

ИНФОРМАЦИОННАЯ ТЕХНОЛОГИЯ

ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ

СПРАВОЧНИК

Ч А С Т Ь 3

ОПРЕДЕЛЕНИЕ АБСТРАКТНЫХ УСЛУГ

Издание официальное

Предисловие

1 РАЗРАБОТАН Московским научно-исследовательским центром (МНИЦ) Государственного Комитета Российской Федерации по связи и информатизации

ВНЕСЕН Техническим Комитетом по стандартизации ТК 22 «Информационные технологии»

2 ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 19 мая 1998 г. № 215

3 Настоящий стандарт содержит полный аутентичный текст международного стандарта ИСО/МЭК 9594-3—95 «Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 3. Определение абстрактных услуг»

4 ВВЕДЕН ВПЕРВЫЕ

© ИПК Издательство стандартов, 1998

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Госстандарта России

Содержание

Введение	IV
1 Область применения	1
2 Нормативные ссылки	1
3 Определения	1
4 Сокращения	2
5 Соглашения	3
6 Общее описание услуг справочника	3
7 Типы информации и общие процедуры	3
8 Операции «связка» и «развязка»	14
9 Операции справочника типа «чтение»	16
10 Операции справочника типа «поиск»	19
11 Операции справочника типа «модификация»	25
12 Ошибки	31
Приложение А. Абстрактные услуги в АСН.1	36
Приложение В. Операционная семантика для управления базовым доступом	43

Введение

Настоящий стандарт разработан с целью обеспечения взаимосвязи систем обработки информации, предназначенных для предоставления услуг справочника. Совокупность подобных систем вместе с содержащейся в них информацией справочника может рассматриваться как единое целое, называемое справочником. Информация, хранимая справочником и называемая в целом «информационной базой справочника» (ИБС), используется обычно для обеспечения обмена данными между такими объектами, как логические объекты прикладного уровня, персонал, терминалы и дистрибутивные списки.

Справочник играет существенную роль во взаимосвязи открытых систем (ВОС), цель которой состоит в том, чтобы при минимуме технических согласований вне стандартов во ВОС обеспечить взаимосвязь систем обработки информации.

- поставляемых от различных изготовителей;
- использующих различные методы административного управления;
- имеющих различные уровни сложности;
- использующих различные технологии.

Стандарт определяет возможности, обеспечиваемые справочником для своих пользователей.

В приложении А представлен модуль АСН.1 для абстрактных услуг справочника. В приложении В приведены диаграммы, которые описывают семантику, связанную с базовым управлением доступом, в том виде, как она используется при выполнении операций справочника.

Информационная технология
ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ
СПРАВОЧНИК

Часть 3.

Определение абстрактных услуг

Information technology. Open Systems Interconnection. The directory.
 Part 3. Abstract service definition

Дата введения 1999—01—01

1 ОБЛАСТЬ ПРИМЕНЕНИЯ

Настоящий стандарт определяет в абстрактном виде внешне наблюдаемые услуги, обеспечиваемые справочником.

2 НОРМАТИВНЫЕ ССЫЛКИ

Настоящий стандарт содержит ссылки на следующие стандарты:

ГОСТ Р ИСО/МЭК 9072-1—93 Системы обработки информации. Передача текста. Удаленные операции. Часть 1. Концепции, модель и нотация

ГОСТ Р ИСО/МЭК 9594-1—98 Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 1. Общее описание принципов, моделей и услуг

ГОСТ Р ИСО/МЭК 9594-5—98 Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 5. Спецификации протокола

ГОСТ Р ИСО/МЭК 9594-6—98 Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 6. Выбранные типы атрибутов

ГОСТ Р ИСО/МЭК 9594-8—98 Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 8. Основы аутентификации

ИСО/МЭК 9594-2—93* Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 2. Модели

ИСО/МЭК 9594-4—93* Информационная технология. Взаимосвязь открытых систем. Справочник. Часть 4. Процедуры распределенных операций

3 ОПРЕДЕЛЕНИЯ

В настоящем стандарте применяют следующие определения.

3.1 Базовые определения справочника

В настоящем стандарте использованы следующие термины, определенные в ГