributeTypes           [1] NULL                                    OPTIONAL,
        attributeType                   [2] SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
        allAttributeValues              [3] SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
        allUserAttributeTypesAndValues  [4] NULL                                    OPTIONAL,
        attributeValue                  [5] SET SIZE (1..MAX) OF AttributeTypeAndValue OPTIONAL,
        selfValue                       [6] SET SIZE (1..MAX) OF AttributeType      OPTIONAL,
        rangeOfValues                   [7] Filter                                  OPTIONAL,
        maxValueCount                   [8] SET SIZE (1..MAX) OF MaxValueCount      OPTIONAL,
        maxImmSub                       [9] INTEGER                                 OPTIONAL,
```

```
        restrictedBy                    [10]  SET SIZE (1..MAX) OF RestrictedValue        OPTIONAL,
        contexts                        [11]  SET SIZE (1..MAX) OF ContextAssertion
        OPTIONAL,
        classes                         [12]  Refinement                                  OPTIONAL
}

MaxValueCount ::= SEQUENCE {
        type            AttributeType,
        maxCount        INTEGER }

RestrictedValue ::= SEQUENCE {
        type            AttributeType,
        valuesIn        AttributeType }

UserClasses ::= SEQUENCE {
        allUsers    [0]  NULL                                          OPTIONAL,
        thisEntry   [1]  NULL                                          OPTIONAL,
        name        [2]  SET SIZE (1..MAX) OF NameAndOptionalUID       OPTIONAL,
        userGroup   [3]  SET SIZE (1..MAX) OF NameAndOptionalUID       OPTIONAL,
                         -- компонентом dn должно быть имя
                         -- статьи из GroupOfUniqueNames
        subtree     [4]  SET SIZE (1..MAX) OF SubtreeSpecification     OPTIONAL }

ItemPermission ::= SEQUENCE {
        precedence              Precedence OPTIONAL,
                         -- по умолчанию к предшествованию в ACIItem
        userClasses             UserClasses,
        grantsAndDenials        GrantsAndDenials }

UserPermission ::= SEQUENCE {
        precedence              Precedence OPTIONAL,
                         -- по умолчанию к предшествованию в ACIItem
        protectedItems          ProtectedItems,
        grantsAndDenials        GrantsAndDenials }

AuthenticationLevel ::= CHOICE {
        basicLevels SEQUENCE {
                level           ENUMERATED { none (0), simple (1), strong (2) },
                localQualifier  INTEGER OPTIONAL,
                signed          BOOLEAN DEFAULT FALSE },
        other           EXTERNAL }

GrantsAndDenials ::= BIT STRING {
        -- разрешения, которые могут использоваться в сочетании
        -- с любым компонентом ProtectedItems
        grantAdd                (0),
        denyAdd                 (1),
        grantDiscloseOnError    (2),
        denyDiscloseOnError     (3),
        grantRead               (4),
        denyRead                (5),
        grantRemove             (6),
        denyRemove              (7),
        -- разрешения, которые могут использоваться только в сочетании
        -- с компонентом этой статьи
        grantBrowse             (8),
        denyBrowse              (9),
        grantExport             (10),
        denyExport              (11),
        grantImport             (12),
        denyImport              (13),
        grantModify             (14),
        denyModify              (15),
        grantRename             (16),
        denyRename              (17),
        grantReturnDN           (18),
        denyReturnDN            (19),
```

```
        -- разрешения, которые могут использоваться в сочетании
        -- с любым компонентом, кроме статьи, ProtectedItems
        grantCompare            (20),
        denyCompare             (21),
        grantFilterMatch        (22),
        denyFilterMatch         (23),
        grantInvoke             (24),
        denyInvoke              (25) }

AttributeTypeAndValue ::= SEQUENCE {
        type        ATTRIBUTE.&id ({SupportedAttributes}),
        value       ATTRIBUTE.&Type({SupportedAttributes}{@type}) }
```

18.4.2 Описание параметров ACIItem

18.4.2.1 Бирка

Бирка **identificationTag** используется для идентификации конкретного **ACIItem**. Это применяется для проведения различия между отдельными **ACIItem** для целей защиты, управления и административного управления.

18.4.2.2 Приоритетность

Приоритетность используется для управления относительным порядком, при котором **ACIItem** рассматриваются в процессе принятия решения по управлению доступом в соответствии с п. 18.8. Параметры **ACIItem**, имеющие значения высшего приоритета, при прочих равных условиях могут превалировать над другими, имеющими значения с более низким приоритетом. Значения приоритета являются целыми величинами и сравниваются как таковые.

Приоритетность может использоваться старшим органом в органе безопасности для разрешения частичной передачи стратегии управления доступом, установленной в пределах ACSA. Этого можно достичь путем присвоения старшим органом высшего приоритета общей стратегии и наделения представляющих подчиненный орган (например, связанный с ACIA) пользователей правом создавать и изменять ACI с более низким приоритетом, с тем чтобы адаптировать общую стратегию к конкретным задачам. Таким образом, частичная передача требует наличия у старшего органа средств, необходимых для ограничения максимального приоритета, который подчиненный орган может присвоить ACI, находящейся под его управлением.

Базовое управление доступом не определяет и не описывает, каким образом ограничивается максимальный приоритет, который может использовать подчиненный орган. Это должно выполняться местными средствами.

18.4.2.3 Уровень аутентификации

Уровень аутентификации **AuthenticationLevel** определяет минимальный уровень безопасности запросчика, требуемый для этого **ACIItem**. Существуют две формы уровня аутентификации:

- **basicLevels**, которая определяет уровень аутентификации, факультативно уточненный положительным или отрицательным целым значением **localQualifier**, и требуется ли подписание запроса;
- **other**: определяемая внешне мера.

При использовании **basicLevels** агент DSA должен присвоить запросчику **AuthenticationLevel**, состоящий из **level** и необязательного **localQualifier**, в соответствии с местной стратегией. Для того чтобы уровень аутентификации запросчика удовлетворял или превышал минимальное требование, **level** запросчика должен соответствовать или превышать уровень, определенный в **ACIItem**, и, кроме того, **localQualifier** запросчика должен быть арифметически больше или равен значению **ACIItem**. Строгой аутентификацией запросчика считается превышающая требования для простой аутентификации или отсутствия аутентификации, а требования простой аутентификации превышают требования для отсутствия аутентификации. Для целей управления доступом "простой" уровень аутентификации требует пароль; случай только идентификации без ввода пароля рассматривается как "никакой". Если **localQualifier** не определяется в **ACIItem**, запросчику не требуется иметь соответствующее значение (если такое значение существует, оно игнорируется). Кроме выполнения и превышения вышеуказанных требований запрос должен быть подписан, если **ACIItem** определяет значение **signed** равным **TRUE**.

При использовании **other** агент DSA должен присвоить запросчику соответствующий **AuthenticationLevel** в соответствии с местной стратегией. Форма этого **AuthenticationLevel** и метод его сравнения с **AuthenticationLevel** в ACI является вопросом местной компетенции.

ПРИМЕЧАНИЕ 1. – Уровень аутентификации, связанный с явным отказом, указывает минимальный уровень, согласно которому должен быть аутентифицирован запросчик, чтобы не получить отказ в доступе. Например, **ACIItem**, который отказывает в доступе конкретному классу пользователей и требует строгой аутентификации, откажет в доступе всем запросчикам, которые не могут подтвердить с помощью идентификационной информации, прошедшей строгую аутентификацию, что они не относятся к этому классу пользователей.

ПРИМЕЧАНИЕ 2. – DSA может определять уровень аутентификации, основываясь не на полученных в результате протокольных обменов значениях, а на иных факторах.

18.4.2.4 Параметры itemFirst и userFirst

Все **ACIItem** содержат выбор вариантов **itemFirst** или **userFirst**. Эти варианты позволяют осуществлять группирование разрешений в зависимости от того, по классам пользователей или по защищенным объектам они наиболее удобно группируются. Варианты **itemFirst** и **userFirst** эквивалентны в том смысле, что они собирают ту же информацию по управлению доступом, но они по-разному организуют эту информацию. Выбор между ними базируется на удобстве с административной точки зрения. Ниже описаны используемые в **itemFirst** или **userFirst** параметры.

a) **ProtectedItems** определяет объекты, к которым применяются конкретные функции управления доступом. Этот параметр определяется как совокупность, составленная из следующих пунктов:

- **entry** означает контент статьи в целом. В случае члена семейства этот элемент также означает контент статьи каждого подчиненного члена семейства в пределах того же составного атрибута. Он необязательно включает информацию в эти статьи. Этот пункт должен игнорироваться, если присутствует элемент **classes**, поскольку последний выбирает защищенные статьи (и подчиненные члены семейства) на основе их классов объектов.
- **allUserAttributeTypes** означает информацию обо всех типах атрибутов пользователя, связанных с данной статьей, но не значения, связанные с этими атрибутами.
- **allUserAttributeTypesAndValues** означает информацию обо всех типах атрибутов пользователя, связанных с данной статьей, включая все значения всех атрибутов пользователя.
- **attributeType** означает информацию о типе атрибутов, относящуюся к определенным атрибутам, но не значения, связанные с этим типом.
- **allAttributeValues** означает информацию обо всех значениях атрибутов, относящуюся к конкретным атрибутам.
- **attributeValue** означает конкретное значение конкретных атрибутов.
- **selfValue** означает утверждение о значении атрибута, соответствующем действующему запросчику. Защищенный объект **selfValue** применяется, только когда должны применяться функции управления доступом в отношении конкретного аутентифицированного пользователя. Он может применяться только в одном конкретном случае, когда определенный атрибут имеет синтаксис **DistinguishedName** или **uniqueMember**, а значение атрибута в пределах указанного атрибута совпадает с выделенным именем инициатора операции.

ПРИМЕЧАНИЕ 1. – **allUserAttributeTypes** и **allUserAttributeTypesAndValues** не включают операционных атрибутов, которые должны определяться на поатрибутной основе с использованием attributeType, **allAttributeValues** или **attributeValue**.

- **rangeOfValues** означает любое значение атрибута, которое соответствует описанному фильтру, т. е. для которого оценка указанного фильтра возвращает значение TRUE.

  ПРИМЕЧАНИЕ 2. – Оценка фильтра не производится по каким-либо статьям в DIB; она осуществляется с применением определенной в п. 7.8 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3 семантики, оперируя с фиктивной статьей, которая содержит исключительно единственное значение атрибута, являющееся защищенным объектом.

Следующие пункты создают ограничения, которые могут блокировать предоставление некоторых разрешений в отношении защищенных объектов в той же последовательности SEQUENCE:

- **maxValueCount** ограничивает максимальное количество значений атрибута, разрешенное для конкретного типа атрибута. Этот пункт проверяется, если защищенным объектом является значение атрибута указанного типа, а разрешение запрашивается для операции добавления. Значения этого атрибута в статье подсчитываются независимо от контекста или управления доступом и как если бы операция, добавляющая значения, завершилась успешно. Если количество значений в атрибуте превышает **maxCount**, пункт ACI интерпретируется как не предоставляющий доступа к операции добавления.
- **maxImmSub** ограничивает максимальное количество непосредственно подчиненных старшей статьи относительно статьи, которая добавляется или импортируется. Этот пункт проверяется, если защищенным объектом является статья, разрешение запрашивается для операции добавления или импортирования, и непосредственно старшая статья находится в том же DSA, что и добавляемая или импортируемая статья. Непосредственно подчиненные старшей статьи подсчитываются независимо от контекста или управления доступом и как если бы операция добавления или импортирования статьи завершилась успешно. Если количество подчиненных превышает **maxImmSub**, пункт ACI интерпретируется как не предоставляющий доступа к операции добавления или импортирования.
- **restrictedBy** ограничивает значения, добавляемые к типу атрибута, значениями, которые уже присутствуют в той же статье как значения атрибута **valuesIn**. Этот пункт проверяется, если защищенным объектом является значение атрибута определенного типа, а разрешение запрашивается для операции добавления. Значения атрибута **valuesIn** проверяются независимо от контекста или управления доступом и как если бы операция, добавляющая значения, завершилась успешно. Если добавляемое значение не присутствует в **valuesIn**, пункт ACI интерпретируется как не предоставляющий доступа к операции добавления.
- **contexts** ограничивает значения, добавляемые к статье, значениями, имеющими перечни контекстов, которые соответствуют всем содержащимся в **contexts** утверждениям о контексте. Этот пункт проверяется, если защищенным объектом является значение атрибута, а разрешение запрашивается для операции добавления. Если добавляемое значение не соответствует

утверждениям о контексте, пункт ACI интерпретируется как не предоставляющий доступа к операции добавления; если оно соответствует всем утверждениям, пункт ACI интерпретируется как не отказывающий в доступе к операции добавления.

ПРИМЕЧАНИЕ 3. – Это актуально, только если разрешение запрашивается для операции добавления, и все утверждения о контексте должны быть выполнены. Оно не предназначено для общего случая использования контекстов в целях проведения различий между защищенными объектами для прочих разрешений.

– **classes** означает контент статьей (возможно, члена семейства), который ограничены контентом, имеющим значения класса объектов, которые соответствуют предикату, определенному пунктом **Refinement** (см. п. 12.3.5), вместе (в случае порождающего или иного члена семейства) с контентом статьи в целом каждой статьи подчиненного члена семейства; этот пункт необязательно включает информацию в эти статьи.

ПРИМЕЧАНИЕ 4. – С помощью правил для статей **entry** и классов **classes** все члены семейства наследуют управление доступом порождающего или старшего члена семейства в пределах того же семейства. Это не исключает возможности того, что члены семейства станут субъектами последующих стратегий, определяемых **entryACI** или **prescriptiveACI**, которые повысят или понизят уровень защиты.

b) **UserClasses** определяет совокупность нуля или более пользователей, к которым применяются данные разрешения. Совокупность пользователей составляется из следующих пунктов:

– **allUsers** означает всех пользователей справочника (с возможными требованиями в отношении **authenticationLevel**).

– **thisEntry** означает пользователя с тем же выделенным именем, что и статья, к которой осуществляется доступ, или, если статья является членом семейства, то в этом случае дополнительно пользователя с выделенным именем порождающего элемента.

– **name** – это имя пользователя с определенным выделенным именем (с необязательным уникальным идентификатором).

– **userGroup** – это совокупность пользователей, являющихся членами статьи **groupOfUniqueNames**, указанной конкретным выделенным именем (с необязательным уникальным идентификатором). Члены группы уникальных имен интерпретируются как индивидуальные имена объектов, а не как имена других групп уникальных имен. Порядок определения членства в группе описывается в п. 18.4.2.5.

– **subtree** – это совокупность пользователей, чьи выделенные имена подпадают под определение поддерева (не имеющего уточнения).

Имена, используемые для указания пользователя, группы или поддерева, должны быть первичными выделенными именами. Не подлежат включению контекст и альтернативные выделенные значения. Для определения первичного выделенного имени для альтернативных имен, с которыми оно поступает, функция принятия решения по управлению доступом не требуется.

ПРИМЕЧАНИЕ 5. – Это означает, что если запросчик передает альтернативное имя, которое далее не преобразуется Справочником в первичное выделенное имя, управление доступом, основанное на первичных выделенных именах, может не опознать запросчика как принадлежащего к классу пользователей, которому предоставлен доступ или которому отказано в доступе.

c) **SubtreeSpecification** используется для описания поддерева относительно статьи корня, имя которой содержится в **base**. **base** представляет выделенное имя корня поддерева. Поддерево распространяется до листьев DIT, если иное не указано в **chop**. Использование компонента **specificationFilter** не разрешается; в случае присутствия компонент должен игнорироваться.

ПРИМЕЧАНИЕ 6. – **SubtreeSpecification** не разрешает уточнение поддерева, поскольку уточнение может потребовать от DSA использования распределенной операции для определения принадлежности данного пользователя к конкретному классу пользователей. Базовое управление доступом построено так, чтобы избегать дистанционной работы в процессе принятия решения по управлению доступом. Члены в поддереве, в определение которого включены только **base** и **chop**. могут оцениваться локально, тогда как члены в поддереве, в определении которого используется **specificationFilter**, могут оцениваться только путем получения информации от пользовательской статьи, которая потенциально находится в другом DSA.

d) **ItemPermission** содержит совокупность пользователей и их разрешений относительно **ProtectedItems** в пределах спецификации **itemFirst**. Разрешения описываются в **grantsAndDenials**, как это указано в п. f) настоящего пункта. Все описанные в **grantsAndDenials** разрешения рассматриваются как имеющие уровень приоритета, определенный в **precedence** для целей оценки информации по управлению доступом, как указано в п. 18.8. Если в **ItemPermission** пункт **precedence** опущен**,** значение приоритета берется из **precedence**, определенного для **ACIItem** (см. п. 18.4.2.2).

e) **UserPermission** содержит набор защищенных объектов и связанных разрешений относительно **userClasses** в пределах спецификации **userFirst**. Защищенные объекты определяются в **protectedItems**, как указано в п. 18.4.2. Связанные разрешения определяются в **grantsAndDenials**, как указано в п. f) настоящего пункта. Все описанные в **grantsAndDenials** разрешения рассматриваются как имеющие уровень приоритета, определенный в **precedence** для целей оценки информации по управлению доступом, как указано в п. 18.8. Если в **UserPermission** пункт **precedence** опущен**,** значение приоритета берется из **precedence**, определенного для **ACIItem** (см. п. 18.4.2.2).

f) **GrantsAndDenials** определяет права доступа, которые предоставлены или в которых отказано в спецификации **ACIItem**. Точная семантика этих разрешений относительно всех защищенных объектов содержится в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

g) **UniqueIdentifier** может использоваться механизмом аутентификации для проведения различия между примерами повторного использования выделенного имени. Значение уникального идентификатора присваивается органом аутентификации согласно его стратегии и предоставляется DSA аутентификации. Если это поле присутствует, то для обращающегося пользователя для совпадения класса пользователей **name** элемента **ACIItem**, который предоставляет разрешения, в дополнение к требованию совпадения выделенного имени пользователя с указанным выделенным именем, аутентификация пользователя должна выдать связанный уникальный идентификатор, и это значение должно успешно пройти сопоставление на равенство с указанным значением.

ПРИМЕЧАНИЕ 7. – Если аутентификация основана на предоставлении **SecurityParameters**, уникальный идентификатор, связанный с пользователем, может быть взят из поля **subjectUniqueIdentifier** сертификата **Certificate** отправителя в необязательном **CertificationPath**.

18.4.2.5 Определение членства в группе

Определение того, является ли данный запросчик членом группы, состоит в проверке двух критериев. Определение может быть также ограничено, если определение группы локально неизвестно. Критерии членства и интерпретация нелокальных групп рассматриваются ниже.

a) От DSA не требуется проведения дистанционной операции для определения того, принадлежит ли запросчик конкретной группе, для целей базового управления доступом. Если членство в группе не может быть оценено, DSA должен предположить, что запросчик не принадлежит к этой группе, если пункт ACI предоставляет запрошенное разрешение, и что он принадлежит к этой группе, если этот пункт отказывает ему в запрошенном разрешении.

ПРИМЕЧАНИЕ 1. – Администраторы управления доступом должны быть осведомлены о том, что функции управления доступом основываются на членстве в группах, недоступных локально, или в группах, которые доступны только через репликацию (и которые могут, таким образом, быть необновленными).

ПРИМЕЧАНИЕ 2. – С функциональной точки зрения, как правило, нецелесообразно осуществлять выборку данных о членстве в группе из удаленных DSA в качестве части оценки функций управления доступом. Однако при некоторых обстоятельствах это может оказаться целесообразным, и DSA разрешается, например, перед применением настоящего пункта дистанционно выполнять операции для получения или обновления локальной копии статьи группы или использовать операцию сравнения для проверки членства.

b) Для того чтобы определить, является ли запросчик членом класса пользователей **userGroup**, применяются следующие критерии:

– Статья, название которой указано в спецификации **userGroup**, должна быть примером **groupOfNames** класса объектов или **groupOfUniqueNames**.

– Имя запросчика должно быть значением атрибута **member** или **uniqueMember** этой статьи.

ПРИМЕЧАНИЕ 3. – Значения атрибутов **member** или **uniqueMember**, которые не совпадают с именем запросчика, игнорируются, даже если они представляют имена групп, членом которых может быть признан инициатор. Следовательно, при оценке функций управления доступом вложенные группы не поддерживаются.

ПРИМЕЧАНИЕ 4. – Имена, используемые в **member** или **uniqueMember**, должны быть первичными выделенными именами. Не подлежат включению контекст и альтернативные значения с контекстом.

18.5 Операционные атрибуты ACI

Информация по управлению доступом хранится в Справочнике как операционный атрибут статей и подстатей. Операционный атрибут содержит множество значений, которые позволяют представлять ACI как совокупность **ACIItems** (определение см. в п. 18.4).

18.5.1 Предписывающая информация по управлению доступом

Атрибут предписывающей ACI определяется как операционный атрибут подстатьи. Он содержит информацию по управлению доступом, применимую к статьям в пределах области действия этой подстатьи:

```
prescriptiveACI  ATTRIBUTE  ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-prescriptiveACI }
```

18.5.2 Информация по управлению доступом статьи

Атрибут ACI статьи определяется как операционные атрибуты статьи. Он содержит информацию по управлению доступом, применимую к статье, в которой находится, и контент этой статьи:

```
entryACI  ATTRIBUTE  ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
```

```
        USAGE                           directoryOperation
        ID                              id-aca-entryACI }
```

18.5.3 ACI подстатьи

Атрибуты ACI подстатьи определяются как операционные атрибуты административных статей и предоставляют информацию по управлению доступом, которая применяется ко всем подстатьям соответствующей административной точки. Предписывающая ACI в пределах подстатей конкретной административной точки никогда не применяется к той же или любой иной подстатье этой административной точки, но может быть применима к подстатьям подчиненных административных точек. Атрибуты ACI подстатьи содержатся только в административных точках и не затрагивают каких бы то ни было элементов DIT, не являющихся непосредственно подчиненными подстатьями.

При оценке управления доступом для определенной подстатьи ACI, которая должна рассматриваться, является:

- **entryACI** в пределах самой подстатьи (если таковая имеется);
- **subentryACI** в пределах связанной административной статьи (если таковая имеется);
- **prescriptiveACI**, связанна с другими соответствующими административными точками в пределах той же специальной области управления доступом (если таковая имеется).

```
subentryACI  ATTRIBUTE  ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-subentryACI }
```

18.6 Защита ACI

Операционные атрибуты ACI могут подвергаться действию тех же механизмов защиты, что и обычные атрибуты. Следует заметить следующие важные обстоятельства:

a) **identificationTag** обеспечивает идентификатор для каждого **ACIItem**. Эта бирка может использоваться для удаления указанного значения **ACIItem** или для защиты его с помощью предписывающей ACI или ACI статьи.

   ПРИМЕЧАНИЕ 1. – Правила Справочника обеспечивают, что только один **ACIItem** на атрибут управления доступом обладает каким-либо определенным значением **identificationTag**.

b) Управление доступом при создании подстатей для административной статьи может осуществляться посредством операционного атрибута **subentryACI** в административной статье.

   ПРИМЕЧАНИЕ 2. – Право создавать предписывающие функции управления доступом может также регулироваться непосредственно стратегией обеспечения безопасности; это положение необходимо для создания функций управления доступом в новых автономных административных областях.

18.7 Управление доступом и операции Справочника

Любая операция Справочника предусматривает принятие ряда *решений по управлению доступом* относительно различных защищенных объектов, к которым эта операция осуществляет доступ.

Для выполнения некоторых операций (например, операции изменения) каждое такое решение по управлению доступом должно предоставлять доступ; если в доступе к какому-либо защищенному объекту отказано, вся операция заканчивается неудачей. Для других операций защищенный объект, в доступе к которому отказано, просто не включается в результат операции и обработка продолжается.

Если в запрошенном доступе отказано, могут потребоваться дальнейшие решения по управлению доступом, для того чтобы определить, имеет ли пользователь разрешения **DiscloseOnError** в отношении этого защищенного объекта. Только если разрешение **DiscloseOnError** предоставлено, Справочник может ответить сообщением об ошибке, которое превалирует над наличием защищенного объекта; во всех иных случаях Справочник действует так, чтобы скрыть существование защищенного объекта.

Требования управления доступом для каждой операции, т. е. защищенные пункты и разрешение на доступ, необходимое для доступа к каждому защищенному пункту, описываются в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

Алгоритм принятия конкретного решения по управлению доступом описывается в п. 18.8.

18.8 Функция принятия решения по управлению доступом

В данном подпункте поясняется порядок принятия решения по управлению доступом для любого конкретного защищенного объекта. Содержится концептуальное описание функции принятия решения по управлению доступом (ACDF) для базового управления доступом **basic-access-control**. Описывается обработка пунктов ACI для

принятия решения о предоставлении или об отказе в предоставлении отдельному запросчику определенного разрешения в отношении данного защищенного объекта.

18.8.1 Вход и выход

Для каждого запроса ACDF входными данными являются:

a) выделенное имя запросчика (см. п. 7.3 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3), уникальный идентификатор и уровень аутентификации или те из перечисленных, которые имеются в наличии;

b) защищенный объект (статья, атрибут или значение атрибута), рассматриваемый в текущей точке принятия решения, для которого запрошена ACDF;

c) запрошенная категория разрешения, определенная для текущей точки принятия решения;

d) пункты ACI, связанные со статьей, содержащей защищенный объект (или которая является им). Защищенные объекты описываются в п. 18.4.2.4. Область воздействия для пунктов ACI в пределах атрибута **prescriptiveACI** описывается в пп. 18.3.2 и 18.5.1. Область воздействия для пунктов ACI в пределах атрибута **entryACI** описывается в пп. 18.3.2 и 18.5.2. Область воздействия для пунктов ACI в пределах атрибута **subentryACI** описывается в п. 18.5.3.

   Если статья является членом семейства, она наследует также управление доступом порождающего элемента или старших членов семейства в пределах того же семейства. Это не исключает возможности того, что члены семейства станут субъектами последующих стратегий, определяемых **entryACI** или **prescriptiveACI**, которые повысят или понизят уровень защиты.

Кроме того, если пункты ACI включают любое из описанных в п. 18.4.2.4 ограничений в отношении защищенного объекта, в качестве входных данных могут потребоваться также вся статья и ряд подчиненных ее старшей статьи.

Выходом является решение о *предоставлении доступа* или об *отказе в доступе* к защищенному объекту.

В любом конкретном случае принятия решения по управлению доступом выход должен быть таким же, как если бы были выполнены шаги с 18.8.2 по 18.8.4.

18.8.2 Кортежи

Для каждого значения ACI в пунктах ACI п. 18.8.1 d) развернуть это значение в совокупность *кортежей*: один кортеж для каждого элемента совокупностей **itemPermissions** и **userPermissions**. Собрать все кортежи из всех значений ACIв единую совокупность. Каждый кортеж содержит следующие пункты:

(userClasses, authenticationLevel, protectedItems, grantsAndDenials, precedence)

Для всех кортежей, **grantsAndDenials** которых определяют и разрешения, и отказы, заменить кортеж двумя кортежами – один определяет только разрешения, другой – только отказы.

18.8.3 Отбрасывание неактуальных кортежей

Для отбрасывания неактуальных кортежей выполнить следующие шаги:

1) Отбросить все кортежи, которые не содержат запросчика в **userClass** кортежа (п. 18.4.2.4 b)), следующим образом:

   – Для кортежей, которые предоставляют доступ: отбросить все кортежи, которые не содержат идентификационную информацию запросчика в элементе **userClasses** кортежа, учитывая в соответствующем случае элементы **uniqueIdentifier**. Если кортеж определяет **uniqueIdentifier**, для того чтобы кортеж не был отброшен, значение сопоставления должно содержаться в идентификационной информации запросчика. Отбросить кортежи, определяющие более высокий уровень аутентификации по сравнению со связанным с запросчиком согласно п. 18.4.2.3.

   – Для кортежей, которые отказывают в доступе: удерживать все кортежи, содержащие запросчика в элементе **userClasses** кортежа, учитывая в соответствующем случае элементы **uniqueIdentifier**. Также удерживать все кортежи, которые отказывают в доступе и которые определяют более высокий уровень аутентификации по сравнению со связанным с запросчиком согласно п. 18.4.2.3. Все прочие кортежи, отказывающие в доступе, отбрасываются.

     ПРИМЕЧАНИЕ 1. – Второе требование во втором подпункте выше (т. е. удерживать любой кортеж, который отказывает в доступе и при этом определяет более высокий уровень аутентификации по сравнению со связанным с запросчиком) отражает факт непредставления запросчиком адекватных доказательств того, что он не является членом класса, которому отказано в доступе.

2) Отбросить все кортежи, не содержащие защищенный объект в **protectedItems** (п. 18.4.2.4 a)).

3) Проанализировать все кортежи, содержащие **maxValueCount**, **maxImmSub**, **restrictedBy** или **contexts**. Отбросить все кортежи, которые предоставляют доступ и которые не соответствуют любому из этих ограничений (п. 18.4.2.4 a)).

4) отбросить все кортежи, которые не содержат запрошенного разрешения в качестве одного из битов совокупности в **grantsAndDenials** (пп. 18.4.1, 18.4.2.4 f)).

ПРИМЕЧАНИЕ 2. – Порядок, в котором выполняется отбрасывание неактуальных кортежей, не изменяет результат ACDF.

18.8.4 Отбор кортежей, имеющих высшие уровни приоритета и специфичности

Выполнить следующие шаги для отбора кортежей, имеющих высшие уровни приоритета и специфичности:

1) Отбросить все кортежи, имеющие **precedence**, меньший по сравнению с наивысшим остающимся.

2) Если остается более одного кортежа, выбрать кортежи, имеющие *класс пользователей с более высоким уровнем специфичности.* Если имеются какие-либо кортежи с совпадением запросчика с **name** или **thisEntry** элемента **UserClasses**, отбросить все прочие кортежи. В противном случае, если имеются какие-либо кортежи с совпадением с **UserGroup**, отбросить все прочие кортежи. В противном случае, если имеются кортежи с совпадением **subtree**, отбросить все прочие кортежи.

3) Если остается более одного кортежа, выбрать кортежи с *защищенным объектом с более высоким уровнем специфичности*. Если защищенным объектом является атрибут и имеются кортежи, определяющие этот тип атрибута явным образом, отбросить все прочие кортежи. Если защищенным объектом является значение атрибута и имеются кортежи, определяющие это значение атрибута явным образом, отбросить все прочие кортежи. Защищенный объект, являющийся **rangeOfValues**, должен интерпретироваться как определяющий значение атрибута явным образом.

Предоставить доступ, если и только если остается один или более кортежей и все они разрешают доступ. В любом ином случае отказать в доступе.

18.9 Упрощенное управление доступом

18.9.1 Введение

В этом пункте описываются функциональные возможности схемы управления доступом, называемой упрощенным управлением доступом, которая построена для предоставления подмножества функциональных возможностей базового управления доступом.

18.9.2 Определение функциональных свойств упрощенного управления доступом

Функциональные свойства упрощенного управления доступом определяются следующим образом:

a) Решения по управлению доступом должны приниматься только на основании значений **ACIItem** операционных атрибутов **prescriptiveACI** и **subentryACI**.

ПРИМЕЧАНИЕ 1. – Информация **entryACI**, если присутствует, не должна использоваться для принятия решения по управлению доступом.

b) Специальные административные области управления доступом должны поддерживаться. Внутренние административные области управления доступом не должны использоваться. Конкретные решения по управлению доступом должны приниматься на основании значений **ACIItem**, полученных от единой административной точки или от подстатей этой административной точки.

ПРИМЕЧАНИЕ 2. – Значения атрибутов **prescriptiveACI**, входящие в подстатьи административных точек, которые не содержат значения **id-ar-accessControlSpecificArea** атрибута административной функции, не должны использоваться для принятия решения по управлению доступом.

c) Все прочие положения должны быть теми же, что и для базового управления доступом.

19 Управление доступом на основе правил

19.1 Область действия и применение

В данном пункте описывается конкретная схема управления доступом (одна из многих возможных) для Справочника. Определенная в настоящем документе схема управления доступом указывается операционным атрибутом **accessControlScheme** путем присвоения ему значения **rule-based-access-control** или **rule-and-basic-access-control** или **rule-and-simple-access-control**, если она используется в с комбинации со схемами базового или упрощенного управления доступом, описанными в п. 18. В п. 17.2.2 описано, какие статьи содержат этот операционный атрибут **accessControlScheme**.

Определенная ниже схема связана только с управлением доступом к информации Справочника в пределах DIB (возможно, включая информацию о структуре дерева и управлении доступом). Схема не касается управления доступом для целей связи с объектом прикладного уровня DSA. Управление доступом к информации означает предотвращение несанкционированного обнаружения, раскрытия или изменения этой информации.

19.2 Модель управления доступом на основе правил

Возможны условия, когда для того чтобы определить, отказать ли в доступе, используется информация, связанная с допуском (вместо идентификационной информации) запросчика. Это называется управлением доступом на основе

правил и для него используются административно введенные правила стратегии управления доступом, служащие для того чтобы определять, когда следует отказывать в доступе к некоторым элементам контента Справочника. Если управлением доступом на основе правил в доступе отказано, доступ не может быть разрешен другими схемами управления доступом. Модель управления доступом на основе правил указывает информацию, используемую при определении того, должен ли быть выдан отказ в доступе. Это применяется к каждой операции. Все решения по управлению доступом включают:

a) Информацию по управлению доступом, связанную со значениями атрибутов, к которым осуществляется доступ. Эта информация по управлению доступом называется меткой безопасности.

b) Информацию управления доступом, связанную с запрашивающим операцию пользователем. Эта информация по управлению доступом называется допуском. Пользователь, запрашивающий информацию, называется запросчиком.

c) Правила, которые исходя из метки безопасности и допуска определяют, разрешается ли доступ, называемые стратегиями обеспечения безопасности.

См. рисунок 18.

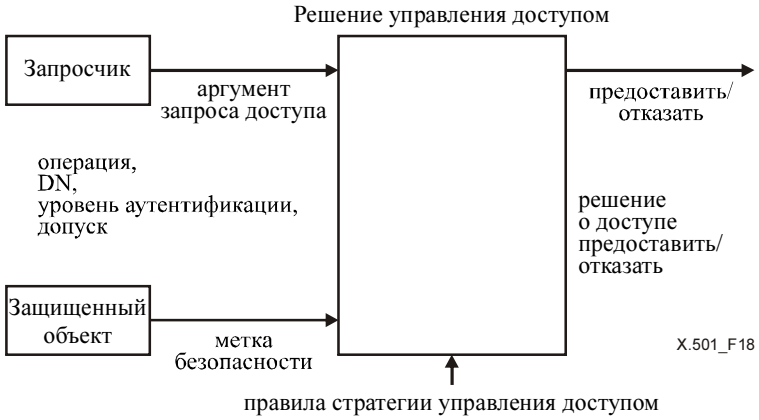


Рисунок 18 – Модель принятия решения по управлению доступом на основе правил

Метка(и) безопасности может быть надежно связана со значениями атрибутов путем привязки метки к информации с помощью цифровой подписи или иного механизма контроля целостности данных. Метка безопасности является свойством значения атрибута и связывается со значением как контекст.

Допуск требуется для получения возможности проведения сравнения с меткой безопасности. Допуск может быть привязан к выделенному имени запросчика через поле расширения сертификата (в зависимости от атрибута Справочника) или через сертификат атрибута. Средства предоставления допуска являются вопросом принятой стратегии обеспечения безопасности.

ПРИМЕЧАНИЕ. – Использование прочей информации допуска (например, информации, связанной с любыми промежуточными DSA, которые могут сцеплять операцию) не входит в область определения настоящей спецификации Справочника.

Правила безопасности, которые должны применяться при принятии решения по управлению доступом, определяются как часть стратегии обеспечения безопасности. Стратегия обеспечения безопасности либо указывается в метке безопасности, либо определяется для среды, содержащей помеченный объект.

19.3 Административные области управления доступом

Для базового управления доступом (см. п. 18.3) DIT разделяется на административные области, включая специальные области управления доступом. Административная статья для ACSA указывает имеющие метку стратегии обеспечения безопасности, применимые для этой административной области, а также применимую схему управления доступом (**rule-based-access-control** или **rule-and-basic-access-control** или **rule-and-simple-access-control** или какую-либо иную схему управления доступом).

19.4 Метка безопасности

19.4.1 Введение

Метки безопасности могут использоваться для связи относящейся к безопасности информации с атрибутами в пределах Справочника.

Метки безопасности могут присваиваться значению атрибута согласно стратегии обеспечения безопасности, принятой для этого атрибута. Стратегия безопасности может также определять порядок использования меток безопасности для обеспечения выполнения этой стратегии безопасности.

Метка безопасности состоит из совокупности элементов, необязательно включающей идентификатор стратегии безопасности, классификацию безопасности, гриф конфиденциальности и совокупность категорий безопасности. Метка безопасности привязана к значению атрибута с помощью цифровой подписи или иного механизма контроля целостности.

19.4.2 Административное управление метками безопасности

Метка безопасности присваивается значению атрибута административной функцией до помещения его в Справочник.

Эта административная функция выполняет присвоение меток безопасности в соответствии с действующей стратегией обеспечения безопасности для ACSA.

Связывание метки безопасности защищается с использованием цифровой подписи или иного механизма обеспечения целостности. Эта защита применяется административной функцией или создателем значения атрибута.

19.4.3 Значения атрибутов, имеющие метку

Контекст метки безопасности связывает метку со значением атрибута. Со значением атрибута может быть связана только одна метка безопасности. Это значит, что контекст метки безопасности является содержащим одно значение. Кроме того, правила сопоставления для контекста метки безопасности не поддерживаются.

ПРИМЕЧАНИЕ. – Понятие контекстов вводится в п. 8.8.

```
attributeValueSecurityLabelContext CONTEXT ::= {
        WITH SYNTAX     SignedSecurityLabel     -- Значению атрибута может быть присвоено
                                                -- не более одного контекста метки безопасности
        ID              id-avc-attributeValueSecurityLabelContext }

SignedSecurityLabel  ::=  SIGNED {SEQUENCE  {
        attHash         HASH {AttributeTypeAndValue},
        issuer          Name            OPTIONAL, -- имя маркирующего органа
        keyIdentifier   KeyIdentifier OPTIONAL,
        securityLabel   SecurityLabel } }

SecurityLabel  ::=  SET {
        security-policy-identifier   SecurityPolicyIdentifier   OPTIONAL,
        security-classification      SecurityClassification      OPTIONAL,
        privacy-mark                 PrivacyMark                 OPTIONAL,
        security-categories          SecurityCategories          OPTIONAL }
                (ALL EXCEPT ( {-- ни одного, должен присутствовать по крайней мере один компонент -- } ) )

SecurityPolicyIdentifier  ::=  OBJECT IDENTIFIER

SecurityClassification ::= INTEGER {
        unmarked        (0),
        unclassified    (1),
        restricted      (2),
        confidential    (3),
        secret          (4),
        top-secret      (5) }

PrivacyMark  ::=  PrintableString (SIZE (1..ub-privacy-mark-length))

SecurityCategories ::= SET SIZE (1..MAX) OF SecurityCategory
```

Этот контекст не используется для фильтрации или отбора отдельных атрибутов, как это происходит в случае других контекстов, а механизмы, связанные с контекстами (исходный режим, значения контекста по умолчанию и т. д.), не используются для применения управления доступом на основе правил.

Компонент **attHash** содержит результирующее значение применения криптографической процедуры хэширования к октетам, кодированным по отличительным правилам кодирования, как это определено в Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8.

Компонент **issuer** передает имя маркирующего органа.

Компонент **keyIdentifier** может быть идентификатором сертифицированного открытого ключа как содержащийся в поле расширения идентификатора открытого ключа владельца, определенном в Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8, или идентификатором симметричного ключа и связанной информации по управлению безопасностью.

Компонент **securityLabel** состоит из совокупности элементов, необязательно включающей идентификатор стратегии безопасности, классификацию безопасности, гриф конфиденциальности и совокупность категорий безопасности, как это определено в п. 8.5.9 Рек. МСЭ-Т X.411 | ИСО/МЭК 10021-4.

19.5 Допуск

Атрибут допуска связывает допуск с именованным объектом, включая агентов DUA.

```
clearance  ATTRIBUTE  ::=  {
        WITH SYNTAX       Clearance
        ID                id-at-clearance }

Clearance  ::=  SEQUENCE  {
        policyId                  OBJECT IDENTIFIER,
        classList                 ClassList                              DEFAULT {unclassified},
        securityCategories        SET SIZE (1..MAX) OF SecurityCategory  OPTIONAL }

ClassList  ::=  BIT STRING {
        unmarked          (0),
        unclassified      (1),
        restricted        (2),
        confidential      (3),
        secret            (4),
        topSecret         (5) }

SecurityCategory  ::=  SEQUENCE {
        type          [0]     SECURITY-CATEGORY.&id ({SecurityCategoriesTable}),
        value         [1]     EXPLICIT SECURITY-CATEGORY.&Type ({SecurityCategoriesTable} {@type}) }

SECURITY-CATEGORY  ::=  TYPE-IDENTIFIER

SecurityCategoriesTable SECURITY-CATEGORY ::= { ... }
```

Компонент **policyId** передает идентификатор, который может использоваться для указания действующей стратегии безопасности, к которой относится допуск **classList** и **securityCategories**.

Компонент **classList** содержит перечень классификаций, которые связаны с этим именованным объектом.

Компонент **securityCategories** (см. п. 8.5.9 Рек. МСЭ-Т X.411 | ИСО/МЭК 10021-4), если присутствует, вводит дополнительные ограничения в пределах контекста **classList**.

ПРИМЕЧАНИЕ. – Допуск надежно привязывается к именованному объекту с помощью сертификата атрибута (Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8), поля расширения сертификата открытого ключа (например, в пределах расширения **SubjectDirectoryAttribute**) (Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8) или с помощью средств, не входящих в область определения настоящей спецификации Справочника.

19.6 Управление доступом и операции Справочника

Любая операция Справочника предусматривает принятие ряда решений по управлению доступом относительно значений атрибутов, к которым эта операция осуществляет доступ.

Для некоторых операций (например, операция удаления статьи), даже если операция может казаться успешно выполненной при отказе в доступе к одному или более значениям атрибутов, скрытые атрибуты останутся в справочнике. Для других операций защищенные объекты, в доступе к которым отказано, просто не включаются в результат операции и обработка продолжается.

Требования управления доступом для каждой операции определяются в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

Алгоритм принятия любого решения по управлению доступом описывается следующим образом:

- Если при **rule-based-access-control** в доступе ко всем значениям атрибутов какой-либо статьи отказано, доступ к этой статье не предоставляется всем операциям.
- Если при **rule-based-access-control** в доступе ко всем значениям атрибута какого-либо атрибута отказано, доступ к этому атрибуту не предоставляется всем операциям.
- Управление доступом на основе правил затрагивает операции, предусматривающие чтение значений атрибутов (например, операции чтения и поиска), таким образом, что если в доступе к значению атрибута отказано, это значение атрибута является невидимым (операция выполняется так, как если бы этот атрибут не присутствовал).

– Управление доступом на основе правил затрагивает операции, предусматривающие удаление статьи (например, операция удаления статьи), таким образом, что эти операции не удаляют те значения атрибутов, в доступе к которым отказано.

– Управление доступом на основе правил затрагивает операции, предусматривающие удаление типа атрибута (например, операции изменения статьи – удаления статьи), таким образом, что эти операции не удаляют те значения атрибутов, в доступе к которым отказано.

– Управление доступом на основе правил затрагивает операции, предусматривающие удаление значения атрибута (например, операции изменения статьи – удаления статьи), таким образом, что при отказе в доступе к значению атрибута эти операции завершаются неудачей.

19.7 Функция принятия решения по управлению доступом

В данном подпункте поясняется порядок принятия решения по управлению доступом для любого конкретного значения атрибута. Содержится концептуальное описание функции принятия решения по управлению доступом (ACDF) для управления доступом на основе правил **rule-based-access-control**. Описывается порядок обработки допуска и метки безопасности для принятия решения о том, предоставлять ли конкретному запросчику определенное разрешение в отношении данного значения атрибута. Функция принятия решения применяет правила стратегии обеспечения безопасности, которые исходя из метки безопасности и допуска запросчика устанавливают, разрешается ли доступ. Описание правил безопасности не входит в область определения настоящих спецификаций Справочника. Упрощенный пример правил стратегии обеспечения безопасности для **rule-based-access-control** приводится в п. M.10.

Для каждого запроса ACDF входными данными являются:

a) допуск запросчика (как он определен в п. 19.5);

b) значение атрибута, рассматриваемое в текущей точке принятия решения, для которого запрошена ACDF;

c) действующая стратегия обеспечения безопасности для специальной области управления доступом;

d) метка безопасности, привязанная в значению атрибута.

Выходными данными является решение о том, отказать ли в доступе к значению атрибута.

В любом конкретном случае принятия решения по управлению доступом выход должен быть таким же, как если бы были выполнены шаги в п. 19.6.

19.8 Использование управления доступа на основе правил и базового управления доступом

Если действуют и управление доступом на основе правил, и базовое управление доступом, порядок, в котором они применяются, является вопросом местной компетенции, за исключением следующего: если каким-либо из этих механизмов отказано в доступе к статье, типу атрибута или значению атрибута, доступ не должен предоставляться другим механизмом. В этом отношении разрешение *DiscloseOnError* (см. пп. 18.2.3 и 18.2.4) схемы **basic-access-control** является разрешением, которое не может отменить отказ в этом разрешении механизма **rule-based-access-control**.

20 Целостность данных при хранении

20.1 Введение

В некоторых ситуациях Справочник не может достаточно уверенно гарантировать неизменность данных при хранении независимо от управления доступом. Целостность хранящихся в Справочнике данных может подтверждаться с помощью цифровых подписей, хранящихся как часть информации Справочника. Цифровая подпись статьи или отобранных в пределах статьи атрибутов может содержаться как атрибут (см. п. 20.1.2), либо цифровая подпись единичного значения атрибута может содержаться как контекст (см. п. 20.1.3).

ПРИМЕЧАНИЕ 1. – DSA должен сопровождать кодирование атрибута, помещенного в Справочник, с тем чтобы обеспечивать верность подписи, рассчитанной в возвращенном результате.

ПРИМЕЧАНИЕ 2. – Конфиденциальность значений атрибута не входит в область определения данной спецификации.

20.2 Защита статьи или отобранных типов атрибута

Целостность данных атрибутов при хранении обеспечивается путем использования цифровых подписей, содержащихся рядом с атрибутами, которые они защищают. Целостность всей статьи или всех значений атрибута для отобранных атрибутов в статье защищается атрибутом, содержащим цифровую подпись всех защищаемых значений атрибута.

Эта цифровая подпись создается органом или пользователем справочника, ответственным за помещение информации в статью справочника. Цифровая подпись может проверяться любым пользователем, считывающим значения атрибута для статьи. Сама служба справочника не участвует в создании и проверки цифровой подписи, хранящейся в атрибуте.

Этот механизм контроля целостности защищает целостность атрибутов справочника как при хранении, так и в процессе передачи между компонентами Справочника (DSA и DUA). Данный механизм контроля целостности не зависит от безопасности самой службы справочника.

Цифровые подписи, применяемые к статье в целом, не включают операционных, коллективных атрибутов или собственно информации, касающейся целостности атрибутов **attributeIntegrityInfo**. Все контексты значений атрибута включаются.

Следующий ниже модуль определяет тип атрибута для содержания цифровой подписи вместе со связанной информацией управления, который обеспечивает целостность статьи целиком или всех значений отобранных типов атрибута.

```
attributeIntegrityInfo ATTRIBUTE  ::= {
        WITH SYNTAX             AttributeIntegrityInfo
        ID                      id-at-attributeIntegrityInfo}

AttributeIntegrityInfo  ::=  SIGNED { SEQUENCE  {

        scope           Scope,                  -- Указывает защищаемые атрибуты

        signer          SignerOPTIONAL,         -- Наименование органа или источников данных

        attribsHash     AttribsHash } }         -- Хэш-значение защищаемых атрибутов

Signer  ::=  CHOICE {
        thisEntry       [0]     EXPLICIT ThisEntry,
        thirdParty      [1]     SpecificallyIdentified }

ThisEntry  ::=  CHOICE  {
        onlyOne         NULL,
        specific        IssuerAndSerialNumber }

IssuerAndSerialNumber  ::=  SEQUENCE {
        issuer          Name,
        serial          CertificateSerialNumber }

SpecificallyIdentified  ::=  SEQUENCE  {
        name            GeneralName,
        issuer          GeneralName                     OPTIONAL,
        serial          CertificateSerialNumber         OPTIONAL }
        ( WITH COMPONENTS { ..., issuer PRESENT, serial PRESENT } |
        ( WITH COMPONENTS { ..., issuer ABSENT, serial ABSENT } ) )

Scope  ::=  CHOICE  {
        wholeEntry      [0]     NULL, -- Подпись защищает все значения атрибута в этой статье
        selectedTypes   [1]     SelectedTypes
                                -- Подпись защищает все значения отобранных типов атрибута
        }

SelectedTypes  ::=  SEQUENCE SIZE (1..MAX) OF AttributeType

AttribsHash  ::=  HASH { SEQUENCE SIZE (1..MAX) OF Attribute }
                  -- Тип и значения атрибута со связанными значениями контекста
                  -- для выбранной области действия
```

Значение **AttributeIntegrityInfo** может быть создано тремя разными способами:

a) Административный орган может создавать и подписывать значение, а открытый ключ для проверки подписи опознается автономными средствами.

b) Владелец статьи, т. е. объект, представленный статьей, может создавать и подписывать значение. Если владелец имеет несколько сертификатов или предполагает, что будет иметь их в будущем, сертификат должен опознаваться по выдающему сертификат органу сертификации вместе с серийным номером сертификата.

c) Третья сторона может создавать и подписывать значение. Потребуются имя подписавшего, наименование выдающего сертификат органа сертификации и серийный номер сертификата.

Если областью применения является статья целиком **wholeEntry**, все применимые атрибуты должны быть упорядочены, как это определяется для совокупности типа в п. 6.1 Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8. Если областью применения являются отобранные типы **selectedTypes**, порядок их следования должен быть тем же, что и указанный в **SelectedTypes**.

ПРИМЕЧАНИЕ. – Если пользователь не осуществляет выборку всего набора атрибутов, который определен в типе данных **Scope**, у него не будет возможности проверить целостность этих атрибутов.

20.3 Контекст для защиты единичного значения атрибута

Следующий ниже модуль определяет контекст для содержания цифровой подписи вместе со связанной информацией управления, который обеспечивает целостность единичного значения атрибута. В проверку целостности включаются все контексты значения атрибута, за исключением контекста, используемого для содержания подписей.

```
attributeValueIntegrityInfoContext  CONTEXT  ::=  {
        WITH SYNTAX       AttributeValueIntegrityInfo
        ID                id-avc-attributeValueIntegrityInfoContext }

AttributeValueIntegrityInfo  ::=  SIGNED { SEQUENCE {

        signer        SignerOPTIONAL,        -- Наименование органа или источников данных

        aVIHash             AVIHash } }              -- Хэш-значение защищаемого атрибута

AVIHash  ::=  HASH { AttributeTypeValueContexts }
                    -- Тип и значения атрибута со связанными значениями контекста

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

Упорядочение **contextList** должно быть таким, как определенное для совокупности типа в п. 6.1 Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8.

РАЗДЕЛ 9 – МОДЕЛИ DSA

21 Модели DSA

Данный пункт посвящен родовым моделям, описывающим разные аспекты компонентов, составляющих Справочник, системных агентов Справочника (DSA). В следующих пунктах рассматриваются дополнительные модели DSA.

21.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

21.1.1 фрагмент DIB: Часть DIB, которую содержит один главный DSA, содержащий один или более контекстов именования.

21.1.2 префикс контекста: Последовательность RDN от корня DIT до начальной вершины контекста именования; соответствует выделенному имени этой вершины.

21.1.3 контекст именования: Поддерево статей, содержащееся в одном главном DSA.

21.2 Функциональная модель Справочника

Справочник объявляется как множество, состоящее из одного или более прикладных процессов, называемых *системными агентами Справочника (DSA)* и/или *серверами LDAP*. Каждый DSA предоставляет нуль, одну или более точек доступа. Каждый сервер LDAP предоставляет одну или более точек доступа. Это показано на рисунке 19. Если Справочник состоит из более чем одного DSA или сервера LDAP, он считается *распределенным*. Процедуры для функционирования Справочника, если он является распределенным, определяются в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4.

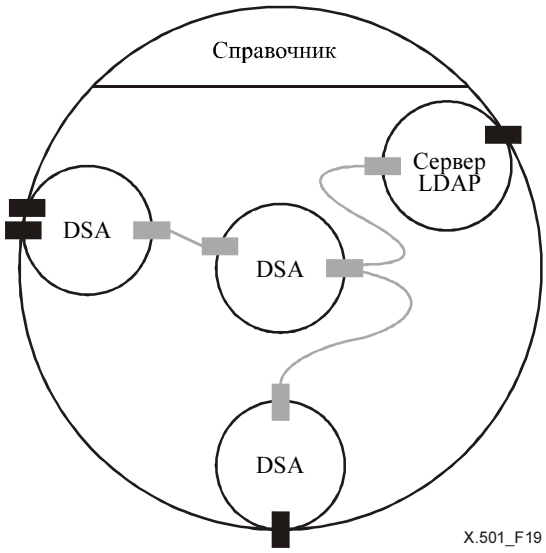


Рисунок 19 – Справочник, образованный множественными DSA

ПРИМЕЧАНИЕ 1. – DSA будет, видимо, характеризоваться локальными вариантами поведения и структурой, что не входит в область действия спецификаций Справочника. Например, DSA, ответственный за содержание части или всей информации в DIB, будет, как правило, выполнять эту функцию с помощью базы данных, интерфейс к которой является вопросом местной компетенции.

Конкретная пара прикладных процессов, которым необходимо взаимодействовать для предоставления услуг, может находиться в разных открытых системах. Такое взаимодействие осуществляется с помощью протоколов Справочника, определенных в Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5, или с помощью упрощенного протокола доступа к Справочнику (LDAP), определенного в RFC 3377 IETF.

ПРИМЕЧАНИЕ 2. – Поведение сервера LDAP определяется в RFC 3377 IETF и может отличаться от определенных в данном пункте вариантов поведения DSA.

В п. 23 описываются модели, которые используются в качестве основы для описания распределенных аспектов справочника. Структура описания операционных моделей, касающихся конкретных аспектов функционирования компонентов Справочника, агентов DSA, представлена в пп. 25–28.

21.3 Модель распределения Справочника

В данном подпункте представлены принципы, на основании которых DIB может быть распределена между множественными DSA.

ПРИМЕЧАНИЕ 1. – DIB также может быть распределена по любому количеству серверов LDAP, которые могут сосуществовать или не сосуществовать с одним или более DSA. Серверы LDAP, их характеристики и варианты поведения описываются в IETF RFC 3377 и могут отличаться от характеристик и вариантов поведения DSA, описанных в данном пункте.

Административное управление каждой статьей в пределах DIB осуществляет один и только один администратор DSA, который считается имеющим административные полномочия в отношении этой статьи. Обслуживание и управление статьей должно выполняться в DSA, который находится под административным управлением административного органа этой статьи. Этот DSA является *главным DSA* для данной статьи.

Каждый главный DSA в пределах Справочника содержит *фрагмент* DIB. Фрагмент DIB, содержащийся главным DSA, описывается в терминах DIT и содержит один или более контекстов именования. *Контекст именования* – это поддерево DIT, все статьи которого имеют общее административное полномочие и содержатся в том же главном DSA. Контекст именования начинается в вершине DIT (не являющейся корнем) и продолжается вниз до вершин, являющихся и не являющихся листьями. Такие вершины образуют границу контекста именования. Старший элемент стартовой вершины контекста именования не содержится в этом главном DSA. Подчиненные элементы не являющихся листьями вершин, которые принадлежат границе, обозначают начало следующих контекстов именования.

ПРИМЕЧАНИЕ 2. – Таким образом DIT разбивается на несовместные контексты именования, и каждый из них находится под управлением административного органа одного главного DSA.

ПРИМЕЧАНИЕ 3. – Сам контекст именования не является административной областью, имеющей административную точку или явную спецификацию поддерева, но он может совпадать с административной областью.

Семейство статей должно находиться в едином контексте именования.

Администратор главного DSA может иметь административные полномочия в отношении нескольких несовместных контекстов именования. Для каждого контекста именования, в отношении которого главный DSA имеет административные полномочия, он логически содержит последовательность имен RDN от корня DIT до начальной вершины поддерева, составляющего контекст именования. Такая последовательность имен RDN называется *префиксом контекста* контекста именования.

ПРИМЕЧАНИЕ 4. – Первичное выделенное имя контекста именования должно использоваться как префикс контекста. В имена RDN могут факультативно включаться контексты и альтернативные значения с контекстом.

Администратор главного DSA может делегировать административные полномочия в отношении любых непосредственно подчиненных элементов любой локально содержащейся статьи другому главному DSA. Главный DSA, который делегировал полномочия, называется *старшим DSA*, а контекст, содержащийся старшей статьей статьи, административные полномочия в отношении которой были делегированы, называется *старшим контекстом именования*. Делегирование административных полномочий начинается от корня и продолжается вниз по DIT; т. е. оно может только проходить от статьи до ее подчиненных элементов.

На рисунке 20 показано гипотетическое DIT, логически разделенное на контексты именования (названные A, B, C, D и E), которые физически распределены по трем DSA (DSA1, DSA2 и DSA3).

На этом примере можно заметить, что контексты именования, содержащиеся отдельными главными DSA, могут быть конфигурироваться таким образом, чтобы удовлетворять широкий диапазон функциональных требований. Некоторые главные DSA могут конфигурироваться для содержания тех статей, которые представляют домены именования высшего уровня в пределах некоторой логической части (логических частей) DIB, например организационную структуру крупной компании, но необязательно всех подчиненных статей. Альтернативным образом, главные DSA могут иметь конфигурацию для содержания только тех контекстов именования, которые представляют первичные статьи-листья.

Согласно приведенным выше определениям, граничным случаем для контекста именования может быть либо единичная статья, либо DIT целиком.

В то время как отображение логической в физическую форму DIT в главные DSA является потенциально произвольным, задача расположения информации и управления ею упрощается, если главные DSA конфигурируются для содержания небольшого количества контекстов именования.

DSA могут содержать статьи-копии так же, как и статьи. Теневые статьи, единственный вид копий статей, рассматриваемый в спецификациях Справочника, обслуживаются средствами теневой службы, которая описывается в Рек. МСЭ-Т X.525 | ИСО/МЭК 9594-9. Кроме этого стандартизированного вида дублируемой информации в Справочнике могут встретиться два дополнительных нестандартизированных вида копии статьи.

- Копии статьи, которые могут храниться в другом(их) DSA согласно двустороннему соглашению.
- Копии статьи, которые могут собираться при хранении (локально и динамически) кэш-копий статьи, являющихся результатом запроса.

ПРИМЕЧАНИЕ 5. – Средства поддержания этих копий и управления ими в настоящих спецификациях Справочника не определяются. Ввиду более точной обработки таких функций, как управление доступом, рекомендуется вместо кэш-копий использовать теневую службу.

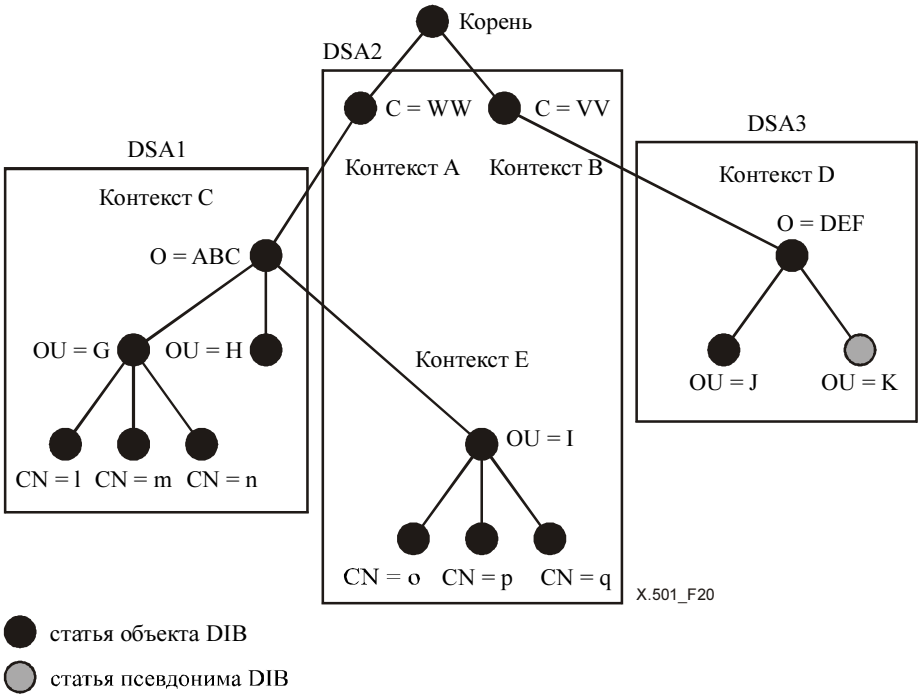


Рисунок 20 – Гипотетическое DIT

DSA, содержащий статьи-копии, является *теневым DSA* для этой статьи. Теневой DSA может содержать копию контекста именования или ее часть. Спецификация части контекста именования, являющейся теневой, называется *блоком репликации*.

Как указано в п. 9.2 Рек. МСЭ-Т X.525 | ИСО/МЭК 9594-9, блок репликации определяется в информационной модели Справочника, а также предусматривается механизм спецификации. Основу механизма теневого копирования в Справочнике составляет определение подмножества DIT, которое будет теневым. Это подмножество называется *блоком репликации.* Блок репликации состоит из спецификации в трех частях, которая определяет область применения части DIT, которая становится теневой, атрибуты, подлежащие репликации в пределах этой области применения, и требования для подчиненного знания. Блок репликации вызывает также неявным образом включение в теневую информацию информации о стратегии в форме операционных атрибутов, содержащихся в статьях и подстатьях (например, информация по управлению доступом), которая должна использоваться для корректного выполнения операций Справочника. Подлежащая включению информация префикса начинается в автономной административной точке и продолжается до базовой статьи репликации.

Инициатору запроса к Справочнику сообщается (через **fromEntry**), является ли информация, возвращаемая в ответ на запрос, информацией статьи-копии. Определяется функция управления службой **dontUseCopy** (не использовать копию), которая дает пользователю возможность запрещать использование статей-копий для выполнения запроса (хотя информация копии может использоваться в разрешении имени).

ПРИМЕЧАНИЕ 6. – Такое разрешение имени в некоторых ситуациях будет заканчиваться неудачей в случае действительного альтернативного имени при разрешении по копии, содержащейся в DSA, созданному до третьего издания, или в DSA, относящемся к более позднему изданию, который содержит копию с неполной информацией имени, когда RDN включает тип атрибута, для которого существуют множественные выделенные значения, отличающиеся контекстом.

Для того чтобы DUA мог начать обработку запроса, он должен содержать некоторую информацию, в частности адрес представления, т. е. по крайней мере адрес одного DSA, с которым он может первоначально установить контакт. Порядок сбора и содержания этой информации является вопросом местной компетенции.

В ходе изменения статей Справочник может оказаться несогласованным. Это будет особенно вероятным, если в процессе изменения будут задействованы псевдонимы или объекты псевдонимов, которые могут находиться в разных DSA. Такое несогласование должно быть устранено специальным административным решением, например удалить псевдонимы, если были удалены соответствующие объекты псевдонимов. В период такой несогласованности Справочник продолжает функционировать.

РАЗДЕЛ 10 – ИНФОРМАЦИОННАЯ МОДЕЛЬ DSA

22 Знания

22.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

22.1.1 категория: Характеристика ссылки на знания, которая определяет ее как указывающую на главный или теневой DSA.

22.1.2 коллективно используемый: Характеристика дублируемой области, которая допускает общее распределение точки доступа содержащего ее DSA; коллективно используемая дублируемая область, как правило, представляет собой полную теневую копию контекста именования.

22.1.3 перекрестная ссылка: Ссылка на знания, содержащая информацию о DSA, который содержит статью или статью-копию. Используется для оптимизации. Для этой статьи не требуется иметь взаимоотношения типа старший или подчиненный с какой-либо статьей в DSA, содержащем перекрестную ссылку.

22.1.4 ссылка на знания – мост между DIT: Ссылка на знания, содержащая информацию о DSA, который содержит статьи в другом DIT. Для этой статьи не требуется иметь взаимоотношения типа старший или подчиненный с какой-либо статьей в DSA, содержащем другое DIT.

22.1.5 непосредственно старшая ссылка: Ссылка на знания, содержащая информацию о DSA, который содержит контекст именования (или являющуюся его производной повсеместно используемую дублируемую область), являющийся непосредственно старшим по отношению к контексту, содержащемуся агентом DSA, для которого актуальна эта ссылка на знания.

22.1.6 знания (информация): Операционная информация DSA, содержащаяся DSA, который использует ее для определения местоположения отдаленной статьи или информации статьи-копии.

22.1.7 ссылка на знания: Знания, которые связывают, прямо или косвенно, статью DIT или статью-копию с DSA, в котором она находится.

22.1.8 главные знания: Знания о главном DSA для контекста именования.

22.1.9 неспецифическая подчиненная ссылка: Ссылка на знания, содержащая информацию о DSA, который содержит одну или более неспецифицированных статей или статей-копий.

22.1.10 траектория ссылки: Непрерывная последовательность ссылок на знания.

22.1.11 теневые знания: Знания об одном или более DSA для контекста именования (если знания специфические) или контекстов (если неспецифические).

22.1.12 подчиненная ссылка: Ссылка на знания, содержащая информацию о DSA, который содержит специфическую подчиненную статью или статью-копию.

22.1.13 старшая ссылка: Ссылка на знания, содержащая информацию о DSA, который считается способным выполнять разрешение (т. е. находить любую статью в пределах) всего DIT.

22.2 Введение

DIB распределяется по большому количеству главных DSA, и каждый из них содержит фрагмент DIB и имеет административные полномочия в отношении этого фрагмента. Принципы, управляющие таким распределением, изложены в п. 21.3.

Кроме того, эти и другие DSA могут содержать копии частей DIB.

Одним из требований Справочника является прозрачность – для определенных режимов взаимодействия с пользователем – распределения справочника, что создает эффект наличия всей DIB целиком в пределах любого отдельного DSA.

Для поддержки этого функционального требования необходимо, чтобы каждый DSA мог получать доступ к информации, которая содержится в DIB, связанной с любым именем (т. е. с любым выделенным именем объекта или псевдонимом объекта). Если сам DSA не содержит статью объекта или статью-копию объекта, связанную с этим именем, он должен иметь возможность взаимодействия с DSA, который ее содержит, прямо или косвенно, путем непосредственного или косвенного взаимодействия с другими DSA.

Если пользователь Справочника указывает, что для выполнения его запроса не должна использоваться информация статьи-копии, DSA, обслуживающий этот запрос, должен иметь возможность получения доступа, прямо или косвенно, к главному DSA, содержащему информацию статьи, которая связана с именем, указанном в запросе пользователя.

В этом пункте определяются знания как та операционная информация DSA, которая необходима для выполнения этих технических задач. В последующих пунктах описывается представление знаний в контексте общей информационной модели DSA.

ПРИМЕЧАНИЕ. – Предыдущие положения отражает технические задачи Справочника. Реализация этих технических задач зависит от других вопросов (например, вопросов стратегии) наряду с согласованной конфигурацией знаний в DSA. Пункты 25–28 образуют основу для решения некоторых из этих вопросов.

Приложение O содержит иллюстрацию моделирования знаний. Основой этой иллюстрации является гипотетическое DIT, представленное на рисунке 20.

22.3 Ссылки на знания

Знания – это та содержащаяся в DSA операционная информация, которая представляет конкретное описание информации статьи и статьи-копии, содержащейся в другом DSA. Знания используются DSA в целях определения, с каким DSA следует связаться, когда полученный от DUA или другого DSA запрос не может быть выполнен с помощью локально содержащейся информации.

Знания состоят из ссылок на знания. *Ссылка на знания* связывает, прямо или косвенно, имя статьи Справочника с DSA, содержащим эту статью или копию этой статьи.

Имена, используемые в ссылках на знания как префиксы контекста, имена DSA или имена статей, должны быть первичными выделенными именами. Контекст и альтернативные значения с контекстом также могут включаться в имена RDN.

ПРИМЕЧАНИЕ. – Разрешение имени для действительного альтернативного имени может окончиться неудачей, если ссылки на знания содержатся DSA до третьего издания, которые не распознают множественные выделенные значения, отличающиеся контекстом, или в DSA, не содержащих все альтернативные выделенные имена в ссылках на знания или в статьях-копиях.

22.3.1 Категории знаний

Существуют две категории ссылок на знания: ссылка на главные знания и ссылка на теневые знания.

Главные знания – это знания о точке доступа главного DSA для контекста именования.

Теневые знания – это знания об агентах DSA, содержащих дублируемую информацию Справочника; она может распределяться теневыми поставщиками между теневыми потребителями с помощью процедур репликации, описанных в Рек. МСЭ-Т X.525 | ИСО/МЭК 9594-9. Теневые знания – это знания о точке доступа совокупности одного или более теневых DSA для дублируемой области (контекст именования или его часть).

DSA, являющийся объектом теневых знаний, должен содержать коллективно используемую дублируемую область. Одной из форм дублируемой области, которая используется коллективно, является полная теневая копия контекста именования. Неполная теневая копия контекста именования, содержащаяся DSA, может быть повсеместно используемой, если она достаточно полна для выполнения запросов, обычно направляемых пользователями к DSA. Обеспечение того, чтобы дублируемая область была повсеместно используемой, является функцией административного органа, который обусловливает распределенный характер теневых знаний DSA, содержащего неполную копию контекста именования.

Данный DSA может содержать и главные, и теневые знания, последние – включая множественные теневые DSA, относительно конкретного контекста именования. Определенные знания, используемые при обработке запроса, полученного от DUA или другого DSA, например в процессе разрешения имени, определяются зависящей от DSA процедурой выбора, на основании чего DSA рассчитывает, базируясь на любом нестандартизированном критерии, который административный орган считает соответствующим, точку доступа DSA, способного обрабатывать запрос.

ПРИМЕЧАНИЕ. – Спецификации Справочника не налагают ограничений на порядок использования агентами DSA главных и теневых знаний (иной, чем косвенным образом путем введения ограничений на поведение DSA, например функциями управления службой **dontUseCopy** и **copyShallDo**, описанных в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3).

22.3.2 Типы ссылок на знания

Знания, которыми владеет DSA, определяются в аспекте множества, состоящего из одной или более ссылок на знания, где каждая ссылка связывает, прямо или косвенно, статьи (или статьи-копии) DIB с DSA, который содержит эти статьи (или статьи-копии).

DSA может содержать следующие типы ссылок на знания:

- старшие ссылки;
- непосредственно старшие ссылки;
- подчиненные ссылки;
- неспецифические подчиненные ссылки; и
- перекрестные ссылки.

Ссылка на знание конкретного типа должна быть ссылкой либо на главные, либо на теневые знания.

Кроме того, DSA, который участвует в процессе теневого копирования как теневой поставщик и/или теневой потребитель, может содержать один или более следующих типов ссылки на знания:

– ссылка на поставщика; и
– ссылка на потребителя.

Эти типы ссылок на знания описываются ниже.

22.3.2.1 Старшие ссылки

Старшую ссылку составляет:

– точка доступа DSA.

Каждый DSA не первого уровня (см. п. 22.5) должен сохранять по крайней мере одну старшую ссылку. Старшая ссылка должна составлять часть траектории ссылки до корня. Если для обеспечения этого не используется какой-либо не входящий в данный стандарт метод, например в пределах DMD, это требование должно выполняться путем ссылки на DSA, который содержит контекст именования или дублируемую область, префикс контекста которого имеет меньшее количество RDN, чем префикс контекста с наименьшим количеством RDN, содержащийся DSA, который содержит эту ссылку.

22.3.2.2 Непосредственно старшие ссылки

Непосредственно старшую ссылку составляют:

– префикс контекста контекста именования, который является непосредственно старшим по отношению к контексту, содержащемуся (в качестве статьи или статьи-копии) в DSA, который содержит эту ссылку;
– точка доступа DSA, содержащего контекст именования (в качестве статьи или статьи-копии).

Непосредственно старшая ссылка является необязательным типом ссылки, который используется только при наличии иерархического операционного связывания с DSA, на который делается эта ссылка (см. п. 24 в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4). В отсутствие таких явных операционных привязок ссылка на непосредственно старший контекст именования может быть сделана посредством перекрестной ссылки.

22.3.2.3 Подчиненные ссылки

Подчиненную ссылку составляют:

– префикс контекста контекста именования, который является непосредственно подчиненным по отношению к контексту, содержащемуся (в качестве статьи или статьи-копии) в DSA, который содержит эту ссылку;
– точка доступа DSA, содержащего контекст именования (в качестве статьи или статьи-копии).

Все контексты именования, являющиеся непосредственно подчиненными по отношению к контекстам именования, которые содержатся в главном DSA, должны быть представлены подчиненными ссылками (или неспецифическими подчиненными ссылками, описанными в п. 22.3.2.4).

В случае если DSA содержит статьи-копии, подчиненные контексты именования могут быть представлены или не представлены в зависимости от действующего соглашения о теневом копировании.

22.3.2.4 Неспецифические подчиненные ссылки

Неспецифическую подчиненную ссылку составляют:

– точка доступа DSA, который содержит статьи (или статьи-копии) одного или более непосредственно подчиненного контекста именования.

Данный тип ссылки является необязательным и предусмотрен для случая, когда известно, что DSA содержит некоторые подчиненные статьи (или статьи-копии), а специфические RDN этих статей (или статей-копий) неизвестны. Этот тип ссылки не может использоваться для ссылки на серверы LDAP.

Для каждого контекста именования, который он содержит, главный DSA может содержать нуль и более неспецифических подчиненных ссылок. DSA, к которым осуществляется доступ по неспецифической ссылке, должны обладать способностью выполнения запроса напрямую (удачно или неудачно). В случае неудачи запросчику возвращается сообщение об ошибке службы **serviceError** с указанием причины **unableToProceed** (невозможно обработать запрос).

В случае если DSA содержит статьи-копии, неспецифические подчиненные ссылки могут быть представлены или не представлены в зависимости от действующего соглашения о теневом копировании.

22.3.2.5 Перекрестные ссылки

Перекрестную ссылку составляют:

– префикс контекста;

– точка доступа DSA, который содержит статьи или статьи-копии для контекста именования.

Данный тип ссылки является необязательным и служит для оптимизации разрешения имени. DSA может содержать любое (включая нуль) количество перекрестных ссылок.

22.3.2.6 Ссылки на поставщика

Ссылку на поставщика, которая содержится в DSA теневого потребителя, составляют:

– префикс контекста контекста именования, из которого выводится получаемая от теневого поставщика дублируемая область;

– идентификатор соглашения о теневом копировании, которое теневой потребитель заключил с теневым поставщиком;

– точка доступа DSA теневого поставщика;

– указание того, является ли теневой поставщик дублируемой области главным или не главным; и

– факультативно, точка доступа главного DSA, если поставщик не является главным.

22.3.2.7 Ссылки на потребителя

Ссылку на потребителя, которую содержит DSA теневого поставщика, составляют:

– префикс контекста контекста именования, из которого выводится предоставляемая теневым поставщиком дублируемая область;

– идентификатор соглашения о теневом копировании, которое теневой поставщик заключил с теневым потребителем; и

– точка доступа DSA теневого потребителя.

22.4 Минимальные знания

Свойством Справочника является возможность доступа ко всем статьям независимо от места генерации запроса.

Также свойством Справочника является предусмотренная для обеспечения адекватных уровней эффективности и готовности возможность выполнения некоторых запросов с использованием копии статьи, в то время как другие запросы могут быть выполнены только с использованием самих статей (т. е. информации, которую главный DSA содержит для этой статьи).

Для реализации этих базирующихся на независимости от местоположения свойств Справочника каждый DSA должен поддерживать минимальный объем знаний, который зависит от конкретной конфигурации данного DSA

Эти минимальные требования необходимы для того, чтобы распределенный процесс разрешения имен мог устанавливать траекторию ссылки – как непрерывную последовательность ссылок на главные знания – ко всем контекстам именования в пределах Справочника.

Сверх этих минимальных требований для установления траекторий ссылок к копиям контекстов именования могут использоваться дополнительные знания. Для установления оптимизированных траекторий ссылок к контекстам именования и копиям контекстов именования могут использоваться знания перекрестных ссылок (главные и теневые).

Требования в отношении минимального объема знаний для агентов DSA определяются в пп. 22.4.1–22.4.4.

22.4.1 Старшие знания

Каждый DSA, не являющийся DSA первого уровня, должен поддерживать по крайней мере одну старшую ссылку. Дополнительные старшие ссылки могут содержаться для операционных задач в качестве альтернативных траекторий к корню DIT.

22.4.2 Подчиненные знания

DSA, который является главным DSA контекста именования, должен поддерживать подчиненные или неспецифические подчиненные ссылки категории главных знаний к каждому главному DSA, содержащему (в качестве главного) непосредственно подчиненный контекст именования.

22.4.3 Знания о поставщике

DSA теневого потребителя должен поддерживать ссылку на теневого поставщика для каждого DSA теневого поставщика, который предоставляет ему дублируемую область. Если подчиненные знания теневого потребителя для копии контекста именования неполны, то для установления траектории ссылок к подчиненной информации он должен использовать свою ссылку на поставщика. Эта процедура описывается в п. 20 Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4.

22.4.4 Знания о потребителе

DSA теневого поставщика должен поддерживать ссылку на потребителя для каждого DSA теневого потребителя, которому он предоставляет дублируемую область.

22.5 DSA первого уровня

DSA на который делается старшая ссылка, принимает на себя бремя установления траектории ссылок ко всему DIT, не известному для обращающегося DSA. DSA, на который ссылаются другие DSA, сам может поддерживать одну и более старших ссылок. Этот рекурсивный процесс старших обращений останавливается на множестве *DSA первого уровня*, на котором лежит окончательная ответственность за установление траекторий ссылок.

DSA первого уровня характеризуется следующим:

a) он не содержит старших ссылок;

b) он может содержать один или более контекстов именования, являющихся непосредственно подчиненными по отношению к корню DIT (как главный или теневой DSA для контекста именования); и

c) он содержит подчиненные ссылки (главной и/или теневой категории) и неспецифические подчиненные ссылки (главной и/или теневой категории), которые отвечают за все контексты именования, являющиеся непосредственно подчиненными по отношению к корню DIT, которое сам он не содержит.

Административные органы для DSA первого уровня совместно осуществляют административное управление непосредственно подчиненными корня DIT. Процедуры, регулирующие это совместное административное управление, определяются многосторонними соглашениями, которые не входят в область определения спецификаций Справочника.

ПРИМЕЧАНИЕ. – В среде связанных статей возможно существование нескольких статей первого уровня, которые имеют то же имя, создавая множественные DIT. Административное управление этими DIT совместно осуществляют административные органы для связанных DSA первого уровня.

Для ограничения количества запросов, которые могут направляться к главному DSA первого уровня (т. е. DSA, главному для контекста именования, являющегося непосредственно подчиненным относительно корня DIT), можно учредить теневые DSA первого уровня для главного DSA первого уровня. Эти теневые DSA хранят копии статей и подчиненных ссылок, которые являются непосредственно подчиненными относительно корня, содержащегося в его главном DSA (или в DSA поставщика) первого уровня. Таким образом, они могут служить старшей ссылкой для DSA не первого уровня.

23 Базовые элементы информационной модели DSA

23.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

23.1.1 информационное дерево DSA: Совокупность всех DSE, содержащихся DSA, с позиции их имен.

23.1.2 совместный атрибут DSA: Операционный атрибут в информационной модели DSA, связанный с конкретным именем, значение или значения которого, если содержатся несколькими DSA, идентичны (за исключением периодов временной несогласованности).

23.1.3 зависящий от DSA атрибут: Операционный атрибут в информационной модели DSA, связанный с конкретным именем, значение или значения которого, если они содержатся в нескольких DSA, не должны обязательно быть идентичными.

23.1.4 зависящая от DSA статья (DSE): Информация, содержащаяся DSA, которая связана с конкретным именем; DSE может (но необязательно) содержать информацию, связанную с соответствующей статьей Справочника.

23.1.5 тип DSE: Указание конкретного назначения DSE; DSE может иметь множество назначений и, следовательно, множество типов.

23.2 Введение

Информационная модель Справочника описывает, каким образом Справочник в целом представляет информацию об объектах, имеющих выделенное имя и факультативно псевдонимы. В этом описании DIT, статей и атрибутов композиция Справочника как совокупности потенциально взаимодействующих DSA выделена из модели.

Информационная модель DSA, с другой стороны, посвящена конкретно DSA и информации, которую должны содержать DSA, для того чтобы совокупность составляющих Справочник DSA могла вместе реализовать информационную модель Справочника. Модель рассматривает:

– как информация Справочника (статьи и подстатьи объектов и псевдонимов) отображаются в DSA;
– как копии информации Справочника могут содержаться в DSA;
– операционную информацию, требуемую DSA для выполнения разрешения имени и оценки операций; и
– операционную информацию, требуемую DSA для задействования в теневом копировании и для использования теневой информации.

Целью моделирования представления операционной информации DSA, такой как знания, является создание общей основы для управлением доступом к операционной информации DSA.

23.3 Зависящие от DSA статьи и их имена

В информационной модели DSA хранилища информации, содержащие информацию, связанную с конкретным именем, называются *зависящими от DSA статьями* (DSE). Статьи Справочника существуют в информационной модели DSA только как информационные элементы, из которых могут состоять DSE. Операционные атрибуты, специфические для информационной модели DSA, образуют другую разновидность информационных элементов, из которых могут состоять DSE.

Если DSA содержит какую-либо информацию, непосредственно относящуюся к имени (т. е. информацию, содержащуюся в хранилище и обозначаемую по имени), считается, что он знает это имя или обладает знаниями о нем.

Для каждого имени, известного DSA, вся информация, хранящаяся этим DSA и непосредственно связанная с именем, кроме самого этого имени, представляется одной DSE. Эта последняя информация (т. е. RDN и его взаимосвязь с DIT) не представляется в информационной модели DSA явным образом в атрибутах; совокупность имен, известных DSA, образует неявную матрицу, в которой связанные DSE могут считаться присоединенными.

ПРИМЕЧАНИЕ 1. – Один из последствий способа обработки имен информационной моделью DSA заключается в том, что в отношении DSE, тип которых не является статьей, псевдонимом или подстатьей, AVA, выражающие RDN этой DSE, не моделируются как содержащиеся в атрибуте(ах).

Если существуют альтернативные имена, вследствие наличия атрибутов именования, имеющих множественные выделенные значения, которые различаются контекстом, единичная DSE представляет всю содержащуюся в DSA информацию обо всех альтернативных именах. Это моделируется в информационной модели DSA как единичное имя с зависящими от контекста вариантами, а не как отдельные имена.

ПРИМЕЧАНИЕ 2. – Для целей согласованного разрешения имен и межсетевого взаимодействия с DSA, относящимися к изданию до третьего, каждый DSA должен иметь информацию по крайней мере о первичных выделенных значениях всех атрибутов, участвующих в имени, и, желательно, о возможно большем количестве альтернативных выделенных значений.

Совокупность всех имен, известных DSA, вместе с информацией, связанной с каждым именем, с позиции этих имен называется *информационным деревом DSA* для этого DSA. Информационное дерево DSA показано на рисунке 21.

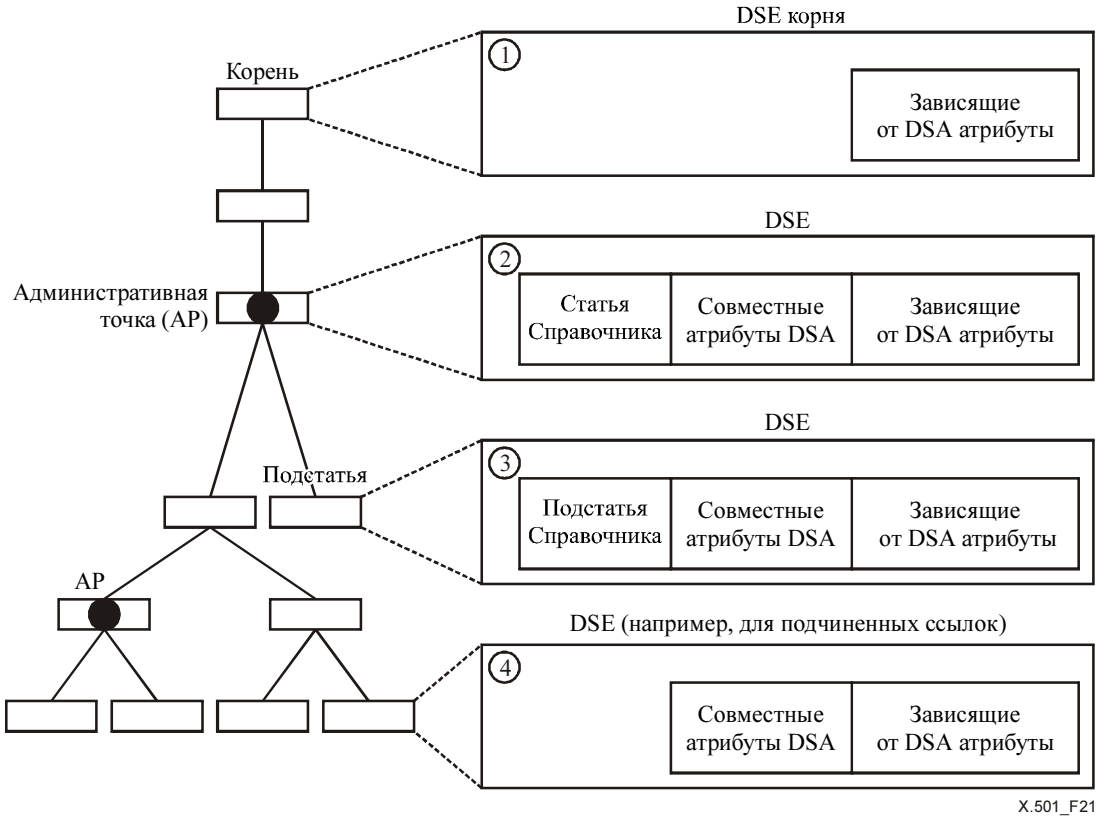


Рисунок 21 – Информационное дерево DSA

Минимальная информация, которую DSA может связывать с именем и, следовательно, знать имя, заключается в выражении назначения, для которого известно это имя (т. е. функции, которую выполняет имя в процессе функционирования знающего его DSA). Это назначение представляется в информационной модели DSA зависящим от DSA атрибутом **dseType**.

Кроме того, DSE может содержать другую информацию, связанную с этим именем, такую как статья или статья-копия, совместные атрибуты DSA и зависящие от DSA атрибуты.

DSE может представлять непосредственно статью Справочника, часть статьи или никакой информации Справочника. Информация, содержащаяся в DSE, варьируется в зависимости от ее типа и назначения. В общем случае в DSA могут существовать следующие виды DSE.

– DSE, непосредственно представляющая статью Справочника, содержит атрибуты пользователя и операционные атрибуты, соответствующие этой статье Справочника (как показано в DSE 2 на рисунке 21). DSE может также содержать совместные атрибуты DSA и зависящие от DSA атрибуты.

– DSE, представляющая часть статьи (как результат создания теневой копии), содержит некоторые атрибуты пользователя и некоторые операционные атрибуты, соответствующие этой статье Справочника, зависящие от DSA атрибуты и может также содержать совместные атрибуты DSA.

– DSE подстатьи, представляющая, например, предписывающую ACI или коллективные атрибуты, содержит релевантные атрибуты пользователя и релевантные операционные атрибуты, соответствующие подстатье Справочника (как показано в DSE 3 на рисунке 21). DSE может также содержать совместные атрибуты DSA и зависящие от DSA атрибуты.

– DSE, не представляющая информацию статьи Справочника, содержит только совместные атрибуты DSA и/или зависящие от DSA атрибуты (как показано в DSE 1 и 4 на рисунке 21). Например, DSE, представляющая подчиненную ссылку, может иметь совместный атрибут DSA, который указывает главную точку доступа, и зависящий от DSA атрибут для указания, что эта DSE является подчиненной ссылкой.

ПРИМЕЧАНИЕ 3. – DSE – это концептуальная статья, которая способствует составлению спецификации и моделированию информационных компонентов согласованным и удобным способом. Хотя говорят, что DSE "содержат" или "хранят" информацию, для реализации это не означает каких-либо конкретных ограничений или структур данных реализации.

23.4 Базовые элементы

DSE состоит из трех базовых элементов: тип DSE, некоторое количество операционных атрибутов DSA (тип DSE – один из них) и факультативно статья или статья-копия.

23.4.1 Операционные атрибуты DSA

В информационной модели DSA существуют две разновидности операционных атрибутов, которые не соответствуют информации в статьях Справочника. Ими являются совместные атрибуты DSA и зависящие от DSA атрибуты.

Совместный атрибут DSA – это операционный атрибут в информационной модели DSA, связанный с конкретным именем, значение или значения которого, если они содержатся в нескольких DSA, идентичны (за исключением периодов временной несогласованности). DSA может содержать теневую копию совместного атрибута DSA.

Зависящий от DSA атрибут – это операционный атрибут в информационной модели DSA, связанный с конкретным именем, значение или значения которого, если они содержатся в нескольких DSA, не должны обязательно быть идентичными. Зависящий от DSA атрибут представляет операционную информацию, которая является характерной для функционирования содержащего его DSA. DSA не может содержать теневую копию зависящего от DSA атрибута.

ПРИМЕЧАНИЕ. – Хотя DSA теневого поставщика может предоставлять DSA теневого потребителя зависящий от DSA атрибут, это концептуально является не теневой копией информации, которой располагает поставщик, но скорее информацией, произведенной поставщиком для потребителя, которую потребитель далее может использовать и изменять.

23.4.2 Типы DSE

Тип DSE, представленный в информационной модели DSA зависящим от DSA атрибутом **dseType**, указывает конкретное назначение (функцию) DSE. Это назначение указывается именованными битами отдельного значения атрибута **dseType**. Поскольку DSE может выполнять несколько функций, для представления этих функций могут быть установлены несколько именованных битов атрибута **dseType**. Число вероятных комбинаций именованных битов определяется в Приложении N.

Выражение "DSE типа **x**" используется в спецификациях Справочника для указания того, что установлен именованный бит **x** атрибута **dseType** DSE. Для DSE типа x прочие именованные биты могут быть или не быть установлены, исходя из необходимости. Также может использоваться альтернативное выражение "тип DSE включает **x**".

Синтаксическая спецификация операционного атрибута **dseType** может быть составлена с использованием нотации атрибутов следующим образом:

```
dseType  ATTRIBUTE  ::= {
        WITH SYNTAX                     DSEType
        EQUALITY MATCHING RULE          bitStringMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-dseType }
```

Этот зависящий от DSA операционный атрибут управляется самим DSA.

Типом ASN.1, который представляет синтаксис возможных значений атрибута **dseType**, является **DSEType**. Он содержит следующее описание:

```
DSEType  ::=  BIT STRING {
        root            (0),    -- DSE корня --
        glue            (1),    -- представляет знания только имени --
        cp              (2),    -- префикс контекста --
        entry           (3),    -- статья объекта --
        alias           (4),    -- статья псевдонима --
        subr            (5),    -- подчиненная ссылка --
        nssr            (6),    -- неспецифическая подчиненная ссылка --
        supr            (7),    -- старшая ссылка --
        xr              (8),    -- перекрестная ссылка --
        admPoint        (9),    -- административная точка --
        subentry        (10),   -- подстатья --
        shadow          (11),   -- теневая копия --
        immSupr         (13),   -- непосредственно старшая ссылка --
        rhob            (14),   -- информация rhob --
        sa              (15),   -- подчиненная ссылка на статью псевдонима --
        dsSubentry      (16),   -- зависящая от DSA подстатья --
        familyMember    (17),   -- член семейства --
        ditBridge       (18),   -- ссылка – мост между DIT --
        writeableCopy   (19) }  -- перезаписываемая копия
```

Значениями **DSEType** являются:

a) **root**: DSE корня содержит зависящие от DSA атрибуты, используемые DSA, которые характеризуют DSA в целом. Именем, соответствующим DSE корня, является вырожденное имя, состоящее из последовательности нулевых RDN.

ПРИМЕЧАНИЕ 1. – Информация, характеризующая DSA, который должен быть доступным через абстрактную службу Справочника, содержится в статье этого DSA. DSA может, но необязательно, содержать собственную статью или копию собственной статьи.

b) **glue**: Стыковочная DSE представляет знания только об имени. DSA, содержащий DSE префикса контекста или DSE перекрестной ссылки, может содержать стыковочные DSE для представления имен старших DSE префикса контекста или перекрестной ссылки, если с этими именами более не связано какой-либо операционной информации (т. е. знаний). Это показано на рисунке 22. DSE стыковочного типа не должна иметь какого-либо другого установленного бита **DSEType**.

c) **cp**: DSE, представляющая префикс контекста контекста именования.

d) **entry**: DSE, которая содержит статью объекта.

e) **alias**: DSE, которая содержит статью псевдонима.

f) **subr**: DSE, которая содержит специфический атрибут знаний для представления подчиненной ссылки.

g) **nssr**: DSE, которая содержит неспецифический атрибут знаний для представления неспецифической подчиненной ссылки.

h) **supr**: DSE, которая содержит специфический атрибут знаний для представления старших ссылок DSA.

i) **xr**: DSE, которая содержит специфический атрибут знаний для представления перекрестной ссылки.

j) **admPoint**: DSE, соответствующая административной точке.

k) **subentry**: DSE, которая содержит подстатью.

l) **shadow**: DSE, которая содержит теневую копию статьи (или части статьи) или иную информацию (например, знания), полученные от теневого поставщика; этот именованный бит устанавливается теневым потребителем.

m) **immSupr**: DSE, которая содержит специфический атрибут знания для представления непосредственно старшей ссылки.

n) **rhob**: DSE, которая содержит административную точку и информацию подстатьи, полученные от старшего DSA в релевантном иерархическом рабочем связывании (т. е. либо в иерархическом рабочем связывании, либо в неспецифическом иерархическом связывании, как описано в пп. 24 и 25 Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4).

o) **sa**: Спецификатор **subr** DSE, указывающий, что подчиненная статья контекста именования является псевдонимом.

p) **dsSubentry**: DSE, которая содержит специфическую подстатью DSA.

q) **familyMember**: DSE, которая содержит член семейства.

r) **ditBridge**: DSE, которая содержит ссылку – мост между DIT.

s) **writeableCopy**: DSE, которая содержит перезаписываемую копию статьи или иной информации (например, знаний), которая тиражируется в реализации с множественными главными.

ПРИМЕЧАНИЕ 2. – Для некоторых операций Справочника требуется способность опознания единичной главной для любой данной статьи. Следовательно, при реализации с множественными главными все кроме одной главные копии каждой статьи должны быть этого типа DSE; строго одна из главных копий не должна быть этого типа DSE, с тем чтобы при необходимости такой операции она функционировала как первичная главная.

Использование этого операционного атрибута для представления аспектов информационной модели DSA описывается в п. 23.

24 Представление информации DSA

В данном пункте поясняется представление информации DSA. Описывается представление операционной информации DSA (знаний), пользовательской информации Справочника и операционной информации Справочника.

24.1 Представление пользовательской и операционной информации Справочника

В данном пункте описывается представление пользовательской и операционной информации Справочника в информационной модели DSA.

24.1.1 Статья объекта

Статья объекта представляется DSE типа **entry**, которая содержит атрибуты пользователя и операционные атрибуты, связанные с этой статьей объекта. Именем DSE является имя статьи объекта (т. е. выделенное имя объекта).

Если DSE содержит копию статьи, тип DSE включает **shadow**.

Если имя статьи объекта включает какие-либо альтернативные выделенные имена, отличающиеся контекстом, имя DSE также может включать эти альтернативные выделенные имена, отличающиеся контекстом. В случае DSE, содержащей теневую копию статьи, имя DSE может также включать подмножество этих альтернативных выделенных имен. В случае DSE, которая не является копией, имя DSE должно включать все выделенные имена.

ПРИМЕЧАНИЕ. – Для целей согласованности и межсетевого взаимодействия с DSA, соответствующими созданным до третьего издания, имя содержащей копию DSE должно включать, по крайней мере, первичное выделенное значение любого атрибута именования. Таким образом, копия имеет, по крайней мере, первичное выделенное имя статьи объекта. Качество разрешения имени повышается, если присутствуют все выделенные значения (и, следовательно, все альтернативные выделенные имена).

24.1.2 Статья псевдонима

Статья псевдонима представляется DSE типа **alias**, которая содержит атрибуты, связанные со статьей псевдонима (т. е. атрибуты RDN и атрибут имени объекта с псевдонимом). Именем DSE является имя статьи псевдонима.

Если DSE содержит копию статьи псевдонима, тип DSE включает **shadow**.

Если имя статьи псевдонима включает какие-либо альтернативные выделенные имена, отличающиеся контекстом, имя DSE также может включать эти альтернативные выделенные имена, отличающиеся контекстом. В случае DSE, содержащей теневую копию статьи псевдонима, имя DSE может также включать подмножество этих альтернативных выделенных имен. В случае DSE, которая не является копией, имя DSE должно включать все выделенные имена.

ПРИМЕЧАНИЕ. – Для целей согласованности и межсетевого взаимодействия с DSA, созданными до третьего издания, имя DSE, содержащей копию, должно включать, по крайней мере, первичное выделенное значение любого атрибута именования. Таким образом, копия имеет, по крайней мере, первичное выделенное имя статьи псевдонима. Качество разрешения имени повышается, если присутствуют все выделенные значения (и, следовательно, все альтернативные выделенные имена).

24.1.3 Административная точка

Административная точка представляется DSE типа **admPoint**, которая содержит атрибуты, связанные с административной точкой. Именем DSE является имя административной точки.

Если DSE представляет статью, тип DSE включает **entry**. Если DSE содержит копию информации административной точки, тип DSE включает **shadow**.

Если имя административной точки включает какие-либо альтернативные выделенные имена, отличающиеся контекстом, имя DSE также может включать эти альтернативные выделенные имена, отличающиеся контекстом. В случае DSE, содержащей теневую копию статьи административной точки, имя DSE может также включать подмножество этих альтернативных выделенных имен. В случае DSE, которая не является копией, имя DSE должно включать все выделенные имена.

ПРИМЕЧАНИЕ. – Для целей согласованности и межсетевого взаимодействия с DSA, соответствующими предварительному третьему изданию, имя DSE, содержащей копию, должно включать, по крайней мере, первичное выделенное значение любого атрибута именования. Таким образом, копия имеет, по крайней мере, первичное выделенное имя административной точки. Качество разрешения имени повышается, если присутствуют все выделенные значения (и, следовательно, все альтернативные выделенные имена).

24.1.4 Подстатья

Подстатья представляется DSE типа **subentry**, которая содержит операционную и пользовательскую информацию, связанную с этой подстатьей. Именем DSE является имя подстатьи.

Если DSE содержит копию подстатьи, типом DSE является **subentry** и **shadow**.

24.1.5 Член семейства

Член семейства (включая порождающий элемент) представляется DSE типа **familyMember**. Порождающий элемент также является DSE типа **entry**; это единственный член семейства, для которого допустимо иметь данный тип DSE.

24.2 Представление ссылок на знания

Ссылка на знания состоит из DSE соответствующего типа, которая содержит соответственным образом подходящий операционный атрибут DSA и которая обозначается по имени, отражающему определенные взаимоотношения с контекстом именования, содержащимся DSA, на который делается эта ссылка.

Именем этой DSE должно быть первичное выделенное имя и оно может включать альтернативные имена и информацию контекста, если они присутствует в префиксе контекста контекста именования, содержащимся DSA, на который делается эта ссылка. В случае DSE, которая содержит теневую копию, имя DSE может включать подмножество этих альтернативных имен. В случае DSE, которая не является копией, имя DSE должно включать все выделенные имена.

ПРИМЕЧАНИЕ. – Качество разрешения имени повышается, если присутствуют все выделенные значения (и, следовательно, все альтернативные выделенные имена).

24.2.1 Типы атрибута знаний

Операционные атрибуты DSA определяются в информационной модели DSA для выражения имеющихся у DSA:

- знаний о его собственной точке доступа;
- старших знаний;
- специфических знаний (его подчиненные ссылки);
- неспецифических знания (его неспецифические подчиненные ссылки);
- знаний о его поставщике(ах), факультативно включая главного, если DSA является теневым потребителем;
- знаний о его потребителе(ях), если DSA является теневым поставщиком;
- знания о вторичных тенях, если DSA является теневым поставщиком; и
- знания о другом DIT.

Значения идентификатора объекта для этих операционных атрибутов задаются в Приложении F.

24.2.1.1 Моя точка доступа

Тип операционного атрибута **myAccessPoint** (моя точка доступа) используется DSA для представления своей собственной точки доступа. Это – зависящий от DSA атрибут. Все DSA должны содержать этот атрибут в своей DSE корня. Этот атрибут является содержащим одно значение и он управляется самими DSA.

```
myAccessPoint  ATTRIBUTE  ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-myAccessPoint }
```

Модуль ASN.1 **AccessPoint** определяется в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4. Его спецификация на языке ASN.1 воспроизводится здесь для удобства читателя.

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

ПРИМЕЧАНИЕ. – **Name** в **ae-title** может быть первичным выделенным именем или альтернативным выделенным именем; однако согласованность и межсетевое взаимодействие с DSA, созданными до третьего издания, улучшаются при использовании первичного выделенного имени.

В настоящей спецификации Справочника не описывается, каким образом DSA получает информацию, содержащуюся в **myAccessPoint**.

Тип атрибута **myAccessPoint** содержится в DSE типа **root**.

Информация, содержащаяся в **myAccessPoint**, может использоваться в DOP при установлении или изменении рабочего связывания.

24.2.1.2 Старшие знания

Операционный атрибут типа **superiorKnowledge** используется DSA не первого уровня для представления старших ссылок. Это атрибут, зависящий от DSA. Все DSA не первого уровня должны содержать данный атрибут в своей DSE корня. Этот атрибут является содержащим множественные значения и он управляется самим DSA.

```
superiorKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-superiorKnowledge }
```

DSA может запрашивать содержащуюся в **superiorKnowledge** информацию методами, которые не описываются в настоящих спецификациях Справочника. Он может также составлять ее из непосредственно старших ссылок, например из своей непосредственно старшей ссылки, префикс которой в своем имени имеет наименьшее число RDN.

Тип атрибута **superiorKnowledge** содержится в DSE корневого типа **root**.

Содержащаяся в **superiorKnowledge** информация может использоваться DSA при построении непрерывной ссылки, возвращаемой в отсылке DAP или DSP, или при выполнении сцепления.

24.2.1.3 Специфические знания

Специфические знания составляют точки доступа для главного DSA контекста именовании и/или теневых DSA для этого контекста именования. Они являются специфическими, потому что префикс контекста контекста именования известен и связан с информацией о точке доступа. Специфические знания представляются операционным атрибутом типа **specificKnowledge**. Это совместный атрибут DSA, он является содержащим единственное значение и управляется самим DSA.

```
specificKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation

        ID                              id-doa-specificKnowledge }
```

Модель на языке ASN.1 **MasterAndShadowAccessPoints** определяется в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4. Его спецификация на языке ASN.1 приводится здесь для удобства читателей.

```
MasterAndShadowAccessPoints  ::=  SET OF MasterOrShadowAccessPoint

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

        chainingRequired  [5]   BOOLEAN  DEFAULT FALSE }
```

DSA может запрашивать содержащуюся в **specificKnowledge** информацию методами, которые не описываются в настоящих спецификациях Справочника. В случае перекрестной ссылки (DSE типа **xr**) он может также составлять ее из информации, полученной в компоненте **crossReference** параметра **ChainingResults** ответа DSP. В случае подчиненной ссылки (DSE типа **subr**) он может составлять ее из информации, полученной в DOP при установлении или изменении HOB.

Атрибут типа **specificKnowledge**, содержится в DSE типа **subr**, **immSupr** или **xr**. Он используется DSA для представления подчиненных, непосредственно старших и перекрестных ссылок.

Содержащаяся в **specificKnowledge** информация может использоваться DSA при построении непрерывной ссылки, возвращаемой в отсылке DAP или DSP (или при выполнении сцепления), а также при построении имеющих тень зависящих от DSA статей (SDSE) типа **subr**, **immSupr** или **xr**, предоставляемых в DISP.

24.2.1.4 Неспецифические знания

Неспецифические знания составляют точки доступа главного DSA одного или более контекстов именования и/или теневых DSA для тех же одного или более контекстов именования. Они являются неспецифическими, потому что префиксы контекстов контекста(ов) именования неизвестен (неизвестны). Непосредственно старший контекста(ов) именования, тем не менее, известен, и информация о точке доступа связана с его именем. Неспецифические знания представляются типом операционного атрибута **nonSpecificKnowledge**. Это совместный атрибут DSA, он является содержащим множественные значения и управляется самим DSA.

```
nonSpecificKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation
        ID                              id-doa-nonSpecificKnowledge }
```

Значение **MasterAndShadowAccessPoints** составляют точка доступа для главного DSA, содержащего один или более подчиненных контекстов именования, и нуль и больше точек доступа DSA, содержащих теневые копии некоторых или всех этих контекстов именования.

DSA может запрашивать содержащуюся в **nonSpecificKnowledge** информацию методами, которые не описываются в настоящих спецификациях Справочника. В случае неспецифической подчиненной ссылки (DSE типа **nssr**) он может также составлять ее из информации, полученной в DOP при установлении или изменении NHOB.

Тип атрибута **nonSpecificKnowledge** содержится в DSE типа **nssr**. Он используется для представления неспецифических подчиненных ссылок.

Содержащаяся в **nonSpecificKnowledge** информация может использоваться DSA при составлении непрерывной ссылки, возвращаемой в отсылке DAP или DSP (или при выполнении сцепления), а также при составлении SDSE типа **nssr**, предоставляемых в DISP.

24.2.1.5 Знания поставщика

Знания поставщика о DSA теневого потребителя составляют точка(и) доступа и идентификатор(ы) соглашения о теневом копировании для его поставщика(ов) копии (или копий) дублируемой области. Факультативно, если этот поставщик не является главным контекста именования, из которого выведена дублируемая область, в знания поставщика может быть включена точка доступа главного. Знания поставщика представляются типом операционного атрибута **supplierKnowledge**. Это зависящий от DSA атрибут, он является содержащим множественные значения и управляется самим DSA.

Синтаксисом ASN.1 для значения **supplierKnowledge** является **SupplierInformation**. Значение этого атрибута составляют точка доступа DSA теневого поставщика и идентификатор соглашения о теневом копировании между DSA поставщика и DSA потребителя, содержащего зависящий от DSA атрибут (выраженный как значение типа **SupplierOrConsumer**); указание того, является ли поставщик дублируемой области главным контекста именования, из которого она выведена; и, если он не является главным, факультативно точка доступа главного DSA.

```
SupplierOrConsumer ::= SET {
        COMPONENTS OF           AccessPoint, -- поставщик или потребитель --

        agreementID       [3]   OperationalBindingID }

SupplierInformation ::= SET {
        COMPONENTS OF           SupplierOrConsumer,     -- поставщик --
        supplier-is-master      [4]   BOOLEAN DEFAULT TRUE,
        non-supplying-master    [5]   AccessPoint OPTIONAL }

supplierKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                   SupplierInformation
        EQUALITY MATCHING RULE        supplierOrConsumerInformationMatch
        NO USER MODIFICATION          TRUE
        USAGE                         dSAOperation
        ID                            id-doa-supplierKnowledge }
```

DSA может запрашивать содержащуюся в **supplierKnowledge** информацию методами, которые не описываются в настоящих спецификациях Справочника. DSA теневого потребителя может также составлять ее из информации, полученной в DOP при заключении или изменении соглашения о теневом копировании.

Тип атрибута **supplierKnowledge** содержится в DSE типа **cp**. Он используется для представления одной или более старших ссылок. Все DSA теневого потребителя должны содержать значение этого атрибута для каждого соглашения о теневом копировании, в котором они участвуют в качестве потребителя.

Содержащаяся в **supplierKnowledge** информация может использоваться DSA при составлении непрерывной ссылки, возвращаемой в отсылке DAP или DSP. Компонент **agreementID** (его тип, **OperationalBindingID**, определяется в п. 28.2) атрибута **supplierKnowledge** требуется в операциях DOP для управления соглашением о теневом копировании и во всех операциях DISP.

24.2.1.6 Знания потребителя

Знания потребителя о DSA теневого поставщика составляют точка(и) доступа и идентификатор(ы) соглашения(й) о теневом копировании для потребителя(ей) копии (или копий) контекста именования, предоставляемого им поставщиком. Знания потребителя представляются типом операционного атрибута **consumerKnowledge**. Это зависящий от DSA атрибут, он является содержащим множественные значения и управляется самим DSA.

Синтаксисом ASN.1 для значения **consumerKnowledge** является **ConsumerInformation** (который имеет тот же синтаксис, что и **SupplierOrConsumer**, но ссылается на точку доступа потребителя).

```
ConsumerInformation ::= SupplierOrConsumer      -- потребитель --

consumerKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                   ConsumerInformation
        EQUALITY MATCHING RULE        supplierOrConsumerInformationMatch
        NO USER MODIFICATION          TRUE
        USAGE                         dSAOperation
        ID                            id-doa-consumerKnowledge }
```

DSA может запрашивать содержащуюся в **consumerKnowledge** методами, которые не описываются в настоящих спецификациях Справочника. DSA теневого поставщика может также составлять ее из информации, полученной в DOP при заключении или изменении соглашения о теневом копировании.

Тип атрибута **consumerKnowledge** содержится в DSE типа **cp**. Он используется для представления одной или более ссылок на потребителя. Все DSA теневого поставщика должны содержать значение этого атрибута для каждого соглашения о теневом копировании, в котором они участвуют в качестве поставщика.

Компонент **agreementID** атрибута **consumerKnowledge** требуется в операциях DOP для управления соглашением о теневом копировании и во всех операциях DISP.

24.2.1.7 Вторичные теневые знания

Вторичные теневые знания составляют информация, которую DSA поставщика (т. е. главный DSA) может выбрать для сопровождения, связанного с DSA потребителей, которые участвуют во вторичном теневом копировании, со своей позиции. Вторичные теневые знания представляются типом операционного атрибута **secondaryShadows**. Это зависящий от DSA атрибут, он является содержащим множественные значения и управляется самим DSA. Синтаксисом ASN.1 для значения **secondaryShadows** является **SupplierAndConsumers**. Он содержит точку доступа теневого поставщика и перечень его прямых потребителей.

```
SupplierAndConsumers ::= SET {
        COMPONENTS OF       AccessPoint, -- поставщик --
        consumers           [3]     SET OF AccessPoint }

secondaryShadows ATTRIBUTE ::= {
        WITH SYNTAX                 SupplierAndConsumers
        EQUALITY MATCHING RULE      supplierAndConsumersMatch
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-secondaryShadows }
```

Компонент **consumers** атрибута **SuppliersAndConsumers** содержит только точки доступа DSA, которые содержат коллективно используемые копии дублируемой области.

DSA поставщика может получать информацию, необходимую для составления значений этого атрибута, от DSA потребителя, следуя процедуре, описанной в п. 23.1.1 Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4.

Тип атрибута **secondaryShadows** содержится в DSE типа **cp**.

Поддержка вторичных теневых знаний является факультативной.

24.2.1.8 Знания о мосте между DIT

Главный DSA контекста именования в другом DIT представляется атрибутом **ditBridgeKnowledge**, который состоит из идентификатора домена и его точки доступа. Операционный атрибут **dITBridgeKnowledge** содержит **DITBridgeKnowledge** всех известных таких DSA. Он является содержащим множественные значения, совместным атрибутом DSA и управляется администратором DSA. Этот атрибут содержится в DSE типа **root**, которая дополнительно получает DSE типа **ditBridge** для ссылки на мост между DIT.

```
ditBridgeKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                 DitBridgeKnowledge
        EQUALITY MATCHING RULE      directoryStringFirstComponentMatch
        NO USER MODIFICATION        TRUE
        USAGE                       dSAOperation
        ID                          id-doa-ditBridgeKnowledge }
```

Модуль ASN.1 **DitBridgeKnowledge** определяется в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4. Его спецификация на языке ASN.1 воспроизводится здесь для удобства читателей.

```
DitBridgeKnowledge ::= SEQUENCE {
        domainLocalID               DirectoryString     OPTIONAL,

        accessPoints                MasterAndShadowAccessPoints }
```

Содержащаяся в **ditBridgeKnowledge** информация должна использоваться DSA при выполнении операции поиска, включающей связанные статьи.

24.2.1.9 Правила сопоставления

Ниже описываются четыре правила сопоставления равенства для представленных ранее атрибутов знаний. Они применяются к атрибутам с синтаксисом типов **AccessPoint**, **MasterAndShadowAccessPoints**, **SupplierInformation**, **ConsumerInformation** и **SuppliersAndConsumers**.

24.2.1.9.1 Совпадение точки доступа

Правило совпадения точки доступа описывается следующим образом:

```
accessPointMatch  MATCHING-RULE  ::= {
        SYNTAX      Name
        ID          id-kmr-accessPointMatch }
```

Правило сопоставления **accessPointMatch** применяется к значениям атрибута типа **AccessPoint**. Значение синтаксиса утверждения выводится из значения синтаксиса атрибута с помощью значения специальной метки контекста **[0]** компонента (**Name**). Два значения считаются совпавшими по признаку равенства, если совпадает каждый компонент **Name** при использовании процедуры сопоставления для значений **DistinguishedName**.

24.2.1.9.2 Совпадение главной и теневой точек доступа

Правило совпадения главной и теневой точек доступа по правилу сопоставления на равенство описывается следующим образом:

```
masterAndShadowAccessPointsMatch  MATCHING-RULE  ::= {
        SYNTAX      SET OF Name
        ID          id-kmr-masterShadowMatch }
```

Правило сопоставления **masterAndShadowAccessPointsMatch** применяется к атрибутам типа **MasterAndShadowAccessPoints**. Значение синтаксиса утверждения выводится из значения синтаксиса атрибута путем удаления компонентов **category** и **address** каждого **SET** в **SET OF MasterOrShadowAccessPoints**. Два таких значения считаются совпавшими по признаку равенства, если оба значения имеют то же количество элементов **SET OF** и, после упорядочения любым удобным образом элементов **SET OF** каждого, компоненты **ae-title** каждой пары элементов **SET OF** совпадают при использовании процедуры сопоставления для **distinguishedNameMatch**.

24.2.1.9.3 Совпадение информации поставщика или потребителя

Правило совпадения информации поставщика или потребителя описывается следующим образом:

```
supplierOrConsumerInformationMatch  MATCHING-RULE  ::= {
        SYNTAX      SET {
                ae-title                [0]     Name,
                agreement-identifier    [2]     INTEGER }

        ID          id-kmr-supplierConsumerMatch }
```

Правило сопоставления **supplierOrConsumerInformationMatch** применяется к значениям атрибутов типа **SupplierInformation** или **ConsumerInformation** (и других атрибутов, имеющих значения, совместимые с **SupplierInformation** или **ConsumerInformation**). Значение синтаксиса утверждения выводится из значения синтаксиса атрибута путем отбора компонентов **SET** с метками, которые совпадают с компонентами **SET** синтаксиса утверждения. Два таких значения считаются совпавшими по признаку равенства, если компоненты **ae-title** каждого (после удаления явного **[0]** информации метки) совпадают при использовании процедуры сопоставления для значений **DistinguishedName,** и компоненты **identifier**, содержащиеся в компоненте **agreement** каждого (после удаления явного **[2]** и информации метки **SEQUENCE**), совпадают при использовании процедуры сопоставления для значений **INTEGER**.

24.2.1.9.4 Совпадение поставщиков и потребителей

Правило совпадения поставщика и потребителя описывается следующим образом:

```
supplierAndConsumersMatch  MATCHING-RULE  ::= {
        SYNTAX      Name
        ID          id-kmr-supplierConsumersMatch }
```

Правило совпадения поставщика и потребителя применяется к значениям атрибута типа **SupplierAndConsumers** (и других атрибутов, имеющих значения, совместимые с **SupplierAndConsumers**). Два таких значения считаются совпавшими по признаку равенства, если компоненты **ae-title** каждого (после удаления явного **[0]** и информации метки) совпадают при использовании процедуры для значений **DistinguishedName**.

24.2.2 Типы ссылок на знания

В этом подпункте описывается представление знаний в информационной модели DSA.

24.2.2.1 Самоотносимость

Самоотносимость представляет знания DSA о своей собственной точке доступа. Она представляется значением атрибута **myAccessPoint**, содержащегося в корневой DSE DSA, т. е. DSE типа **root**.

24.2.2.2 Старшая ссылка

Старшая ссылка представляется DSE типа **supr** и **root**, которая содержит атрибут **superiorKnowledge**. Поскольку значение атрибута **superiorKnowledge** может содержать точки доступа нескольких DSA, следовательно, оно может представлять несколько старших ссылок.

24.2.2.3 Непосредственно старшая ссылка

Непосредственно старшая ссылка представляется DSE типа **immSupr**, которая содержит атрибут **specificKnowledge**. Имя DSE, которая содержит этот атрибут, соответствует префиксу контекста именования, содержащемуся старшим DSA, на который указывает эта ссылка.

Поскольку значение атрибута **specificKnowledge** может содержать точки доступа нескольких DSA, следовательно, оно может представлять несколько непосредственно старших ссылок – не более чем одной категории **master** и нуля или более категория **shadow**.

Если содержащая непосредственно старшую ссылку DSE получена от теневого поставщика, тип DSE включает **shadow**.

24.2.2.4 Подчиненная ссылка

Подчиненная ссылка представляется DSE типа **subr**, которая содержит атрибут **specificKnowledge**. Имя DSE, которая содержит этот атрибут, соответствует префиксу контекста релевантного контекста именования, содержащегося подчиненным DSA, на который указывает эта ссылка.

Поскольку значение атрибута **specificKnowledge** может содержать точки доступа нескольких DSA, следовательно, оно может представлять несколько подчиненных ссылок – не более чем одной категории **master** и нуля или более категории **shadow**.

Если содержащая подчиненную ссылку DSE получена от теневого поставщика, тип DSE включает **shadow**.

DSE может также включать **immSupr** в DSA, содержащем два контекста именования, один из которых старший по отношению к другому, которые разделяются третьим контекстом именования в единичной статье, содержащимся в другом DSA. Пример такой ситуации приводится в Приложении O.

24.2.2.5 Неспецифическая подчиненная ссылка

Неспецифическая подчиненная ссылка представляется DSE типа **nssr** (и, как правило, **entry**), которая содержит атрибут **nonSpecificKnowledge**. Имя DSE, содержащей атрибут, соответствует имени, сформированному путем исключения окончательного RDN префиксов контекста контекста именования, содержащегося подчиненными DSA, на которые указывают ссылки.

ПРИМЕЧАНИЕ. – NSSR не могут делать ссылку на серверы LDAP.

Поскольку значение атрибута **nonSpecificKnowledge** может содержать точки доступа нескольких DSA, следовательно, оно может представлять несколько неспецифических подчиненных ссылок – не более чем одной категории **master** и нуля или более категории **shadow**. Каждое значение атрибута **nonSpecificKnowledge** представляет связанную совокупность неспецифических подчиненных ссылок, DSA категории **shadow** содержат одну и более дублируемые области, выведенные из контекста(ов) именования, содержащегося (содержащихся) в DSA категории **master**.

Если DSE, содержащая неспецифическую подчиненную ссылку, является теневой информацией, полученной от теневого поставщика, тип DSE включает **shadow**.

DSE включает **shadow** в случае теневого DSA, если DSE соответствует статье, для которой главный DSA имеет неспецифические подчиненные знания и для которой выполняется теневое копирование только атрибута **nonSpecificKnowledge** для неспецифической подчиненной ссылки.

DSE включает **cp** и **shadow** в случае теневого DSA, дублируемая область которого не включает статью префикса контекста, а главный DSA для контекста именования имеет неспецифические подчиненные знания для этого префикса контекста.

DSE включает **admPoint** и **shadow** в случае теневого DSA, если DSE соответствует административной точке, информация статьи для административной точки не имеет теневой копии, и главный DSA для контекста именования имеет неспецифические подчиненные знания для этой административной точки.

Если административная точка в двух предыдущих случаях совпадает с префиксом контекста, DSE может включать **admPoint**, **cp** и **shadow**.

24.2.2.6 Перекрестная ссылка

Перекрестная ссылка представляется DSE типа **xr**, которая содержит атрибут **specificKnowledge**. Имя DSE, содержащей этот атрибут, соответствует префиксу контекста контекста именования, содержащемуся DSA, на который указывает эта ссылка.

Поскольку значение атрибута **specificKnowledge** может содержать точки доступа нескольких DSA, следовательно, оно может представлять несколько перекрестных ссылок – не более одной категории **master** и нуля или более категории **shadow**.

24.2.2.7 Ссылка на поставщика

Ссылка на поставщика представляется DSE типа **cp**, которая содержит атрибут **supplierKnowledge**. Имя DSE, содержащей этот атрибут, соответствует префиксу контекста контекста именования, который имеет теневую копию.

Поскольку атрибут **supplierKnowledge** может иметь несколько значений, он может представлять несколько ссылок на поставщиков. Каждое значение атрибута представляет одну ссылку на поставщика.

24.2.2.8 Ссылка на потребителя

Ссылка на потребителя представляется DSE типа **cp**, которая содержит атрибут **consumerKnowledge**. Имя DSE, содержащей этот атрибут, соответствует префиксу контекста контекста именования, который имеет теневую копию.

Поскольку атрибут **consumerKnowledge** может иметь несколько значений, он может представлять несколько ссылок на потребителей. Каждое значение атрибута представляет одну ссылку на потребителя.

24.3 Представление имен и контекстов именования

24.3.1 Имена и стыковочные DSE

Как поясняется в п. 23.3, минимальной информацией, которую DSA может связывать с именем, является цель, для которой он содержит это имя, представленная DSE, которая содержит значение атрибута **dseType**. Если DSE содержит только такой минимум информации, ее тип DSE должен быть **glue** (стыковочная). В этом случае DSE не должна содержать статью или подстатью (или теневую копию статьи или подстатьи) или зависящий от DSA атрибут.

Стыковочные DSE возникают в информационной модели DSA для представления известных DSA имен в виде последовательности содержащейся информации, связанной с другими именами. Например, рассмотрим перекрестную ссылку, показанную на рисунке 22. Содержащий эту перекрестную ссылку DSA также "знает" (в указанном в п. 23.3 смысле) имена, которые являются старшими по отношению к имени префикса контекста, связанному с этой перекрестной ссылкой. Если с такими старшими именами не связана более никакая информация, они представляются в информационной модели DSA стыковочными DSE.

24.3.2 Контексты именования

Контекст именования образуют префикс контекста, поддерево, состоящее из нуля или более статей, подчиненных по отношению к этому префиксу контекста (корень поддерева), и, в случае наличия подчиненных по отношению к нему контекстов именования, подчиненные и/или неспецифические подчиненные ссылки, достаточные для формирования полного подчиненного знания.

Префикс контекста представляется DSE типа **cp**. Если префикс контекста соответствует статье, тип DSE включает **entry**. Если он соответствует псевдониму, тип DSE включает **alias**. Если префикс контекста соответствует административной точке, тип DSE включает **admPoint**.

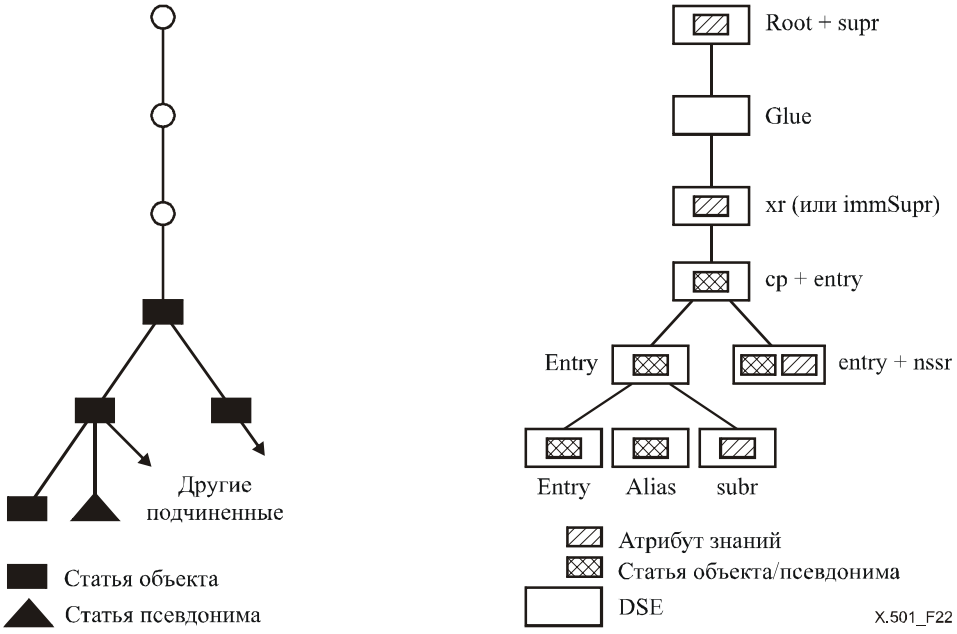
Поддерево статей и подстатей, подчиненных по отношению к префиксу контекста, представляется DSE, как описано в пп. 24.1.1–24.1.5.

Представление подчиненных знаний контекста именования выполняется статьями DSE, как описано в п. 24.2.2.

Дублируемая область (теневая копия всего или части контекста именования) представляется как указано выше, за исключением того, что тип DSE включает **shadow** в каждой DSE, для которой от теневого поставщика получены атрибуты пользователя и операционные атрибуты. В случае неполных дублируемых областей DSE типа **glue** могут представлять мост между отдельными фрагментами информации, которая имеет теневую копию. С этими (или любыми) стыковочными DSE не связываются ни атрибуты пользователей, ни операционные атрибуты.

24.3.3 Пример

На рисунке 22 представлен пример отображения части DIT (соответствующей контексту именования) в информационное дерево DSA. Кроме самой информации контекста именования показаны корневая DSE DSA, содержащая старшую ссылку (это не информационное дерево DSA для DSA первого уровня), стыковочная DSE и DSE, представляющая ссылку (либо перекрестную ссылку, либо непосредственно старшую ссылку) на непосредственно старший контекст именования.



Поддерево DIT, соответствующее контексту именования

Информационное дерево DSA для контекста именования

Рисунок 22 – DSE для контекста именования

РАЗДЕЛ 11 – ФУНКЦИОНАЛЬНАЯ ОСНОВА DSA

25 Обзор

25.1 Определения

Для целей настоящей спецификации Справочника применяются следующие определения:

25.1.1 кооперативное состояние: По отношению ко второму DSA – состояние DSA, для которого образован и не уничтожен экземпляр рабочего связывания.

25.1.2 функциональная основа справочника: Обеспечивает основу, при применении которой из нее могут быть выведены специальные рабочие модели, относящиеся к конкретным аспектам (например, теневое копирование или создание контекста именования) функционирования компонентов Справочника (DSA). Она факторизирует общие элементы. которые присутствуют при любых взаимодействиях между компонентами Справочника.

25.1.3 некооперативное состояние: По отношению ко второму DSA – состояние DSA до образования или после уничтожения экземпляра рабочего связывания.

25.1.4 рабочее связывание: Взаимная договоренность между двумя DSA, которая после введения в действие выражает их "соглашение" вступить впоследствии в какой-либо вид взаимодействия.

25.1.5 образование рабочего связывания: Процесс образования экземпляра рабочего связывания.

25.1.6 экземпляр рабочего связывания: Рабочее связывание специального типа между двумя DSA.

25.1.7 управление рабочим связыванием: Процесс образования, уничтожения или изменения экземпляра рабочего связывания. Такое управление может осуществляться через обмены информацией, определенные настоящими спецификациями Справочника, через обмены, определенные в других спецификациях, или иными средствами.

25.1.8 изменение рабочего связывания: Процесс изменения экземпляра рабочего связывания.

25.1.9 уничтожение рабочего связывания: Процесс уничтожения экземпляра рабочего связывания.

25.1.10 тип рабочего связывания: Конкретный типа рабочего связывания, определенный для конкретной цели, который выражает "соглашение" между двумя DSA участвовать с конкретном виде взаимодействия (например, в теневом копировании).

25.2 Введение

Спецификации Справочника определяют информационные обмены по протоколу прикладного уровня и связанные процедуры DSA, которые обусловливают распределенное функционирование Справочника. В пунктах 25–28 описывается функциональная основа DSA, которая моделирует некоторые общие элементы в таких информационных обменах и процедурах.

Два DSA взаимодействуют на кооперативных началах, поскольку каждый из них, наряду со своим техническим потенциалом для обменов информацией и выполнения связанных с этими обменами процедур, конфигурирован для вступления в определенные виды взаимодействия с другим DSA.

Эти пункты связаны с представлением общей основы для спецификации структуры элементов сотрудничества между двумя DSA.

Одной из задач представления этой основы является обеспечение достаточно общего ее характера для определении всех форм сотрудничества между DSA в данном и последующих изданиях спецификаций Справочника. Эта основа используется в пределах спецификаций Справочника для определения типов теневого копирования и иерархического рабочего связывания.

26 Рабочее связывание

26.1 Общие положения

Данный пункт посвящен определению общей основы, функциональной основы DSA, в рамках которых может быть построено описание характера кооперативного взаимодействия компонентов Справочника (DSA) для достижения взаимно согласованной цели.

Общая основа факторизирует общие функции, которые характеризуют все виды взаимодействия между DSA. При применении функциональной основы DSA к конкретным аспектам кооперативного взаимодействия DSA в результате можно получить конкретные и согласованные спецификации, обусловливающие сокращение общего количества механизмов, которые должен поддерживать тот или иной DSA.

Взаимная договоренность между двумя DSA, которая после введения в действие выражает их "соглашение" вступить впоследствии в какой-либо вид взаимодействия, называется *рабочим связыванием*. Два DSA могут коллективно использовать столько экземпляров рабочего связывания конкретного типа, сколько потребуется.

Функциональная основа DSA обеспечивает общий подход к определению *типа рабочего связывания.* Тип рабочего связывания – это конкретный тип рабочего связывания для какой-либо определенной цели, который выражает "соглашение" между двумя DSA участвовать в конкретном типе взаимодействия (например, в теневом копировании). Это взаимодействие позволяет активизировать одной или другой стороной соглашения операции из строго определенной совокупности.

Два конкретных DSA, заключившие такое "соглашение", совместно используют экземпляр рабочего связывания определенного типа рабочего связывания. Они считаются находящимися в *кооперативном состоянии* этого экземпляра типа рабочего связывания.

До образования или после уничтожения экземпляра рабочего связывания два DSA считаются находящимися в *некооперативном состоянии.*

Управление рабочим связыванием – это процесс образования, уничтожения или изменения экземпляра рабочего связывания. Такое управление может осуществляться через обмены информацией, определенные спецификациями Справочника, через обмены, определенные в других спецификациях, или иными средствами.

Эти общие понятия представлены на рисунке 23.

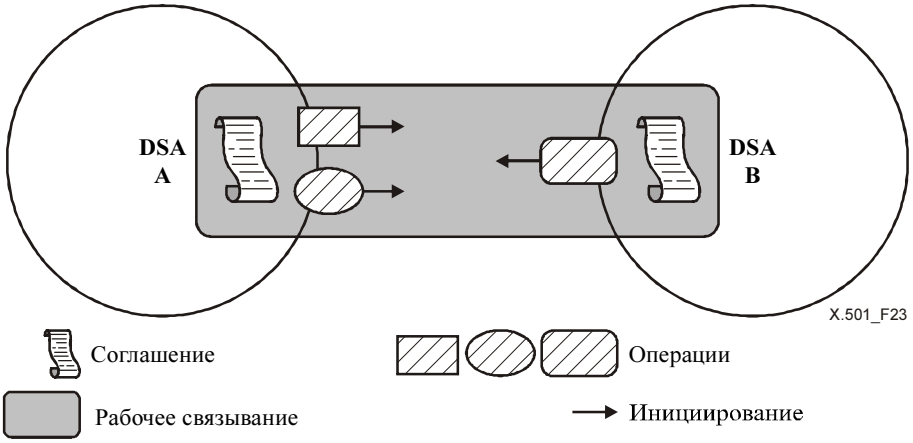


Рисунок 23 – Рабочее связывание

26.2 Применение функциональной основы

Применение функциональной основы DSA для определения типа рабочего связывания задействует следующие базовые элементы:

a) два DSA;

b) "соглашение" об услуге, которую один DSA будет предоставлять другому DSA;

c) совокупность одной или более операций вместе с сопровождающими процедурами, которым должен следовать DSA и посредством которых может быть реализована эта услуга;

d) спецификация взаимодействия DSA, необходимая для управления соглашением.

Взаимоотношение этих базовых элементов выражается рабочим связыванием. Рабочее связывание состоит из совокупности этих базовых элементов, которые участвуют в представлении абстрактного соглашения в технических терминах. Рабочее связывание представляет среду, управляемую "соглашением", в которой один DSA предоставляет определенную услугу другому (и наоборот).

26.2.1 Два DSA

Функциональная основа DSA обеспечивает структуру, в рамках которой могут быть специфицированы взаимодействие одного DSA с другим и процедуры, которые они вследствие этого выполняют.

В рабочем связывании каждый из двух DSA может выполнять идентичные функции, в каковом случае оба DSA могут управлять рабочим связыванием, оба DSA могут активизировать те же операции в отношении друг друга и оба DSA ограничены одной и той же совокупностью процедур. Это называется симметричным рабочим связыванием.

И наоборот, каждый DSA может выполнять в рабочем связывании разные функции, так что к каждому DSA применяются разные совокупности операций и процедур. В управлении рабочим связыванием могут участвовать один или оба DSA. Это называется асимметричным рабочим связыванием.

26.2.2 Соглашение

"Соглашение" – это взаимная договоренность, достигнутая административными органами двух DSA об услуге, которую один DSA будет оказывать другому (и/или наоборот). Порядок первоначального обсуждения "соглашения" административными органами не входит в область действия спецификаций Справочника.

Параметры этого "соглашения" могут быть формализованы путем записи их в DSA с виде типа данных ASN.1 для использования в протокольном обмене в рамках управления рабочим связыванием. Таким образом оба DSA достигают взаимной договоренности об услуге, которую каждый предоставляет другому.

26.2.3 Операции

Операции являются основной средой связи, которую DSA используют для взаимодействия. Для обеспечения согласованной услуги пара DSA должна осуществлять между собой одну или более операций.

Хотя DSA технически может быть способен поддерживать большое количество операций, он может изъявить желание сотрудничать с другим DSA в обработке лишь небольшого количества этих операций или в обработке только тех операций, которые имеют конкретное множество значений для определенных параметров.

Определение типа рабочего связывания требует перечисления операций, которыми можно обмениваться. Оно также разрешает введение ограничений на значения параметров, определенных в рамках этих операций.

26.2.4 Управление соглашением

Основа обеспечивает обобщенные операции для управления экземпляром рабочего связывания. Эти операции предусмотрены для образования, изменения и уничтожения рабочего связывания.

Применение основы для составления спецификации конкретного типа рабочего связывания требует определения инициатора каждой из этих трех операций управления, а также требует определения процедур для каждого образования, изменения и уничтожения. Всякий раз когда какая-либо операция управления применяется к рабочему связыванию специфицированного типа, DSA должен выполнять соответствующую процедуру.

26.3 Состояния сотрудничества

Обобщенная функциональная модель определяет два состояния сотрудничества как управляемые экземпляром конкретного типа рабочего связывания между двумя DSA с позиции одного DSA в отношении другого DSA, и три перехода между этими состояниями. Каждый обозначенный экземпляр типа рабочего связывания, совместно используемый двумя DSA, имеет собственные состояния сотрудничества. Состояниями сотрудничества являются:

a) *Некооперативное состояние*: Конкретный обозначенный экземпляр типа рабочего связывания не образован или уничтожен между двумя DSA. Взаимодействие между двумя DSA (в отношении обозначенного экземпляра типа рабочего связывания) не определено. DSA, к которому обращается другой DSA, с которым первый находится в некооперативном состоянии, может, например, отказаться от какого бы то ни было взаимодействия или он может быть готов обслужить запрос.

b) *Кооперативное состояние*: Это экземпляр рассматриваемого рабочего связывания между двумя DSA. Их кооперативное поведение управляется определением типа рабочего связывания, их конкретными параметрами и связанными процедурами.

Переходы между этими двумя состояниями могут быть активизированы двумя способами: путем взаимодействия по стандартному протоколу или иными средствами.

Взаимодействие между двумя DSA в целях управления экземпляром рабочего связывания (например, для заключения и прекращения действия соглашения о теневом копировании) отличается от возможного взаимодействия между ними, управляемого связыванием (например, взаимодействие для обновления блока репликации).

Состояниями перехода являются следующие:

a) Переход *образование* создает экземпляр рабочего связывания конкретного типа между двумя DSA, в результате чего происходит переход из некооперативного в кооперативное состояние.

b) Переход *уничтожение* разрушает экземпляр рабочего связывания конкретного типа между двумя DSA, в результате чего происходит переход из кооперативного в некооперативное состояние.

c) Переход *изменение* модифицирует параметры экземпляра рабочего связывания между двумя DSA, в результате чего происходит переход из кооперативного в кооперативное состояние.

Эти наиболее общие состояния и переходы графически представлены на рисунке 24.

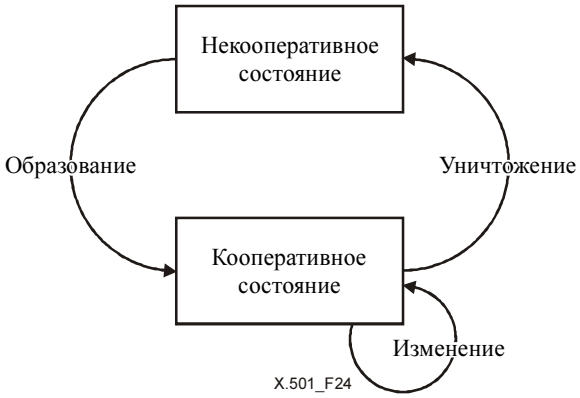


Рисунок 24 – Состояния сотрудничества

27 Спецификация рабочего связывания и управление им

27.1 Спецификация типа рабочего связывания

При применении основы для определения типа рабочего связывания должны быть специфицированы следующие характеристики типа:

a) *Симметрия*

Спецификация соответствующих функций DSA, которые являются сторонами рабочего связывания.

Рабочее связывание может быть симметричным, в каковом случае функции одного DSA являются взаимозаменяемыми с другим и оба DSA проявляют те же виды внешнего взаимодействия. Оно может также быть несимметричным, и в этом случае каждый DSA выполняет особую функцию, и оба DSA проявляют разные виды внешнего взаимодействия. В последнем случае функциональная основа Справочника различает эти две абстрактные функции как "ROLE-A" (функция A) и "ROLE-B" (функция B).

Каждая из этих абстрактных функций – "ROLE-A" и "ROLE-B" – должна быть связана с конкретной функцией с определенной семантикой (например, "ROLE-A" как теневой поставщик, "ROLE-B" как теневой потребитель).

b) *Соглашение*

Определение семантики и представление компонентов "соглашения". Эта информация параметризует конкретный экземпляр рабочего связывания между двумя DSA.

c) *Инициатор*

Определение того, какой из двух абстрактных функций – "ROLE-A" и "ROLE-B" – разрешено инициировать образование или уничтожение рабочего связывания данного типа.

d) *Процедуры управления*

Совокупность процедур, которые должен выполнять DSA, когда рабочее связывание данного типа образуется, изменяется или уничтожается.

e) *Тип обозначения*

Обозначает тип взаимодействия DSA, который определяется рабочим связыванием. Эти идентификаторы являются значениями идентификатора объекта.

f) *Контексты применения, операции и процедуры*

Обозначает совокупность контекстов применения, операции которых (или их подгруппа) могут использоваться в течение кооперативной фазы рабочего связывания.

Для каждой операции, на которую указывает тип рабочего связывания, требуется описание процедур, которые должен выполнять DSA при активизации этой операции (это может быть выполнено путем ссылки на какую-либо иную часть настоящих спецификаций Справочника).

Для рабочего связывания, управление которым должно осуществляться с помощью обобщенных операций управления рабочим связыванием, представленных в данном пункте, тип связывания должен специфицироваться с использованием трех информационных классов объектов **OPERATIONAL-BINDING**, **OP-BINDING-COOP** и **OP-BIND-ROLE**, которые определены в этом пункте.

27.2 Управление рабочим связывание

В целом, управление рабочим связыванием требует прежде всего образования экземпляра рабочего связывания. За этим может факультативно следовать одно или более изменений некоторых или всех параметров первоначального соглашения и в заключение может включать уничтожение экземпляра рабочего связывания. Детально порядок управления экземпляром определяется в процессе определения типа рабочего связывания. Определение типа требует спецификации:

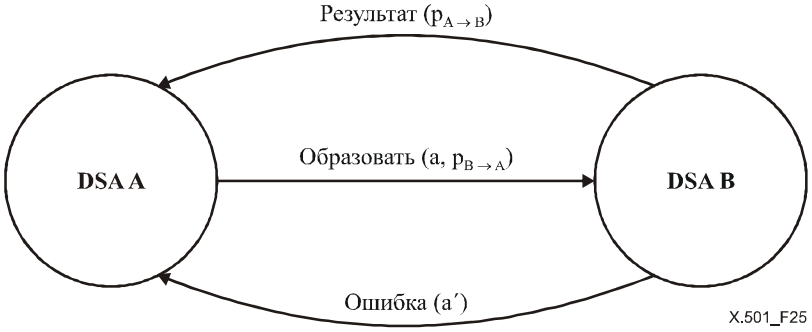
a) инициатора каждой из операций управления (это может быть один, оба или ни один из двух DSA);

b) параметров каждой из операций управления; и

c) процедур, которым должен следовать каждый DSA для выполнения каждой из операций управления.

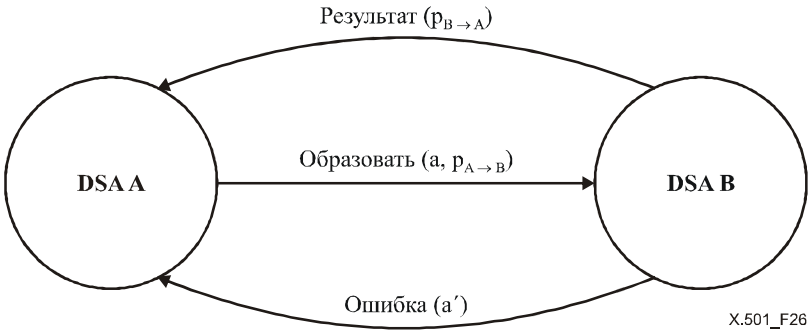
В процессе образования экземпляра рабочего связывания создается идентификатор экземпляра рабочего связывания (**id** связывания). Этот идентификатор в сочетании с выделенными именами двух DSA, участвующих в рабочем связывании, образует уникальный идентификатор для этого экземпляра связывания. Все операции управления, следующие после образования экземпляра рабочего связывания, должны использовать **id** связывания для обозначения экземпляра рабочего связывания, который должен быть изменен или уничтожен.

Инициатор операции образования всегда передает параметры "соглашения" второму DSA. Кроме того, инициатор может также передавать некоторые параметры образования, характерные для его функции в рабочем связывании. Если отвечающий DSA готов вступить в рабочее связывание, он может вернуть в результате параметры образования, характерные для его функции. Если отвечающий DSA не готов вступать в рабочее связывание, он должен вернуть ошибку, которая может необязательно содержать соглашение с совокупностью пересмотренных параметров. Это показано для случаев, когда инициаторами операции образования являются функция A, рисунок 25, и функция B, рисунок 26.



a Соглашение
b Параметр образования

Рисунок 25 – DSA с функцией A инициирует образование



a Соглашение
b Параметр образования

Рисунок 26 – DSA с функцией B инициирует образование

27.3 Шаблоны спецификации рабочего связывания

Для определения конкретного типа рабочего связывания в качестве шаблонов могут использоваться следующие три информационные классы объектов ASN.1. Они делают возможным описание с использованием языка ASN.1 тех частей типа рабочего связывания, которые могут быть формализованы. Другие аспекты типа рабочего связывания, такие как процедуры, которым должен следовать DSA при образовании или уничтожении рабочего связывания, должны специфицироваться какими-либо иными средствами (это можно сделать аналогично тому, как выполняется информационное описание процедур DSA в процессе разрешения имен, которое описано в Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4).

27.3.1 Информационный класс объектов рабочего связывания

```
OPERATIONAL-BINDING ::= CLASS {
        &Agreement,
        &Cooperation        OP-BINDING-COOP,
        &both               OP-BIND-ROLE OPTIONAL,
        &roleA              OP-BIND-ROLE OPTIONAL,
        &roleB              OP-BIND-ROLE OPTIONAL,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        AGREEMENT                       &Agreement
        APPLICATION CONTEXTS            &Cooperation
        [ SYMMETRIC                     &both ]
        [ ASYMMETRIC
                [ ROLE-A                &roleA ]
                [ ROLE-B                &roleB ] ]

        ID                              &id }
```

Информационный класс объектов **OPERATIONAL-BINDING** служит шаблоном спецификации для типа рабочего связывания. С тем чтобы упростить использование этого класса в качестве шаблона, для него определяется переменная нотация. Между определением типа рабочего связывания и полями переменной нотации существует следующее соответствие:

a) Типом ASN.1 параметра аргумента, который используется для этого типа рабочего связывания, является тип, на который указывает поле **AGREEMENT**.

b) Контексты применения и операции этих контекстов применения, которые используются в течение кооперативной фазы экземпляра рабочего связывания определенного типа, перечисляются в поле **APPLICATION-CONTEXTS**. Выбираются все операции перечисленного контекста применения, если не присутствует поле **APPLIES TO**, за которым следует перечень ссылок на операции, которые выбираются из контекста применения. Этим перечнем является множество класса объектов, составленное из экземпляров информационного класса объектов **OPERATION**.

c) Класс рабочего связывания определяется полями **SYMMETRIC** или **ASYMMETRIC**. В случае симметричного рабочего связывания после термина **SYMMETRIC** следует один информационный объект класса **OP-BIND-ROLE**, который действителен для обеих функций рабочего связывания. В случае несимметричного рабочего связывания после **ASYMMETRIC** следуют два информационных объекта класса **OP-BIND-ROLE**, на один из них указывает подполе **ROLE-A**, а на другой – подполе **ROLE-B**.

d) Значение идентификатора объекта, которое служит для обозначения этого типа рабочего связывания, определяется полем **ID**.

27.3.2 Информационный класс объектов сотрудничества для рабочего связывания

```
OP-BINDING-COOP ::= CLASS {
        &applContext        APPLICATION-CONTEXT,
        &Operations         OPERATION OPTIONAL }
WITH SYNTAX {
        &applContext
        [ APPLIES TO        &Operations ] }
```

Информационный класс объектов **OP-BINDING-COOP** служит в качестве шаблона спецификации для обозначения операций именованного контекста применения, некоторые аспекты которого определяются рабочим связыванием. Экземпляр этого класса является значимым только в пределах контекста конкретного типа рабочего связывания. Переменная нотация, которая определяется для этого класса, упрощает его использование в качестве шаблона. Между определением типа рабочего связывания и полями переменной нотации существует следующее соответствие:

a) Поле **applContext** обозначает контекст применения, все или часть операций которого каким-либо образом определяются рабочим связыванием.

b) Поле **APPLIES TO**, если оно присутствует, обозначает конкретные операции, к которым применяется это рабочее связывание. Если это поле отсутствует, рабочее связывание применяется ко всем операциям контекста применения.

27.3.3 Информационный класс объектов функции рабочего связывания

```
OP-BIND-ROLE ::= CLASS {
        &establish                  BOOLEAN   DEFAULT FALSE,
        &EstablishParam                       OPTIONAL,
        &modify                     BOOLEAN   DEFAULT FALSE,
        &ModifyParam                          OPTIONAL,
        &terminate                  BOOLEAN   DEFAULT FALSE,
        &TerminateParam                       OPTIONAL }
WITH SYNTAX {
        [ ESTABLISHMENT-INITIATOR           &establish ]
        [ ESTABLISHMENT-PARAMETER           &EstablishParam ]
        [ MODIFICATION-INITIATOR            &modify ]
        [ MODIFICATION-PARAMETER            &ModifyParam ]
        [ TERMINATION-INITIATOR             &terminate ]
        [ TERMINATION-PARAMETER             &TerminateParam ] }
```

Информационный класс объектов **OP-BIND-ROLE** служит шаблоном спецификации функций типа рабочего связывания. Экземпляр этого класса является значимым только в пределах контекста конкретного типа рабочего связывания. Для упрощения использования этого класса в качестве шаблона для него определяется переменная нотация. Между определением типа рабочего связывания и полями переменной нотации существует следующее соответствие:

a) Поле **ESTABLISHMENT INITIATOR** указывает, может ли DSA, принимающий определенную функцию, инициировать образование рабочего связывания конкретного типа.

b) Поле **ESTABLISHMENT PARAMETER** определяет тип ASN.1, которым обменивается DSA, принимающий определенную функцию, при образовании экземпляра типа рабочего связывания.

c) Поле **MODIFICATION INITIATOR** указывает, может ли DSA, принимающий определенную функцию, инициировать изменение рабочего связывания конкретного типа.

d) Поле **MODIFICATION PARAMETER** определяет тип ASN.1, которым обменивается DSA, принимающий определенную функцию, при изменении экземпляра типа рабочего связывания.

e) Поле **TERMINATION INITIATOR** указывает, может ли DSA, принимающий определенную функцию, уничтожить образование рабочего связывания конкретного типа.

f) Поле **TERMINATION PARAMETER** определяет тип ASN.1, которым обменивается DSA, принимающий определенную функцию, при уничтожении экземпляра типа рабочего связывания.

28 Операции для управления рабочим связыванием

В данном пункте определяется совокупность операций. которые могут использоваться для образования, изменения и уничтожения рабочего связывания различных типов. Эти операции являются обобщенными в том смысле, что они могут применяться для управлением любым типом рабочего связывания. Для спецификации этих операций используются определения, предоставляемые для известного типа рабочего связывания при применении в качестве шаблона информационного класса объектов **OPERATIONAL-BINDING**.

ПРИМЕЧАНИЕ. – При использовании этого средства может осуществляться управление произвольными типами рабочего связывания. Эти операции (совместно со связанным контекстом применения) образуют средство расширяемости в аспекте взаимодействия DSA. В будущем могут определяться новые типы рабочего связывания, что позволит расширить функциональную взаимосвязь между DSA.

28.1 Определение контекста применения

Совокупность операций для управления экземплярами рабочего связывания может использоваться для определения контекста применения следующими двумя способами:

1) Контекст применения может содержать только операции для управления рабочим связыванием. Контекст применения для обобщенного управления рабочим связыванием определяется в Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5.

   Операции, которые могут заменяться в течение кооперативной фазы рабочего связывания, образуют один или более отдельных контекстов применения.

2) Совокупность операций может импортироваться в модуль, который используется для определения специального контекста применения. Операции управления рабочим связыванием могут затем использоваться совместно с операциями кооперативной фазы в пределах единичного контекста применения.

   ПРИМЕЧАНИЕ. – Первый подход эффективен в случае, если какой-либо специализированный компонент DSA желает использовать ассоциацию исключительно для управления совокупностью рабочих связываний этого DSA и не готов к принятию каких бы то ни было операций, определенных для кооперативной фазы (например, операций обновления теневой копии updateShadow).

28.2 Операция образования рабочего связывания

Операция образования рабочего связывания позволяет создавать экземпляр рабочего связывания заранее определенного типа между двумя DSA. Это достигается путем передачи параметров образования и условий соглашения, которые указаны в определении типа рабочего связывания. Аргументы операции могут быть подписаны (см. п. 17.3) запросчиком. Если требуется, подписывать результаты может отвечающая сторона.

В случае симметричного рабочего связывания любой из двух DSA может быть инициатором образования экземпляра рабочего связывания заранее определенного типа. .

В случае асимметричного рабочего связывания образование рабочего связывания выполняет DSA, принимающий на себя функцию "ROLE-A" или "ROLE-B", в зависимости от конкретного определения типа рабочего связывания.

```
establishOperationalBinding  OPERATION  ::=  {
        ARGUMENT        EstablishOperationalBindingArgument
        RESULT          EstablishOperationalBindingResult
        ERRORS          { operationalBindingError | securityError | serviceError }
        CODE            id-op-establishOperationalBinding }

EstablishOperationalBindingArgument  ::=  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType             [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID               [1]     OperationalBindingID OPTIONAL,
        accessPoint             [2]     AccessPoint,
                -- симметричное, инициирует Role A или инициирует Role B --
        initiator CHOICE {
                symmetric                       [3]     OPERATIONAL-BINDING.&both.&EstablishParam
                                                        ({OpBindingSet}{@bindingType}),
                roleA-initiates                 [4]     OPERATIONAL-BINDING.&roleA.&EstablishParam
                                                        ({OpBindingSet}{@bindingType}),
                roleB-initiates                 [5]     OPERATIONAL-BINDING.&roleB.&EstablishParam
                                                        ({OpBindingSet}{@bindingType}) } OPTIONAL,
        agreement               [6]     OPERATIONAL-BINDING.&Agreement
                                                ({OpBindingSet}{@bindingType}),
        valid                   [7]     Validity DEFAULT { },
        securityParameters      [8]     SecurityParameters OPTIONAL } }
```

```
OpBindingSet        OPERATIONAL-BINDING ::= {
        shadowOperationalBinding |
        hierarchicalOperationalBinding |
        nonSpecificHierarchicalOperationalBinding }

OperationalBindingID ::= SEQUENCE {
        identifier      INTEGER,
        version         INTEGER }
```

Компонент **bindingType** объявляет, какой тип рабочего связывания должен быть создан. Типы рабочего связывания определяются с помощью используемого в качестве шаблона информационного класса объектов **OPERATIONAL-BINDING**, который присваивает типу рабочего связывания значение идентификатора объекта. Тип связывания **bindingType** берется из поля **ID** одного из экземпляров типа рабочего связывания, на который указывает **OpBindingSet**. Эта совокупность является параметром параметризованного типа аргумента **EstablishOperationalBindingArgument**.

DSA-инициатор может присвоить обозначение экземпляру рабочего связывания через компонент **bindingID**. Если **bindingID** отсутствует в пределах аргумента операции, отвечающий DSA должен присвоить идентификатор ID экземпляру рабочего связывания и вернуть его в компоненте **bindingID** результата **establishOperationalBindingResult**. В любом случае при образовании рабочего связывания оба компонента **identifier** и **version** значения **OperationalBindingID** должны быть присвоены и выданы выполняющим присвоение DSA.

Компонент **accessPoint** описывает точку доступа инициатора для последующего взаимодействия.

Функция, которую принимает на себя DSA, выдающий операцию образования рабочего связывания, указывается записью **CHOICE** с опциями **symmetric**, **roleA-initiates** и **roleB-initiates**. Опция **CHOICE** управляет конкретными параметрами образования, используемыми инициирующим и отвечающим DSA. Семантика функций определяется как часть определения типа рабочего связывания. Запись **CHOICE** на языке ASN.1 определяется **ESTABLISHMENT PARAMETER** информационного класса объектов **OP-BIND-ROLE** инициатора, используемого в качестве шаблона. Запись **CHOICE** опускается, если образование этого типа рабочего связывания не требует от инициатора параметров образования.

Компонент **agreement** содержит условия соглашения, управляющие экземпляром рабочего связывания. Его фактический контент зависит от типа рабочего связывания, которое должно быть образовано. Запись на языке ASN.1 для этого параметра определяется полем **AGREEMENT** используемого в качестве шаблона информационного класса объектов **OPERATIONAL-BINDING** этого типа рабочего связывания.

Продолжительность существования экземпляра рабочего связывания определяется опцией **valid**. Время начала существования экземпляра рабочего связывания определяется в **validFrom**, а время, когда экземпляр рабочего связывания должен быть уничтожен, передается в **validUntil**.

```
Validity ::= SEQUENCE {
        validFrom               [0]     CHOICE {
                now             [0]     NULL,
                time            [1]     Time } DEFAULT now : NULL,
        validUntil              [1]     CHOICE {
                explicitTermination             [0]     NULL,
                time                            [1]     Time } DEFAULT explicitTermination : NULL }

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

До использования значения **Time** в какой-либо операции сравнения и если синтаксис **Time** выбран как тип **UTCTime**, значение двузначного поля года должно быть приведено к четырехзначному значению года следующим образом:

- если диапазон двузначного значения 00–49 включительно, к значению следует прибавить 2000.
- если диапазон двузначного значения 50–99 включительно, к значению следует прибавить 1999.

Использование **GeneralizedTime** может препятствовать межсетевому взаимодействию с реализациями, не осведомленными о возможности выбора либо **UTCTime**, либо **GeneralizedTime**. Определение случаев использования **GeneralizedTime** является обязанностью лиц, определяющих домены, в которых будет использоваться настоящая спецификация Справочника, например групп по составлению профилей. Ни в каком случае опция **UTCTime** не должна использоваться для представления даты после 2049 года.

Если операция образования рабочего связывания выполнена успешно, возвращается следующий результат, который может быть подписан (см. п. 17.3) отвечающей стороной.

```
EstablishOperationalBindingResult ::= OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID   [1]     OperationalBindingID OPTIONAL,
        accessPoint [2]     AccessPoint,
        -- симметричное, отвечает Role A или отвечает Role B
        initiator CHOICE {
                symmetric           [3]     OPERATIONAL-BINDING.&both.&EstablishParam
                                            ({OpBindingSet}{@bindingType}),
                roleA-replies       [4]     OPERATIONAL-BINDING.&roleA.&EstablishParam
                                            ({OpBindingSet}{@bindingType}),
                roleB-replies       [5]     OPERATIONAL-BINDING.&roleB.&EstablishParam
                                            ({OpBindingSet}{@bindingType}) } OPTIONAL,
        COMPONENTS OF  CommonResultsSeq } }
```

Компонент **bindingType** содержится в результате для указания типа рабочего связывания для использования в пределах элемента **CHOICE**. Его значение то же, что и предоставленное инициатором образования, и берется из поля **ID** одного из экземпляров типа рабочего связывания, на которые указывает **OpBindingSet**. Эта совокупность является параметром параметризованного типа результата **EstablishOperationalBindingResult**.

Обозначение образованного экземпляра рабочего связывания может быть возвращено в **bindingID**. Оно должно использоваться для обозначения этого экземпляра рабочего связывания в любой последующей операции изменения или уничтожения рабочего связывания, а также может использоваться в любой другой операции, которая выполняется в течение кооперативной фазы образованного экземпляра рабочего связывания.

Компонент **accessPoint** описывает точку доступа отвечающей стороны для последующего взаимодействия.

DSA-инициатор может присвоить обозначение экземпляру рабочего связывания через компонент **bindingID**. Если **bindingID** отсутствует в пределах аргумента операции, отвечающий DSA должен присвоить идентификатор ID этому экземпляру рабочего связывания и вернуть его в компонент **bindingID** результата **establishOperationalBindingResult**.

Функция, которую принимает на себя DSA, выдающий операцию образования рабочего связывания, указывается записью **CHOICE** с опциями **symmetric**, **roleA-initiates** и **roleB-initiates**. Семантика функций определяется как часть определения типа рабочего связывания. Запись **CHOICE** на языке ASN.1 определяется **ESTABLISHMENT PARAMETER** используемого в качестве шаблона информационного класса объектов **OP-BIND-ROLE** отвечающей стороны. Запись **CHOICE** опускается, если образование этого типа рабочего связывания не требует от отвечающей стороны параметров образования.

28.3 Операция изменения рабочего связывания

Операция изменения рабочего связывания используется для изменения образованного рабочего связывания. Право на осуществление изменения указывается полем(ями) **MODIFICATION INITIATOR** в определении типа рабочего связывания с использованием в качестве шаблонов информационных классов объектов **OP-BIND-ROLE** и **OPERATIONAL-BINDING**.

Компонентами рабочего связывания, которые могут быть изменены, являются контент аргумента для рабочего связывания и период его действия. Кроме того, параметр изменения может определяться инициирующей функцией. Аргументы операции могут быть подписаны (см. п. 17.3) запросчиком. Если требуется, отвечающая сторона может подписать результат.

```
modifyOperationalBinding  OPERATION  ::= {
        ARGUMENT  ModifyOperationalBindingArgument
        RESULT    ModifyOperationalBindingResult
        ERRORS    { operationalBindingError | securityError | serviceError }
        CODE      id-op-modifyOperationalBinding }

ModifyOperationalBindingArgument  ::= OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType         [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID           [1]     OperationalBindingID,
        accessPoint         [2]     AccessPoint OPTIONAL,
        -- симметричное, инициирует Role A или инициирует Role B --
        initiator CHOICE {
                symmetric   [3]     OPERATIONAL-BINDING.&both.&ModifyParam
                                    ({OpBindingSet}{@bindingType}),
                roleA-initiates     [4]     OPERATIONAL-BINDING.&roleA.&ModifyParam
                                    ({OpBindingSet}{@bindingType}),
                roleB-initiates [5] OPERATIONAL-BINDING.&roleB.&ModifyParam
                                    ({OpBindingSet}{@bindingType}) } OPTIONAL,
        newBindingID [6]    OperationalBindingID,
        newAgreement        [7]     OPERATIONAL-BINDING.&Agreement
```

```
                                ({OpBindingSet}{@bindingType})  OPTIONAL,
    valid               [8]     Validity OPTIONAL,
    securityParameters
                        [9]     SecurityParameters OPTIONAL } }
```

Компонент **bindingType** объявляет, какой тип рабочего связывания должен быть изменен. Тип связывания **bindingType** берется из поля **ID** одного из экземпляров типа рабочего связывания, на который указывает **OpBindingSet**. Эта совокупность является параметром параметризованного типа аргумента **ModifyOperationalBindingArgument**.

Обозначение подлежащего изменению экземпляра рабочего связывания передается идентификатором **bindingID**. Исправленный идентификатор экземпляра рабочего связывания дается **newBindingID**. Компонент **version** нового идентификатора **newBindingID** должен быть больше идентификатора **bindingID**.

Операционный компонент **accessPoint** присутствует, если должна быть изменена точка доступа инициатора для последующего взаимодействия.

Функция, которую принимает на себя DSA, выдающий операцию изменения рабочего связывания, указывается записью **CHOICE** с опциями **symmetric**, **roleA-initiates** и **roleB-initiates**. Семантика функций определяется как часть определения типа рабочего связывания. Запись **CHOICE** на языке ASN.1 определяется параметром изменения **MODIFICATION PARAMETER** информационного класса объектов **OP-BIND-ROLE** инициатора, используемого в качестве шаблона. Запись **CHOICE** опускается, если изменение этого типа рабочего связывания не требует от инициатора параметров изменения.

Компонент **newAgreement**, если он присутствует, содержит измененные условия соглашения, управляющие экземпляром рабочего связывания. Запись на языке ASN.1 для этого параметра определяется полем **AGREEMENT** используемого в качестве шаблона информационного класса объектов **OPERATIONAL-BINDING** этого типа рабочего связывания. Если компонент **newAgreement** не присутствует, эта операция не меняет параметры соглашения.

Для указания измененного периода действия пересмотренного соглашения может использоваться необязательный компонент **valid.** Если компонент **valid** отсутствует, предполагается, что компонент **validFrom** имеет значение **now**, а компонент **validUntil** считается неизмененным. Если компонент **validFrom** присутствует и указывает на момент времени в будущем, текущее соглашение остается в силе до этого времени

Если операция изменения рабочего связывания выполнена успешно, возвращается следующий результат, который может быть подписан (см. п. 17.3) отвечающей стороной:

```
ModifyOperationalBindingResult ::= CHOICE {
    null            [0]     NULL,
    protected       [1]     OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        newBindingID                OperationalBindingID,
        bindingType                 OPERATIONAL-BINDING.&id ({OpBindingSet}),
        newAgreement                    OPERATIONAL-BINDING.&Agreement
                                            ({OpBindingSet}{@.bindingType}),
        valid                       Validity OPTIONAL,
        COMPONENTS OF  CommonResultsSeq } }
```

Отвечающий DSA не может возвращать инициатору изменения параметр изменения, определенный для его функции.

28.4 Операция уничтожения рабочего связывания

Операция уничтожения рабочего связывания используется для запроса уничтожения образованного экземпляра рабочего связывания. Право на запрос уничтожения указывается полем(ями) **TERMINATION INITIATOR** в определении типа рабочего связывания с использованием в качестве шаблонов информационных классов объектов **OP-BIND-ROLE** и **OPERATIONAL-BINDING**. Аргументы операции могут быть подписаны (см. п. 17.3) запросчиком. Если требуется, отвечающая сторона может подписывать результаты.

```
terminateOperationalBinding  OPERATION  ::= {
    ARGUMENT        TerminateOperationalBindingArgument
    RESULT          TerminateOperationalBindingResult
    ERRORS          { operationalBindingError | securityError | serviceError }
    CODE            id-op-terminateOperationalBinding }

TerminateOperationalBindingArgument  ::=  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
    bindingType             [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
    bindingID               [1]     OperationalBindingID,
    -- симметричное, инициирует Role A или инициирует Role B --
```

```
initiator CHOICE {
        symmetric                [2]  OPERATIONAL-BINDING.&both.&TerminateParam
                                      ({OpBindingSet}{@bindingType}),
        roleA-initiates          [3]  OPERATIONAL-BINDING.&roleA.&TerminateParam
                                      ({OpBindingSet}{@bindingType}),
        roleB-initiates          [4]  OPERATIONAL-BINDING.&roleB.&TerminateParam
                                      ({OpBindingSet}{@bindingType})} OPTIONAL,
terminateAt          [5]  Time OPTIONAL,
securityParameters   [6]  SecurityParameters OPTIONAL } }
```

Компонент **bindingType** объявляет, какой тип рабочего связывания должен быть уничтожен. Тип связывания **bindingType** берется из поля **ID** одного из экземпляров типа рабочего связывания, на который указывает **OpBindingSet**. Эта совокупность является параметром параметризованного типа аргумента **ModifyOperationalBindingArgument**.

Обозначение подлежащего уничтожению экземпляра рабочего связывания передается идентификатором **bindingID**. Компонент **version**, присутствующий в **bindingID**, игнорируется.

Функция, которую принимает на себя DSA, выдающий операцию уничтожения рабочего связывания, указывается записью **CHOICE** с опциями **symmetric**, **roleA-initiates** и **roleB-initiates**. Семантика функции определяется как часть определения типа рабочего связывания. Запись **CHOICE** на языке ASN.1 определяется параметром уничтожения **TERMINATION PARAMETER** информационного класса объектов **OP-BIND-ROLE** инициатора, используемого в качестве шаблона. Запись **CHOICE** опускается, если уничтожение этого типа рабочего связывания не требует от инициатора параметров уничтожения.

Если рабочее связывание не должно быть уничтожено немедленно, может быть определено время задержки уничтожения **terminateAt**.

Если операция уничтожения рабочего связывания выполнена успешно, возвращается следующий результат, который может быть подписан (см. п. 17.3) отвечающей стороной:

```
TerminateOperationalBindingResult ::= CHOICE {
        null                 [0]  NULL,
        protected            [1]  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
                bindingID                    OperationalBindingID,
                bindingType          OPERATIONAL-BINDING.&id ({OpBindingSet}),
                terminateAt                  GeneralizedTime OPTIONAL,
                COMPONENTS OF        CommonResultsSeq } } }
```

Отвечающий DSA не может возвращать инициатору уничтожения параметр уничтожения, определенный для его функции.

28.5 Ошибка рабочего связывания

Ошибка рабочего связывания сообщает о проблеме, относящейся к использованию операций для управления рабочим связыванием. Параметр ошибки *error* может быть подписан (см. п. 17.3) отвечающей стороной.

```
operationalBindingError ERROR ::= {
        PARAMETER        OPTIONALLY-PROTECTED-SEQ {

                                 OpBindingErrorParam }
        CODE             id-err-operationalBindingError }

OpBindingErrorParam ::= SEQUENCE {
        problem                  [0]  ENUMERATED {
                invalidID                    (0),
                duplicateID                  (1),
                unsupportedBindingType       (2),
                notAllowedForRole            (3),
                parametersMissing            (4),
                roleAssignment               (5),
                invalidStartTime             (6),
                invalidEndTime               (7),
                invalidAgreement             (8),
                currentlyNotDecidable        (9),
                modificationNotAllowed       (10) },
        bindingType          [1]  OPERATIONAL-BINDING.&id ({OpBindingSet}) OPTIONAL,
        agreementProposal    [2]  OPERATIONAL-BINDING.&Agreement
                                  ({OpBindingSet}{@bindingType}) OPTIONAL,
        retryAt              [3]  Time OPTIONAL,
        COMPONENTS OF             CommonResultsSeq }
```

Значения **problem** имеют следующее содержание:

a) **invalidID**: Идентификатор рабочего связывания, переданный в запросе, неизвестен получающему запрос DSA или находится в неверном состоянии для запрошенной операции.

b) **duplicateID**: Идентификатор рабочего связывания, переданный в запросе образования, уже существует у отвечающей стороны. Причиной этого может быть предшествующая попытка образования экземпляра рабочего связывания, при которой результат был потерян, а инициатор повторил запрос на образование.

c) **unsupportedBindingType**: Запрошенный тип рабочего связывания не поддерживается данным DSA.

d) **notAllowedForRole**: Запрошена операция управления экземпляром рабочего связывания, не разрешенная для функции, которую выполняет запросчик (например, операция уничтожения рабочего связывания выдана DSA, имеющему функцию, которой не разрешено инициирование уничтожения экземпляра рабочего связывания).

e) **parametersMissing**: Отсутствуют какие-либо требуемые параметры образования или уничтожения, которые определены для данного типа рабочего связывания.

f) **roleAssignment**: Запрошенное присвоение функции для экземпляра асимметричного рабочего связывания не принимается.

g) **invalidStartTime**: Специфицированное время начала для экземпляра рабочего связывания не принимается.

h) **invalidEndTime**: Специфицированное время уничтожения для экземпляра рабочего связывания не принимается.

i) **invalidAgreement**: Условия соглашения для запрошенного экземпляра рабочего связывания не принимаются. Условия соглашения, которые были бы приняты отвечающим DSA, могут быть возвращены в **agreementProposal**.

j) **currentlyNotDecidable**: DSA не способен принять решение в режиме он-лайн об образовании или изменении запрошенного экземпляра рабочего связывания. Время возможного повторения запроса может быть выдано в **retryAt**.

k) **modificationNotAllowed**: Операция изменения рабочего связывания отклоняется, поскольку изменение не разрешено для данного экземпляра связывания.

Компонент **bindingType** должен быть таким же, как и переданный активизирующей стороной неудачно завершившейся операцией управления рабочим связыванием.

Компонент **agreementProposal** должен использоваться только для ответа на запрос образования **EstablishOperationalBinding** для предложения пересмотренной совокупности параметров соглашения, как описано в п. 28.2.

Компонент **retryAt** должен использоваться только совместно со значением **currentlyNotDecidable** параметра **problem** для указания времени возможного повтора запроса **EstablishOperationalBinding** или **ModifyOperationalBinding**.

Компонент **CommonResultsSeq** (см. п. 7.4 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3) включает компонент **SecurityParameters**. Компонент **SecurityParameters** (см. п. 7.10 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3) должен включаться в **CommonResultsSeq**, если параметр ошибки должен быть подписан отвечающей стороной.

28.6 Привязывание и развязывание управления рабочим связыванием

Операции привязывания управления рабочим связыванием DSA и развязывания управления рабочим связыванием DSA, определенные в пп. 28.6.1 и 28.6.2, используются DSA в начале и конце конкретного периода действий по управлению рабочим связыванием.

Защита для **dSAOperationalBindingManagementBind** и **dSAOperationalBindingManagementUnbind** должна быть эквивалентной защите, применяемой в отношении операций **DSABind** и **DSAUnbind**.

ПРИМЕЧАНИЕ. – Требуемые для аутентификации полномочия могут переноситься обменным сервисным элементом защиты (см. Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5), и в этом случае они не присутствуют в аргументах или результатах привязывания.

28.6.1 Операция привязывания управления рабочим связыванием DSA

Операция привязывания управления рабочим связыванием DSA используется для того, чтобы начать период управления рабочим связыванием.

dSAOperationalBindingManagementBind OPERATION ::= directoryBind

Компоненты **dSAOperationalManagementBind** идентичны их эквивалентам в **directoryBind** (см. Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3) со следующими отличиями.

ПРИМЕЧАНИЕ. – Требуемые для аутентификации полномочия могут переноситься обменным сервисным элементом защиты (см. Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5), и в этом случае они не присутствуют в аргументах или результатах привязывания.

28.6.1.1 Удостоверения инициатора

Удостоверение **Credentials** параметра **DirectoryBindArgument** позволяет отправлять отвечающему DSA информацию, идентифицирующую AE-Title инициирующего DSA. AE-title должен быть в форме выделенного имени Справочника.

28.6.1.2 Удостоверения отвечающей стороны

Удостоверение **Credentials** параметра **DirectoryBindResult** позволяет отправлять инициирующему DSA информацию, идентифицирующую AE-Title отвечающего DSA. AE-title должен быть в форме выделенного имени Справочника.

28.6.2 Развязывание управления рабочим связыванием DSA

Развязывание в конце периода обеспечения управления рабочим связыванием определяется для среды OSI в пп. 7.6.4 и 7.6.5 Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5 и для среды TCP/IP в п. 9.3.2 Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5.

Приложение A

Использование идентификатора объекта

(|)

Данное приложение является документальным описанием верхних ветвей поддерева идентификатора объекта, в котором размещаются все идентификаторы объектов, присвоенные в настоящих спецификациях Справочника. Это делается с помощью модуля на языке ASN.1, называемого **UsefulDefinitions**, в котором все не являющиеся листьями узлы в поддереве являются присвоенными именами.

```
UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

ID          ::=   OBJECT IDENTIFIER

ds    ID    ::=   {joint-iso-itu-t ds(5)}

-- категории информационного объекта --

module                          ID    ::=   {ds 1}
serviceElement                  ID    ::=   {ds 2}
applicationContext              ID    ::=   {ds 3}
attributeType                   ID    ::=   {ds 4}
attributeSyntax                 ID    ::=   {ds 5}
objectClass                     ID    ::=   {ds 6}
-- attributeSet                 ID    ::=   {ds 7}
algorithm                       ID    ::=   {ds 8}
abstractSyntax                  ID    ::=   {ds 9}
-- object                       ID    ::=   {ds 10}
-- port                         ID    ::=   {ds 11}
dsaOperationalAttribute         ID    ::=   {ds 12}
matchingRule                    ID    ::=   {ds 13}
knowledgeMatchingRule           ID    ::=   {ds 14}
nameForm                        ID    ::=   {ds 15}
group                           ID    ::=   {ds 16}
subentry                        ID    ::=   {ds 17}
operationalAttributeType        ID    ::=   {ds 18}
operationalBinding              ID    ::=   {ds 19}
schemaObjectClass               ID    ::=   {ds 20}
schemaOperationalAttribute      ID    ::=   {ds 21}
administrativeRoles             ID    ::=   {ds 23}
accessControlAttribute          ID    ::=   {ds 24}
--rosObject                     ID    ::=   {ds 25}
--contract                      ID    ::=   {ds 26}
--package                       ID    ::=   {ds 27}
accessControlSchemes            ID    ::=   {ds 28}
certificateExtension            ID    ::=   {ds 29}
managementObject                ID    ::=   {ds 30}
attributeValueContext           ID    ::=   {ds 31}
-- securityExchange             ID    ::=   {ds 32}
idmProtocol                     ID    ::=   {ds 33}
problem                         ID    ::=   {ds 34}
notification                    ID    ::=   {ds 35}
matchingRestriction             ID    ::=   {ds 36}  -- Сейчас в настоящей спецификации
                                                     -- не определено ни одного
controlAttributeType            ID    ::=   {ds 37}
keyPurposes                     ID    ::=   {ds 38}
```

-- модули --

```
usefulDefinitions                ID    ::=    {module usefulDefinitions(0) 5}
informationFramework             ID    ::=    {module informationFramework(1) 5}
directoryAbstractService         ID    ::=    {module directoryAbstractService(2) 5}
distributedOperations            ID    ::=    {module distributedOperations(3) 5}
-- protocolObjectIdentifiers     ID    ::=    {module protocolObjectIdentifiers(4) 5}
selectedAttributeTypes           ID    ::=    {module selectedAttributeTypes(5) 5}
selectedObjectClasses            ID    ::=    {module selectedObjectClasses(6) 5}
authenticationFramework          ID    ::=    {module authenticationFramework(7) 5}
algorithmObjectIdentifiers       ID    ::=    {module algorithmObjectIdentifiers(8) 5}
directoryObjectIdentifiers       ID    ::=    {module directoryObjectIdentifiers(9) 5}
upperBounds                      ID    ::=    {module upperBounds(10) 5}
-- dap                           ID    ::=    {module dap(11) 5}
-- dsp                           ID    ::=    {module dsp(12) 5}
distributedDirectoryOIDs         ID    ::=    {module distributedDirectoryOIDs(13) 5}
directoryShadowOIDs              ID    ::=    {module directoryShadowOIDs(14) 5}
directoryShadowAbstractService   ID    ::=    {module directoryShadowAbstractService(15) 5}
-- disp                          ID    ::=    {module disp(16) 5}
-- dop                           ID    ::=    {module dop(17) 5}
opBindingManagement              ID    ::=    {module opBindingManagement(18) 5}
opBindingOIDs                    ID    ::=    {module opBindingOIDs(19) 5}
hierarchicalOperationalBindings  ID    ::=    {module hierarchicalOperationalBindings(20) 5}
dsaOperationalAttributeTypes     ID    ::=    {module dsaOperationalAttributeTypes(22) 5}
schemaAdministration             ID    ::=    {module schemaAdministration(23) 5}
basicAccessControl               ID    ::=    {module basicAccessControl(24) 5}
directoryOperationalBindingTypes ID    ::=    {module directoryOperationalBindingTypes(25) 5}
certificateExtensions            ID    ::=    {module certificateExtensions(26) 5}
directoryManagement              ID    ::=    {module directoryManagement(27) 5}
enhancedSecurity                 ID    ::=    {module enhancedSecurity (28) 5}
-- directorySecurityExchanges    ID    ::=    {module directorySecurityExchanges (29) 5}
iDMProtocolSpecification         ID    ::=    {module iDMProtocolSpecification(30) 5}
directoryIDMProtocols            ID    ::=    {module directoryIDMProtocols(31) 5}
attributeCertificateDefinitions  ID    ::=    {module attributeCertificateDefinitions(32) 5}
serviceAdministration            ID    ::=    {module serviceAdministration(33) 5}
-- следующее определение предназначено для модуля, который содержит внешне определенные элементы
-- схемы, которые не определены с использованием формальной нотации ASN.1
-- (см. последнюю версию Руководства для разработчиков)
externalDefinitions              ID    ::=    {module externalDefinitions(34) }
commonProtocolSpecification      ID    ::=    {module commonProtocolSpecification (35) 5}
oSIProtocolSpecification         ID    ::=    {module oSIProtocolSpecification (36) 5}
directoryOSIProtocols            ID    ::=    {module directoryOSIProtocols (37) 5}
```

-- синонимы --

```
id-oc                            ID    ::=    objectClass
id-at                            ID    ::=    attributeType
id-as                            ID    ::=    abstractSyntax
id-mr                            ID    ::=    matchingRule
id-nf                            ID    ::=    nameForm
id-sc                            ID    ::=    subentry
id-oa                            ID    ::=    operationalAttributeType
id-ob                            ID    ::=    operationalBinding
id-doa                           ID    ::=    dsaOperationalAttribute
id-kmr                           ID    ::=    knowledgeMatchingRule
id-soc                           ID    ::=    schemaObjectClass
id-soa                           ID    ::=    schemaOperationalAttribute
id-ar                            ID    ::=    administrativeRoles
id-aca                           ID    ::=    accessControlAttribute
id-ac                            ID    ::=    applicationContext
-- id-rosObject                  ID    ::=    rosObject
-- id-contract                   ID    ::=    contract
-- id-package                    ID    ::=    package
id-acScheme                      ID    ::=    accessControlSchemes
id-ce                            ID    ::=    certificateExtension
id-mgt                           ID    ::=    managementObject
id-avc                           ID    ::=    attributeValueContext
-- id-se                         ID    ::=    securityExchange
```

```
id-idm                                  ID      ::=     idmProtocol
id-pr                                   ID      ::=     problem
id-not                                  ID      ::=     notification
id-mre                                  ID      ::=     matchingRestriction
id-cat                                  ID      ::=     controlAttributeType
id-kp                                   ID      ::=     keyPurposes

-- устаревшие идентификаторы модуля --

--      usefulDefinition                ID      ::=     {module 0}
--      informationFramework            ID      ::=     {module 1}
--      directoryAbstractService        ID      ::=     {module 2}
--      distributedOperations           ID      ::=     {module 3}
--      protocolObjectIdentifiers       ID      ::=     {module 4}
--      selectedAttributeTypes          ID      ::=     {module 5}
--      selectedObjectClasses           ID      ::=     {module 6}
--      authenticationFramework         ID      ::=     {module 7}
--      algorithmObjectIdentifiers      ID      ::=     {module 8}
--      directoryObjectIdentifiers      ID      ::=     {module 9}
--      upperBounds                     ID      ::=     {module 10}
--      dap                             ID      ::=     {module 11}
--      dsp                             ID      ::=     {module 12}
--      distributedDirectoryObjectIdentifiersID ::=     {module 13}

-- неиспользуемые идентификаторы модуля --

--      directoryShadowOIDs             ID      ::=     {module 14}
--      directoryShadowAbstractService  ID      ::=     {module 15}
--      disp                            ID      ::=     {module 16}
--      dop                             ID      ::=     {module 17}
--      opBindingManagement             ID      ::=     {module 18}
--      opBindingOIDs                   ID      ::=     {module 19}
--      hierarchicalOperationalBindings ID      ::=     {module 20}
--      dsaOperationalAttributeTypes    ID      ::=     {module 22}
--      schemaAdministration            ID      ::=     {module 23}
--      basicAccessControl              ID      ::=     {module 24}
--      operationalBindingOIDs          ID      ::=     {module 25}

END -- UsefulDefinitions
```

Приложение B

Информационная основа в ASN.1

(|)

Данное Приложение содержит сводку всех определений типов, значений и макросов на языке ASN.1, содержащихся в настоящей спецификации Справочника. Эти определения образуют модуль ASN.1 **InformationFramework**.

```
InformationFramework {joint-iso-itu-t ds(5) module(1) informationFramework(1) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

IMPORTS

-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        directoryAbstractService, id-ar, id-at, id-mr, id-nf, id-oa, id-oc, id-sc,
        selectedAttributeTypes, serviceAdministration, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        SearchRule
                FROM ServiceAdministration serviceAdministration

-- из Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3

        TypeAndContextAssertion
                FROM DirectoryAbstractService directoryAbstractService

-- из Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6

        booleanMatch, commonName, DirectoryString {}, generalizedTimeMatch,
        generalizedTimeOrderingMatch, integerFirstComponentMatch, integerMatch,
        integerOrderingMatch, objectIdentifierFirstComponentMatch
                FROM SelectedAttributeTypes selectedAttributeTypes

        ub-search
                FROM UpperBounds upperBounds      ;

-- типы данных атрибутов --

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

AttributeType  ::=  ATTRIBUTE.&id

AttributeValue  ::=  ATTRIBUTE.&Type

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

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

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

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


-- Определение следующей совокупности информационных объектов передается, возможно,
-- в стандартизированные профили или в свидетельства о соответствии протокольной реализации.
-- Эта совокупность необходима для описания табличного ограничения, накладываемого на компонент
-- значений Attribute, компонент значений AttributeTypeAndValue и компонент утверждения
-- AttributeValueAssertion.

SupportedAttributes  ATTRIBUTE  ::=  { objectClass | aliasedEntryName, ... }

-- Определение следующей совокупности информационных объектов передается, возможно,
-- в стандартизированные профили или в свидетельства о соответствии протокольной реализации.
-- Эта совокупность необходима для описания табличного ограничения, накладываемого на спецификации
-- контекста.

SupportedContexts  CONTEXT  ::=  { ... }


-- типы данных именования --

Name  ::=  CHOICE { -- сейчас только одна возможность --  rdnSequence  RDNSequence }

RDNSequence  ::=  SEQUENCE OF RelativeDistinguishedName

DistinguishedName  ::=  RDNSequence

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

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

-- типы данных поддерева --

SubtreeSpecification  ::=  SEQUENCE {
        base                    [0]     LocalName DEFAULT { },
                                        COMPONENTS OF  ChopSpecification,
        specificationFilter     [4]     Refinement OPTIONAL }
        -- пустая последовательность описывает целую административную область

LocalName  ::=  RDNSequence

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

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

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


-- спецификация информационного класса объектов OBJECT-CLASS --

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

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


-- классы объектов --

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

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

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

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


-- спецификация информационного класса объектов ATTRIBUTE --

ATTRIBUTE ::= CLASS {
        &derivation                 ATTRIBUTE OPTIONAL,
        &Type                       OPTIONAL,   -- требуется либо &Type или &derivation --
        &equality-match             MATCHING-RULE OPTIONAL,
        &ordering-match             MATCHING-RULE OPTIONAL,
        &substrings-match           MATCHING-RULE OPTIONAL,
        &single-valued              BOOLEAN DEFAULT FALSE,
        &collective                 BOOLEAN DEFAULT FALSE,
        &dummy                      BOOLEAN DEFAULT FALSE,
        -- операционные расширения --
        &no-user-modification       BOOLEAN DEFAULT FALSE,
        &usage                      AttributeUsage DEFAULT userApplications,
        &id                         OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        [ SUBTYPE OF                &derivation ]
```

```
        [ WITH SYNTAX                   &Type ]
        [ EQUALITY MATCHING RULE        &equality-match ]
        [ ORDERING MATCHING RULE        &ordering-match ]
        [ SUBSTRINGS MATCHING RULE &substrings-match ]
        [ SINGLE VALUE                  &single-valued ]
        [ COLLECTIVE                    &collective ]
        [ DUMMY                         &dummy ]
        [ NO USER MODIFICATION          &no-user-modification ]
        [ USAGE                         &usage ]
        ID                              &id }

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


-- атрибуты --

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

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


-- спецификация информационного класса объектов MATCHING-RULE --

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


-- правила сопоставления --

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

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

MAPPING-BASED-MATCHING
        { SelectedBy, BOOLEAN:combinable, MappingResult, OBJECT IDENTIFIER:matchingRule }  ::=
CLASS  {
        &selectBy               SelectedBy              OPTIONAL,
        &ApplicableTo           ATTRIBUTE,
        &subtypesIncluded       BOOLEAN                 DEFAULT  TRUE,
        &combinable             BOOLEAN                 (combinable),
        &mappingResults         MappingResult           OPTIONAL,
        &userControl            BOOLEAN                 DEFAULT  FALSE,
        &exclusive              BOOLEAN                 DEFAULT  TRUE,
        &matching-rule          MATCHING-RULE.&id       (matchingRule),
```

```
        &id                     OBJECT IDENTIFIER       UNIQUE }
WITH SYNTAX  {
        [ SELECT BY             &selectBy ]
        APPLICABLE TO           &ApplicableTo
        [ SUBTYPES INCLUDED     &subtypesIncluded ]
        COMBINABLE              &combinable
        [ MAPPING RESULTS       &mappingResults ]
        [ USER CONTROL          &userControl ]
        [ EXCLUSIVE             &exclusive ]
        MATCHING RULE           &matching-rule
        ID                      &id }


-- спецификация информационного класса объектов NAME-FORM --

NAME-FORM ::= CLASS  {
        &namedObjectClass       OBJECT-CLASS,
        &MandatoryAttributes    ATTRIBUTE,
        &OptionalAttributes     ATTRIBUTE OPTIONAL,
        &id                     OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
        NAMES                   &namedObjectClass
        WITH ATTRIBUTES         &MandatoryAttributes
        [ AND OPTIONALLY        &OptionalAttributes ]
        ID                      &id }

-- класс STRUCTURE-RULE и типы данных правила структуры DIT --

STRUCTURE-RULE ::= CLASS {
        &nameForm               NAME-FORM,
        &SuperiorStructureRules STRUCTURE-RULE OPTIONAL,
        &id                     RuleIdentifier }
WITH SYNTAX {
        NAME FORM               &nameForm
        [ SUPERIOR RULES        &SuperiorStructureRules ]
        ID                      &id }

DITStructureRule ::= SEQUENCE {
        ruleIdentifier          RuleIdentifier ,
                                -- должен быть уникальным в пределах области действия подсхемы
        nameForm                NAME-FORM.&id,
        superiorStructureRules  SET SIZE (1..MAX) OF RuleIdentifier OPTIONAL }

RuleIdentifier ::= INTEGER


-- класс CONTENT-RULE и типы данных правила контента DIT --

CONTENT-RULE ::= CLASS {
        &structuralClass                OBJECT-CLASS.&id        UNIQUE,
        &Auxiliaries                    OBJECT-CLASS            OPTIONAL,
        &Mandatory                      ATTRIBUTE               OPTIONAL,
        &Optional                       ATTRIBUTE               OPTIONAL,
        &Precluded                      ATTRIBUTE               OPTIONAL }
WITH SYNTAX {
        STRUCTURAL OBJECT-CLASS         &structuralClass
        [ AUXILIARY OBJECT-CLASSES      &Auxiliaries ]
        [ MUST CONTAIN                  &Mandatory ]
        [ MAY CONTAIN                   &Optional ]
        [ MUST-NOT CONTAIN              &Precluded ] }

DITContentRule ::= SEQUENCE {
        structuralObjectClass   OBJECT-CLASS.&id,
        auxiliaries             SET SIZE (1..MAX) OF OBJECT-CLASS.&id   OPTIONAL,
        mandatory       [1]     SET SIZE (1..MAX) OF ATTRIBUTE.&id      OPTIONAL,
        optional        [2]     SET SIZE (1..MAX) OF ATTRIBUTE.&id      OPTIONAL,
        precluded       [3]     SET SIZE (1..MAX) OF ATTRIBUTE.&id      OPTIONAL }
```

```
CONTEXT ::= CLASS {
        &Type,
        &DefaultValue               OPTIONAL,
        &Assertion                  OPTIONAL,
        &absentMatch                BOOLEAN DEFAULT TRUE,
        &id                         OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        WITH SYNTAX       &Type
        [ DEFAULT-VALUE  &DefaultValue ]
        [ ASSERTED AS     &Assertion ]
        [ ABSENT-MATCH   &absentMatch ]
        ID                &id }

DITContextUse ::= SEQUENCE {
        attributeType                   ATTRIBUTE.&id,
        mandatoryContexts [1]     SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL,
        optionalContexts   [2]    SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL }

DIT-CONTEXT-USE-RULE ::= CLASS {
        &attributeType              ATTRIBUTE.&id     UNIQUE,
        &Mandatory                  CONTEXT           OPTIONAL,
        &Optional                   CONTEXT           OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE                  &attributeType
        [ MANDATORY CONTEXTS        &Mandatory ]
        [ OPTIONAL CONTEXTS         &Optional ] }

FRIENDS ::= CLASS {
        &anchor                     ATTRIBUTE.&id UNIQUE,
        &Friends                    ATTRIBUTE }
WITH SYNTAX {
        ANCHOR                      &anchor
        FRIENDS                     &Friends }

-- информамционные объекты схемы системы --
-- классы объектов --

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

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

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

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

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

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

```
-- атрибуты --

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

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

createTimestamp ATTRIBUTE ::= {
        WITH SYNTAX                         GeneralizedTime
                        -- согласно п. 42.3 b) или c) Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1
        EQUALITY MATCHING RULE              generalizedTimeMatch
        ORDERING MATCHING RULE              generalizedTimeOrderingMatch
        SINGLE VALUE                        TRUE
        NO USER MODIFICATION                TRUE
        USAGE                               directoryOperation
        ID                                  id-oa-createTimestamp }

modifyTimestamp ATTRIBUTE ::= {
        WITH SYNTAX                         GeneralizedTime
                        -- согласно п. 42.3 b) или c) Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1
        EQUALITY MATCHING RULE              generalizedTimeMatch
        ORDERING MATCHING RULE              generalizedTimeOrderingMatch
        SINGLE VALUE                        TRUE
        NO USER MODIFICATION                TRUE
        USAGE                               directoryOperation
        ID                                  id-oa-modifyTimestamp }

subschemaTimestamp ATTRIBUTE ::= {
        WITH SYNTAX                         GeneralizedTime
                        -- согласно п. 42.3 b) или c) Рек. МСЭ-Т X.680 | ИСО/МЭК 8824-1
        EQUALITY MATCHING RULE              generalizedTimeMatch
        ORDERING MATCHING RULE              generalizedTimeOrderingMatch
        SINGLE VALUE                        TRUE
        NO USER MODIFICATION                TRUE
        USAGE                               directoryOperation
        ID                                  id-oa-subschemaTimestamp }

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

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

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

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

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

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

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

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

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

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

searchRules ATTRIBUTE ::= {
        WITH SYNTAX                     SearchRuleDescription
        EQUALITY MATCHING RULE          integerFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-oa-searchRules }

SearchRuleDescription ::= SEQUENCE {
        COMPONENTS OF                   SearchRule,
        name                    [28]    SET SIZE (1 .. MAX) OF DirectoryString { ub-search } OPTIONAL,
        description             [29]    DirectoryString { ub-search } OPTIONAL }

hierarchyLevel ATTRIBUTE ::= {
        WITH SYNTAX                     HierarchyLevel
        EQUALITY MATCHING RULE          integerMatch
        ORDERING MATCHING RULE          integerOrderingMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyLevel }
```

```
HierarchyLevel  ::=  INTEGER

hierarchyBelow ATTRIBUTE  ::=  {
        WITH SYNTAX                     HierarchyBelow
        EQUALITY MATCHING RULE          booleanMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-oa-hierarchyBelow }

HierarchyBelow  ::=  BOOLEAN

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

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

-- присвоения идентификаторов объектов --

-- классы объектов --

id-oc-top                               OBJECT IDENTIFIER::=    {id-oc 0}
id-oc-alias                             OBJECT IDENTIFIER::=    {id-oc 1}
id-oc-parent                            OBJECT IDENTIFIER::=    {id-oc 28}
id-oc-child                             OBJECT IDENTIFIER::=    {id-oc 29}

-- атрибуты --

id-at-objectClass                       OBJECT IDENTIFIER::=    {id-at 0}
id-at-aliasedEntryName                  OBJECT IDENTIFIER::=    {id-at 1}

-- правила сопоставления --

id-mr-objectIdentifierMatch             OBJECT IDENTIFIER::=    {id-mr 0}
id-mr-distinguishedNameMatch            OBJECT IDENTIFIER::=    {id-mr 1}


-- операционные атрибуты --

id-oa-excludeAllCollectiveAttributes    OBJECT IDENTIFIER       ::=     {id-oa 0}
id-oa-createTimestamp                   OBJECT IDENTIFIER       ::=     {id-oa 1}
id-oa-modifyTimestamp                   OBJECT IDENTIFIER       ::=     {id-oa 2}
id-oa-creatorsName                      OBJECT IDENTIFIER       ::=     {id-oa 3}
id-oa-modifiersName                     OBJECT IDENTIFIER       ::=     {id-oa 4}
id-oa-administrativeRole                OBJECT IDENTIFIER       ::=     {id-oa 5}
id-oa-subtreeSpecification              OBJECT IDENTIFIER       ::=     {id-oa 6}
id-oa-collectiveExclusions              OBJECT IDENTIFIER       ::=     {id-oa 7}
id-oa-subschemaTimestamp                OBJECT IDENTIFIER       ::=     {id-oa 8}
id-oa-hasSubordinates                   OBJECT IDENTIFIER       ::=     {id-oa 9}
id-oa-subschemaSubentryList             OBJECT IDENTIFIER       ::=     {id-oa 10}
id-oa-accessControlSubentryList         OBJECT IDENTIFIER       ::=     {id-oa 11}
id-oa-collectiveAttributeSubentryList   OBJECT IDENTIFIER       ::=     {id-oa 12}
id-oa-contextDefaultSubentryList        OBJECT IDENTIFIER       ::=     {id-oa 13}
id-oa-contextAssertionDefault           OBJECT IDENTIFIER       ::=     {id-oa 14}
id-oa-serviceAdminSubentryList          OBJECT IDENTIFIER       ::=     {id-oa 15}
id-oa-searchRules                       OBJECT IDENTIFIER       ::=     {id-oa 16}
id-oa-hierarchyLevel                    OBJECT IDENTIFIER       ::=     {id-oa 17}
```

```
id-oa-hierarchyBelow                OBJECT IDENTIFIER    ::=    {id-oa 18}
id-oa-hierarchyParent               OBJECT IDENTIFIER    ::=    {id-oa 19}
id-oa-hierarchyTop                  OBJECT IDENTIFIER    ::=    {id-oa 20}


-- классы подстатей --

id-sc-subentry                      OBJECT IDENTIFIER    ::=    {id-sc 0}
id-sc-accessControlSubentry         OBJECT IDENTIFIER    ::=    {id-sc 1}
id-sc-collectiveAttributeSubentry   OBJECT IDENTIFIER    ::=    {id-sc 2}
id-sc-contextAssertionSubentry      OBJECT IDENTIFIER    ::=    {id-sc 3}
id-sc-serviceAdminSubentry          OBJECT IDENTIFIER    ::=    {id-sc 4}

-- формы имени --

id-nf-subentryNameForm              OBJECT IDENTIFIER    ::=    {id-nf 16}

-- административные функции --

id-ar-autonomousArea                OBJECT IDENTIFIER    ::=    {id-ar 1}
id-ar-accessControlSpecificArea     OBJECT IDENTIFIER    ::=    {id-ar 2}
id-ar-accessControlInnerArea        OBJECT IDENTIFIER    ::=    {id-ar 3}
id-ar-subschemaAdminSpecificArea    OBJECT IDENTIFIER    ::=    {id-ar 4}
id-ar-collectiveAttributeSpecificArea OBJECT IDENTIFIER  ::=    {id-ar 5}
id-ar-collectiveAttributeInnerArea  OBJECT IDENTIFIER    ::=    {id-ar 6}
id-ar-contextDefaultSpecificArea    OBJECT IDENTIFIER    ::=    {id-ar 7}
id-ar-serviceSpecificArea           OBJECT IDENTIFIER    ::=    {id-ar 8}

END -- InformationFramework
```

Приложение C

Схема административного управления подсхемой на языке ASN.1

(|)

Данное Приложение содержит определения типов, значений и информационных объектов на языке ASN.1 для административного управления подсхемой, определенного в п. 15, в форме модуля ASN.1 **SchemaAdministration**.

```
SchemaAdministration {joint-iso-itu-t ds(5) module(1) schemaAdministration(23) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

IMPORTS

-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        id-soa, id-soc, informationFramework, selectedAttributeTypes, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        ATTRIBUTE, AttributeUsage, CONTEXT, DITContentRule, DITStructureRule, MATCHING-RULE,
        NAME-FORM, OBJECT-CLASS, ObjectClassKind, objectIdentifierMatch
                FROM InformationFramework informationFramework

-- из Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6

        DirectoryString {}, integerFirstComponentMatch, integerMatch,
        objectIdentifierFirstComponentMatch
                FROM SelectedAttributeTypes selectedAttributeTypes

        ub-schema
                FROM UpperBounds upperBounds;


-- типы --

DITStructureRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF  DITStructureRule,
        name                [1]     SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description             DirectoryString { ub-schema } OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE }

DITContentRuleDescription  ::=  SEQUENCE {
        COMPONENTS OF           DITContentRule,
        name                [4]     SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description                 DirectoryString { ub-schema }       OPTIONAL,
        obsolete                    BOOLEAN DEFAULT FALSE }

MatchingRuleDescription  ::=  SEQUENCE {
        identifier              MATCHING-RULE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }           OPTIONAL,
        obsolete                BOOLEAN                                 DEFAULT FALSE,
        information   [0]       DirectoryString { ub-schema }           OPTIONAL }
                                -- описывает синтаксис ASN.1

AttributeTypeDescription  ::=  SEQUENCE  {
```

```
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }           OPTIONAL,
        obsolete                BOOLEAN                                 DEFAULT FALSE,
        information   [0]       AttributeTypeInformation }

AttributeTypeInformation ::= SEQUENCE {
        derivation              [0]     ATTRIBUTE.&id OPTIONAL,
        equalityMatch           [1]     MATCHING-RULE.&id                       OPTIONAL,
        orderingMatch           [2]     MATCHING-RULE.&id                       OPTIONAL,
        substringsMatch         [3]     MATCHING-RULE.&id                       OPTIONAL,
        attributeSyntax         [4]     DirectoryString { ub-schema }           OPTIONAL,
        multi-valued            [5]     BOOLEAN                                 DEFAULT TRUE,
        collective              [6]     BOOLEAN                                 DEFAULT FALSE,
        userModifiable          [7]     BOOLEAN                                 DEFAULT TRUE,
        application                     AttributeUsage                          DEFAULT  userApplications }

ObjectClassDescription ::= SEQUENCE {
        identifier              OBJECT-CLASS.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN                                                 DEFAULT FALSE,
        information   [0]       ObjectClassInformation }

ObjectClassInformation ::= SEQUENCE {
        subclassOf              SET SIZE (1..MAX) OF OBJECT-CLASS.&id    OPTIONAL,
        kind                    ObjectClassKind                          DEFAULT structural,
        mandatories [3]         SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL,
        optionals     [4]       SET SIZE (1..MAX) OF ATTRIBUTE.&id       OPTIONAL }

NameFormDescription ::= SEQUENCE {
        identifier              NAME-FORM.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema } OPTIONAL,
        description             DirectoryString { ub-schema } OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE,
        information   [0]       NameFormInformation }

NameFormInformation ::= SEQUENCE {
        subordinate             OBJECT-CLASS.&id,
        namingMandatoriesSET OF ATTRIBUTE.&id,
        namingOptionals         SET SIZE (1..MAX) OF ATTRIBUTE.&id OPTIONAL }

MatchingRuleUseDescription ::= SEQUENCE {
        identifier              MATCHING-RULE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN                                                 DEFAULT FALSE,
        information   [0]       SET OF ATTRIBUTE.&id }

ContextDescription ::= SEQUENCE {
        identifier              CONTEXT.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString {ub-schema}        OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN                                                 DEFAULT FALSE,
        information   [0]       ContextInformation }

ContextInformation ::= SEQUENCE {
        syntax                  DirectoryString { ub-schema } ,
        assertionSyntax         DirectoryString { ub-schema } OPTIONAL }

DITContextUseDescription ::= SEQUENCE {
        identifier              ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN                                                 DEFAULT FALSE,
        information   [0]       DITContextUseInformation }
```

```
dITContextUseInformation  ::=  SEQUENCE  {
        mandatoryContexts [1]     SET SIZE (1..MAX) OF CONTEXT.&id OPTIONAL,
        optionalContexts     [2]     SET SIZE (1..MAX) OF CONTEXT.&id  OPTIONAL }


-- классы объектов --

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


-- атрибуты --

dITStructureRules ATTRIBUTE  ::=  {
        WITH SYNTAX                     DITStructureRuleDescription
        EQUALITY MATCHING RULE          integerFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITStructureRule }

dITContentRules  ATTRIBUTE  ::=  {
        WITH SYNTAX                     DITContentRuleDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContentRules }

matchingRules  ATTRIBUTE  ::=  {
        WITH SYNTAX                     MatchingRuleDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-matchingRules }

attributeTypes ATTRIBUTE ::=  {
        WITH SYNTAX                     AttributeTypeDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-attributeTypes }

objectClasses ATTRIBUTE  ::=  {
        WITH SYNTAX                     ObjectClassDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-objectClasses }

nameForms  ATTRIBUTE  ::=  {
        WITH SYNTAX                     NameFormDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-nameForms }

matchingRuleUse ATTRIBUTE  ::=  {
        WITH SYNTAX                     MatchingRuleUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-matchingRuleUse }
```

```
structuralObjectClass ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-soa-structuralObjectClass }

governingStructureRule ATTRIBUTE  ::= {
        WITH SYNTAX                     INTEGER
        EQUALITY MATCHING RULE          integerMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           directoryOperation
        ID                              id-soa-governingStructureRule }

contextTypes ATTRIBUTE  ::= {
        WITH SYNTAX                     ContextDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-contextTypes }

dITContextUse ATTRIBUTE  ::= {
        WITH SYNTAX                     DITContextUseDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-dITContextUse }

friends ATTRIBUTE  ::= {
        WITH SYNTAX                     FriendsDescription
        EQUALITY MATCHING RULE          objectIdentifierFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-soa-friends }

FriendsDescription  ::= SEQUENCE {
        anchor                  ATTRIBUTE.&id,
        name                    SET SIZE (1..MAX) OF DirectoryString { ub-schema }      OPTIONAL,
        description             DirectoryString { ub-schema }                           OPTIONAL,
        obsolete                BOOLEAN DEFAULT FALSE,
        friends         [0]     SET OF ATTRIBUTE.&id }

-- присвоения идентификаторов объектов --

-- классы объектов схемы --

id-soc-subschema                OBJECT IDENTIFIER       ::=     {id-soc 1}

-- операционные атрибуты схемы --

id-soa-dITStructureRule         OBJECT IDENTIFIER       ::=     {id-soa 1}
id-soa-dITContentRules          OBJECT IDENTIFIER       ::=     {id-soa 2}
id-soa-matchingRules            OBJECT IDENTIFIER       ::=     {id-soa 4}
id-soa-attributeTypes           OBJECT IDENTIFIER       ::=     {id-soa 5}
id-soa-objectClasses            OBJECT IDENTIFIER       ::=     {id-soa 6}
id-soa-nameForms                OBJECT IDENTIFIER       ::=     {id-soa 7}
id-soa-matchingRuleUse          OBJECT IDENTIFIER       ::=     {id-soa 8}
id-soa-structuralObjectClass    OBJECT IDENTIFIER       ::=     {id-soa 9}
id-soa-governingStructureRule   OBJECT IDENTIFIER       ::=     {id-soa 10}
id-soa-contextTypes             OBJECT IDENTIFIER       ::=     {id-soa 11}
id-soa-dITContextUse            OBJECT IDENTIFIER       ::=     {id-soa 12}
id-soa-friends                  OBJECT IDENTIFIER       ::=     {id-soa 13}

END -- SchemaAdministration
```

Приложение D

Административное управление службой на языке ASN.1

(|)

Данное Приложение содержит определения типов, значений и информационных объектов на языке ASN.1 для административного управления подсхемой, определенного в п. 16, в форме модуля ASN.1, **ServiceAdministration**.

```
ServiceAdministration {joint-iso-itu-t ds(5) module(1) serviceAdministration(33) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

IMPORTS

-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        directoryAbstractService, informationFramework
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        ATTRIBUTE, AttributeType, CONTEXT, MATCHING-RULE, OBJECT-CLASS,
        SupportedAttributes, SupportedContexts
                FROM InformationFramework informationFramework

-- из Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3

        FamilyGrouping, FamilyReturn, HierarchySelections, SearchControlOptions,
        ServiceControlOptions
                FROM DirectoryAbstractService directoryAbstractService      ;


-- типы --

SearchRule ::= SEQUENCE {
        COMPONENTS OF           SearchRuleId,
        serviceType         [1] OBJECT IDENTIFIER                           OPTIONAL,
        userClass           [2] INTEGER                                     OPTIONAL,
        inputAttributeTypes [3] SEQUENCE SIZE (0..MAX) OF RequestAttribute  OPTIONAL,
        attributeCombination[4] AttributeCombination                        DEFAULT and : { },
        outputAttributeTypes[5] SEQUENCE SIZE (1..MAX) OF ResultAttribute   OPTIONAL,
        defaultControls     [6] ControlOptions                              OPTIONAL,
        mandatoryControls   [7] ControlOptions                              OPTIONAL,
        searchRuleControls  [8] ControlOptions                              OPTIONAL,
        familyGrouping      [9] FamilyGrouping                              OPTIONAL,
        familyReturn        [10] FamilyReturn                               OPTIONAL,
        relaxation          [11] RelaxationPolicy                           OPTIONAL,
        additionalControl   [12] SEQUENCE SIZE (1..MAX) OF AttributeType    OPTIONAL,
        allowedSubset       [13] AllowedSubset                              DEFAULT '111'B,
        imposedSubset       [14] ImposedSubset                              OPTIONAL,
        entryLimit          [15] EntryLimit                                 OPTIONAL }

SearchRuleId ::= SEQUENCE {
        id                      INTEGER,
        dmdId               [0] OBJECT IDENTIFIER }

AllowedSubset ::= BIT STRING { baseObject (0), oneLevel (1), wholeSubtree (2) }
```

```
ImposedSubset ::= ENUMERATED { baseObject (0), oneLevel (1), wholeSubtree (2) }

RequestAttribute ::= SEQUENCE {
        attributeType                   ATTRIBUTE.&id ({ SupportedAttributes }),
        includeSubtypes         [0]     BOOLEAN                                         DEFAULT FALSE,
        selectedValues          [1]     SEQUENCE SIZE (0..MAX) OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType })     OPTIONAL,
        defaultValues           [2]     SEQUENCE SIZE (0..MAX) OF SEQUENCE {
                entryType               OBJECT-CLASS.&id  OPTIONAL,
                values                  SEQUENCE OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType }) }   OPTIONAL,
        contexts                [3]     SEQUENCE SIZE (0..MAX) OF ContextProfile        OPTIONAL,
        contextCombination      [4]     ContextCombination                              DEFAULT and : { },
        matchingUse             [5]     SEQUENCE SIZE (1..MAX) OF MatchingUse  OPTIONAL }

ContextProfile ::= SEQUENCE {
        contextType         CONTEXT.&id({SupportedContexts}),
        contextValue        SEQUENCE SIZE (1..MAX) OF CONTEXT.&Assertion
                                ({SupportedContexts}{@contextType})  OPTIONAL }

ContextCombination ::= CHOICE {
        context                 [0]     CONTEXT.&id({SupportedContexts}),
        and                     [1]     SEQUENCE OF ContextCombination,
        or                      [2]     SEQUENCE OF ContextCombination,
        not                     [3]     ContextCombination }

MatchingUse ::= SEQUENCE {
        restrictionType         MATCHING-RESTRICTION.&id ({SupportedMatchingRestrictions}),
        restrictionValue        MATCHING-RESTRICTION.&Restriction
                                        ({SupportedMatchingRestrictions}{@restrictionType}) }

-- Определение следующей совокупности информационных объектов передается, возможно,
-- в стандартизированные профили или в свидетельства о соответствии протокольной реализации.
-- Эта совокупность необходима для описания табличного ограничения, накладываемого на компоненты
-- SupportedMatchingRestrictions.

SupportedMatchingRestrictions  MATCHING-RESTRICTION  ::= { ... }

AttributeCombination ::= CHOICE {
        attribute               [0]     AttributeType,
        and                     [1]     SEQUENCE OF AttributeCombination,
        or                      [2]     SEQUENCE OF AttributeCombination,
        not                     [3]     AttributeCombination }

ResultAttribute ::= SEQUENCE {
        attributeType       ATTRIBUTE.&id ({ SupportedAttributes }),
        outputValues        CHOICE {
                selectedValues      SEQUENCE OF ATTRIBUTE.&Type
                                        ({ SupportedAttributes }{ @attributeType }),
                matchedValuesOnly   NULL } OPTIONAL,
        contexts                [0]     SEQUENCE SIZE (1..MAX) OF ContextProfile  OPTIONAL }

ControlOptions ::= SEQUENCE {
        serviceControls     [0]     ServiceControlOptions       DEFAULT { },
        searchOptions       [1]     SearchControlOptions        DEFAULT { searchAliases },
        hierarchyOptions    [2]     HierarchySelections         OPTIONAL }

EntryLimit ::= SEQUENCE {
        default             INTEGER,
        max                 INTEGER }

RelaxationPolicy ::= SEQUENCE {
        basic           [0]     MRMapping DEFAULT { },
        tightenings     [1]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        relaxations     [2]     SEQUENCE SIZE (1..MAX) OF MRMapping OPTIONAL,
        maximum         [3]     INTEGER OPTIONAL,       -- обязательно, если присутствует tightenings
        minimum         [4]     INTEGER DEFAULT 1 }
```

```
MRMapping ::= SEQUENCE {
        mapping       [0]   SEQUENCE SIZE (1..MAX) OF Mapping          OPTIONAL,
        substitution  [1]   SEQUENCE SIZE (1..MAX) OF MRSubstitution   OPTIONAL }

Mapping ::= SEQUENCE {

        mappingFunction   OBJECT IDENTIFIER (CONSTRAINED BY { -- должен быть идентификатор

                        -- объекта из mapping-based matching algorithm -- } ),
        level                       INTEGER  DEFAULT 0 }

MRSubstitution ::= SEQUENCE {
        attribute                   AttributeType,
        oldMatchingRule   [0]       MATCHING-RULE.&id  OPTIONAL,
        newMatchingRule   [1]       MATCHING-RULE.&id  OPTIONAL }


-- информационные классы объектов ASN.1 --

SEARCH-RULE ::= CLASS {
        &dmdId                      OBJECT IDENTIFIER,
        &serviceType                OBJECT IDENTIFIER                   OPTIONAL,
        &userClass                  INTEGER                             OPTIONAL,
        &InputAttributeTypes        REQUEST-ATTRIBUTE                   OPTIONAL,
        &combination                AttributeCombination                OPTIONAL,
        &OutputAttributeTypes       RESULT-ATTRIBUTE                    OPTIONAL,
        &defaultControls            ControlOptions                      OPTIONAL,
        &mandatoryControls          ControlOptions                      OPTIONAL,
        &searchRuleControls         ControlOptions                      OPTIONAL,
        &familyGrouping             FamilyGrouping                      OPTIONAL,
        &familyReturn               FamilyReturn                        OPTIONAL,
        &additionalControl          AttributeType                       OPTIONAL,
        &relaxation                 RelaxationPolicy                    OPTIONAL,
        &allowedSubset              AllowedSubset                       DEFAULT '111'B,
        &imposedSubset              ImposedSubset                       OPTIONAL,
        &entryLimit                 EntryLimit                          OPTIONAL,
        &id                         INTEGER UNIQUE }
WITH SYNTAX {
        DMD ID                      &dmdId
        [ SERVICE-TYPE              &serviceType ]
        [ USER-CLASS                &userClass ]
        [ INPUT ATTRIBUTES          &InputAttributeTypes ]
        [ COMBINATION               &combination ]
        [ OUTPUT ATTRIBUTES         &OutputAttributeTypes ]
        [ DEFAULT CONTROL           &defaultControls ]
        [ MANDATORY CONTROL         &mandatoryControls ]
        [ SEARCH-RULE CONTROL       &searchRuleControls ]
        [ FAMILY-GROUPING           &familyGrouping ]
        [ FAMILY-RETURN             &familyReturn ]
        [ ADDITIONAL CONTROL        &additionalControl ]
        [ RELAXATION                &relaxation ]
        [ ALLOWED SUBSET            &allowedSubset ]
        [ IMPOSED SUBSET            &imposedSubset ]
        [ ENTRY LIMIT               &entryLimit ]
        ID                          &id }

REQUEST-ATTRIBUTE ::= CLASS {
        &attributeType              ATTRIBUTE.&id,
        &SelectedValues             ATTRIBUTE.&Type                          OPTIONAL,
        &DefaultValues              SEQUENCE {
                                    entryType    OBJECT-CLASS.&id            OPTIONAL,
                                    values       SEQUENCE OF ATTRIBUTE.&Type } OPTIONAL,
        &contexts                   SEQUENCE OF ContextProfile               OPTIONAL,
        &contextCombination         ContextCombination                       OPTIONAL,
        &MatchingUse                MatchingUse                              OPTIONAL,
        &includeSubtypes            BOOLEAN                                  DEFAULT  FALSE }
WITH SYNTAX {
        ATTRIBUTE TYPE              &attributeType
        [ SELECTED VALUES           &SelectedValues ]
```

```
        [ DEFAULT VALUES            &DefaultValues ]
        [ CONTEXTS                  &contexts ]
        [ CONTEXT COMBINATION       &contextCombination ]
        [ MATCHING USE              &MatchingUse ]
        [ INCLUDE SUBTYPES          &includeSubtypes ] }

RESULT-ATTRIBUTE  ::=  CLASS {
        &attributeType              ATTRIBUTE.&id,
        &outputValues               CHOICE {
                selectedValues      SEQUENCE OF ATTRIBUTE.&Type,
                matchedValuesOnly   NULL }                          OPTIONAL,
        &contexts                   ContextProfile                  OPTIONAL }
WITH SYNTAX {
        ATTRIBUTE TYPE          &attributeType
        [ OUTPUT VALUES         &outputValues ]
        [ CONTEXTS              &contexts ] }

MATCHING-RESTRICTION ::= CLASS {
        &Restriction,
        &Rules              MATCHING-RULE.&id,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX  {
        RESTRICTION         &Restriction
        RULES               &Rules
        ID                  &id }

END  -- ServiceAdministration
```

Приложение E

Базовое управление доступом на языке ASN.1

(|)

Данное Приложение содержит сводку всех описаний типов и значений для базового управления доступом. Эти определения образуют модуль ASN.1 **BasicAccessControl**.

```
BasicAccessControl {joint-iso-itu-t ds(5) module(1) basicAccessControl(24) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

IMPORTS

-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        directoryAbstractService, id-aca, id-acScheme, informationFramework,
        selectedAttributeTypes, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        ATTRIBUTE, AttributeType, ContextAssertion, DistinguishedName, MATCHING-RULE,
        objectIdentifierMatch, Refinement, SubtreeSpecification, SupportedAttributes
                FROM InformationFramework informationFramework

-- из Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3

        Filter
                FROM DirectoryAbstractService directoryAbstractService

-- из Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6

        DirectoryString{}, directoryStringFirstComponentMatch, NameAndOptionalUID,
        UniqueIdentifier
                FROM SelectedAttributeTypes selectedAttributeTypes

        ub-tag
                FROM UpperBounds upperBounds      ;

-- типы --

ACIItem ::= SEQUENCE {
        identificationTag               DirectoryString { ub-tag },
        precedence                      Precedence,
        authenticationLevel             AuthenticationLevel,
        itemOrUserFirst                 CHOICE {
                itemFirst               [0]     SEQUENCE {
                        protectedItems                  ProtectedItems,
                        itemPermissions                 SET OF ItemPermission },
                userFirst               [1]     SEQUENCE {
                        userClasses                     UserClasses,
                        userPermissions                 SET OF UserPermission } } }

Precedence ::= INTEGER (0..255)
```

```
ProtectedItems ::= SEQUENCE {
        entry                               [0]   NULL                                          OPTIONAL,
        allUserAttributeTypes               [1]   NULL                                          OPTIONAL,
        attributeType                       [2]   SET SIZE (1..MAX) OF AttributeType            OPTIONAL,
        allAttributeValues                  [3]   SET SIZE (1..MAX) OF AttributeType            OPTIONAL,
        allUserAttributeTypesAndValues      [4]   NULL                                          OPTIONAL,
        attributeValue                      [5]   SET SIZE (1..MAX) OF AttributeTypeAndValue    OPTIONAL,
        selfValue                           [6]   SET SIZE (1..MAX) OF AttributeType            OPTIONAL,
        rangeOfValues                       [7]   Filter                                        OPTIONAL,
        maxValueCount                       [8]   SET SIZE (1..MAX) OF MaxValueCount            OPTIONAL,
        maxImmSub                           [9]   INTEGER                                       OPTIONAL,
        restrictedBy                        [10]  SET SIZE (1..MAX) OF RestrictedValue          OPTIONAL,
        contexts                            [11]  SET SIZE (1..MAX) OF ContextAssertion         OPTIONAL,
        classes                             [12]  Refinement                                    OPTIONAL
}

MaxValueCount ::= SEQUENCE {
        type            AttributeType,
        maxCount        INTEGER }

RestrictedValue ::= SEQUENCE {
        type            AttributeType,
        valuesIn        AttributeType }

UserClasses ::= SEQUENCE {
        allUsers        [0]   NULL                                          OPTIONAL,
        thisEntry       [1]   NULL                                          OPTIONAL,
        name            [2]   SET SIZE (1..MAX) OF NameAndOptionalUID       OPTIONAL,
        userGroup       [3]   SET SIZE (1..MAX) OF NameAndOptionalUID       OPTIONAL,
                              -- компонентом dn должно быть имя
                              -- статьи GroupOfUniqueNames
        subtree         [4]   SET SIZE (1..MAX) OF SubtreeSpecification     OPTIONAL }

ItemPermission ::= SEQUENCE {
        precedence          Precedence OPTIONAL,
                            -- по умолчанию к предшествующим ACIItem
        userClasses         UserClasses,
        grantsAndDenials    GrantsAndDenials }

UserPermission ::= SEQUENCE {
        precedence          Precedence OPTIONAL,
                            -- по умолчанию к предшествующим ACIItem
        protectedItems      ProtectedItems,
        grantsAndDenials    GrantsAndDenials }

AuthenticationLevel ::= CHOICE {
        basicLevels  SEQUENCE {
                level           ENUMERATED { none (0), simple (1), strong (2) },
                localQualifier  INTEGER OPTIONAL,
                signed          BOOLEAN DEFAULT FALSE },
        other        EXTERNAL }

GrantsAndDenials ::= BIT STRING {
        -- разрешения, которые могут использоваться совместно с
        -- любым компонентом из ProtectedItems
        grantAdd                (0),
        denyAdd                 (1),
        grantDiscloseOnError    (2),
        denyDiscloseOnError     (3),
        grantRead               (4),
        denyRead                (5),
        grantRemove             (6),
        denyRemove              (7),
```

```
        -- разрешения, которые могут использоваться совместно
        -- с этим компонентом статьи
        grantBrowse             (8),
        denyBrowse              (9),
        grantExport             (10),
        denyExport              (11),
        grantImport             (12),
        denyImport              (13),
        grantModify             (14),
        denyModify              (15),
        grantRename             (16),
        denyRename              (17),
        grantReturnDN           (18),
        denyReturnDN            (19),
        -- разрешения, которые могут использоваться совместно
        -- с любым компонентом из ProtectedItems, кроме статьи
        grantCompare            (20),
        denyCompare             (21),
        grantFilterMatch        (22),
        denyFilterMatch         (23),
        grantInvoke             (24),
        denyInvoke              (25) }

AttributeTypeAndValue ::= SEQUENCE {
        type            ATTRIBUTE.&id ({SupportedAttributes}),
        value           ATTRIBUTE.&Type({SupportedAttributes}{@type}) }

-- атрибуты --

accessControlScheme ATTRIBUTE ::= {
        WITH SYNTAX                     OBJECT IDENTIFIER
        EQUALITY MATCHING RULE          objectIdentifierMatch
        SINGLE VALUE                    TRUE
        USAGE                           directoryOperation
        ID                              id-aca-accessControlScheme }

prescriptiveACI ATTRIBUTE ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-prescriptiveACI }

entryACI ATTRIBUTE ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-entryACI }

subentryACI ATTRIBUTE ::= {
        WITH SYNTAX                     ACIItem
        EQUALITY MATCHING RULE          directoryStringFirstComponentMatch
        USAGE                           directoryOperation
        ID                              id-aca-subentryACI }

-- присвоения идентификаторов объектов --

-- атрибуты --

id-aca-accessControlScheme      OBJECT IDENTIFIER       ::=     { id-aca 1 }
id-aca-prescriptiveACI          OBJECT IDENTIFIER       ::=     { id-aca 4 }
id-aca-entryACI                 OBJECT IDENTIFIER       ::=     { id-aca 5 }
id-aca-subentryACI              OBJECT IDENTIFIER       ::=     { id-aca 6 }
```

-- схемы управления доступом –

```
basicAccessControlScheme          OBJECT IDENTIFIER    ::=    { id-acScheme 1 }
simplifiedAccessControlScheme     OBJECT IDENTIFIER    ::=    { id-acScheme 2 }
rule-based-access-control         OBJECT IDENTIFIER    ::=    { id-acScheme 3 }
rule-and-basic-access-control     OBJECT IDENTIFIER    ::=    { id-acScheme 4 }
rule-and-simple-access-control    OBJECT IDENTIFIER    ::=    { id-acScheme 5 }
```

END -- *BasicAccessControl*

Приложение F

Типы операционных атрибутов DSA на языке ASN.1

(|)

Данное Приложение содержит все определения типов и значений на языке ASN.1, представленных в пп. 23 и 24, в форме модуля ASN.1 **DSAOperationalAttributeTypes**.

```
DSAOperationalAttributeTypes {joint-iso-itu-t ds(5) module(1) dsaOperationalAttributeTypes(22) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

IMPORTS

-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        distributedOperations, id-doa, id-kmr, informationFramework, opBindingManagement,
        selectedAttributeTypes, upperBounds
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5 }

        ATTRIBUTE, MATCHING-RULE, Name
                FROM InformationFramework informationFramework

        OperationalBindingID
                FROM OperationalBindingManagement opBindingManagement

-- из Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4

        AccessPoint, DitBridgeKnowledge, MasterAndShadowAccessPoints
                FROM DistributedOperations distributedOperations

-- из Рек. МСЭ-Т X.520 | ИСО/МЭК 9594-6

        bitStringMatch, directoryStringFirstComponentMatch
                FROM SelectedAttributeTypes selectedAttributeTypes  ;

-- типы данных --

DSEType  ::=  BIT STRING {

        root            (0),    -- DSE корня --
        glue            (1),    -- представляет знания только имени --
        cp              (2),    -- префикс контекста --
        entry           (3),    -- статья объекта --
        alias           (4),    -- статья псевдонима --
        subr            (5),    -- подчиненная ссылка --
        nssr            (6),    -- неспецифическая подчиненная ссылка --
        supr            (7),    -- старшая ссылка --
        xr              (8),    -- перекрестная ссылка --
        admPoint        (9),    -- административная точка --
        subentry        (10),   -- подстатья --
        shadow          (11),   -- теневая копия --
        immSupr         (13),   -- непосредственно старшая ссылка --
        rhob            (14),   -- информация rhob --
        sa              (15),   -- подчиненная ссылка на статью псевдонима --
        dsSubentry      (16),   -- зависящая от DSA подстатья --
        familyMember    (17),   -- член семейства --
```

```
        ditBridge           (18),       -- ссылка – мост между DIT --
        writeableCopy       (19) }      -- перезаписываемая копия --

SupplierOrConsumer ::= SET {
        COMPONENTS OF       AccessPoint, -- поставщик или потребитель --
        agreementID         [3]     OperationalBindingID }

SupplierInformation ::= SET {
        COMPONENTS OF       SupplierOrConsumer,     -- поставщик --
        supplier-is-master  [4]     BOOLEAN DEFAULT TRUE,
        non-supplying-master [5] AccessPoint OPTIONAL }

ConsumerInformation ::= SupplierOrConsumer      -- потребитель --

SupplierAndConsumers ::= SET {
        COMPONENTS OF       AccessPoint,        -- поставщик --
        consumers           [3]     SET OF AccessPoint }

-- типы атрибутов --

dseType ATTRIBUTE ::= {
        WITH SYNTAX                     DSEType
        EQUALITY MATCHING RULE          bitStringMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-dseType }

myAccessPoint ATTRIBUTE ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-myAccessPoint }

superiorKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     AccessPoint
        EQUALITY MATCHING RULE          accessPointMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-superiorKnowledge }

specificKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        SINGLE VALUE                    TRUE
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation
        ID                              id-doa-specificKnowledge }

nonSpecificKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     MasterAndShadowAccessPoints
        EQUALITY MATCHING RULE          masterAndShadowAccessPointsMatch
        NO USER MODIFICATION            TRUE
        USAGE                           distributedOperation
        ID                              id-doa-nonSpecificKnowledge }

supplierKnowledge ATTRIBUTE ::= {
        WITH SYNTAX                     SupplierInformation
        EQUALITY MATCHING RULE          supplierOrConsumerInformationMatch
        NO USER MODIFICATION            TRUE
        USAGE                           dSAOperation
        ID                              id-doa-supplierKnowledge }
```

```
consumerKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                         ConsumerInformation
        EQUALITY MATCHING RULE              supplierOrConsumerInformationMatch
        NO USER MODIFICATION                TRUE
        USAGE                               dSAOperation
        ID                                  id-doa-consumerKnowledge }

secondaryShadows  ATTRIBUTE  ::= {
        WITH SYNTAX                         SupplierAndConsumers
        EQUALITY MATCHING RULE              supplierAndConsumersMatch
        NO USER MODIFICATION                TRUE
        USAGE                               dSAOperation
        ID                                  id-doa-secondaryShadows }

ditBridgeKnowledge  ATTRIBUTE  ::= {
        WITH SYNTAX                         DitBridgeKnowledge
        EQUALITY MATCHING RULE              directoryStringFirstComponentMatch
        NO USER MODIFICATION                TRUE
        USAGE                               dSAOperation
        ID                                  id-doa-ditBridgeKnowledge }

-- правила сопоставления --

accessPointMatch  MATCHING-RULE  ::= {
        SYNTAX      Name
        ID          id-kmr-accessPointMatch }

masterAndShadowAccessPointsMatch  MATCHING-RULE  ::= {
        SYNTAX      SET OF Name
        ID          id-kmr-masterShadowMatch }

supplierOrConsumerInformationMatch  MATCHING-RULE  ::= {
        SYNTAX      SET {
                ae-title                [0]     Name,
                agreement-identifier    [2]     INTEGER }
        ID          id-kmr-supplierConsumerMatch }

supplierAndConsumersMatch  MATCHING-RULE  ::= {
        SYNTAX      Name
        ID          id-kmr-supplierConsumersMatch }

-- присвоения идентификаторов объектов --

-- операционные атрибуты dsa --

id-doa-dseType                  OBJECT IDENTIFIER       ::=     {id-doa 0}
id-doa-myAccessPoint            OBJECT IDENTIFIER       ::=     {id-doa 1}
id-doa-superiorKnowledge        OBJECT IDENTIFIER       ::=     {id-doa 2}
id-doa-specificKnowledge        OBJECT IDENTIFIER       ::=     {id-doa 3}
id-doa-nonSpecificKnowledge     OBJECT IDENTIFIER       ::=     {id-doa 4}
id-doa-supplierKnowledge        OBJECT IDENTIFIER       ::=     {id-doa 5}
id-doa-consumerKnowledge        OBJECT IDENTIFIER       ::=     {id-doa 6}
id-doa-secondaryShadows         OBJECT IDENTIFIER       ::=     {id-doa 7}
id-doa-ditBridgeKnowledge       OBJECT IDENTIFIER       ::=     {id-doa 8}

-- правила сопоставления знаний --

id-kmr-accessPointMatch         OBJECT IDENTIFIER       ::=     {id-kmr 0}
id-kmr-masterShadowMatch        OBJECT IDENTIFIER       ::=     {id-kmr 1}
id-kmr-supplierConsumerMatch    OBJECT IDENTIFIER       ::=     {id-kmr 2}
id-kmr-supplierConsumersMatch   OBJECT IDENTIFIER       ::=     {id-kmr 3}

END  -- DSAOperationalAttributeTypes
```

Приложение G

Управление рабочим связыванием на языке ASN.1

(|)

Данное Приложение включает описания всех имеющих отношение к управлению рабочим связыванием типов, значений и информации на языке ASN.1, актуальных для настоящей спецификации Справочника, в форме модуля ASN.1 **OperationalBindingManagement**.

```
OperationalBindingManagement {joint-iso-itu-t ds(5) module(1) opBindingManagement(18) 5}
DEFINITIONS ::=
BEGIN

-- EXPORTS All --
-- Типы и значения, определенные в данном модуле, экспортируются для использования
-- в других модулях ASN.1, содержащихся в настоящих спецификациях Справочника, и для использования
-- в других приложениях, которые будут применять их для доступа к службам Справочника.
-- Другие приложения могут использовать их для собственных целей, но это не должно ограничивать
-- расширения и модификации, необходимые для поддержания или совершенствования службы Справочника.

IMPORTS

-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        directoryAbstractService, directoryShadowAbstractService, distributedOperations,
        directoryOSIProtocols, enhancedSecurity, hierarchicalOperationalBindings,
        commonProtocolSpecification
                FROM UsefulDefinitions {joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5}

        OPTIONALLY-PROTECTED-SEQ
                FROM EnhancedSecurity enhancedSecurity

        hierarchicalOperationalBinding, nonSpecificHierarchicalOperationalBinding
                FROM HierarchicalOperationalBindings hierarchicalOperationalBindings

-- из Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3

        CommonResultsSeq, directoryBind, securityError, SecurityParameters
                FROM DirectoryAbstractService directoryAbstractService

-- из Рек. МСЭ-Т X.518 | ИСО/МЭК 9594-4

        AccessPoint
                FROM DistributedOperations distributedOperations

-- из Рек. МСЭ-Т X.519 | ИСО/МЭК 9594-5

        id-err-operationalBindingError, id-op-establishOperationalBinding,
        id-op-modifyOperationalBinding, id-op-terminateOperationalBinding,
        OPERATION, ERROR
                FROM CommonProtocolSpecification commonProtocolSpecification

        APPLICATION-CONTEXT
                FROM DirectoryOSIProtocols directoryOSIProtocols

-- из Рек. МСЭ-Т X.525 | ИСО/МЭК 9594-9

        shadowOperationalBinding
                FROM DirectoryShadowAbstractService directoryShadowAbstractService        ;

-- привязывание и развязывание --

dSAOperationalBindingManagementBind OPERATION        ::=        directoryBind
```

```
-- операции, аргументы и результаты --

establishOperationalBinding  OPERATION  ::=  {
        ARGUMENT        EstablishOperationalBindingArgument
        RESULT          EstablishOperationalBindingResult
        ERRORS          {operationalBindingError | securityError}
        CODE            id-op-establishOperationalBinding }

EstablishOperationalBindingArgument  ::= OPTIONALLY-PROTECTED-SEQ  { SEQUENCE {
        bindingType         [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID           [1]     OperationalBindingID OPTIONAL,
        accessPoint         [2]     AccessPoint,
                -- симметричное, инициирует Role A или инициирует Role B --
        initiator CHOICE {
                symmetric                   [3]     OPERATIONAL-BINDING.&both.&EstablishParam
                                                        ({OpBindingSet}{@bindingType}),
                roleA-initiates             [4]     OPERATIONAL-BINDING.&roleA.&EstablishParam
                                                        ({OpBindingSet}{@bindingType}),
                roleB-initiates             [5]     OPERATIONAL-BINDING.&roleB.&EstablishParam
                                                        ({OpBindingSet}{@bindingType}) } OPTIONAL,
        agreement           [6]     OPERATIONAL-BINDING.&Agreement
                                            ({OpBindingSet}{@bindingType}),
        valid               [7]     Validity DEFAULT { },
        securityParameters  [8]     SecurityParameters OPTIONAL } }

OperationalBindingID  ::=  SEQUENCE {
        identifier      INTEGER,
        version         INTEGER }

Validity  ::=  SEQUENCE {
        validFrom       [0]     CHOICE {
                now             [0]     NULL,
                time            [1]     Time } DEFAULT now : NULL,
        validUntil      [1]     CHOICE {
                explicitTermination [0]     NULL,
                time                [1]     Time } DEFAULT explicitTermination : NULL }

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

EstablishOperationalBindingResult  ::=  OPTIONALLY-PROTECTED-SEQ  { SEQUENCE {
        bindingType [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID   [1]     OperationalBindingID OPTIONAL,
        accessPoint [2]     AccessPoint,
        -- симметричное, отвечает Role A или отвечает Role B --
        initiator CHOICE {
                symmetric           [3]     OPERATIONAL-BINDING.&both.&EstablishParam
                                            ({OpBindingSet}{@bindingType}),
                roleA-replies       [4]     OPERATIONAL-BINDING.&roleA.&EstablishParam
                                            ({OpBindingSet}{@bindingType}),
                roleB-replies       [5]     OPERATIONAL-BINDING.&roleB.&EstablishParam
                                            ({OpBindingSet}{@bindingType}) } OPTIONAL,
        COMPONENTS OF  CommonResultsSeq } }

modifyOperationalBinding  OPERATION  ::=  {
        ARGUMENT        ModifyOperationalBindingArgument
        RESULT          ModifyOperationalBindingResult
        ERRORS          { operationalBindingError | securityError }
        CODE            id-op-modifyOperationalBinding }

ModifyOperationalBindingArgument  ::=  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType         [0]     OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID           [1]     OperationalBindingID,
        accessPoint         [2]     AccessPoint OPTIONAL,
```

```
        -- симметричное, инициирует Role A или инициирует Role B --
        initiator CHOICE {
            symmetric           [3]  OPERATIONAL-BINDING.&both.&ModifyParam
                                     ({OpBindingSet}{@bindingType}),
            roleA-initiates     [4]  OPERATIONAL-BINDING.&roleA.&ModifyParam
                                     ({OpBindingSet}{@bindingType}),
            roleB-initiates     [5]  OPERATIONAL-BINDING.&roleB.&ModifyParam
                                     ({OpBindingSet}{@bindingType}) } OPTIONAL,
        newBindingID            [6]  OperationalBindingID,
        newAgreement            [7]  OPERATIONAL-BINDING.&Agreement
                                     ({OpBindingSet}{@bindingType})  OPTIONAL,
        valid                   [8]  Validity OPTIONAL,
        securityParameters
                                [9]  SecurityParameters OPTIONAL } }

ModifyOperationalBindingResult ::= CHOICE {
        null            [0]  NULL,
        protected       [1]  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
            newBindingID            OperationalBindingID,
            bindingType             OPERATIONAL-BINDING.&id ({OpBindingSet}),
            newAgreement            OPERATIONAL-BINDING.&Agreement
                                    ({OpBindingSet}{@.bindingType}),
            valid                   Validity OPTIONAL,
            COMPONENTS OF           CommonResultsSeq } } }

terminateOperationalBinding OPERATION ::= {
        ARGUMENT        TerminateOperationalBindingArgument
        RESULT          TerminateOperationalBindingResult
        ERRORS          {operationalBindingError | securityError}
        CODE            id-op-terminateOperationalBinding }

TerminateOperationalBindingArgument ::= OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
        bindingType             [0]  OPERATIONAL-BINDING.&id ({OpBindingSet}),
        bindingID               [1]  OperationalBindingID,
        -- симметричное, инициирует Role A или инициирует Role B --
        initiator CHOICE {
            symmetric           [2]  OPERATIONAL-BINDING.&both.&TerminateParam
                                     ({OpBindingSet}{@bindingType}),
            roleA-initiates     [3]  OPERATIONAL-BINDING.&roleA.&TerminateParam
                                     ({OpBindingSet}{@bindingType}),
            roleB-initiates     [4]  OPERATIONAL-BINDING.&roleB.&TerminateParam
                                     ({OpBindingSet}{@bindingType})} OPTIONAL,
        terminateAt             [5]  Time OPTIONAL,
        securityParameters      [6]  SecurityParameters OPTIONAL } }

TerminateOperationalBindingResult ::= CHOICE {
        null            [0]  NULL,
        protected       [1]  OPTIONALLY-PROTECTED-SEQ { SEQUENCE {
            bindingID               OperationalBindingID,
            bindingType             OPERATIONAL-BINDING.&id ({OpBindingSet}),
            terminateAt             GeneralizedTime OPTIONAL,
            COMPONENTS OF           CommonResultsSeq } } }

-- ошибки и параметры --

operationalBindingError ERROR ::= {
        PARAMETER       OPTIONALLY-PROTECTED-SEQ {
                            OpBindingErrorParam }
        CODE            id-err-operationalBindingError }

OpBindingErrorParam ::= SEQUENCE {
        problem                 [0]  ENUMERATED {
            invalidID               (0),
            duplicateID             (1),
            unsupportedBindingType  (2),
            notAllowedForRole       (3),
            parametersMissing       (4),
```

```
                roleAssignment          (5),
                invalidStartTime        (6),
                invalidEndTime          (7),
                invalidAgreement        (8),
                currentlyNotDecidable   (9),
                modificationNotAllowed  (10) },
        bindingType             [1]     OPERATIONAL-BINDING.&id ({OpBindingSet}) OPTIONAL,
        agreementProposal       [2]     OPERATIONAL-BINDING.&Agreement
                                        ({OpBindingSet}{@bindingType}) OPTIONAL,
        retryAt                 [3]     Time OPTIONAL,
        COMPONENTS OF                   CommonResultsSeq }


-- информационные классы объектов --

OPERATIONAL-BINDING ::= CLASS {
        &Agreement,
        &Cooperation        OP-BINDING-COOP,
        &both               OP-BIND-ROLE OPTIONAL,
        &roleA              OP-BIND-ROLE OPTIONAL,
        &roleB              OP-BIND-ROLE OPTIONAL,
        &id                 OBJECT IDENTIFIER UNIQUE }
WITH SYNTAX {
        AGREEMENT                       &Agreement
        APPLICATION CONTEXTS            &Cooperation
        [ SYMMETRIC                     &both ]
        [ ASYMMETRIC
                [ ROLE-A                &roleA ]
                [ ROLE-B                &roleB ] ]
        ID                              &id }

OP-BINDING-COOP ::= CLASS {
        &applContext        APPLICATION-CONTEXT,
        &Operations         OPERATION OPTIONAL }
WITH SYNTAX {
        &applContext
        [ APPLIES TO        &Operations ] }

OP-BIND-ROLE ::= CLASS {
        &establish                      BOOLEAN    DEFAULT FALSE,
        &EstablishParam                            OPTIONAL,
        &modify                         BOOLEAN    DEFAULT FALSE,
        &ModifyParam                               OPTIONAL,
        &terminate                      BOOLEAN    DEFAULT FALSE,
        &TerminateParam                            OPTIONAL }
WITH SYNTAX {
        [ ESTABLISHMENT-INITIATOR       &establish ]
        [ ESTABLISHMENT-PARAMETER       &EstablishParam ]
        [ MODIFICATION-INITIATOR        &modify ]
        [ MODIFICATION-PARAMETER        &ModifyParam ]
        [ TERMINATION-INITIATOR         &terminate ]
        [ TERMINATION-PARAMETER         &TerminateParam ] }

OpBindingSet        OPERATIONAL-BINDING ::= {
        shadowOperationalBinding |
        hierarchicalOperationalBinding |
        nonSpecificHierarchicalOperationalBinding }

END -- OperationalBindingManagement
```

Приложение H

Усиленная безопасность

(|)

Известно, что данный модуль содержит неправильные спецификации. Вследствие этого часть данного модуля не рекомендуется к применению. Не рекомендуемая часть отмечена комментариями ASN.1. В будущем издании либо будут исключены неправильные спецификации, либо будут включены обновленные спецификации.

```
EnhancedSecurity { joint-iso-itu-t ds(5) modules(1) enhancedSecurity(28) 5 }
DEFINITIONS IMPLICIT TAGS  ::=
BEGIN

-- EXPORTS All --

IMPORTS


-- из Рек. МСЭ-Т X.501 | ИСО/МЭК 9594-2

        authenticationFramework, basicAccessControl, certificateExtensions, id-at, id-avc, id-mr,
        informationFramework, upperBounds
                FROM UsefulDefinitions { joint-iso-itu-t ds(5) module(1) usefulDefinitions(0) 5 }

        Attribute, ATTRIBUTE, AttributeType, Context, CONTEXT, MATCHING-RULE, Name,
        objectIdentifierMatch, SupportedAttributes
                FROM InformationFramework informationFramework

        AttributeTypeAndValue
                FROM BasicAccessControl basicAccessControl

-- из Рек. МСЭ-Т X.509 | ИСО/МЭК 9594-8

        AlgorithmIdentifier, CertificateSerialNumber, HASH{}, SIGNED{}
                FROM AuthenticationFramework authenticationFramework

        GeneralName, KeyIdentifier
                FROM CertificateExtensions certificateExtensions

        ub-privacy-mark-length
                FROM UpperBounds upperBounds      ;

OPTIONALLY-PROTECTED { Type }  ::=  CHOICE {
        unsigned                Type,
        signed                  SIGNED {Type} }

OPTIONALLY-PROTECTED-SEQ { Type } ::= CHOICE {
        unsigned                Type,
        signed  [0]             SIGNED { Type } }

attributeValueSecurityLabelContext CONTEXT ::= {
        WITH SYNTAX     SignedSecurityLabel     -- Значению атрибута может быть присвоено не более
                                                -- одного контекста метки безопасности
        ID              id-avc-attributeValueSecurityLabelContext }

SignedSecurityLabel  ::=  SIGNED {SEQUENCE  {
        attHash         HASH {AttributeTypeAndValue},
        issuer          Name            OPTIONAL,  -- имя присваивающего метку органа
        keyIdentifier   KeyIdentifier OPTIONAL,
        securityLabel   SecurityLabel } }
```

```
SecurityLabel ::= SET {
        security-policy-identifier   SecurityPolicyIdentifier   OPTIONAL,
        security-classification      SecurityClassification     OPTIONAL,
        privacy-mark                 PrivacyMark                OPTIONAL,
        security-categories          SecurityCategories         OPTIONAL }
                (ALL EXCEPT ( {-- никакой, должен присутствовать, по крайней мере, один компонент -- } ) )

SecurityPolicyIdentifier ::= OBJECT IDENTIFIER

SecurityClassification ::= INTEGER {
        unmarked     (0),
        unclassified (1),
        restricted   (2),
        confidential (3),
        secret       (4),
        top-secret   (5) }

PrivacyMark ::= PrintableString (SIZE (1..ub-privacy-mark-length))

SecurityCategories ::= SET SIZE (1..MAX) OF SecurityCategory

clearance ATTRIBUTE ::= {
        WITH SYNTAX      Clearance
        ID               id-at-clearance }

Clearance ::= SEQUENCE {
        policyId         OBJECT IDENTIFIER,
        classList        ClassList                              DEFAULT {unclassified},
        securityCategories SET SIZE (1..MAX) OF SecurityCategory OPTIONAL }

ClassList ::= BIT STRING {
        unmarked     (0),
        unclassified (1),
        restricted   (2),
        confidential (3),
        secret       (4),
        topSecret    (5) }

SecurityCategory ::= SEQUENCE {
        type         [0]  SECURITY-CATEGORY.&id ({SecurityCategoriesTable}),
        value        [1]  EXPLICIT SECURITY-CATEGORY.&Type ({SecurityCategoriesTable} {@type}) }

SECURITY-CATEGORY ::= TYPE-IDENTIFIER

SecurityCategoriesTable SECURITY-CATEGORY ::= { ... }

attributeIntegrityInfo ATTRIBUTE ::= {
        WITH SYNTAX      AttributeIntegrityInfo
        ID               id-at-attributeIntegrityInfo }

AttributeIntegrityInfo ::= SIGNED { SEQUENCE {
        scope        Scope,                  -- Обозначает защищенные атрибуты
        signer       SignerOPTIONAL,         -- Имя держателя полномочий или источника данных
        attribsHash  AttribsHash } }         -- Хэш-значение защищенного атрибута

Signer ::= CHOICE {
        thisEntry    [0]  EXPLICIT ThisEntry,
        thirdParty   [1]  SpecificallyIdentified }

ThisEntry ::= CHOICE {
        onlyOne      NULL,
        specific     IssuerAndSerialNumber }

IssuerAndSerialNumber ::= SEQUENCE {
        issuer       Name,
        serial       CertificateSerialNumber }
```

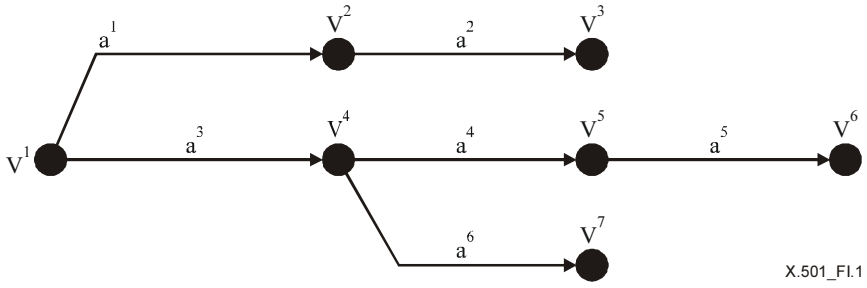
```
SpecificallyIdentified ::= SEQUENCE {
        name            GeneralName,
        issuer          GeneralName                 OPTIONAL,
        serial          CertificateSerialNumber     OPTIONAL }
        ( WITH COMPONENTS { ..., issuer PRESENT, serial PRESENT } |
        ( WITH COMPONENTS { ..., issuer ABSENT, serial ABSENT } ) )

Scope ::= CHOICE {
        wholeEntry   [0]    NULL,       -- Подпись защищает все значения атрибутов в данной статье
        selectedTypes       [1]     SelectedTypes
                            -- Подпись защщает все значения атрибутов избранных типов атрибутов
        }

SelectedTypes ::= SEQUENCE SIZE (1..MAX) OF AttributeType

AttribsHash ::= HASH { SEQUENCE SIZE (1..MAX) OF Attribute }
                    -- Тип и значения атрибута со связанными значениями контекста для избранной
                    -- области действия Scope

attributeValueIntegrityInfoContext CONTEXT ::= {
        WITH SYNTAX         AttributeValueIntegrityInfo
        ID                  id-avc-attributeValueIntegrityInfoContext }

AttributeValueIntegrityInfo ::= SIGNED { SEQUENCE {
        signer          SignerOPTIONAL,         -- Имя держателя полномочий или источника данных
        aVlHash         AVlHash } }             -- Хэш-значение защищенного атрибута

AVlHash ::= HASH { AttributeTypeValueContexts }
                    -- Тип и значение атрибута со связанными значениями контекста

AttributeTypeValueContexts ::= SEQUENCE {
        type            ATTRIBUTE.&id ({SupportedAttributes}),
        value           ATTRIBUTE.&Type ({SupportedAttributes}{@type}),
        contextList     SET SIZE (1..MAX) OF Context OPTIONAL }

-- Присвоения идентификаторов объектов --
-- атрибуты --

id-at-clearance                             OBJECT IDENTIFIER   ::=     {id-at 55}
-- id-at-defaultDirQop                      OBJECT IDENTIFIER   ::=     {id-at 56}
id-at-attributeIntegrityInfo                OBJECT IDENTIFIER   ::=     {id-at 57}
-- id-at-confKeyInfo                        OBJECT IDENTIFIER   ::=     {id-at 60}

-- правила сопоставления --

-- id-mr-readerAndKeyIDMatch                OBJECT IDENTIFIER   ::=     {id-mr 43}

-- контексты--

id-avc-attributeValueSecurityLabelContext   OBJECT IDENTIFIER   ::=     {id-avc 3}
id-avc-attributeValueIntegrityInfoContext   OBJECT IDENTIFIER   ::=     {id-avc 4}

END -- EnhancedSecurity
```

Приложение I

Математическое представление деревьев

(I)

Дерево – это множество точек, называемых *вершинами*, и множество направленных линий, называемых *дугами*, каждая дуга a ведет от вершины V к вершине V'. Например, дерево на рисунке I.1 имеет семь вершин, обозначенных от V^1 до V^7, и шесть дуг обозначенных от a^1 до a^6.



Две вершины V' и V считаются *начальной* и *конечной* вершинами, соответственно, дуги, идущей от V к V'. Например, V^2 и V^3 являются соответственно начальной и конечной вершинами дуги a^2. Несколько разных дуг могут иметь ту же начальную вершину, но не ту же конечную вершину. Например дуги a^1 и a^3 имеют ту же начальную вершину V^1, но никакие две из представленных на рисунке дуги не имеют ту же конечную вершину.

Вершина, которая не является конечной вершиной какой-либо дуги, часто называется *корневой* вершиной и, более неформально, "корнем" дерева. Например, на рисунке I.1 V^1 является корнем.

Вершина, которая не является начальной вершиной какой-либо дуги, часто называется *листовой* вершиной и, более неформально, "листом" дерева графа. Например, вершины V^3, V^6 и V^7 являются листьями.

Ориентированный маршрут от вершины V к вершине V' – это множество дуг $(a^1, a^2, ..., a^n)$ $(n \geq 1)$, так что V является начальной вершиной дуги a^1, V' является конечной вершиной дуги a^n, а конечная вершина дуги a^k также является начальной вершиной дуги a^{k+1} при $1 \leq k < n$. Например, ориентированным маршрутом от вершины V^1 до вершины V^6 является множество дуг (a^3, a^4, a^5). Термин "маршрут" должен пониматься как обозначение ориентированного маршрута от корня до вершины.

Приложение J

Критерии составления имени

(|)

Информационная инфраструктура имеет довольно общий характер и предусматривает произвольное разнообразие статей и атрибутов, существующих в пределах DIT. В силу того, что, как определено в настоящем документе, имена тесно связаны с маршрутами по DIT, это означает, что возможно произвольное разнообразие имен. В данном приложении предлагаются критерии, которые следует учитывать при составлении имен. Соответствующие критерии использовались при составлении рекомендованных форм имен, которые представлены в Рек. МСЭ-Т X.521 | ИСО/МЭК 9594-7. Предлагается также использовать эти критерии в соответствующих случаях при составлении имен объектов, к которым не применяются рекомендованные формы имен.

В настоящее время рассматривается только один критерий – критерий удобства для пользователя.

ПРИМЕЧАНИЕ. – Не все имена должны обязательно быть удобными для пользователя.

Далее в настоящем приложении рассматривается концепция удобства для пользователя применительно к именам.

Имена, с которыми непосредственно сталкивается человек, должны быть удобны для пользователя. Удобным для пользователя именем является имя, составленное с учетом точки зрения пользователя, а не компьютера. Это имя, которое является простым для выведения, запоминания и понимания с позиции человека, а не простым для понимания с позиции компьютера.

Цель концепции удобства для пользователя можно сформулировать более точно в форме двух следующих принципов:

– В общем случае для человека должно быть возможным правильно предугадать удобное для пользователя имя объекта на основе информации об объекте, которую он приобрел естественным образом. Например, должно быть возможным предугадать имя предпринимателя, имея о нем только информацию, непреднамеренно полученную в ходе обычных деловых отношений.

– Если имя объекта определено неоднозначно, Справочник должен выявить этот факт, а не делать вывод о том, что это имя обозначает один конкретный объект. Например, если два человека имеют одинаковую фамилию, эта фамилия сама по себе должна считаться неадекватным обозначением любого из объектов.

Цель удобства для пользователя подразделяется на следующие подцели:

a) Имена не должны искусственно устранять естественную неопределенность. Например, если два человека носят одинаковую фамилию "Jones", ни один из них не должен указываться как "WJones" или "Jones2". Вместо этого, соглашение об именовании должно предусматривать удобные для пользователя средства различения объектов. Например, это может быть запрос имени и второго инициала в дополнение к фамилии.

b) Имена должны допускать общие сокращения и общие варианты написания. Например, если индивидуум является сотрудником Conway Steel Corporation и наименование работодателя входит в его имя, любой вариант наименования – "Conway Steel Corporation", "Conway Steel Corp.", "Conway Steel" и "CSC" – должен быть достаточным для обозначения организации, о которой идет речь.

c) В некоторых случаях могут использоваться псевдонимы: для направления поиска к определенной статье, для большего удобства для пользователя или для сокращения области поиска. В следующих примерах показано использование псевдонимов в этих целей: Как показано на рисунке J.1, филиал в Осаке может также указываться именем { C = Япония, L = Осака, O = ABC, OU = отделение в Осаке }.

d) Если имена содержат несколько частей, и количество обязательных частей, и количество необязательных частей должно быть относительно небольшим и, следовательно, обеспечивать простоту запоминания.

e) Если имена содержат несколько частей, точный порядок следования частей не должен в общем случае иметь значения.

f) Удобные для пользователя имена не должны включать адресов компьютеров.

g) В определенных случаях для предоставления альтернативных имен может использоваться контекст. Например, как показано на рисунке J.2, индивидуум Jones может обозначаться как {O = "XYZ", OU = "Research", CN = "Jones"}, где контекстом является Язык = английский, и как {O = "XYZ", OU = "Recherche", CN = "Jones"}, где контекстом является Язык = французский.

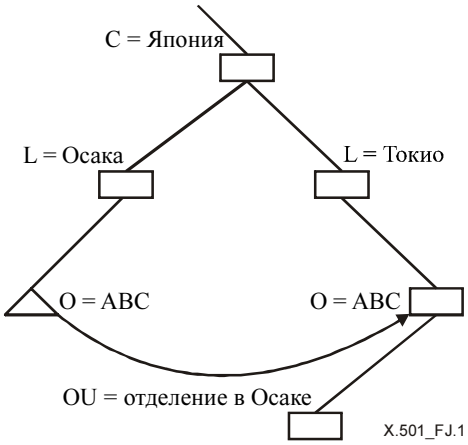


Рисунок J.1 – Пример синонимии

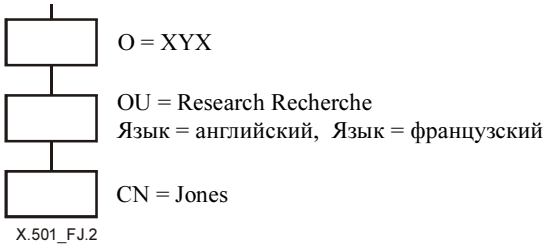


Рисунок J.2 – Пример вариантов контекста имени

Приложение K

Примеры различных аспектов схемы

(|)

K.1 Пример иерархии атрибутов

На рисунке K.1 показана простая иерархия значений родового атрибута **telephoneNumber**, и значения этого атрибута представлены содержащимися во внешнем множестве. Из родового типа выводятся два видовых типа атрибутов **workTelephoneNumber** и **homeTelephoneNumber**. Значения этих типов представлены содержащимися во внутренних множествах.

Значение типа **homeTelephoneNumber** содержится в обоих множествах: во внутреннем множестве, представляющем значения **homeTelephoneNumber**, и во внешнем множестве, представляющем значения **telephoneNumber**, но не во внутреннем множестве, представляющем значения **workTelephoneNumber**.

Может быть определено правило структуры DIT, которое разрешает статьям содержать значения всех трех типов, показанных на рисунке K.1. Может быть определено другое правило, позволяющее статьям содержать только значения типа **telephoneNumber**.

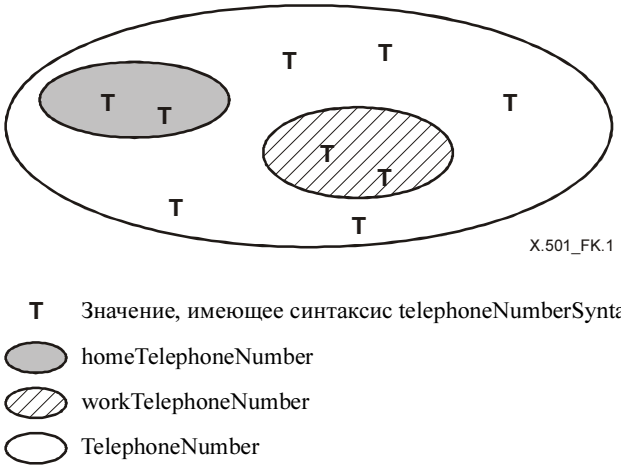


Рисунок K.1 – Иерархия значений атрибутов телефонных номеров

K.2 Пример спецификации поддерева

Ниже представлен пример иллюстрирующий спецификацию поддеревьев. Рассмотрим часть DIT, представленную на рисунке K.2.

Поддерево 1 и поддерево 2 специфицированы относительно административной точки, имя которой a. Идентификаторы b1, c2, d3 и т. д. представляют значения локальных имен относительно административной точки a.

Поддерево 1 может быть описано следующим образом:

```
subtree1 SubtreeSpecification ::= {
                        specificExclusions { chopBefore b1 } }
```

Поддерево 2 может быть описано следующим образом:

```
subtree2 SubtreeSpecification    ::= {
                        base b1 }
```

Предположим, что статьи, обозначенные на рисунке K.2 локальными именами e1, e2 и т. д., представляют статьи сотрудников организации. Для включения всех этих статей в административную область может быть определено уточнение поддерева:

```
subtree-refinement1 SubtreeSpecification ::= {
        specificationFilter
                item            id-oc-organizationalPerson }
```

Оно может быть далее уточнено для включения в поддерево 2 только сотрудников из этой организации:

```
subtree2-refinement SubtreeSpecification      ::=     {
        base                            b1,
        specificationFilter
                item            id-oc-organizationalPerson }
```

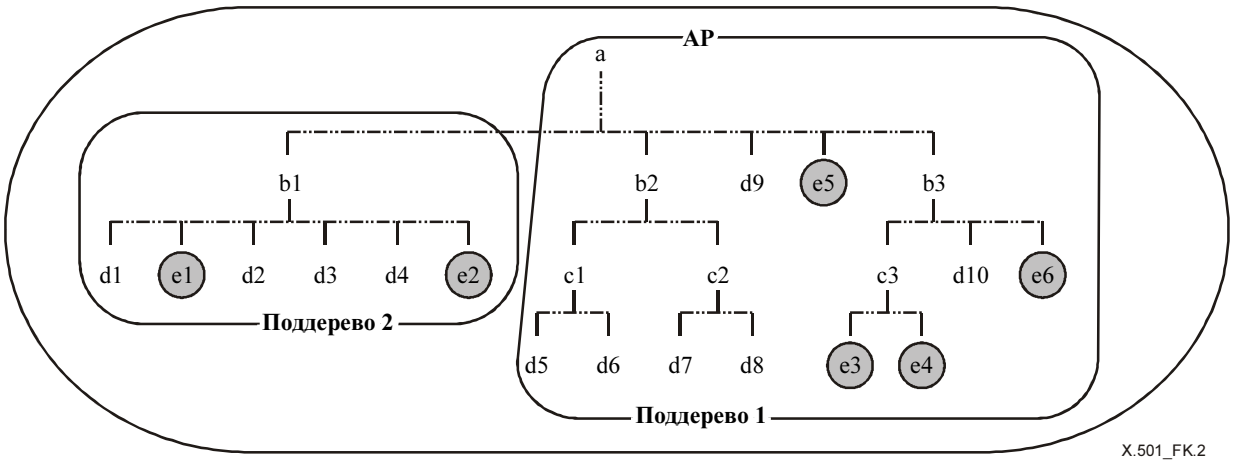


Рисунок K.2 – Пример спецификации поддерева

K.3 Спецификация схемы

K.3.1 Классы объектов и формы имен

В пределах конкретной административной области подсхемы используются следующие определенные в Рек. МСЭ-Т X.521 | ИСО/МЭК 9594-7 классы объектов:

- **organization**;
- **organizationalUnit**;
- **organizationalPerson**.

Форма имени не требуется для административной статьи, которая будет единственной статьей подсхемы класса объектов **organization**. Следующие определенные в Рек. МСЭ-Т X.521 | ИСО/МЭК 9594-7 формы имен используются для включения статей класса **organizationalUnit** и **organizationalPerson**:

- **orgNameForm**;
- **orgUnitNameForm**;
- **orgPersonNameForm**.

K.3.2 Правила структуры DIT

Для описания структуры дерева, показанной на рисунке K.3, определены следующие правила структуры. На рисунке K.3 показано, какое правило может использоваться для добавления статей в различных точках DIT.

```
rule-0 STRUCTURE-RULE::= {
        NAME FORM                       orgNameForm
        ID                              0 }

rule-1 STRUCTURE-RULE::= {
        NAME FORM                       orgUnitNameForm
        SUPERIOR RULES                  { rule-0 }
        ID                              1 }
```

```
rule-2 STRUCTURE-RULE::= {
        NAME FORM                   orgUniNameForm
        SUPERIOR RULES              { rule-1 }
        ID                          2 }

rule-3 STRUCTURE-RULE::= {
        NAME FORM                   orgUniNameForm
        SUPERIOR RULES              { rule-2 }
        ID                          3 }

rule-4 STRUCTURE-RULE::= {
        NAME FORM                   orgPersonNameForm
        SUPERIOR RULES              { rule-1, rule-2, rule-3 }
        ID                          4 }
```

Рисунок K.3 – Пример подсхемы

K.4 Правила контента DIT

Администратором подсхемы обязан выполнить следующие два условия в отношении включения дополнительной информации в статьи в административной области подсхемы:

- все статьи сотрудников организации **organizationalPerson** и подразделений организации **organizationalUnit** должны иметь атрибут **organizationalTelephoneNumber**. Этот атрибут должен возвращаться при получении Справочником запроса телефонного номера telephoneNumbers;
- все статьи **organizationalPerson** получают нового администратора атрибутов.

Для выполнения этих условий определяются следующие типы атрибутов:

```
manager  ATTRIBUTE  ::= {
        WITH SYNTAX                 BOOLEAN
        EQUALITY MATCHING RULE      booleanMatch
        SINGLE VALUE                TRUE
        ID                          id-ex-managerAttribute }

organizationalTelephoneNumber  ATTRIBUTE  ::= {
        SUBTYPE OF              telephoneNumber
        COLLECTIVE              TRUE
        ID                      id-ex-organizationalTelephoneNumber }
```

Для выполнения этих условий определяются следующие правила контента:

```
organizationRule  CONTENT-RULE  ::=  {
        STRUCTURAL OBJECT CLASS   id-oc-organization }

organizationalUnitRule  CONTENT-RULE  ::=  {
        STRUCTURAL OBJECT CLASS   id-oc-organizationalUnit
        MAY CONTAIN               { organizationalTelephoneNumber } }

organizationalPersonRule  CONTENT-RULE  ::=  {
        STRUCTURAL OBJECT CLASS   id-oc-organizationalPerson
        MUST CONTAIN              { manager }
        MAY CONTAIN               { organizationalTelephoneNumber } }
```

K.5 Использование контента DIT

Администратор подсхемы обязан реализовать стратегию международной организации, обязательным элементом которой является использование контекста **locale** для проведения различий между значениями типов должностей и атрибутов описания в пределах административной области организации. Кроме того, поскольку в организации регулярно происходит ротация обязанностей, желательно применение временного контекста в отношении должностей в статьях, относящихся к определенным сотрудникам.

Для выполнения этих условий определяются следующие правила контекста DIT:

```
descriptionContextRule          DIT-CONTEXT-USE-RULE ::= {
        ATTRIBUTE TYPE                  description
        MANDATORY CONTEXTS              { locale } }

titleContextRule                        DIT-CONTEXT-USE-RULE ::= {
        ATTRIBUTE TYPE                  title
        MANDATORY CONTEXTS              { localeContext }
        OPTIONAL CONTEXTS               { temporalContext } }
```

Приложение L

Обзор разрешений базового управления доступом

(|)

L.1 Введение

Данное Приложение носит информативный характер и его целью является обзор значений различных комбинаций операций, защищенных объектов и категорий разрешений. В случаях расхождения между данным обзором и спецификацией, представленной в теле настоящей спецификации Справочника, окончательным является нормативный текст тела спецификации.

Таблица L.1 относится к операциям Справочника, связанным с управлением доступом к статьям и атрибутам, и содержит обзор категорий разрешений, которые должны быть получены для успешного выполнения операций.

Таблица L.2 содержит обзор категорий разрешений **returnDN** и **discloseOnError**, и в ней показано, как предоставление разрешения или отказ в нем связаны с различными протокольными элементами.

Таблица L.3 содержит обзор семантики, связанной с разрешениями и отказами, выдаваемыми функциями управления доступом к статьям.

Таблица L.4 содержит обзор семантики, связанной с разрешениями и отказами, выдаваемыми функциями управления доступом к атрибутам.

L.2 Разрешения, требуемые для выполнения операций

L.1 –

Операция Справочника	**Требуемые разрешения на доступ к защищенному объекту статьи**	**Требуемые разрешения на доступ к защищенному объекту атрибута и значения атрибута**
Сравнение	*Read*	*Compare* для сравниваемого атрибута *Compare* для сравниваемого значения атрибута
Чтение	*Read* и *ReturnDN* для выделенного имени	*Read* для любой возвращаемой информации о типах атрибута *Read* для любых возвращаемых значений атрибута
Список	*Browse* и *ReturnDN* для всех подчиненных статей для которых возвращается RDN	Никакого разрешения не требуется
Поиск	*Browse* для статей, находящихся в области поиска, которые являются потенциальными кандидатами для отбора; *ReturnDN* для каждого возвращаемого выделенного имени	*FilterMatch* для информации о типе и значении атрибута, если имеется, используемых для оценки пункта фильтра как TRUE или FALSE *Read* для любой возвращаемой информации о типах атрибута *Read* для любых возвращаемых значений атрибута
Добавить статью	*Add*	*Add* для всех специфицированных типов атрибутов *Add* для всех специфицированных значений атрибутов
Удалить статью	*Remove*	Никакого разрешения не требуется
Изменить статью	*Modify*	*Add* для всех добавляемых атрибутов *Add* для всех добавляемых значений атрибутов *Remove* для всех удаляемых атрибутов *Remove* для всех удаляемых значений атрибутов
Изменить DN	*Rename* в исходном местоположении, если меняется только последнее RDN *Export* для перемещения поддерева из исходного местоположения *Import* для перемещения поддерева в местоположении назначения	Никакого разрешения не требуется

L.3 Разрешения, влияющие на ошибки

Таблица L.2 – Разрешения, влияющие на возвращаемые ошибки и имена

	Затронутые протокольные элементы	Значение
ReturnDN	**EntryInformation** **CompareResult** **ListResult** **SearchResult** **NameError** **ContinuationReference**	Если предоставлено, разрешает возвращение фактического выделенного имени. Если не предоставлено, запрещает возвращение фактического выделенного имени. Согласно местной стратегии вместо этого может возвращаться действительное имя псевдонима.
DiscloseOnError	**NameError** **UpdateError** **AttributeError** **SecurityError**	Если предоставлено, разрешает возвращение ошибки, которая может раскрыть факт существования защищенного объекта. Если не предоставлено, требует от Справочника скрыть факт существования защищенного объекта.

L.4 Разрешения уровня статьи

Таблица L.3 – Разрешения уровня статьи и их значение

Разрешение	Значение
Read	Если предоставлено, допускает выполнение с данной статьей операций Справочника чтения и сравнения, но не разрешает само по себе возвращения какой бы то ни было информации об атрибутах из этой статьи. Если не предоставлено, запрещает выполнение с данной статьей операций чтения и сравнения.
Browse	Если предоставлено, разрешает статье участвовать в качестве кандидата на отбор в области действия операции составления списка или поиска. Если не предоставлено, исключает данную статью из области действия операции составления списка или поиска.
Add	Если предоставлено, допускает добавление самой статьи, исключая ее атрибуты. Добавление имеет смысл только как предписывающая ACI. Если не предоставлено, запрещает добавление статьи.
Modify	Если предоставлено, допускает выполнение операций изменения со статьей. Если не предоставлено, запрещает выполнение операций изменения со статьей.
Remove	Если предоставлено, допускает удаление статьи, независимо от каких бы то ни было соображений, связанных с атрибутами. Если не предоставлено, запрещает удаление статьи.
Rename	Если предоставлено, разрешает, независимо от защиты атрибута или защиты значения атрибута, которая может быть применима к этой статье, изменение RDN статьи и, факультативно, удаление старого значения и добавление нового значения посредством операции изменения DN в зависимости от разрешений *Import* и *Export*, в соответствующем случае. Если не предоставлено, запрещает изменение RDN статьи.
Import	Если предоставлено, разрешает перемещение статьи, включая все подчиненные, на местоположение назначения в DIT в операции изменения DN. *Import* означает только предписывающую ACI. Если не предоставлено, запрещает перемещение статьи вместе с ее подчиненными в указанной точке в DIT с использованием операции изменения DN.
Export	Если предоставлено, допускает операцию изменения DN для перемещения статьи, включая все ее подчиненные, в обозначенную точку в ином месте в DIT. Запросчик должен иметь разрешение *import* в целевом местоположении. Если не предоставлено, запрещает перемещение статьи и ее подчиненных в единичной операции изменения DN.
ReturnDN	Если предоставлено, допускает возвращение выделенного имени статьи в результате операции. Если не предоставлено, запрещает возвращение выделенного имени. Согласно местной стратегии вместо этого может возвращаться действительное имя псевдонима.
DiscloseOnError	Если предоставлено, допускает возвращение ошибки, которая может раскрыть факт существования статьи. Если не предоставлено, требует, чтобы Справочник скрыл факт существования статьи. Разрешение *DiscloseOnError* само по себе не налагает запрета на возможность обнаружения статьи другими средствами, в отношении которых получены соответствующие разрешения.

L.5 Разрешения уровня атрибута

Таблица L.4 – Разрешения уровня атрибута и их значение

Разрешение	Категория защищенного объекта	Значение
Read	Тип атрибута	Если предоставлено, разрешает возвращение информации об этом типе атрибута в операции чтения или поиска. Являясь предписывающим в отношения чтения значений для этого атрибута, тем не менее, само по себе не предоставляет никаких прав в отношении каких бы то ни было значений этого атрибута. Если не предоставлено, запрещает возвращение информации об этом типе атрибута в операции чтения или поиска. Фактически, это также запрет и в отношении всех значений.
Read	Значение атрибута	Если предоставлено, разрешает возвращение обозначенного(ых) значения(ий) какого-либо типа атрибута в операции чтения или поиска. Само по себе не предоставляет каких бы то ни было прав в отношении этого типа атрибута. Для чтения значения также требуется разрешение *Read* для этого типа атрибута. Если не предоставлено, запрещает возвращение обозначенного(ых) значения(ий) этого типа атрибута в операции чтения или поиска. Само по себе не запрещает доступа к другим значениям или к самому типу атрибута.
Compare	Тип атрибута	Если предоставлено, разрешает тест типа атрибута операциями сравнения. Являясь предписывающим в отношении сравнения значений, само по себе не разрешает сравнения каких бы то ни было значений. Если не предоставлено, запрещает тест этого типа атрибута операциями сравнения. Запрещает также тест всех значений.
Compare	Значение атрибута	Если предоставлено, разрешает тест указанного значения указанного типа операциями сравнения. Не предоставляет каких бы то ни было прав в отношении самого типа атрибута. Для сравнения значения также требуется разрешение *Compare* для этого типа атрибута. Если не предоставлено, запрещает тест указанного значения операциями сравнения.
FilterMatch	Тип атрибута	Если предоставлено, допускает использование типа атрибута в оценке пункта фильтра поиска. Является предписывающим в отношении включения значений этого типа в оценки фильтра, но само по себе не предоставляет прав в отношении каких бы то ни было значений. Если не предоставлено, запрещает использование этого типа атрибута, включая все его значения, в оценке пункта фильтра.
FilterMatch	Значение атрибута	Если предоставлено, допускает использование значения(ий) атрибута в оценке пункта фильтра поиска. Для успешной оценки также требуется разрешение *FilterMatch* для типа атрибута. Если не предоставлено, запрещает использование значения(ий) в оценке пункта фильтра.
Add	Тип атрибута	Если предоставлено, допускает добавление обозначенного типа атрибута. Не предоставляет прав на добавление каких бы то ни было значений. Если не предоставлено, запрещает добавление обозначенного типа атрибута и, следовательно, каких бы то ни было значений.
Add	Значение атрибута	Если предоставлено, допускает добавление обозначенных значений атрибута. Не предоставляет прав на добавление самого типа. Напротив, не требуется прав на добавление типа атрибута для добавления значения к существующему атрибуту. Если не предоставлено, запрещает добавление обозначенных значений атрибута.
Remove	Тип атрибута	Если предоставлено, допускает удаление обозначенного типа атрибута и всех его значений в операции изменения. Само по себе не предоставляет права удалять отдельные значения. Если не предоставлено, запрещает удаление типа атрибута в операции изменения.
Remove	Значение атрибута	Если предоставлено, допускает удаление обозначенных значений атрибута в операции изменения. Для удаления последнего значения атрибута также требуется разрешение *Remove* для типа атрибута. Если не предоставлено, запрещает удаление обозначенных значений атрибута в операции изменения.

Таблица L.4 – Разрешения уровня атрибута и их значение

Разрешение	Категория защищенного объекта	Значение
DiscloseOnError	Тип атрибута	Если предоставлено, допускает возвращение ошибки, которая может раскрыть факт существования атрибута. Если не предоставлено, требует, чтобы Справочник скрыл факт существования атрибута. Разрешение *DiscloseOnError* само по себе не налагает запрета на возможность обнаружения типа атрибута другими средствами, в отношении которых получены соответствующие разрешения.
DiscloseOnError	Значение атрибута	Если предоставлено, допускает возвращение ошибки, которая может раскрыть факт существования значения. Если не предоставлено, требует, чтобы Справочник скрыл факт существования значения атрибута. Разрешение *DiscloseOnError* само по себе не налагает запрета на возможность обнаружения значения атрибута другими средствами, в отношении которых получены соответствующие разрешения.

Приложение M

Примеры управления доступом

(|)

M.1 Введение

Данное Приложение носит исключительно информативный и учебный характер. Оно включает три основные части: принципы построения, представляющие важность для архитектуры механизма базового управления доступом; подробный пример базового управления доступом; и краткий пример управления доступом на основе правил. Подробная информация о базовом управлении доступом и управлении доступом на основании правил представлена в пп. 18 и 19 настоящей спецификации Справочника и в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

M.2 Принципы построения для базового управления доступом

В данном подпункте представлены несколько наиболее важных принципов построения, используемых в архитектуре базового управления доступом. Для удобства ссылок каждый принцип имеет свое обозначение (например, PR-1).

PR-1: В общем случае разрешения, связанные с классами пользователей **UserClasses** более высокого уровня специфичности, переопределяют разрешения, связанные с **UserClasses** более низкого уровня специфичности. Этот принцип применяется, когда разрешения имеют тот же уровень приоритетности. Специфичность в контексте данного принципа является мерой того, насколько точно имя запросчика сопоставляется со спецификацией конкретного **UserClasses**; **allUsers** имеет самый низкий уровень специфичности, а **name** – самый высокий. Этот принцип изложен в п. 18.8.4 2). Он упрощает ситуации, когда стратегия в отношении разрешений по умолчанию (выраженная в форме имеющего более низкую специфичность **UserClasses**) селективно переопределяется разрешениями, связанными со спецификацией **UserClasses**, обладающего большей специфичностью.

PR-2: В общем случае разрешения, связанные с защищенными объектами **ProtectedItems** более высокого уровня специфичности, переопределяют разрешения, связанные с **ProtectedItems** более низкого уровня специфичности. Этот принцип применяется, когда разрешения имеют тот же уровень приоритетности и тот же уровень специфичности **UserClasses**. Специфичность в контексте данного принципа является мерой того, насколько точно спецификация **ProtectedItems** сопоставляется с конкретным объектом, к которому запрашивается доступ. Например, когда целевой защищенный объект является конкретным значением атрибута, **allAttributeValues** и **allUserAttributeTypesAndValues** имеют более низкий уровень специфичности по сравнению с **attributeValue**. Этот принцип изложен в п. 18.8.4 3). Он упрощает ситуации, когда стратегия в отношении разрешений по умолчанию (выраженная в форме имеющего более низкую специфичность **ProtectedItems**) селективно переопределяется разрешениями, связанными со спецификацией **ProtectedItems**, обладающего большей специфичностью.

PR-3: Базовое управление доступом моделируется как полностью независимое от процесса разрешения имен, за исключением случая разыменования псевдонимов. Решения по управлению доступом принимаются, за исключением случая разыменования псевдонимов, только после того как Справочник успешно определит местоположение подходящего DSA, содержащего целевой защищенный объект. Следствием этого является принцип, согласно которому базовое управление доступом не влияет на то, как Справочник генерирует подзапросы, и не влияет на то, как Справочник выполняет разрешение имен, связанное с подзапросами (за исключением случая разыменования псевдонимов).

PR-4: Пункт **Precedence** может использоваться для обеспечения реализации взаимосвязи между старшим и подчиненным органом таким образом, чтобы старший мог переопределять функции управления, установленные подчиненным. Например, пусть SE1 обозначает подстатью административной статьи для ACSA, например ACSA-1; аналогичным образом, пусть SE2 обозначает подстатью административной статьи для ACIA внутри ACSA-1. Ограничения на **Precedence** в SE2 могут быть специфицированы органом ACSA-1 таким образом, чтобы **prescriptiveACI** в SE2 не мог отменять предписывающую ACI в SE1. Также ограничения на **Precedence** для **entryACI** (в пределах ACSA-1) могут быть специфицированы таким образом, чтобы **entryACI** не мог отменять предписывающие функции управления, установленные в SE1. Этот принцип упрощает реализацию частичного делегирования полномочий.

ПРИМЕЧАНИЕ. – В настоящей спецификации Справочника предполагается, что реализуется метод ограничения приоритетности для полномочий, связанных с внутренними областями. Вместе с тем, в спецификации Справочника не определяется (и не описывается), как ограничивается приоритетность.

PR-5: Базовое управление доступом никогда не выдает разрешения на доступ пассивно, каждое решение о разрешении доступа основывается на описанной явным образом информации управления доступом. Следствием этого является принцип, согласно которому предоставление одной формы доступа ни при каких условиях не подразумевает разрешения на выполнение доступа другой формы. Эти принципы согласуются с более общим принципом построения системы безопасности, называемым *минимальная привилегия*.

PR-6: В отсутствие какого-либо **prescriptiveACI**, **entryACI** или **subentryACI**, служащего основанием для принятия решения, ACDF отказывает в доступе. При прочих равных параметрах решения отказы переопределяют разрешения (например, в ситуации, когда присутствуют **ACIItems**, которые предоставляют доступ, и другие, отказывающие в нем, и когда **Precedence** и специфичность одинаковы, преимущественную силу имеет отказ).

M.3 Введение к примеру

На рисунке M.1 показано поддерево DIT вымышленной компании – Z Computer Corporation (ZCC), которое используется на протяжении всего примера. Структура именования на рисунке M.1 соответствует предложениям,

изложенным в Приложении B Рек. МСЭ-Т X.521 | ИСО/МЭК 9594-7. Узел, имеющий выделенное имя {C=US, O=ZCC}, является административной статьей и автономной административной точкой для ZCC; следовательно, он определяет начало автономной административной области (AAA). Контентом AAA является неявно определенное поддерево, начинающееся в автономной административной точке и заканчивающееся либо в узлах-листьях, либо при появлении другой автономной административной точки. Поскольку других автономных административных точек ниже {C=US, O=ZCC} не существует, AAA содержит все узлы ниже {C=US} на рисунке M.1. Структурным классом объектов для {C=US, O=ZCC} является **organization**; он также имеет дополнительный класс объектов **certificationAuthority**. Дополнительный класс объектов присутствует для содействия поддержке жесткой аутентификации, если таковая необходима.

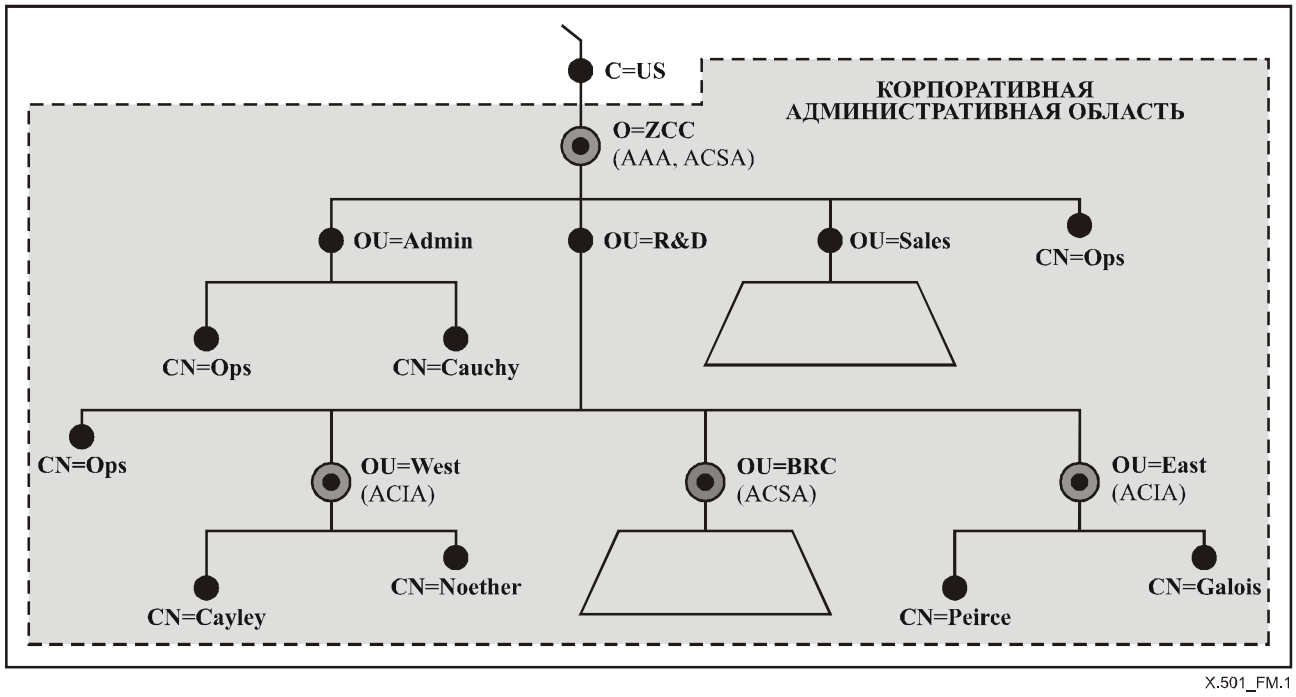


Рисунок M.1 – Ветвь DIT для Z Computer Corporation (ZCC)

Ниже автономной административной точки существуют три поддерева: администрация (Admin), НИОКР (R&D), и сбыт (Sales). Корнем каждого из поддеревьев является статья со структурным классом объектов **organizationalUnit** и дополнительным классом объектов **certificationAuthority**. Поддерево R&D содержит статьи структурного класса объектов **organizationalUnit**, соответствующего отдаленным офисам, под которыми существуют листовые объекты структурного класса **organizationalPerson**. Показаны только несколько репрезентативных объектов класса **organizationalPerson**. Все объекты структурного класса **organizationalUnit** имеют дополнительный класс объектов **certificationAuthority**. Все объекты структурного класса **organizationalPerson** имеют дополнительный класс объектов **strongAuthenticationUser**. Этот дополнительный класс объектов содействует поддержке жесткой аутентификации, если таковая требуется.

Объект с выделенным именем {C=US, O=ZCC, OU=Admin, CN=Ops} является объектом структурного класса объектов **groupOfUniqueNames**; значение его атрибута **uniqueMember** включает администраторов области имен. Одним из имен, которое он содержит, является имя {C=US, O=ZCC, OU=Admin, CN=Cauchy}. Существуют два других таких объектов: {C=US, O=ZCC, OU=R&D, CN=Ops} содержит членов, ответственных за поддержание статей в поддереве R&D; и {C=US, O=ZCC, CN=Ops} содержит членов, ответственных за статьи, являющиеся непосредственно подчиненными относительно {C=US, O=ZCC}. Пользователем с выделенным именем {C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley} является членом двух последних групп.

Две трапеции на рисунке M.1 представляют частичные поддеревья, детали которых не важны для данного примера.

M.4 Стратегия, влияющая на определение специальных и внутренних областей

Для поддержки базового управления доступом в пределах AAA могут быть образованы два типа административных областей: специальная область управления доступом (ACSA) и внутренняя область управления доступом (ACIA). Административная область любого из этих типов образуется путем присвоения соответствующего значения атрибуту **administrative-role** в административной статье, которая должна служить корневой вершиной для этой области. Контентом ACSA является неявным образом определенное поддерево, которое начинается в корневой вершине и продолжается до листовых объектов или до корня другой ACSA. Также, граница ACSA никогда не

распространяется за нижнюю границу охватывающей ее ААА. В случае ACIA нижняя граница определяется либо встретившейся листовой статьей, либо границей охватывающей ACSA. Вложенные ACIA имеют ту же нижнюю границу и эта граница та же, что и нижняя граница охватывающей ACSA.

Корпорация ZCC внедрила стратегию, которая влияет на количество и типы административных областей, требуемых в пределах ААА. Первая такая стратегия заключается в том, что подразделению организации, называемому Консорциумом базовых исследований (BRC), делегируются всеобъемлющие полномочия на установление предписывающих атрибутов управления доступом для управления статьями в поддереве с корневой вершиной {C=US, O=ZCC, OU=R&D, OU=BRC}. Для упрощения реализации этой стратегии корень {C=US, O=ZCC, OU=R&D, OU=BRC} обозначен как административная статья с административной функцией **id-ar-accessControlSpecificArea**. Нижняя граница результирующей ACSA определяется неявным образом появлением листовых статей.

ПРИМЕЧАНИЕ. – ACSA воплощает концепцию всеобъемлющего делегирования полномочий, поскольку решения о доступе зависят от ACI, появляющейся внутри ACSA, которая содержит целевой защищенный объект, и не затрагиваются ACI, появляющейся за пределами ACSA.

Кроме того, описанная выше ACSA – это единственный пример всеобъемлющего делегирования полномочий по управлению доступом в пределах ZCC. Вместе с тем, одним из следствий административной модели Справочника является то, что если в ААА существует по крайней мере одна ACSA, все (и каждый) объекты в ААА должны содержаться в одной (и только одной) ACSA. Это требование может быть сформулировано более четко на языке теории множеств, где каждая ACSA и связанная ААА рассматриваются как множества статей: пересечение множеств каждой пары ACSA является пустым, а объединение множеств всех ACSA равняется этой ААА. Следовательно, в примере необходима по крайней мере одна дополнительная ACSA для содержания объектов, находящихся в ААА, но за пределами поддерева BRC. В силу того, что это единственный пример всеобъемлющего делегирования в пределах ААА, корень ААА также представляет собой начало ACSA, которая содержит все статьи в ААА за исключением содержащихся в поддереве BRC.

Результирующие ACSA на рисунке M.2 обозначены ACSA-1 и ACSA-2. Следует также отметить на рисунке M.2, что поскольку административные области (неявно определенные) являются поддеревьями, каждая область включает свою корневую вершину. Контент ACSA-1 распространяется вниз от ее корня до листовых объектов или до появления корневой вершины другой ACSA (как в {C=US, O=ZCC, OU=R&D, OU=BRC}). В этом примере не существует автономных административных точек ниже {C=US, O=ZCC} и, следовательно, нижняя граница ААА полностью определяется листовыми объектами. Последующая часть данного примера будет посвящена управлению доступом в пределах ACSA-1 (ACSA-2 более не рассматривается). Также для простоты в данном примере не рассматривается управление подчиненными под {C=US, O=ZCC, OU=Sales}.

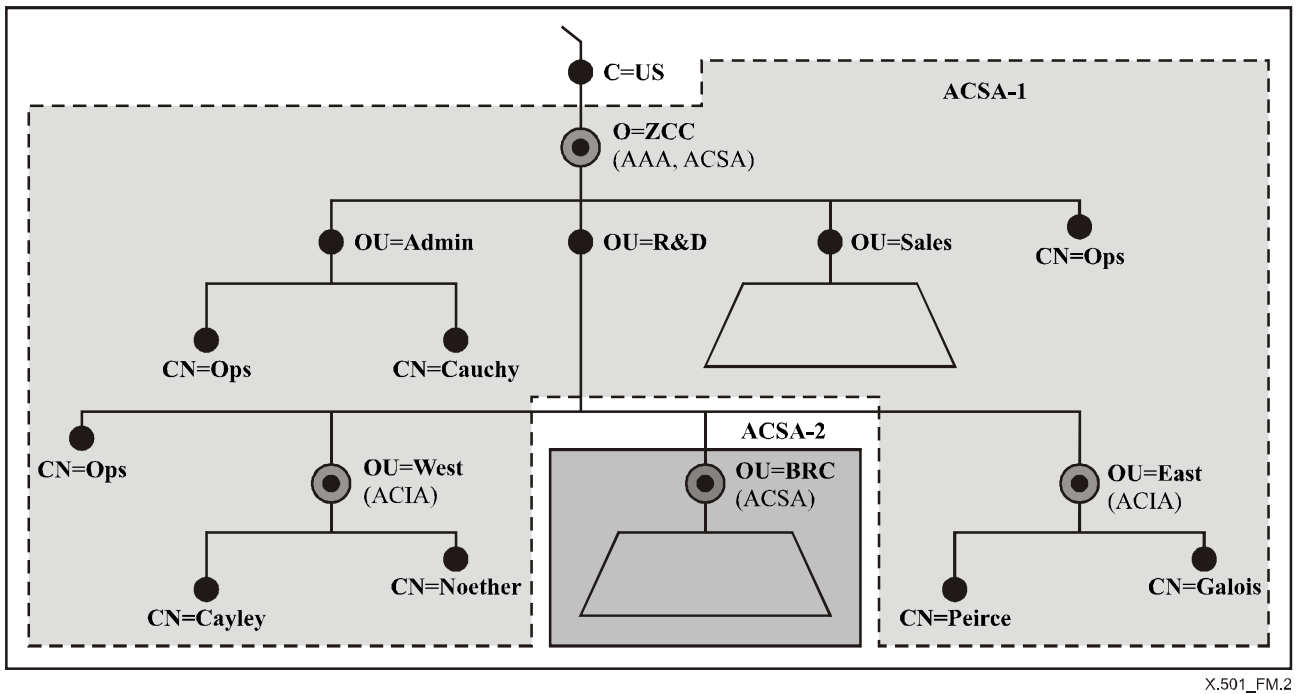


Рисунок M.2 – Специальные области управления доступом

Еще одна стратегия ZCC, затрагивающая определение административных областей, состоит в том, что подразделению организации Western R&D делегируются частичные полномочия в отношении операционных атрибутов управления доступом, влияющие на статьи в поддереве с корневой вершиной {C=US, O=ZCC, OU=R&D, OU=West}. Стратегия оптимально реализуется путем превращения корня поддерева R&D West в административную точку с административной функцией **id-ar-accessControlInnerArea**. Это означает, что предписывающее управление доступом для этого поддерева в общем случае будет сочетанием функций управления, определенных в подстатьях корня этого поддерева, и функций управления, определенных в подстатьях корня охватывающей ACSA (ACSA-1). Контентом результирующей ACIA является неявно определенное поддерево с корнем в {C=US, O=ZCC, OU=R&D, OU=West}, которое продолжается вниз до появления корневых объектов. Поскольку ACIA – это поддерево, ее контент включает корневую вершину этого поддерева.

Аналогичная стратегия применяется в отношении подразделения организации R&D East. Соответствующая ACIA имеет корневую вершину в {C=US, O=ZCC, OU=R&D, OU=East}. На рисунке M.3 показаны две ACIA в пределах ACSA-1. ACIA для R&D West обозначена как ACIA-1; ACIA для R&D East обозначена как ACIA-2.

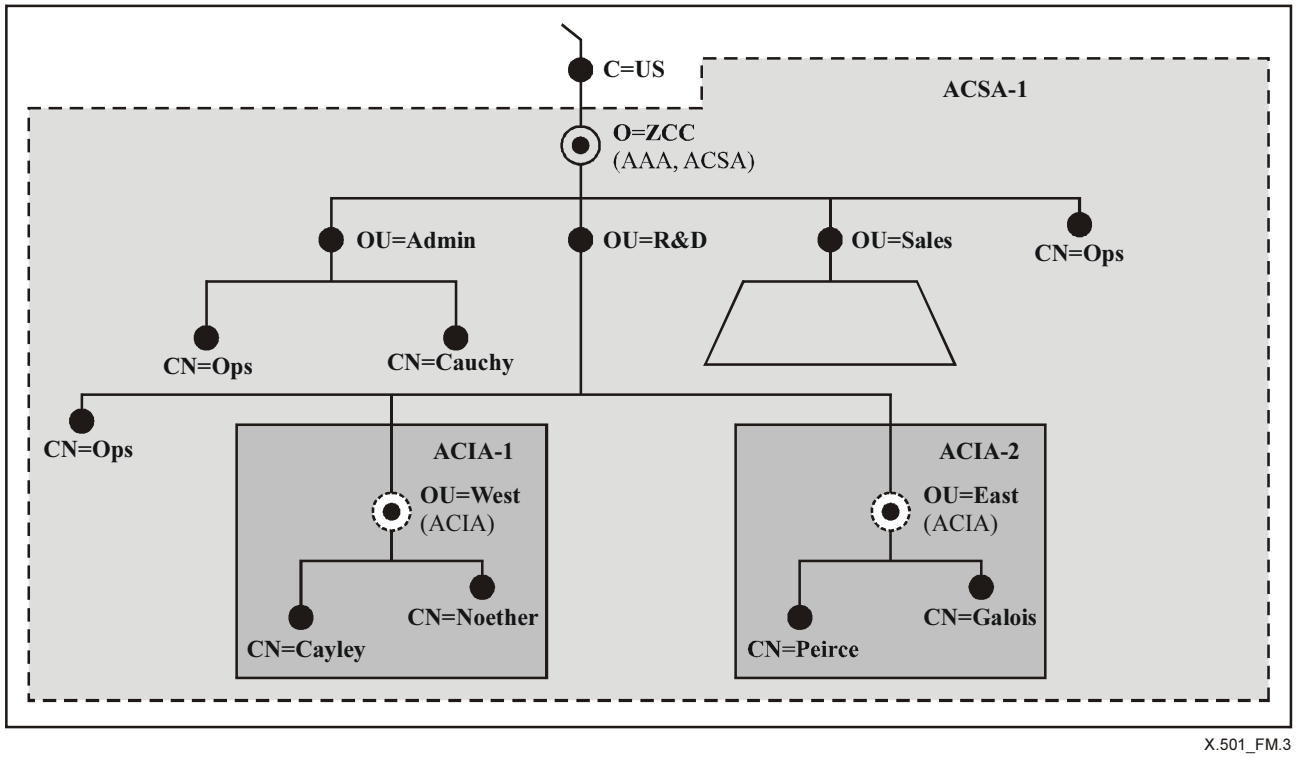


Рисунок M.3 – Внутреннее управление доступом

M.5 Стратегия, влияющая на определение DACD

Предписывающие функции управления определяются в подстатьях (с классом объектов **accessControlSubentry**) административных статей управления доступом. Каждая такая подстатья имеет связанный атрибут **subtreeSpecification**, который определяет совокупность статей в пределах области действия этой подстатьи. Содержащиеся в этой области действия статьи могут образовать поддерево или уточнение поддерева. В контексте базового управления доступом область действия подстатьи управления доступом называется доменом управления доступом к Справочнику (DACD). Органы безопасности, использующие базовое управление доступом, должны не допускать смешивания понятия административной области и понятия DACD. Данный подпункт начинается с анализа разницы и взаимосвязи между административными областями и DACD, после чего рассматривается стратегия ZCC, вызывающая появление отдельных DACD.

Основные различия между административными областями и DACD можно свести к следующему.

– Административная область – это *неявно определенное* поддерево с корнем в административной статье, продолжающееся вниз, как описано в M.4. Такую область называют определенной неявным образом в силу отсутствия в Справочнике стандартизованного атрибута, описывающего ее границу; для определения границы административной области логически анализируется DIT. Административная область никогда не бывает уточнением поддерева.

ПРИМЕЧАНИЕ 1. – Одним из следствий способа определения административных областей является то, что для каждой статьи, затрагиваемой базовым управлением доступом, должна существовать строго одна ACSA, содержащая эту статью (даже если эта статья не включена ни в один DACD в пределах ACSA).

- DACD является поддеревом или уточнением поддерева, явно описанным в атрибуте **subtreeSpecification** подстатьи с классом объектов **accessControlSubentry**.
- ACSA и ACIA используются функцией ACDF для определения того, какие предписывающие функции управления доступом (т. е. какие подстатьи управления доступом) потенциально влияют на результат данного решения по управлению доступом. ACSA используются для реализации всеобъемлющего делегирования полномочий в отношении управления доступом. ACIA используются для реализации частичного делегирования полномочий в отношении управления доступом.
- DACD используется для определения того, какие статьи (или потенциальные статьи) могут быть затронуты связанной подстатьей управления доступом.

К важным аспектам административных областей и DACD и взаимосвязи между ними также относятся следующие замечания.

- Каждый DACD определяется в подстатье конкретной административной статьи, которая, в свою очередь, является корневой вершиной некоторой административной области. Такая связь между DACD, подстатьей, административной статьей и административной областью позволяет составлять для данного DACD определение *связанной административной области* (см. п. M.5.1). Множество статей, содержащихся в DACD, может быть собственным или несобственным подмножеством статей, которые содержатся в связанной административной области.

  ПРИМЕЧАНИЕ 2. – Термины *собственное подмножество* и *несобственное подмножество* заимствованы из математической теории множеств. Множество *A* является собственным подмножеством множества *B*, если и только если каждый элемент множества *A* также является элементом множества *B* и существует по крайней мере один элемент множества *B*, который не является элементом множества *A*. Множество *A* является несобственным подмножеством множества *B*, если и только если оба множества содержат абсолютно те же элементы.

- В случае если множество статей в DACD является несобственным подмножеством статей в связанной административной области, считается, что DACD и административная область *конгруэнтны*. Вместе с тем, даже при наличии такой конгруэнтности DACD и административная область служат принципиально разным целям (области определяют, каким подстатьям разрешается потенциально влиять на результат конкретного решения по управлению доступом, в то время как каждый DACD описывает, какие именно статьи затрагиваются предписывающими функциями управления в данной подстатье).
- DACD никогда не может содержать статей, которые находятся за пределами связанной административной области.
- Функция ACDF создается устойчивой в том смысле, что даже если **subtreeSpecification**, определяющая DACD, имеет в рамках своей области действия статьи за пределами связанной административной области, решения по управлению доступом, касающиеся этих статей, не затрагиваются. Этот аспект устойчивости присутствует в процедуре ACDF для определения того, какие подстатьи потенциально затрагивают данное решение (см. пп. 18.3.2 и 18.8.1 d)).
- DACD, определенные в подстатьях той же административной статьи, могут произвольно перекрываться в пределах общей связанной административной области.
- ACSA никогда не перекрываются; каждая ACIA является *надлежаще вложенной* в пределах ACSA. Надлежаще вложенная означает, что статьи в охваченной области образуют собственное подмножество статей в охватывающей области. ACIA также может содержать одну или более надлежаще вложенных ACIA.
- Если административные области являются вложенными, DACD, связанные с охватывающей областью, могут произвольно перекрывать DACD, связанные с любой охваченной областью. Охватывающей областью может быть ACSA или ACIA, в то время как охваченной областью всегда является ACIA.

Каждый DACD связан с аспектом стратегии, который влияет на одну или более статей или потенциальных статей. Статьи, затрагиваемые конкретным аспектом стратегии, образуют DACD. DACD для конкретного аспекта стратегии должен быть связан с административной областью, управляемой органом, ответственным за реализацию этого аспекта стратегии.

В примере имеются несколько аспектов стратегии, подлежащие реализации органом, который управляет ACSA-1. Существуют, например, функции управления "по умолчанию", которые применяются к объектам на протяжении всей ACSA-1. Таким функциям управления присваиваются приоритет и уровень специфичности, которые позволяют легко переопределять их другими предписывающими функциями управления или атрибутами **entryACI**. Существует также стратегия, которая применяется только к непосредственно подчиненным относительно {C=US, O=ZCC} (в пределах ZCC такие статьи называются статьями *административного уровня*). Существует и стратегия, которая применяется только к статьям, имеющим структурный класс объектов **organizationalPerson**.

Все статьи в ACSA-1 включаются в DACD, связанный с функциями управления по умолчанию. Следовательно, DACD определяется как поддерево с вершиной **base** в {C=US, O=ZCC} и спецификацией **chop**, которая исключает поддерево с корнем в {C=US, O=ZCC, OU=R&D, OU=BRC}. Результирующий DACD конгруэнтен ACSA-1 и показан на рисунке M.4 как DACD-1.

ПРИМЕЧАНИЕ 3. – Значение в данном контексте термина *конгруэнтный* см. в. п. 18.3.2 g).

Также в пределах ACSA-1 DACD для управления статьями **organizationalPerson** является уточнением поддерева с вершиной **base** в {C=US, O=ZCC} и **specificationFilter,** включающим только статьи с **objectClass** класса **organizationalPerson** (см. **subtree-refinement1** в п. K.2). Этот DACD на рисунке M.4 показан как DACD-2

Третий DACD в пределах ACSA-1 связан с управляющими статьями административного уровня (т. е. непосредственно подчиненными, не являющимися подстатьями статьями корня организации). Этот DACD является (усеченным) поддеревом с вершиной **base** в {C=US, O=ZCC} и спецификацией **chop**, которая включает только непосредственно подчиненных, не являющихся подстатьями, относительно {C=US, O=ZCC}. Этот DACD на рисунке M.4 показан как DACD-5.

Для ACIA-1 требуется DACD для работы с аспектом стратегии, который был делегирован органу, управляющему внутренней областью. Делегированные полномочия затрагивают только подчиненных относительно {C=US, O=ZCC, OU=R&D, OU=West}, и, следовательно, этот DACD неконгруэнтен ACIA-1. На рисунке M.4 этот DACD обозначен как DACD-3.

Для ACIA-2 требуется только один DACD; вместе с тем делегированные полномочия затрагивают все статьи в ACIA-2, и, следовательно, этот DACD конгруэнтен ACIA-2. На рисунке M.4 этот DACD обозначен как DACD-4.

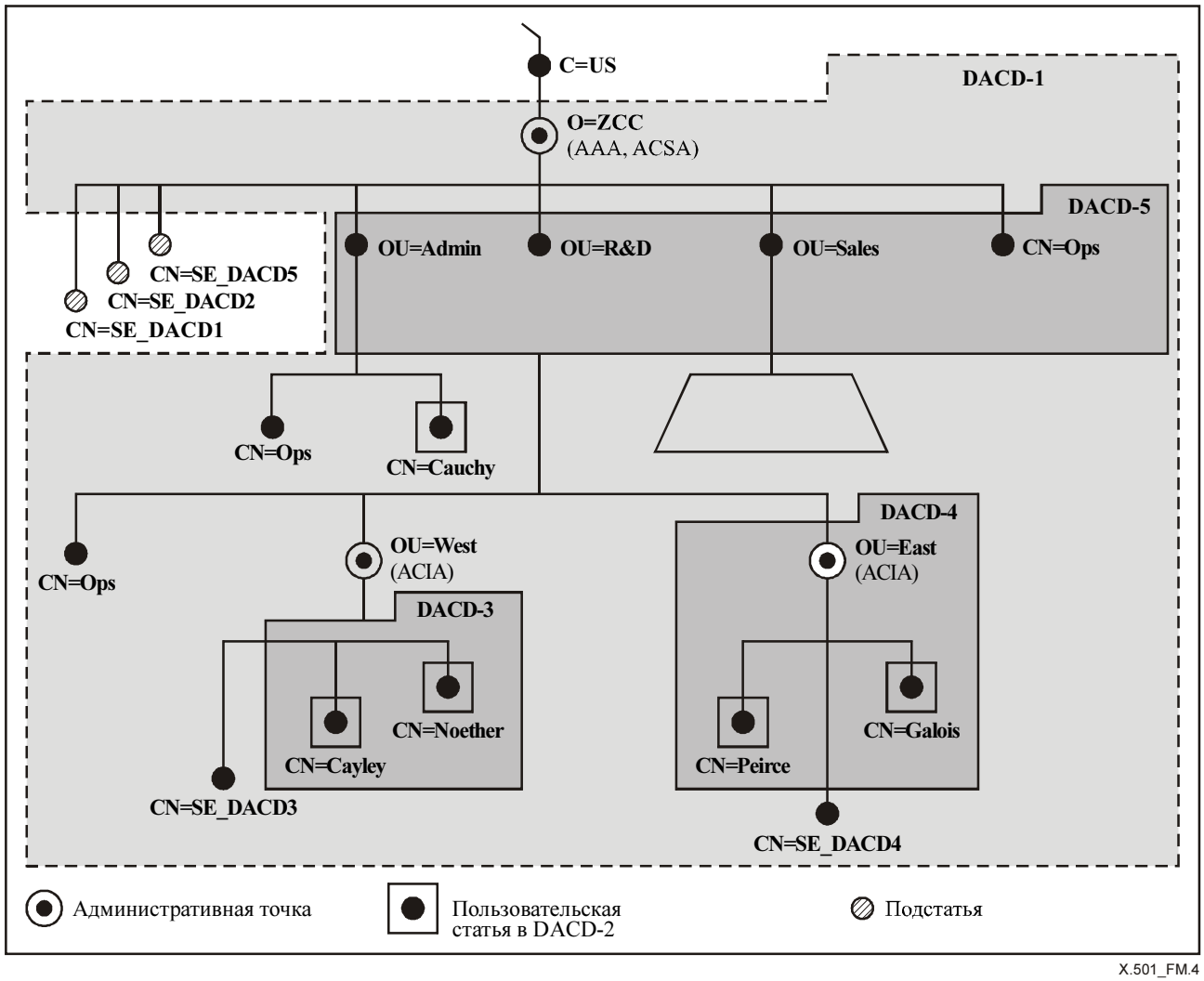


Рисунок M.4 – Домены управления доступом к Справочнику

M.5.1 Административная область, связанная с каждым DACD

Все используемые в примере подстатьи показаны на рисунке M.4. В данном подпункте содержится резюме местоположения всех подстатей и указана административная область, связанная с каждым DACD.

DACD-1, DACD-2 и DACD-5 определяются в подстатьях {C=US, O=ZCC}, которая является административной статьей, определяющей корневую вершину ACSA-1. Следовательно, существуют три DACD, которые считаются

связанными с ACSA-1. Именем подстатьи, определяющей DACD-1, является {C=US, C=ZCC, CN=SE_DACD1}. Другие подстатьи имеют аналогичные имена, которые указывают, какой DACD они определяют.

DACD-3 определяется в подстатье {C=US, O=ZCC, OU=R&D, OU=West}, которая представляет собой административную статью, являющуюся корневой вершиной ACIA-1. Следовательно, DACD-4 является связанным с ACIA-1.

DACD-4 определяется в подстатье {C=US, O=ZCC, OU=R&D, OU=East}, которая является административной статьей, определяющей корневую вершину ACIA-2. Следовательно, DACD-4 является связанным с ACIA-2.

M.6 Стратегия, выраженная в атрибутах prescriptiveACI

В данном подпункте содержится подробное описание стратегии управления доступом, применимой к каждому DACD в ACSA-1. Обсуждаемая в этом примере стратегия должна рассматриваться как неполная, т. е. упрощенная в целях простоты представления. В частности, не рассматривается управление паролями, поскольку в целом пароли представляют специальный случай управления доступом; также не рассматриваются разрешения категории *DiscloseOnError* или *ReturnDN* permissions.

Стратегия, рассматриваемая в данном подпункте, представлена в виде *фрагментов стратегии*, что упрощает понимание использования атрибутов **prescriptiveACI** для коллективного обеспечения реализации общей стратегии. Каждый фрагмент снабжен меткой для ссылок, которая используется в последующих подпунктах; метки имеют форму PF-*n*. где *n* – следующее целое число. Для каждого DACD существует также индикация того, как применимые фрагменты стратегии могут быть выражены в виде одной или более подстатей (содержащих атрибут **prescriptiveACI**).

M.6.1 Атрибут prescriptiveACI для DACD-1

Одной из основных задач DACD-1 является обеспечение реализации фрагментов стратегии, которые связаны с функциями управления доступом "по умолчанию". Такие фрагменты стратегии обеспечивают ограничительные функции управления, которые применяются в отсутствие иной функции управления, имеющей более высокий уровень приоритетности или специфичности. Специфичность поясняется в принципах построения PR-1 и PR-2 в п. M.2.

Корпорация ZCC объявила свою стратегию в отношении общего доступа в форме правил стратегии по умолчанию, которая может быть переопределена для некоторых статей, требующих более жесткого управления. Стратегия по умолчанию излагается в PF-1 и PF-2. Заметим, что, согласно стратегии ZCC, на реализующих эту стратегию лежит ответственность за обеспечение того, чтобы любое отклонение от правил по умолчанию было более ограничительным, чем правила по умолчанию.

PF-1: Персонал организации следует отличать от населения в целом. Права доступа общего пользования должны в целом ограничиваться согласно пп. a) и b), ниже; однако, общий доступ может быть более ограниченным для конкретных статей (он ни при каких условиях не бывает менее ограниченным).

a) Статьи могут обнаруживаться по общему имени. Поиск по общему имени разрешается для согласования приблизительного совпадения и альтернативных имен. В частности, поиск по телефонным номерам не разрешен населению в целом, но разрешен находящимся внутри организации. Результаты поиска могут раскрывать все значения **commonName**.

b) Единственными атрибутами общего пользования являются **commonName**, **telephoneNumber**, компоненты из **postalAttributeSet** и **facsimileTelephoneNumber**.

PF-2: Доступ общего пользования может не предусматривать аутентификацию, но идентификационная информация должна быть представлена.

ZCC также применяет правила стратегии по умолчанию для выражения своей общей стратегии в отношении доступа для персонала. Отклонения от правил стратегии по умолчанию могут быть более ограничительными или менее ограничительными. Стратегия по умолчанию изъявляется в PF-3 и PF-4.

PF-3: Персонал в общем случае имеет доступ для чтения и поиска в отношении большинства атрибутов и большинства статей.

PF-4: Для доступа персонала требуется простая аутентификация, которая не изменяет (никаким образом) контент ACSA-1.

Также существуют фрагменты стратегии, применяемые к DACD-1, которые не интерпретируются как стратегия по умолчанию. В PF-5 и PF-6 приводятся два примера таких фрагментов; они относятся к административному управлению статьями.

PF-5: {C=US, O=ZCC, CN=Cauchy} является "суперпользователем", уполномоченным получать доступ ко всем данным и выполнять любые необходимые операции.

PF-6: Для какого бы то ни было изменения контента ACSA-1 требуется строгая аутентификация.

Для реализации фрагментов стратегии для DACD-1 могут использоваться одна или более подстатей {C=US, O=ZCC}. Каждая такая подстатья имеет ту же спецификацию **subtreeSpecification** с **base** в {C=US, O=ZCC} и спецификацию **chop** для исключения поддерева OU=BRC. Каждая такая статья должна также содержать атрибут **prescriptiveACI**, который реализует некоторое подмножество фрагментов стратегии для DACD-1. Для целей

данного примера предполагается, что для сбора всех предписывающих функций управления, связанных с DACD-1 используется одна подстатья (нет обоснованной технической причины использовать больше одной). Для упрощения ссылок эта подстатья обозначается как SE_DACD1. Атрибут **prescriptiveACI** в SE_DACD1 имеет несколько значений; структура каждого значения рассматривается далее в этом подпункте.

Количество значений, которые может принимать атрибут **prescriptiveACI**, зависит частично от того, как фрагменты стратегии для удобства группируются в значения **itemFirst** и **userFirst** (в любой данной ситуации может использоваться любой из этих стилей); оно также зависит от того, как следует применять управление доступом для самих предписывающих функций управления.

Например, часть реализации PF-1 требует предоставления пользователям из населения (т. е. **allUsers**) все из следующих разрешений:

a) *Browse* для защищенного объекта **entry**;

b) *FilterMatch* и *Read* для защищенного объекта **attributeType {commonName}**;

c) *FilterMatch* и *Read* для защищенного объекта **allAttributeValues {commonName}**.

Эти разрешения необходимы (но недостаточны – см. Примечание 1) для реализации PF-1. В силу наличия трех защищенных объектов (**entry**, **attributeType** и **allAttributeValues**) и только одного класса пользователей (**allUsers**) более естественным представляется использовать единый **ACIItem** стиля **userFirst**, но вместо него может также использоваться стиль **itemFirst**.

ПРИМЕЧАНИЕ 1. – Разрешений, рассматриваемых выше, было бы также достаточно для разрешения поиска по **commonName**, если одновременно выполняются следующие два условия:

a) не существует других релевантных **ACIItems** с более высоким уровнем приоритета и специфичности, которые отказывают в предоставлении какого-либо из разрешений *Browse* или *FilterMatch*, перечисленных выше; и

b) не существует других значений для атрибута **prescriptiveACI** в SE_DACD1, которые отказывают в предоставлении какого-либо из разрешений *Browse*, *Read* или *FilterMatch*, перечисленных выше.

В ином случае могут использоваться три отдельных **ACIItem**: по одному для каждого из защищенных объектов. Этот вариант разрешает отдельное управление доступом для каждого **ACIItem**; каждый имеет являющуюся уникальной метку **identificationTag** (по отношению к другим **identificationTag** для других значений в том же атрибуте **prescriptiveACI**), на которую может быть сделана ссылка в другом **ACIItem**, где защищенным объектом является **attributeValue**, а утверждение о значении связанного атрибута определяет **identificationTag** значения, подлежащего защите. Отметим, что использование **attributeValue** таким образом применяет конкретное правило сопоставления равенства, определенное для атрибутов **prescriptiveACI** (см. п. 18.5.1). Примеры защиты ACI подробно рассматриваются далее в этом примере.

Для целей данного примера для реализации фрагментов стратегии PF-1–PF-4 используются шесть значений для атрибута **prescriptiveACI** в SE_DACD1. Сводка по составу каждого из этих трех значений представлена ниже.

ПРИМЕЧАНИЕ 2. – Все защищенные объекты в сводке по составу, ниже, имеют метку для упрощения ссылок. Метка заключена в круглые скобки и выделена курсивом (например, *A1*, *A2*, *B1*).

ПРИМЕЧАНИЕ 3. – В примере используются четыре уровня приоритетности: 10, 20, 30 и 40.

identificationTag: "Общий доступ – Разрешает доступ к статье для операций составления списка и поиска по общему имени common name"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*A1*) **entry }**
grantsAndDenials: **{ grantBrowse }**
----

identificationTag: "Общий доступ – Разрешает доступ к фильтру для операции поиска"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*B1*) **attributeType { commonName },**
(*B2*) **allAttributeValues { commonName },**
(*B3*) **attributeType { objectClass },**
(*B4*) **allAttributeValues { objectClass } }**
grantsAndDenials: **{ grantFilterMatch }**
----

identificationTag: "Общий доступ – Разрешает доступ к статье для операций чтения и сравнения"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*C1*) **entry }**
grantsAndDenials: **{ grantRead }**
----

identificationTag: "Общий доступ – Разрешает доступ к атрибуту для операции запроса"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**
ProtectedItems: **{** (*D1*) **attributeType {** **commonName,**
postalAttributeSet,
telephoneNumber,
facsimileTelephoneNumber } ,
(*D2*) **allAttributeValues { commonName,**
postalAttributeSet,
telephoneNumber,
facsimileTelephoneNumber } }
grantsAndDenials: **{ grantRead, grantCompare }**
----

identificationTag: "Доступ для персонала – Разрешает доступ к атрибуту для операций запроса"
Precedence: 10
UserClasses: **subtree** with **base {** C=US, O=ZCC **}** and **chop** to
exclude O=BRC subtree
authenticationLevel: **simple**
ProtectedItems: **{** (*E1*) **allUserAttributeTypesAndValues }**
grantsAndDenials: **{ grantRead, grantCompare }**
----

identificationTag: "Доступ для персонала – Разрешает доступ к фильтру для операции поиска"
Precedence: 10
UserClasses: **subtree** with **base {** C=US, O=ZCC **}** and **chop** to
exclude O=BRC subtree
authenticationLevel: **simple**
ProtectedItems: **{** (*F1*) **allUserAttributeTypesAndValues }**
grantsAndDenials: **{ grantFilterMatch }**

ПРИМЕЧАНИЕ 4. – Разрешения для персонала являются объединением разрешений для общего доступа и разрешений конкретно для персонала. Вышеуказанные значения ACIItem для доступа персонала жестко соединяются со значениями, связанными с общим доступом. Такого жесткого соединения можно при необходимости избежать, повторяя каждое из значений для общего доступа (каждое повторенное значение будет иметь новый UserClasses, который определяет только персонал).

Существуют еще два значения этого атрибута, которые связаны с реализацией стратегии административного управления статьями (PF-5 и PF-6). Для простоты в данном примере предполагается, что атрибутами управления доступом являются только операционные атрибуты, присутствующие в AAA. Состав обоих значений представлен ниже.

identificationTag: "Cauchy суперпользователь (Часть 1)"
Precedence: 40
UserClasses: **user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **}**
uniqueIdentifier = 12345
authenticationLevel: **strong**
ProtectedItems: **{** (*G1*) **entry }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantBrowse, grantModify,**
grantRename}
----

identificationTag: "Cauchy суперпользователь (Часть 2)"
Precedence: 40
UserClasses: **user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **}**
uniqueIdentifier = 12345
authenticationLevel: **strong**
ProtectedItems: **{** (*H1*) **allUserAttributeTypesAndValues,**

(*H2*) **attributeType { entryACI },**
(*H3*) **allAttributeValues { entryACI } }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantCompare, grantFilterMatch }**

Заметим, что для того чтобы сделать Cauchy суперпользователем, два вышеуказанные значения необходимы, но недостаточны. Они недостаточны, потому что не позволяют Cauchy управлять подстатьями административной точки для ACSA-1; этому есть две причины. Во-первых, предписывающая ACI не применяется к подстатье, в которую она входит. Во-вторых, предписывающая ACI, размещенная в подстатье, например в подстатье-1, не может использоваться для управления подстатьями, которые являются дочерними относительно подстатьи-1. Следовательно, необходимо поместить **subentryACI** в статью, соответствующую административной точке для ACSA-1, с тем чтобы Cauchy получил разрешение осуществлять свои полномочия в отношении подстатей этой административной точки. Необходимая **subentryACI** рассматривается в п. M.7.

Отметим также, что полномочия, предоставленные в двух вышеуказанных значениях предписывающей ACI, разрешают Cauchy осуществлять полный контроль над подстатьями, связанными с административными точками, которые являются подчиненными по отношению к административной точке для ACSA-1.

M.6.2 Атрибут prescriptiveACI для DACD-2

DACD-2 определяется в подстатье административной статьи для ACSA-1. DACD-2 имеет отношение к управлению статьями с классом объектов **organizationalPerson**. Релевантным является следующий фрагмент стратегии.

PF-7: Только члены административной группы области имен {C=US, O=ZCC, OU=Admin, CN=Ops} могут добавлять, удалять или переименовывать пользовательские статьи. Вместе с тем, им только разрешено добавлять обязательные атрибуты к новой статье (статья, содержащая только обязательные атрибуты, называется *минимальная статья*).

Следующие два значения в атрибуте **prescriptiveACI** SE_DACD2 реализуют PF-7.

ПРИМЕЧАНИЕ. – Переименование статей в контексте PF-7 понимается как переименование без изменения непосредственно старшего. Для простоты в этом примере не рассматривается более сложный случай, когда переименование включает изменение непосредственно старшего переименовываемой статьи (и ее подчиненных), в каковом случае должны учитываться разрешения *Import* и *Export*.

identificationTag: "Административное управление минимальной листовой статьей (Часть 1)"
Precedence: 20
UserClasses: **userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*J1*) **entry,**
(*J2*) **attributeType {commonName, surname },**
(*J3*) **allAttributeValues {commonName, surname } }**
grantsAndDenials: **{ grantAdd, grantRemove }**
----

identificationTag: "Административное управление минимальной листовой статьей (Часть 2)"
Precedence: 20
UserClasses: **userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*K1*) **entry }**
grantsAndDenials: **{ grantRename }**

M.6.3 Атрибут prescriptiveACI для DACD-3

DACD-3 определяется в подстатье административной статьи для ACIA-1. Он реализует фрагменты стратегии, касающиеся стратегии, которая частично делегирована ACIA-1. В примере стратегия для ACIA-1 в отношении **telephoneNumber** отличается от стратегии, представленной в стратегии по умолчанию в пределах DACD-1. В пределах DACD-3 **telephoneNumber** не рассматривается как объект общего доступа. Это отражено в следующем фрагменте стратегии.

PF-8: Единственными атрибутами общего пользования в пределах ACIA-1 являются **commonName**, компоненты из **postalAttributeSet** и **facsimileTelephoneNumber**.

Следующее значение в атрибуте **prescriptiveACI** подстатьи {C=US, O=ZCC, OU=R&D, OU=West, CN=SE_DACD3} реализует PF-8.

identificationTag: "Делегированное управление общим доступом"
Precedence: 10
UserClasses: **{ allUsers }**
authenticationLevel: **none**

ProtectedItems: **{** (*L1*) **attributeType { telephoneNumber } }**
grantsAndDenials: **{ denyRead, denyCompare, denyFilterMatch }**

Организации R&D West также делегируются полномочия для реализации самоуправления статьями класса объектов **organizationalPerson**. Стратегия отражается в следующем фрагменте.

PF-9: Персонал R&D West может осуществлять административное управление значениями только в пределах собственной статьи Справочника для следующих типов атрибутов: **telephoneNumber**, **commonName** и **facsimileNumber**; вместе с тем, они не могут изменять или удалять значение номера телефона, представленное администрацией.

Первая часть PF-9 отражена в двух **ACIItem** ниже. Ограничение на удаление конкретного значения атрибута **telephoneNumber** реализуется с использованием **entryACI**, как описано в п. M.8.

identificationTag: "Самоуправление статьями персонала R&D West (Часть 1)"
Precedence: 20
UserClasses: **thisEntry**
authenticationLevel: **strong**
ProtectedItems: **{** (*M1*) **entry }**
grantsAndDenials: **{ grantModify }**
----

identificationTag: "Самоуправление статьями персонала R&D West (Часть 2)"
Precedence: 20
UserClasses: **thisEntry**
authenticationLevel: **strong**
ProtectedItems: **{** (*N1*) **attributeType { commonName, postalAttributeSet, telephoneNumber, facsimileTelephoneNumber },**
(*N2*) **allAttributeValues { commonName, postalAttributeSet, telephoneNumber, facsimileTelephoneNumber } }**
grantsAndDenials: **{ grantAdd, grantRemove }**

PF-10: В обязанности группы, члены которой указаны в {C=US, O=ZCC, OU=R&D, CN=Ops}, входит общее сопровождение атрибутов пользователей для статей в ACIA-1; вместе с тем, они не могут изменять подстатьи, находящиеся внутри ACIA-1.

Первая часть этой стратегии отражается в следующем **ACIItem**:

identificationTag: "Общее административное управление R&D (Часть 1)"
Precedence: 20
UserClasses: **userGroup {** C=US, O=ZCC, OU=R&D, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*P1*) **entry }**
grantsAndDenials: **{ grantModify, grantAdd, grantRemove, grantBrowse, grantRead, grantRename }**
----

identificationTag: "Общее административное управление R&D (Часть 2)"
Precedence: 20
UserClasses: **userGroup {** C=US, O=ZCC, OU=R&D, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*Q1*) **allUserAttributeTypesAndValues }**
grantsAndDenials: **{ grantAdd, grantRemove, grantRead, grantFilterMatch, grantCompare}**

Ограничение в отношении подстатей обрабатывается путем невключения каких бы то ни было значений **subentryACI**, разрешающих доступ, в административную статью для ACIA-1.

M.6.4 Атрибут prescriptiveACI для DACD-4

DACD-4 определяется в подстатье административной статьи ACIA-2. в этом качестве он реализует фрагменты, касающиеся стратегии, которая была частично делегирована ACIA-2.

Для простоты DACD-4 далее не рассматривается.

M.6.5 Атрибут prescriptiveACI для DACD-5

DACD-5 определяется в подстатье административной точки для ACSA-1. Этот DACD используется для управления доступом ко всем непосредственно подчиненным, не являющимся подстатьями, относительно корня организации. В частности применяется следующая стратегия.

PF-11: В обязанности оперативной группы {C=US, O=ZCC, CN=Ops} входит административное управление всеми статьями, которые являются непосредственно подчиненными относительно {C=US, O=ZCC}.

PF-11 выражается в следующих значениях **ACIItem**.

identificationTag: "Управление административными листовыми статьями (Часть 1)"
Precedence: 40
UserClasses: **userGroup {** C=US, O=ZCC, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*R1*) **entry }**
grantsAndDenials: **{ grantRead, grantBrowse, grantRemove, grantAdd, grantRename, grantModify }**
---

identificationTag: "Управление административными листовыми статьями (Часть 2)"
Precedence: 40
UserClasses: **userGroup {** C=US, O=ZCC, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{ (*S1*) allUserAttributeTypesAndValues, (*S2*) attributeType { entryACI }, (*S3*) allAttributeValues { entryACI } }**
grantsAndDenials: **{ grantRead, grantRemove, grantAdd, grantCompare, grantFilterMatch }**

M.7 Стратегия, выраженная в атрибутах subentryACI

M.7.1 Атрибут subentryACI в административной статье для ACSA-1

PF-5 проявляется в комбинации **prescriptiveACI** и **subentryACI**; связанная **prescriptiveACI** уже была описана в п. M.6.1. Для того чтобы разрешить Cauchy осуществлять административное управление подстатьями административной точки для ACSA-1 (и всеми статьями для административных точек, подчиненных по отношению к административной точке для ACSA-1), необходимо поместить следующие значения **subentryACI** в статью, соответствующую административной точке для ACSA-1.

identificationTag: "Cauchy суперпользователь (Часть 3)"
Precedence: 40
UserClasses: **user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **} uniqueIdentifier = 12345**
authenticationLevel: **strong**
ProtectedItems: **{** (*G1*) **entry }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantBrowse, grantModify, grantRename}**
---

identificationTag: "Cauchy суперпользователь (Часть 4)"
Precedence: 40
UserClasses: **user {** C=US, O=ZCC, OU=Admin, CN=Cauchy **} uniqueIdentifier = 12345**
authenticationLevel: **strong**
ProtectedItems: **{** (*H1*) **allUserAttributeTypesAndValues,** (*H2*) **attributeType { entryACI },** (*H3*) **allAttributeValues { entryACI } }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantCompare, grantFilterMatch }**

M.7.2 Атрибут subentryACI в административной статье для ACIA-1

Атрибут **subentryACI** помещается в корневую вершину ACIA-1 для реализации следующих фрагментов стратегии.

PF-12: В обязанности пользователя с общим именем Cayley входит управление всеми атрибутами **prescriptiveACI**, определенными в ACIA-1.

Следующие два значения атрибута **subentryACI** реализуют PF-12.

identificationTag: "Cayley управляет подстатьями в ACIA-1 (Часть 1)"
Precedence: 20
UserClasses: **user {** C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*T1*) **entry }**
grantsAndDenials: **{ grantRead, grantBrowse, grantRemove, grantAdd, grantRename, grantModify }**
---

identificationTag: "Cayley управляет подстатьями в ACIA-1 (Часть 2)"
Precedence: 20
UserClasses: **user {** C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*U1*) **attributeType { prescriptiveACI },**
(*U2*) **allAttributeValues { prescriptiveACI } }**
grantsAndDenials: **{ grantAdd, grantRead, grantRemove, grantCompare, grantFilterMatch }**

M.8 Стратегия, выраженная в атрибутах entryACI

PF-9 требует, чтобы всем сотрудникам R&D West было разрешено управлять всеми значениями **telephoneNumber** в своей статье Справочника с ограничением в том, что они не могут изменять или удалять конкретное значение, представленное администрацией. Для обеспечения выполнения этого ограничения администрация добавляет значение **entryACI** к каждой статье в момент введения являющегося объектом ограничения телефонного номера в эту статью. Сводка значений **entryACI** представлена ниже:

identificationTag: "Ограничить самоуправление телефонными номерами"
Precedence: 30
UserClasses: **thisEntry**
authenticationLevel: **none**
ProtectedItems: **{** (*V1*) **attributeValue { telephoneNumber** = *значение, обеспечиваемое администрацией* **} }**
grantsAndDenials: **{ denyRemove }**

Отметим, что поскольку пользователи не могут изменять атрибут **entryACI** (это не является частью самоуправления, определенного в PF-9), вышеприведенная функция управления не может быть переопределена пользователем.

Следующий фрагмент стратегии является примером использования **entryACI** для реализации самоуправления в отношении статьи группы.

PF-13: Статья {C=US, O=ZCC, OU=Admin, CN=Ops} является "самоуправляемой" статьей группы; это значит, что каждый член этой группы может удалять свое имя или менять свое имя в данной группе. Они не могут удалить или переименовать саму группу.

PF-13 реализуется атрибутом **entryACI** в статье {C=US, O=ZCC, OU=Admin, CN=Ops}, два значения которого представлены ниже.

identificationTag: "самоуправление административной группы Ops (Часть 1)"
Precedence: 30
UserClasses: **userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel: **strong**
ProtectedItems: **{** (*W1*) **entry }**
grantsAndDenials: **{ grantModify }**
---

identificationTag: "самоуправление административной группы Ops (Часть 2)"
Precedence: *medium*
UserClasses: **userGroup {** C=US, O=ZCC, OU=Admin, CN=Ops **}**
authenticationLevel: **strong**

ProtectedItems: **{ (*X1*) selfValue { uniqueMember } }**
grantsAndDenials: **{ grantRemove, grantAdd }**

M.9 Примеры ACDF

M.9.1 Общий доступ для чтения

Член группы общего доступа, имеющий выделенное имя {C=GB, O=XC, CN=Smith}, предпринимает попытку операции чтения, запрашивая значения телефонного номера для пользователя Cayley. Решения по управлению доступом для этой операции описываются в Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3. Предполагая, что разрешение имени не предусматривает разыменования псевдонима, первой точкой принятия решения является определение, предоставлено ли разрешение *Read* для целевой статьи; это решение основывается на следующих входах ACDF:

- запрошенное разрешение: *Read*;
- инициатор: {C=GB, O=XC, CN=Smith}, не имеющий уникального идентификатора;
- уровень аутентификации: отсутствует;
- защищенный объект: статья {C=US, O=ZCC, OU=R&D, OU=West, CN=Cayley};
- кортежи показаны в таблице M.1.

Таблица M.1

Пользователь	Объект	Разрешение	Предоставлено (G) или отказано (D)	Приоритет	Уровень аутентификации
allUsers	(*A1*)**entry**, статья	*Browse*	G	10	Отсутствует
allUsers	(*B1*)**commonName**, тип	*FilterMatch*	G	10	Отсутствует
allUsers	(*B2*)**commonName**, значения	*FilterMatch*	G	10	Отсутствует
allUsers	(*B3*)**objectClass**, тип	*FilterMatch*	G	10	Отсутствует
allUsers	(*B4*)**objectClass**, значения	*FilterMatch*	G	10	Отсутствует
allUsers	(*C1*)**entry**, статья	*Read*	G	10	Отсутствует
allUsers	(*D1*)**commonName**, тип	*Read*	G	10	Отсутствует
allUsers	(*D1*)**postalAttributeSet**, тип	*Read*	G	10	Отсутствует
allUsers	(*D1*)**telephoneNumber**, тип	*Read*	G	10	Отсутствует
allUsers	(*D1*)**facsimileTelephoneNo**, тип	*Read*	G	10	Отсутствует
allUsers	(*D2*)**commonName**, значения	*Read*	G	10	Отсутствует
allUsers	(*D2*)**postalAttributeSet**, значения	*Read*	G	10	Отсутствует
allUsers	(*D2*)**telephoneNumber**, значения	*Read*	G	10	Отсутствует
allUsers	(*D2*)**facsimileTelephoneNo**, значения	*Read*	G	10	Отсутствует
allUsers	(*L1*)**telephoneNumber**, тип	*Read*	D	10	Отсутствует
allUsers	(*L1*)**telephoneNumber**, тип	*Compare*	D	10	Отсутствует
allUsers	(*L1*)**telephoneNumber**, тип	*FilterMatch*	D	10	Отсутствует

Защищенная целевая статья находится в области действия DACD-1, DACD-2 и DACD-3 (см. рисунок M.4). Не имеет **entryACI**. Эти три DACD участвуют в кортежах (применимых к указанному инициатору), показанных в таблице M.1 к процедуре ACDF, описание см. в п. 18.8.

После отбрасывания нерелевантных строк ACDF имеет всего две строки для рассмотрения: строка 4, предоставляющая разрешение *Read* для статьи, и строка 13, отказывающая в предоставлении разрешения *Read* для статьи. Следовательно, ACDF отказывает в доступе.

ПРИМЕЧАНИЕ. – Для простоты в данном примере не рассматриваются разрешения и процедуры, связанные с условиями сообщений об ошибках. Вместе с тем, в вышеприведенном случае отказа в доступе поведение отвечающего DSA управлялось бы согласно п. 18.2.3 или п. 18.4.1 и включало бы повторное применение ACDF для того чтобы определить, предоставлено ли разрешение *DiscloseOnError* для целевой статьи.

M.9.2 Общий доступ для поиска

Член группы общего доступа, имеющий выделенное имя {C=GB, O=XC, CN=Smith}, предпринимает попытку операции поиска, запрашивая все значения всех атрибутов для всех пользователей (**wholeSubtree**), подчиненных по отношению к базовому объекту {C=US, O=ZCC, OU=R&D, OU=West}; **filter** содержит спецификацию **FilterItem equality**: **objectClass = organizationalPerson**. Точки принятия решения по управлению доступом для этой операции определяются в п. 10.2.6 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3.

M.9.2.1 Проверка каждой статьи в области действия операции поиска на соответствующее разрешение в отношении этой статьи

Для каждой статьи в области поиска, предполагая, что разрешение имени не предусматривает разыменования псевдонима, первой точкой принятия решения является определение, выдается ли разрешение *Browse* для этой статьи. Для первой такой статьи входами ACDF являются:

- запрошенное разрешение: *Browse*;
- инициатор: {C=GB, O=XC, CN=Smith};
- уникальный идентификатор: отсутствует;
- уровень аутентификации: никакой;
- защищенный объект: статья{C=US, O=ZCC, OU=R&D, OU=West};
- кортежи показаны в таблице M.2.

Поскольку проверяемая статья включена только в DACD-1, начальное множество кортежей, собранных функцией ACDF, показано в таблице M.2. Заметим, что для рассмотрения отсутствует **entryACI**.

В результате выполняемой функцией ACDF процедуры отбрасывания строк из таблицы M.2 остается только одна строка; следовательно ACDF предоставляет запрошенный доступ.

Аналогичным образом, ACDF предоставит разрешение *Browse* для всех статей в области действия операции поиска.

Таблица M.2

Пользователь	Объект	Разрешение	Предоставлено (G) или отказано (D)	Приоритет	Уровень аутентификации
allUsers	(*A1*)**entry**, статья	*Browse*	G	10	Отсутствует
allUsers	(*B1*)**commonName**, тип	*FilterMatch*	G	10	Отсутствует
allUsers	(*B2*)**commonName**, значения	*FilterMatch*	G	10	Отсутствует
allUsers	(*B3*)**objectClass**, тип	*FilterMatch*	G	10	Отсутствует
allUsers	(*B4*)**objectClass**, значения	*FilterMatch*	G	10	Отсутствует
allUsers	(*C1*)**entry**, статья	*Read*	G	10	Отсутствует
allUsers	(*D1*)**commonName**, тип	*Read*	G	10	Отсутствует
allUsers	(*D1*)**postalAttributeSet**, тип	*Read*	G	10	Отсутствует
allUsers	(*D1*)**telephoneNumber**, тип	*Read*	G	10	Отсутствует
allUsers	(*D1*)**facsimileTelephoneNo**, тип	*Read*	G	10	Отсутствует
allUsers	(*D2*)**commonName**, значения	*Read*	G	10	Отсутствует
allUsers	(*D2*)**postalAttributeSet**, значения	*Read*	G	10	Отсутствует
allUsers	(*D2*)**telephoneNumber**, значения	*Read*	G	10	Отсутствует
allUsers	(*D2*)**facsimileTelephoneNo**, значения	*Read*	G	10	Отсутствует

M.9.2.2 Проверка на соответствие фильтру

Для каждой статьи в области действия поиска, в отношении которой получено разрешение *Browse*, следующей точной принятия решения является определение, выдано ли разрешение *FilterMatch* в отношении атрибута **objectClass**. Для первой такой статьи входами ACDF являются:

- запрошенное разрешение: *Browse*;
- инициатор: {C=GB, O=XC, CN=Smith};
- уникальный идентификатор: отсутствует;
- уровень аутентификации: отсутствует;
- защищенный объект: статья {C=US, O=ZCC, OU=R&D, OU=West};
- кортежи показаны в таблице M.2.

Функция ACDF должна отбросить все строки таблицы M.2 кроме строки 4; следовательно, доступ должен быть предоставлен. Далее операция поиска должна проверить, являются ли какие-нибудь значения атрибута **objectClass** равными **organizationalPerson**. Поскольку проверяемая статья является статьей подразделения организации, оценкой фильтра **Filter** должно быть значение **FALSE**.

Аналогично, **Filter** должен выдать оценку **FALSE** для статьи с CN=SE_DACD3.

Для двух других статей в области действия операции поиска (CN=Cayley, CN=Noether) значением оценки **Filter** должно быть **TRUE**. Для каждой из этих статей следующим решением управления доступом является определение того, предоставляется ли разрешение *FilterItem* для значения атрибута, в отношении которого **Filter** выдает оценку **TRUE**. Поскольку обе эти статьи включены в DACD-1, DACD-2 и DACD-3, начальное множество кортежей, являющихся входом для ACDF, содержится в таблице M.1. Строка 5 таблицы M.1 предоставляет запрошенный доступ для обеих статей.

Таким образом. результат операции поиска содержит информацию из статей Cayley и Noether. Дополнительные решения по управлению доступом для этих двух статей по существу те же, что и показанные в примере общего доступа для чтения в п. M.9.1.

M.10 Управление доступом на основе правил

Для иллюстрации управления доступом на основе правил определен следующий пример правил безопасности (следует заметить, что пример носит исключительно иллюстративный характер и необязательно представляет какую-либо полную реальную стратегию).

Возможными значениями метки безопасности является иерархическое множество: немаркированный, неклассифицированный, ограниченный, конфиденциальный, секретный, сверхсекретный.

Значениями допуска является иерархические максимальные значения класса: немаркированный, неклассифицированный, ограниченный, конфиденциальный, секретный, сверхсекретный.

ПРИМЕЧАНИЕ. – Эти правила могут быть расширены специалистами для охвата последующей информации о привилегиях, переносимой в грифе конфиденциальности или категориях защиты.

Правилами доступа являются следующие:

a) доступ предоставляется, если уровень допуска выше или равен уровню метки;

b) допуск не предоставляется, если уровень допуска ниже уровня метки.

Приложение N

Комбинации типов DSE

(|)

В таблице N.1 описан ряд комбинаций типов DSE (т. е. комбинаций именованных битов атрибута **dseType**), которые весьма вероятно существуют при применении информационной модели DSA к DSA в отсутствие теневого копирования. Таблица представлена для пояснения информационной модели DSA. Поддержка перечисленных (или иных комбинаций типов DSE) комбинаций не предписывается настоящей спецификацией Справочника.

Таблица N.1 – Определенные комбинации типов DSE в отсутствие теневого копирования

Тип DSE	admPoint	cp	supr	nssr	sa	член семейства	Комментарии
Root			✓	✓			Корневая DSE для DSA первого уровня. DSA первого уровня с nssr, если установлен бит **nssr**. Корневая DSE для DSA не первого уровня, если установлен бит **supr**.
Glue							Стыковочная DSE.
Entry	✓	✓		✓		✓	DSE статьи объекта; также административная точка, если установлен бит **admPoint**; префикс контекста, если установлен бит **cp**; nssr, если установлен бит **nssr**.
Alias							DSE статьи псевдонима.
Subentry		✓					DSE подстатьи.
subr					✓		DSE подчиненной ссылки; подчиненная ссылка указывает на псевдоним, если установлен бит **sa**.
immSupr						✓	Непосредственно старшая ссылка.
xr							DSE перекрестной ссылки.
ПРИМЕЧАНИЕ. – Тип DSE **subr** и **immSupr** также может существовать (возможно с дополнительным битом **admPoint**), хотя представлять их в таблице неудобно. Информация подстатьи и административной точки, сопровождаемых связываниями RHOB, указывается наличием бита **rhob**.							

В первом столбце таблицы обозначены типы DSE, которые нет необходимости комбинировать с каким-либо иным типом DSE для выражения функции DSE. Например, может существовать DSE только с установленным битом **entry**. В столбцах со второго по шестой значком (✓) указаны дополнительные биты типов DSE, которые также могут быть установлены в дополнение к биту, обозначенному в первом столбце. Эти биты могут устанавливаться независимо. Например, DSE **entry** может также иметь бит **nssr**, биты **admPoint** и **cp** или несколько иных комбинаций установленных битов **admPoint**, **cp** и **nssr**. В последнем столбце описываются различные комбинации типов DSE, указанные в этой строке таблицы.

В таблице N.2 описывается ряд дополнительных комбинаций типов DSE, которые весьма возможны при выполнении теневого копирования. Как и в предыдущей таблице, в первом столбце этой таблицы указаны типы DSE, которые не требуется комбинировать, в теневом DSA для этой DSE, с каким-либо иным типом DSE для выражения функции DSE. В столбцах со второго по шестой значком (✓) указаны дополнительные биты типов DSE, которые также могут быть установлены в дополнение к биту, обозначенному в первом столбце. Эти биты могут устанавливаться независимо.

Таблица N.2 – Определенные комбинации типов DSE при выполнении теневого копирования

Тип DSE	admPoint	cp	supr	nssr	sa	член семейства	Комментарии
Root				✓			Корневая DSE для теневого DSA первого уровня с nssr.
Entry	✓	✓		✓		✓	DSE статьи объекта; также административная точка, если установлен бит **admPoint**; префикс контекста, если установлен бит **cp**; nssr если установлен бит **nssr**.
Alias		✓					DSE статьи псевдонима.
Subentry							DSE подстатьи.
subr					✓		DSE подчиненной ссылки; подчиненная ссылка указывает на псевдоним, если установлен бит **sa**.
immSupr	✓					✓	Непосредственно старшая ссылка.
admPoint		✓		✓		✓	DSE административной точки без атрибута пользователя (статья не имеет теневой копии); также префикс контекста, если установлен бит **cp**; также nssr, если установлен **nssr**.
Cp			✓	✓		✓	DSE префикса контекста (статья не имеет теневой копии); также nssr, если установлен **nssr**.
nssr						✓	DSE nssr (статья не имеет теневой копии).

ПРИМЕЧАНИЕ. – Бит **shadow** устанавливается во всех случаях в этой таблице (и, поэтому не представлен явным образом). Как и в случае таблицы N.1, тип DSE **subr**, **immSupr** и **shadow** также может существовать (возможно с дополнительным битом **admPoint**). Наконец, для статей DSE с установленными битами **subr** и/или **immSupr** также могут существовать биты **entry** и **shadow**, поскольку информация имеющей теневую копию статьи, накладывается на информацию знаний, сохраняемую либо связываниями RHOB, либо теневым копированием.

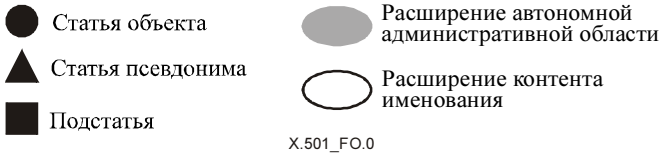
Приложение O

Моделирование знаний

(|)

Следующий пример иллюстрирует гипотетическое DIT, его потенциальное отображение на три DSA и информацию, которую должен был бы сопровождать DSA (включая информацию знаний) для поддержки отображения.

На рисунках O.1 и O.2, ниже, используются следующие символы.



На рисунке O.1 показано гипотетическое DIT. Оно разбито на четыре автономные административные области: вырожденные случаи единичных статей {C=WW} и {C=VV} и два поддерева с вершинами в {C=WW, O=ABC} и {C=VV, O=DEF}. Одна статья, {C=VV, O=DEF, OU=K}, является псевдонимом статьи объекта {C=WW, O=ABC, OU=I}.

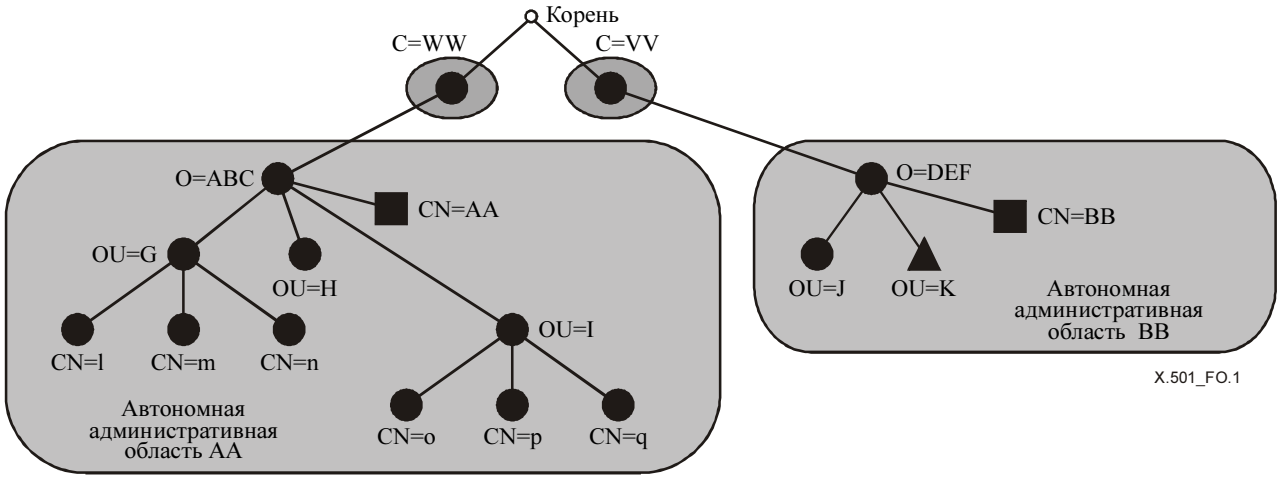


Рисунок O.1 – Гипотетическое DIT

На рисунке O.2 показано разбиение гипотетического DIT на пять контекстов именования (A, B, C, D и E) и их отображение в три DSA (DSA1, DSA2 и DSA3). На рисунке DSA1 содержит контекст C, DSA2 содержит контексты A, B и E и DSA3 содержит контекст D.

Знания, которые содержат три DSA, представляют следующее: DSA1 использует DSA2 в качестве своей старшей ссылки и имеет неспецифическую подчиненную ссылку на DSA2 для информации, подчиненной по отношению к {C=WW, O=ABC}. DSA2 является DSA первого уровня и сохраняет подчиненную ссылку на DSA1 для контекста C и непосредственно старшую ссылку на него для контекста, непосредственно старшего по отношению к контексту E. DSA2 сохраняет подчиненную ссылку на DSA3 для контекста D. DSA3 также использует DSA2 в качестве своей старшей ссылки и имеет перекрестную ссылку на DSA2 для контекста E.

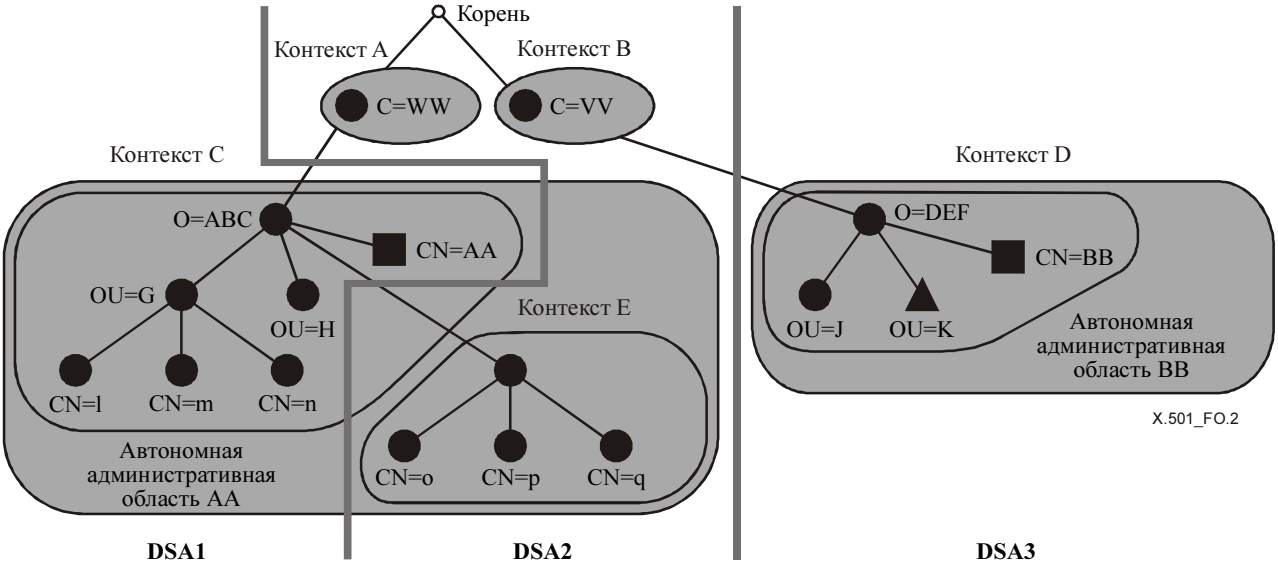
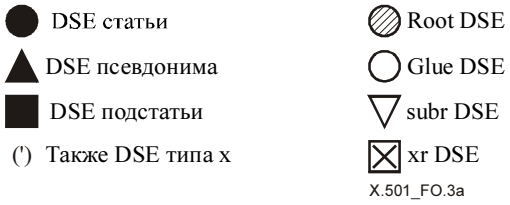


Рисунок O.2 – Гипотетическое дерево, отображенное в три DSA

На рисунках O.3–O.6 показана информация, которую содержит каждый DSA (т. е. информационное дерево DSA каждого DSA) для поддержки этой конфигурации. На этих рисунках используются следующие символы.



На рисунке O.3 показано информационное дерево DSA для DSA1.

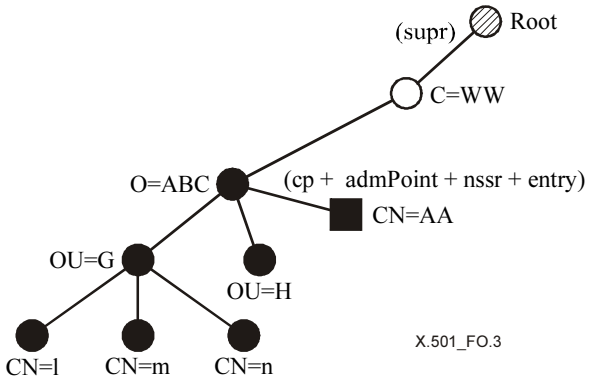


Рисунок O.3 – Информационное дерево DSA для DSA1

Поскольку DSA1 не является DSA первого уровня, его корневая DSE содержит старшую ссылку, которая в примере является точкой доступа для DSA2. DSE имеет тип **root + supr**.

DSA1 содержит одну стыковочную DSE для представления своих знаний с именем {C=WW}.

Автономная административная область AA подразделяется на два контекста именования C и E, при этом контекст C содержится в DSA1. Для простоты представления в примере предполагается, что специальные административные области, относящиеся к управлению доступом и информации подсхемы, совпадают, и что имеется один домен управления доступом и одна подсхема для всей автономной административной области. Следствием этого является то, что для каждой автономной административной области в примере требуется только одна (многоцелевая) подстатья.

Для DSA1 DSE в {C=WW, O=ABC}, представляющая административную точку для AA, префикс контекста для контекста C и неспецифическую подчиненную ссылку на DSA2, имеет тип **entry + cp + admPoint + nssr**. Операционная информация области содержится в подстатье {C=WW, O=ABC, CN=AA}.

DSA1 содержит следующие статьи, входящие в контекст C: {C=WW, O=ABC, OU=G}, {C=WW, O=ABC, OU=H}, {C=WW, O=ABC, OU=G, CN=l}, {C=WW, O=ABC, OU=G, CN=m} и {C=WW, O=ABC, OU=G, CN=n}.

На рисунке O.4 показано одно потенциальное информационное дерево DSA для DSA2.

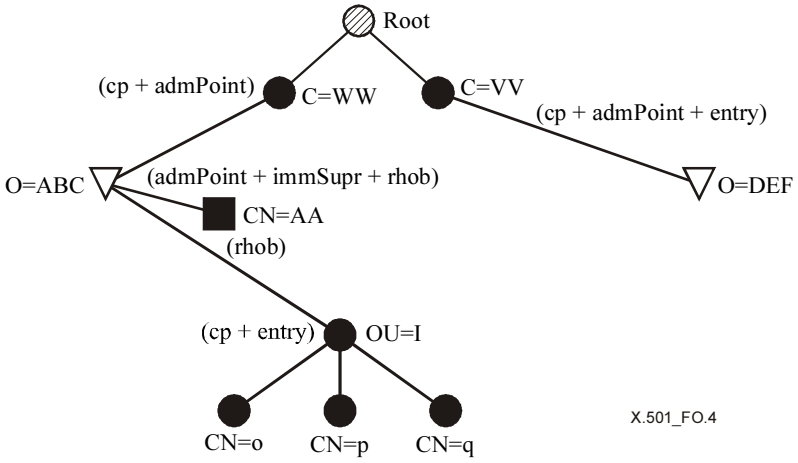


Рисунок O.4 – Информационное дерево DSA для DSA2

В этой гипотетической ситуации DSA2 является DSA первого уровня, поэтому его корневая DSE не содержит старшей ссылки.

Две вырожденные автономные административные области, {C=WW} и {C=VV}, представлены DSE типа **cp + entry + admPoint**.

Подчиненные знания DIT представлены двумя DSE подчиненных ссылок, {C=WW, O=ABC} и {C=VV, O=DEF}. В первом случае это DSE типа **subr + admPoint + immSupr + rhob**, что обусловлено причинами, приведенными ниже.

На рисунке O.4 конфигурация DSA2 выполнена при допущении, что операционную информацию области, касающуюся AA, содержит одиночная подстатья. Это требует, чтобы копия этой подстатьи присутствовала в DSA2 (для оптимальных рабочих характеристик). Один из способов выполнения этого требования заключается в образовании NHOB между DSA1 и DSA2 для сопровождения копии этой подстатьи. В этом случае операционная информация области содержится в DSE с именем {C=WW, O=ABC, CN=AA}, типом которой является **subentry + rhob**. Атрибут **administrative-role**, содержащийся в DSE в {C=WW, O=ABC}, предоставляется в DSA2 из DSA1 как часть NHOB. По этой причине типом DSE является тип **admPoint + rhob**.

Наконец, контекст именования E содержится как DSE префикса контекста {C=WW, O=ABC, OU=I), типом которой является **cp + entry** и три DSE статей {C=WW, O=ABC, OU=I, CN=o}, {C=WW, O=ABC, OU=I, CN=p} и {C=WW, O=ABC, OU=I, CN=q}.

Альтернативные способы конфигурирования DSA2 показаны на рисунке O.5.

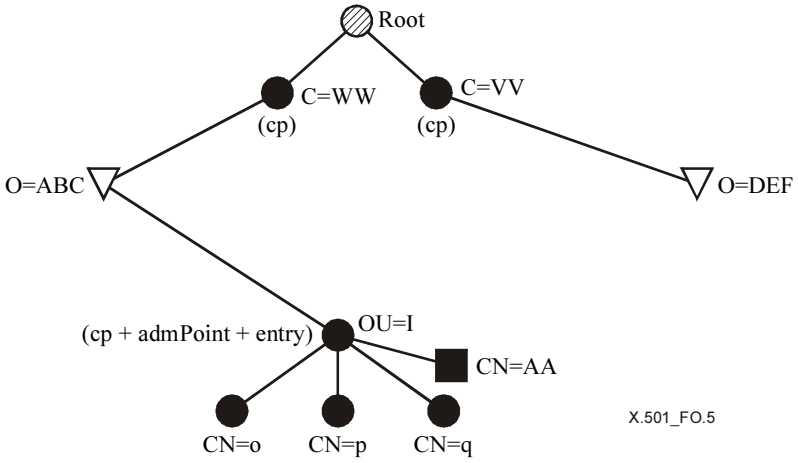


Рисунок O.5 – Альтернативное информационное дерево DSA для DSA2

Отличие от конфигурации, показанной на рисунке O.4, заключается лишь в обработке операционной информации области и обусловлено, возможно, желанием избежать необходимости сопровождать NHOB с DSA1.

Стратегией в данном случае является разбиение AA (т. е. разбиение информации управления доступом домена и аналогично информации подсхемы) на две автономные административные области, при этом одна совпадает с контекстом C, другая – с контекстом E.

В этом случае DSE префикса контекста {C=WW, O=ABC, OU=I} становится также административной точкой, тип DSE – **cp + admPoint + entry**. Вместо имеющей теневую копию подстатьи, обеспечиваемой DSA1 в качестве части NHOB, в подстатье {C=WW, O=ABC, OU=I, CN=AA} содержится операционная информация сокращенной области.

На рисунке O.6 показано информационное дерево DSA для DSA3.

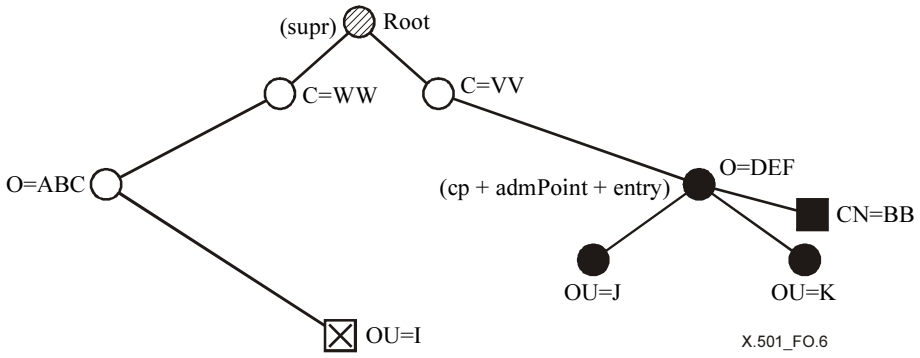


Рисунок O.6 – Информационное дерево DSA для DSA3

Как и DSA1, DSA3 не является DSA первого уровня. Его корневая DSE содержит старшую ссылку, которая в данном примере является точкой доступа для DSA2. Эта DSE имеет тип **root + supr**.

DSA2 содержит одну стыковочную DSE для представления своих знаний об имени {C=VV}.

Автономная административная область BB совпадает с контекстом именования D. Для простоты представления в примере предполагается, что как и в примере автономной административной области AA, специальные административные области, относящиеся к управлению доступом и информации подсхемы, совпадают, и что имеется один домен управления доступом и одна подсхема для всей автономной административной области. Таким образом, для каждой автономной административной области в примере требуется только одна (многоцелевая) подстатья.

Для DSA3 DSE в {C=VV, O=DEF}, представляющая административную точку для BB и префикс контекста для контекста D, имеет тип **entry + cp + admPoint**. Операционная информация области содержится в подстатье {C=VV, O=DEF, CN=BB}.

DSA3 содержит одну статью объекта и одну статью псевдонима, входящие в контекст D: {C=VV, O=DEF, OU=J}, (типа **entry**) и {C=VV, O=DEF, OU=K} (типа **alias** и содержащая атрибут **aliasedEntryName**, имеющий значение {C=WW, O=ABC, OU=I}).

Наконец, DSA3 содержит перекрестную ссылку на контекст E, DSE типа **xr** с именем {C=WW, O=ABC, OU=I}.

Приложение P

Имена, содержащиеся как значения атрибутов или используемые как параметры

(|)

Если имя содержится как значение атрибута в пределах какого-либо другого атрибута, или проходит в каком-либо обмене как значение атрибута (например, указатель псевдонимов), всегда возникает вопрос, может ли содержащееся имя быть альтернативным выделенным именем или первичным выделенным именем, может ли оно содержать альтернативные значения и может ли оно включать контекстную информацию. Там где необходимо, в настоящих спецификациях Справочника упоминаются конкретные ограничения. В общем случае не существует ограничений в отношении имен, хранящихся как значения атрибутов, но вместе с тем, в целях упрощения межсетевого взаимодействия с предыдущими системами и обеспечения прогнозируемых результатов, предлагается следующее:

Если значением операционного атрибута является имя объекта (например, **creatorsName**), это имя должно быть первичным выделенным именем объекта. Альтернативные значения и контекстная информация необязательны, но могут быть включены.

Если пара тип и значение атрибута в RDN в пределах имени включает множественные выделенные значения с использованием **valuesWithContext**, в качестве **value** в **AttributeTypeAndDistinguishedValue** должно использоваться первичное выделенное значение, с тем чтобы обеспечить прогнозируемость межсетевого взаимодействия с предыдущими системами.

Если значением атрибута пользователя является имя (например, член группы имен, см. "кроме того"), оно может быть любым альтернативным выделенным именем, множественными альтернативными именами или всеми альтернативными именами, однако рекомендуется использование первичного выделенного имени, с тем чтобы обеспечить прогнозируемость межсетевого взаимодействия с предыдущими системами. Кроме того, контексты и альтернативные значения не являются в целом полезными, если включаются в такие атрибуты, на которые делается ссылка.

Если атрибут является частью информационного дерева DSA и используется в разрешении имен (т. е. ссылки на знания), это должно быть первичное выделенное имя, и каждое RDN должно нести контексты и все альтернативные выделенные значения в **AttributeTypeAndDistinguishedValue** для каждого атрибута, с тем чтобы оптимизировать разрешение имен, а также межсетевое взаимодействие с предыдущими системами.

Приложение Q

Подфильтры

(|)

Фильтр может быть преобразован во множество подфильтров путем последовательного расширения по правилам де Моргана. (Эти правила действуют для используемой для фильтра трехзначной логики.) Рассмотрим фильтр как дерево, где нелистовые узлы соответствуют каждому **and**{}, **or**{}, **not**{}, а все листовые узлы являются пунктами фильтра. Каждая дуга представляет элемент в **and**{}, **or**{}, **not**{}; в случае **not**{} может быть только одна такая дуга.

Сначала расширяем каждый **not**{} до листьев, используя правила:

not{**and**{x,y,z}} то же, что **or**{**not**{x}, **not**{y}, **not**{z}}
not{**or**{x,y,z}} то же, что **and**{**not**{x}, **not**{y}, **not**{z}}
not{**not**{x}} то же, что x,

оставляя эти **not** для применения непосредственно к пунктам фильтра.

Затем сокращаем дерево, комбинируя **and** и **or**, и перемещаем **and** в направлении листьев, используя правила:

and{**and**{x,y,z}, p, q} то же, что **and**{ x,y,z,p,q}
or{**or**{x,y,z}, p, q} то же, что **or**{ x,y,z,p,q}
and{**or**{x,y,z}, p, q} то же, что **or**{**and**{x,p,q}, **and**{y,p,q}, **and**{z,p,q}}
and(x,y,z} то же, что **and**{любой порядок следования x,y,z}
or(x,y,z} то же, что **or**{ любой порядок следования x,y,z}
and{} является TRUE, таким образом **or**{**and**{},x,y,z} всегда TRUE и **and{and**{},x,y,z} то же, что **and**{x,y,z}
or{} является FALSE, таким образом **and**{**or**{},x,y,z} всегда FALSE и **or**{**or**{},x,y,z} то же, что **or**{x,y,z}

ПРИМЕЧАНИЕ. – Используемая здесь нотация {x,y,z} (и т. д.) означает множество, состоящее из нуля, одного или более членов, таких как x, y и z.

В результате последовательного применения этих правил фильтр, в конечном счете, приобретает каноническую форму:

or{**and**{p_1, p_2 … }, **and**{q_1, q_2 … } …},

где все p_i или q_i являются либо пунктом фильтра F, либо инвертированным пунктом фильтра **not**{F}.

Все **and**{p_1, p_2 … }, таким образом, являются подфильтрами исходного фильтра.

Приложение R

Схемы имен составных статей и их применение

(|)

Понятие локальных имен членов вводится в п. 9.3. Настоящая спецификация Справочника не накладывает каких-либо ограничений на то, как могут распределяться имена, за исключением положений, содержащихся в правилах структуры. Вместе с тем, в некоторых ситуациях для достижения желаемого результата важным является создание схемы именования для членов семейства. В самой простой форме аналогичные члены семейства из разных составных статей могут иметь идентичные локальные имена членов. Например, член семейства, содержащий телефонный номер и связанные с ним характеристики (использование, тариф, ограничения и т. д.), может иметь одинаковое локальное имя члена в различных составных статьях. Это имеет значение, когда составные статьи являются членами иерархических групп (см. п. 7.13 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3). Схема может также быть образована путем отражения в RDN члена семейства того, какую информацию он содержит. Например, адрес для связи (телефонный номер или адрес электронной почты) может иметь RDN равный { comAdressName = telephone1 } или { comAdressName = emailAddress3 }. Местоположение всех, например членов семейства телефонных номеров, может далее определяться по совпадению начальной подстроки в RDN.

Приводимый далее пример использования схемы именования также является примером использования атрибутов управления, на которые делает ссылку компонент правила поиска **additionalControl** (см. п. 16.10.8). Этот пример должен восприниматься исключительно как пример, а не как спецификация, которая может быть реализована или на которую могут официально ссылаться другие спецификации. Это всего лишь пример того, как можно составить атрибут управления и какие спецификации могут быть связаны с таким атрибутом управления.

Правило поиска управляет процессом поиска в специальной области DIT. Эта служба адаптируется к конкретному пользователю, осуществляющему доступ. Однако "владельцы" статей, например абоненты, представленные статьями абонентов, могут иметь индивидуальные, возможно законные, требования в отношении того, как служба, связанная с этой конкретной статьей, должна ограничиваться и регулироваться. Такими индивидуальными требованиями могут быть следующие:

a) Информация в статье может быть представлена на разных языках. Вместе с тем, владелец статьи может потребовать, чтобы, например, адресная информация возвращалась на каком-либо конкретном языке, независимо от языка, который используется обращающимся пользователем в запросе поиска **search**, и содержания запроса обращающегося пользователя. Эта функция не может предоставляться посредством контекстной функции.

b) Владелец статьи может потребовать, чтобы возвращался фиктивный или альтернативный адрес, даже если обращающийся пользователь осуществляет подбор по реальному адресу.

c) Если обращающийся пользователь осуществляет подбор по одному телефонному номеру, он/она получает все или выборку телефонных номеров вместе со связанной информацией.

Такие индивидуальные ограничения и настройки могут выполняться с помощью атрибута управления выборкой **markingRules**. Этот атрибут предназначен для содержания статьей или порождающим элементом составной статьи в пределах специальной административной области службы. Атрибут имеет следующее определение:

```
markingRules ATTRIBUTE  ::=  {
        WITH SYNTAX                 MarkingRule
        USAGE                       directoryOperation
        ID                          id-oa-xx }

MarkingRule  ::=  SEQUENCE {
        searchRules                 SEQUENCE SIZE (1 .. MAX) OF INTEGER     OPTIONAL,
        markingStrands              [0]  Filter                             DEFAULT  and : { },
        localName                   [1]  SEQUENCE SIZE (1 .. MAX) OF FilterItem     OPTIONAL,
        explicitUnmark              [2]  Filter                             OPTIONAL }
```

Значение атрибута управления **markingRules** представляет правило для маркирования и снятия маркирования членов составной статьи, которые подбираются по результатам оценки операции поиска, и для исключения из результата операции прошедших совпадение статей, не являющихся членами данного семейства.

Компонент **searchRules** указывает, к каким правилам поиска применяется конкретное значение этого атрибута. Если управляющее правило поиска имеет идентификатор **id**, равный одному из значений в этом компоненте, должно применяться перемаркирование согласно определению этого значения атрибута управления. То или иное правило поиска может быть представлено в нескольких значениях данного типа атрибута. Если этот компонент отсутствует, правило маркирования применяется для всех правил поиска.

Компонент **markingStrands** актуален, только если **familyGrouping** в процессе сопоставления являлся либо **strand,** либо **multiStrand**. Он указывает, какое условие должно действовать для возможного маркирования цепочек. Фильтр этого компонента выполняет оценку по каждой цепочке, члены которой все были маркированы как участвующие члены в результате соответствия фильтру поиска. Значением оценки является TRUE, если, по крайней мере, одна цепочка выдает оценку TRUE. Сопоставление выполняется по правилам, описанным в п. 7.8 Рек. МСЭ-Т X.511 | ИСО/МЭК 9594-3. Если этот компонент отсутствует, это означает по умолчанию фильтр, который всегда выдает оценку TRUE.

Компонент **localName** актуален, только если **familyGrouping** в процессе сопоставления являлся либо **strand**, либо **multiStrand**, а **markingStrands** выдает оценку TRUE. Он затем указывает, какие цепочки имеют членов его семейства, маркированных как участвующие члены, отбирая нуль или больше членов семейства. Член семейства выбирается, если его локальное имя члена имеет то же количество RDN, что и количество пунктов фильтров в этом компоненте, и если все пункты фильтра дают совпадение, один за одним, соответствующего RDN. Пункт фильтра дает совпадение RDN, если он дает совпадение AVA этого RDN. Все члены семейства любой цепочки, проходящей через отобранный член семейства, маркируются как участвующие.

Компонент **explicitUnmark** определяет фильтр, который при совпадении статьи или члена семейства вызывает снятие маркировки явным образом с этой статьи или члена семейства. Снятие маркирования явным образом актуально только для статей и членов семейства, которые были отобраны для возвращения в результате поиска **search**. Если член семейства демаркируется явным образом и если группирование семейства в процессе сопоставления фильтром поиска не являлось **entryOnly**, все статьи семейства, подчиненные по отношению к явно демаркированному члену, также являются явным образом демаркированными. Снятие явным образом маркирования статьи, не являющейся членом семейства, означает удаление этой статьи из результата, как если бы она не прошла сопоставление. Снятие явным образом маркирования члена семейства означает, что этот член не включается в результат.

Оценка атрибута управления **markingRules** выполняется как процесс, состоящий из двух этапов.

Первый этап выполняется, только если **familyGrouping** в процессе сопоставления являлось либо **strand**, либо **multiStrand**, и **familyReturn** в выборе информации статьи не является **contributingEntriesOnly**.

На первом этапе рассматриваются только составные статьи, в отношении которых было выдано совпадение в результате оценки фильтром поиска, при выполнении следующих условий:

a) порождающий элемент содержит атрибут управления **markingRules**;

b) одно или более значений применимы для управляющего правила поиска и включают компонент **localName**.

Затем дополнительные члены маркируются как участвующие члены, как указано выше.

На втором этапе все члены семейства уже маркированы как участвующие члены и все статьи, не являющиеся членами семейства, проверены на наличие типа атрибута управления **markingRules**, и затем проверяется, имеет ли этот атрибут одно или более значений, применимых для управляющего правила поиска. Если это так, компонент **explicitUnmark**, если он присутствует, проходит оценку. Если результатом оценки является TRUE для какого-либо члена семейства, он демаркируется явным образом, т. е. он не маркируется ни как участвующий, ни как входящий в состав. Все подчиненные члены семейства аналогично явным образом демаркируются. В случае статьи – не члена семейства явное снятие маркировки имеет тот же результат, как если бы фильтр не выдал совпадения статьи.

Приложение S

Концепции и принципы именования

(|)

S.1 История учит нас …

Первое издание настоящих спецификаций Справочника было впервые опубликовано в 1988 году, и с тех пор в отрасли информатики происходили бесчисленные перемены. Некоторые из них прогнозировались и ожидались, другие – нет. Вследствие этого значительная часть того, что сейчас публикуется в этих спецификациях Справочника, применима так же, как это было в 1988 году, в то время как остальные части, очевидно, неприменимы. В данном приложении мы обозначим ключевые концепции, как остающиеся в силе, так и неприменимые, которые в настоящее время требуют рассмотрения.

S.1.1 Первоначальные концепции, которые остаются в силе

К счастью, многие из первоначальных концепций Справочника, которые продолжают оставаться в силе, относятся к наиболее фундаментальным из исходных спецификаций, а именно:

- до сих пор справедливо рассматривать Справочник как набор статей, где каждая содержит информацию в форме атрибутов, описывающих конкретный объект реального мира;
- также по-прежнему справедливо представлять статьи Справочника как именованные объекты, имена которых образуют иерархию, представляющую определенную логическую систематизацию, согласно которой могут быть организованы связанные с ней объекты реального мира;
- сохраняется необходимость обеспечения гибкости в именовании и возможности делегирования полномочий в отношении именования по иерархическим каналам;
- по-прежнему правильно предполагать, что статьи распределяются среди множества (потенциально весьма обширного) серверов справочника;
- также верно полагать, что в отношении любого фрагмента информации об объекте реального мира, Справочник быстро отыщет статью, описывающую этот объект; и
- продолжает оставаться справедливым, что этот фрагмент информации может быть именем статьи или любым атрибутом, который не обозначает имя и содержится в статье.

S.1.2 Первоначальные концепции, которые утратили силу

Несмотря на перечень фундаментальных концепций, сохраняющих свою актуальность, существуют также фундаментальные концепции, которые, как показывает опыт последнего десятилетия, не могут более считаться действительными. Некоторые из них уже были адаптированы в рамках данных спецификаций Справочника, с другими этого не произошло. К претерпевшим изменения относятся следующие:

- уже неверно было бы считать, что любой объект реального мира может быть описан исключительно одной статьей (т. е. существуют связанные статьи);
- невзирая на принципы безопасности, неверно полагать, что знания именования, содержащиеся в пределах Справочника, окажутся достаточными для обнаружения всех именованных статей Справочника (т. е. существуют множественные DIT);
- неверным более является положение, согласно которому знания именования, содержащиеся в пределах Справочника, – это единственный путь к конкретной именованной статье (т. е. можно использовать службы, внешние по отношению к Справочнику, для содействия в установлении местоположения именованной статьи);
- ошибочно полагать, что выделенные имена всегда уникальным образом именуют единичную статью (т. е. то же DN может использоваться для именования статей, содержащихся в двух и более DIT).
- при наличии произвольного фрагмента (если предполагается, что он является единственным случаем) не относящейся к имени информации об объекте, который может находиться в одном из нескольких серверов справочника, уже несправедливо представлять, что распределенный поиск является единственным механизмом, который можно использовать для обнаружения искомой статьи (т. е. действует требование, согласно которому единичный сервер должен локально и детерминистически опознавать связанную статью, независимо от того, содержится ли эта статья в этом сервере).

S.2 Новый взгляд на разрешение имен

В силу того, что именование является столь важным фактором успешного функционирования службы Справочника и что некоторые основополагающие предположения о природе службы Справочника поставлены в настоящее время под сомнение, обратимся в этом подпункте к теме разрешения имен. Вначале в подпункте приводится критический

анализ действующего порядка разрешения имен и делается предположение, что существующая модель разрешения имен уже недостаточна для удовлетворения всех требований справочника. Далее в подпункте предлагается альтернативный метод расширения модели для удовлетворения этих потребностей при обеспечении обратной совместимости с существующими системами.

S.2.1 Модель явных знаний

С момента первого опубликования эти спецификации Справочника используются для распределенного разрешения имен. Концептуально каждый участвующий в данной области имен DSA должен сохранять минимальные знания именования для обеспечения прогнозируемости выполнения распределенного по всему DIT разрешения имен (в зависимости, разумеется, от способности действительно обратиться ко всем участвующим серверам). Конкретно, минимальные знания включают старшие и младшие ссылки на знания, что придает DIT то, что можно назвать "жесткой связанностью", за неимением более удачного термина. При использовании этой модели любой DSA, участвующий в разрешении потенциального имени, должен достоверно знать, какие из следующих трех условий выполняются:

- потенциальное имя попадает в пределы контекста именования, содержащегося данным DSA;
- потенциальное имя попадает в подчиненную область имен, известную этому DSA; или
- ни одно из вышеуказанных условий не выполняется.

В первом случае этот DSA должен завершить процесс разрешения имени, либо опознав статью, либо определив, что таковой не существует. Во втором случае разрешение имени должно продолжаться следованием по подчиненной ссылке на другой DSA. В третьем случае разрешение имени должно продолжаться следованием по старшей ссылке, если таковая старшая ссылка существует, в противном случае процесс завершается. При условии строгой связанности DIT разрешение имен всегда приводит к окончательному ответу. Либо статья существует в конкретном DSA, либо не существует.

На рисунке S.1 показан пример сценария, в котором разрешение имени выполняется на основе имени статьи, содержащейся в DSA 2. На рисунке ссылки на знания показаны пунктирными линиями со стрелками. Заметим, что DSA 3, хотя и содержит контекст именования, подчиненный по отношению к DSA 2, имеет старшую ссылку на DSA 1, который содержит корневой контекст именования. В зависимости от участвующего DSA разрешение имени продолжается следующим образом:

- для DSA 1 разрешение имени следует по подчиненной ссылке на DSA 2;
- для DSA 2 разрешение имени находит именованную статью;
- для DSA 3 разрешение имени следует по старшей ссылке на DSA 1 и выполняется, как указано выше;
- для DSA 4 разрешение имени следует по старшей ссылке на DSA 1 и выполняется, как указано выше.

Во всех случаях разрешение имени находит именованную статью.

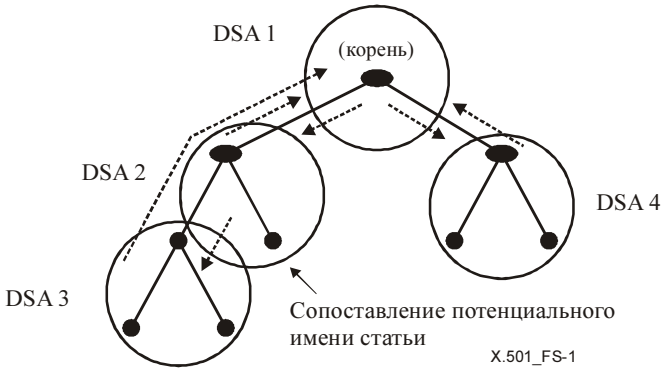


Рисунок S.1

Следует заметить, что, несмотря на возможность определенной оптимизации, определенность содержания ответа не меняется. Две очевидные возможности оптимизации заключаются в использовании непосредственно старшей ссылки в DSA 3 (устраняя необходимость пересечения DSA 1 на пути к DSA 2) и включении перекрестной ссылки в DSA 4, делая возможным выполнение разрешения имени непосредственно из DSA 4 к DSA 2 (опять-таки избегая пересечения DSA 1). В любом случае разрешение имени в этом примере, независимо от точки начала, всегда приводит к одному и тому же ответу.

К сожалению, как отмечалось выше, предположение о жесткой связанности DIT уже утратило силу. Существуют множественные DIT, зачастую включающие дублирующие DN. Оставив пока за скобками возможность наличия дублирующих имен, мы имеем ситуацию, аналогичную представленной на рисунке S.2. В этом примере мы имеем

два DIT, каждое характеризуется строгой связанностью в пределах самого себя, но ни одно из них не обладает знаниями о другом. Одно DIT, как в предыдущем примере, состоит из статей, которые содержатся в DSA 1–DSA 4. Второе DIT состоит из статей, содержащихся в DSA 5 и DSA 6. Отметим, что может быть по-прежнему справедливым считать это единым DIT, поскольку все DN являются обособленными относительно концептуального корня. Вместе с тем от единичного DIT со строгой связанностью это отличается тем, что DSA 1 и DSA 5 не имеют полных знаний о контекстах именования, подчиненных по отношению к корню.

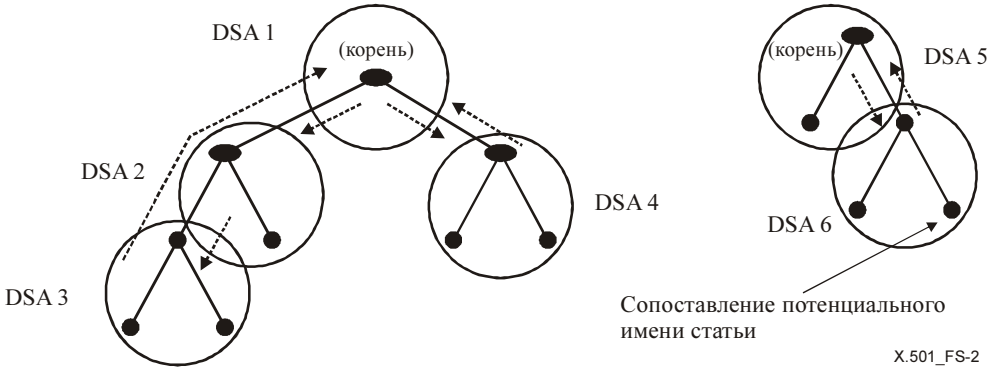


Рисунок S.2

Как показано на этом рисунке, при обозначенном имени статьи разрешение имени выполняется следующим образом:

– из DSA 1–DSA 4 ни один не находит эту статью;
– DSA 5–DSA 6 успешно находят эту статью.

Неудача в обнаружении статьи может быть или не быть проблемой, в зависимости от существующих требований. Далее рассматриваются только те ситуации, когда эта неудача представляет проблему.

В поисках решения разумным на первый взгляд представляется изучение использования перекрестной ссылки или некоей аналогичной структуры. Рассмотрим, например, применение перекрестной ссылки при знаниях о контексте именования DSA 4 в DSA 6. Это показано на рисунке S.3.

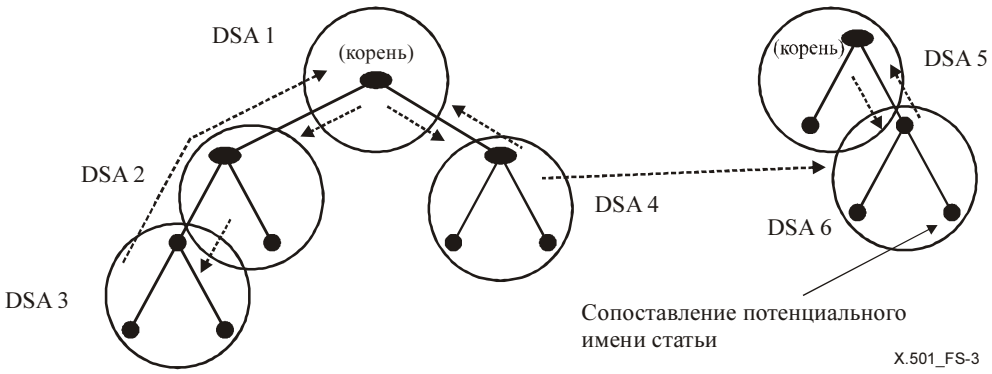


Рисунок S.3

Экспресс-анализ этого подхода показывает следующие сценарии:

– разрешение имени в DSA 1–DSA 3 завершается неудачей;
– разрешение имени в DSA 4–DSA 6 завершается успешно.

Несмотря на то, что на первый взгляд это может показаться приблизительно столь же приемлемым, что и предыдущий сценарий, между ними существует принципиальное различие: разрешение имени в области действия DIT со строгой связанностью теперь выдает противоречивые результаты.

Для обеспечения последовательных результатов имеются два варианта, предусматривающие использование существующих структур знаний. Одним подходом является использование множественных перекрестных ссылок, когда каждый DSA, как взгляд "из", имеет перекрестную ссылку на искомый контекст именования. Эта концепция иллюстрируется на рисунке S.4. Заметим, что в этом сценарии разрешение имени при взгляде слева последовательно

обнаружит любое имя в пределах контекста именования, содержащегося в DSA 6. Также заметим, что имена в DSA 5 не могут быть найдены этим способом, и заметим, что имена в DSA 1–DSA 4 по-прежнему недоступны для DSA 5–DSA 6.

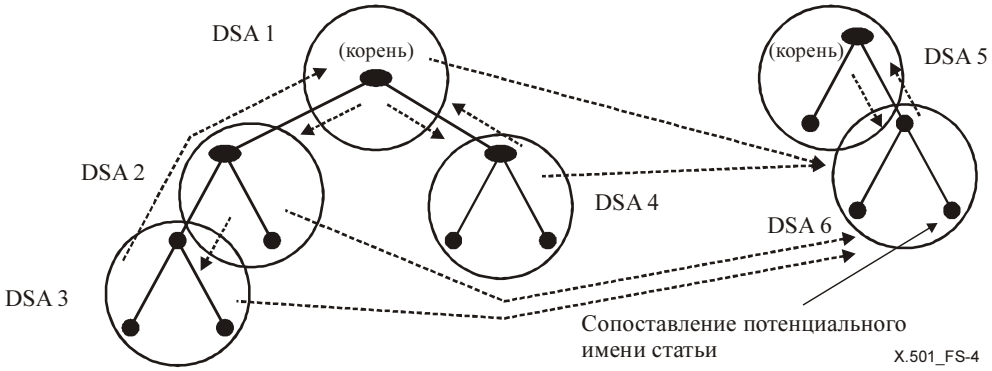


Рисунок S.4

Проблема, однако, может заключаться в том, как перекрестная ссылка реализуется в DSA 1. То есть, с позиции DSA 1 контекст именования, о котором идет речь в DSA 5, может фактически быть подчиненным по отношению к статье, которую он, согласно своей информации, содержит. В частности, если DSA 1 считает себя обладающим полномочиями в отношении корневого контекста, может потребоваться, чтобы эта ссылка фактически была подчиненной ссылкой, что приводит нас ко второму варианту.

Вторым вариантом обеспечения достоверного разрешения имен при взгляде слева в контекст именования DSA 6 является создание подчиненной ссылки корневого уровня в DSA 1. Это показано на рисунке S.5. Реализованные таким образом любые перекрестные ссылки в DSA 2, DSA 3 или DSA 4 будут просто вариантами оптимизации.

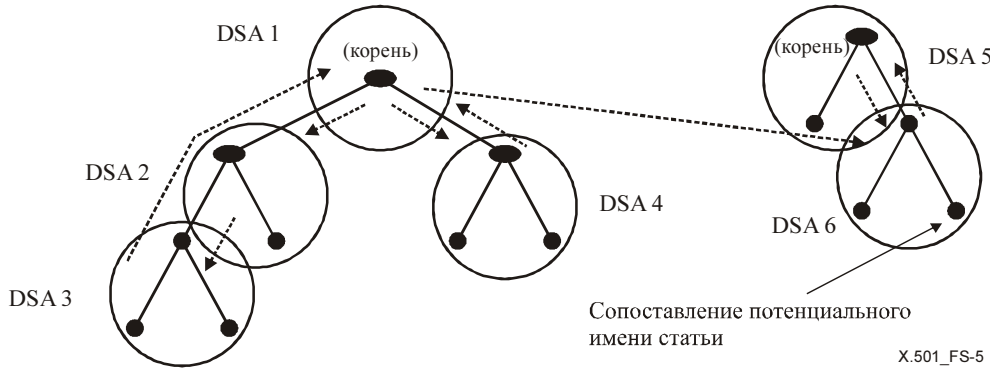


Рисунок S.5

При расширении этой концепцию еще на один шаг, как показано на рисунке S.6, возникают интересные вопросы. На этом рисунке DSA 1 имеет полные подчиненные знания о контекстах именования, содержащихся всеми шестью DSAs, в то время как DSA 5 обладает знаниями только о контекстах именования, содержащихся DSA 5 или DSA 6. Заметим, что если DSA 5 снабдить подчиненными знаниями о контекстах именования, содержащихся DSA 2 и DSA 4, вся ситуация опять будет представлять строго связанный взгляд на DIT. Однако это не так. По сути произошло размывание различия между строго связанным, одиночным DIT и множественными DIT, в результате чего создалась ситуация, которая недостаточно смоделирована в действующей спецификации справочника. Сугубо практически это отражение того, что происходит весьма часто.

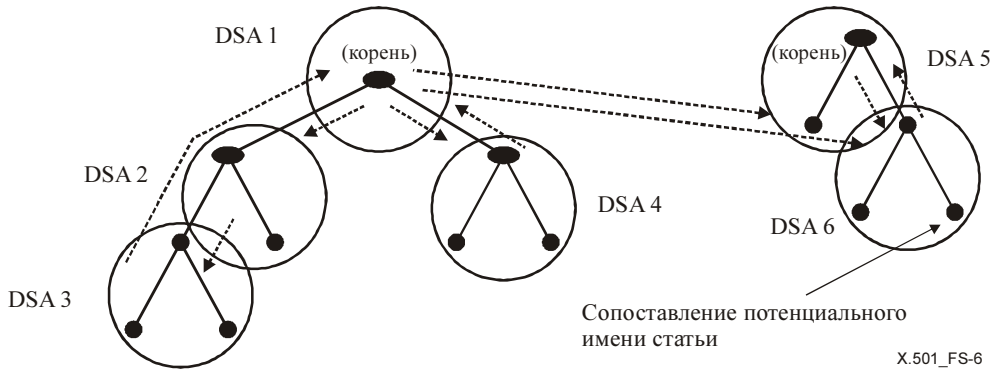


Рисунок S.6

Для того чтобы понять, как еще трудности нас ожидают, рассмотрим теперь случай, когда одно DN существует в более чем одном DIT. На рисунке S.7 показан простой сценарий. В этом примере DIT представлено как существующее в DSA 5. Область имен в этом новом DIT частично перекрывает область предыдущего примера, но также вводит некоторые новые имена. В частности, стрелка указывает на статью, которая вместе со своим родителем имеет то же имя, что и одна из статей в DSA 2. Пары статей, имеющих одно и то же имя, могут содержать или не содержать одинаковую информацию, поэтому они не должны рассматриваться как одна и та же статья.

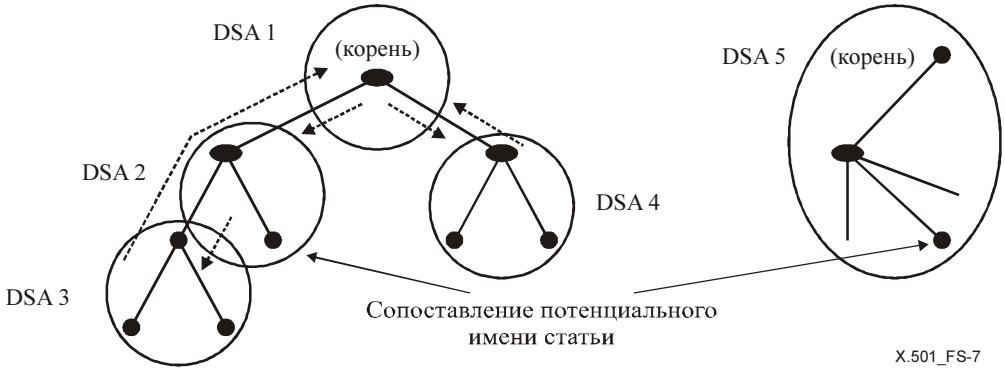


Рисунок S.7

В отсутствие каких бы то ни было ссылок между этими двумя DIT разрешение имени вполне прогнозируемо. В пределах конкретного DIT оно всегда выдаст один и тот же результат, Введение ссылок создает определенные проблемы:

- перекрестная ссылка из DSA 2 на DSA 5 или из DSA 5 на DSA 2 ни при каких условиях не вызовет переход, поскольку каждый DSA считает, что он содержит представляющий интерес контекст именования;
- перекрестная ссылка из либо DSA 3 или DSA 4 на DSA 5 будет иметь приоритет относительно старших ссылок;
- поведение процесса при наличии как перекрестной ссылки из DSA 3 на DSA 5 и непосредственно старшей ссылки из DSA 3 на DSA 2 недетерминировано;
- поведение процесса при наличии подчиненной ссылки из DSA 1 и в DSA 2, и в DSA 5 недетерминировано.

Очевидно, что эти проблемы являются нежелательными. Существуют дополнительные сценарии, которые можно рассмотреть, однако перечисленных выше проблем достаточно для признания этого подхода неприемлемым. К сожалению, ситуации, приводящие к этому конкретному типу сценария именования и распределения знаний, в реальности возникают слишком часто, чтобы их игнорировать. Следовательно, необходимо какая-то форма расширения. В следующей части подпункта рассматривается альтернативный подход.

S.2.2 Разрешение имен при наличии неявных знаний

Во всех рассмотренных выше примерах разрешение имен целиком основывалось на ссылках на явные знания, содержащихся в DSA. За пределами спецификаций Справочника (в основном в IETF) несколько лет назад начались работы над концепцией разрешения имен посредством использования неявных знаний. Другими словами, имеется комплекс работ, который использует информацию, содержащуюся в пределах самого DN, для частичного разрешения имени до первоначального обращения клиента к DSA. Концептуально, если имя содержит достаточно информации, DSA, к которому обращаются первоначально, будет в состоянии дать окончательный ответ: или он содержит именованную статью, или достоверно знает что такая статья не существует.

Данная концепция проиллюстрирована на рисунке S.8, ниже. Этот рисунок аналогичен рисунку S.1, за исключением того, что DSA на этом рисунке не содержат ссылок на знания. Вместо этого знания неявно содержатся в DN, и разрешение происходит с помощью службы, внешней по отношению к справочнику, которая показана на рисунке в виде пресловутого черного ящика. Заметим, что этот черный ящик способен предоставлять указатели на все контексты именования, кроме корня. Местоположение корня не может быть определено таким способом, поскольку нулевое DN, связанное с корнем, не обладает какими-либо знаниями о своем местоположении.

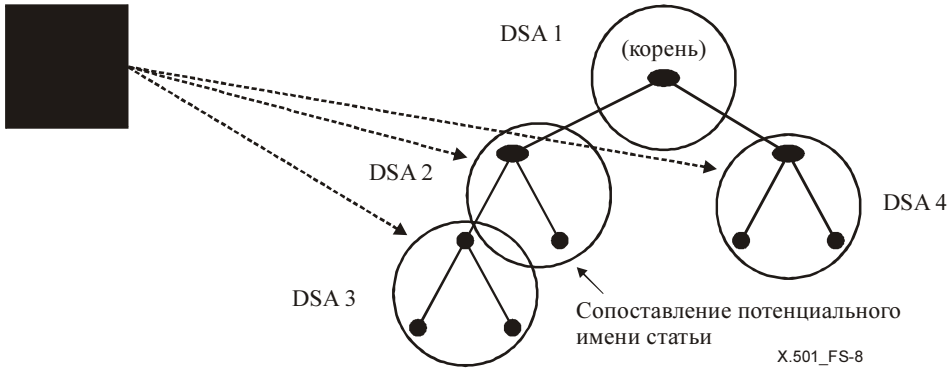


Рисунок S.8

Далее рассмотрим схему на рисунке S.9, которая соответствует представлениям DIT, показанным на рисунке S.2. В этом сценарии, предполагая ту же модель неявных знаний, та же служба – черный ящик способна указывать на контексты именования, добавленные к правой части рисунка. В противоположность ситуации на рисунке S.2 контексты именования в DSA 5 и DSA 6 не создают совершенно другого представления. Предполагая необходимую связность реализованной, все шесть DSA появляются в одной проекции, даже в отсутствие явных знаний среди всех DSA.

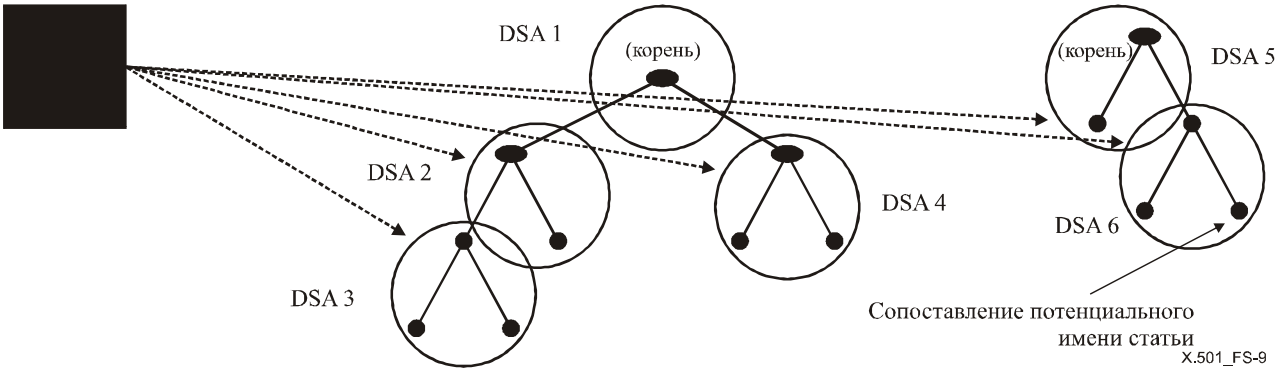


Рисунок S.9

Самой ранней из опубликованных на эту тему работ является RFC 2247[1], в котором определяется отображение между выделенными именами и системой именования доменов (DNS). С тех пор были опубликованы другие работы, еще ряд готовится к публикации. К настоящему времени все опубликованные работы по этой теме базируются на использовании специального атрибута именования, называемого атрибутом domainComponent (dc).

Упрощенная для целей рассмотрения работа по этой проблематике привела к созданию концепции, согласно которой DN, составленные с использованием атрибута dc в своих наиболее значащих RDN, могут разрешаться неявным образом с использованием DNS в качестве внешней службы – черного ящика до DSA, содержащего контекст именования. После этого происходит обращение к DSA, и разрешение имени завершается в этом DSA.

[1] IETF RFC 2247 (1998 г.), *Использование доменов в выделенных именах LDAP/X.500.*

Приложение T

Алфавитный указатель определений

(|)

В данном Приложение в алфавитном порядке перечисляются все термины, определенные в настоящей спецификации Справочника, с указанием ссылок на разделы, в которых они опредлены.

Приложение U

Поправки и исправления

(|)

Настоящее издание спецификации Справочника включает следующие проекты поправок к предыдущему изданию, по которым было проведено голосование и которые были утверждены ИСО/МЭК:

- Поправка 1 для расширений в целях поддержки страниц с результатами в DSP;
- Поправка 2 для расширений в целях поддержки концепции дружественных атрибутов;
- Поправка 3 для большей согласованности X.500 и LDAP между собой;
- Поправка 4 для совершенствования сертификатов открытых ключей и атрибутов.

Настоящее издание этой спецификации Справочника включает технические исправления, которые учитывают следующие сообщения о дефектах: 306 и 312.

СЕРИИ РЕКОМЕНДАЦИЙ МСЭ-Т

Серия A	Организация работы МСЭ-Т
Серия D	Общие принципы тарификации
Серия E	Общая эксплуатация сети, телефонная служба, функционирование служб и человеческие факторы
Серия F	Нетелефонные службы электросвязи
Серия G	Системы и среда передачи, цифровые системы и сети
Серия H	Аудиовизуальные и мультимедийные системы
Серия I	Цифровая сеть с интеграцией служб
Серия J	Кабельные сети и передача сигналов телевизионных и звуковых программ и других мультимедийных сигналов
Серия K	Защита от помех
Серия L	Конструкция, прокладка и защита кабелей и других элементов линейно-кабельных сооружений
Серия M	Управление электросвязью, включая СУЭ и техническое обслуживание сетей
Серия N	Техническое обслуживание: международные каналы передачи звуковых и телевизионных программ
Серия O	Требования к измерительной аппаратуре
Серия P	Качество телефонной передачи, телефонные установки, сети местных линий
Серия Q	Коммутация и сигнализация
Серия R	Телеграфная передача
Серия S	Оконечное оборудование для телеграфных служб
Серия T	Оконечное оборудование для телематических служб
Серия U	Телеграфная коммутация
Серия V	Передача данных по телефонной сети
Серия X	**Сети передачи данных, взаимосвязь открытых систем и безопасность**
Серия Y	Глобальная информационная инфраструктура, аспекты межсетевого протокола и сети последующих поколений
Серия Z	Языки и общие аспекты программного обеспечения для систем электросвязи

Отпечатано в Швейцарии
Женева, 2007 г.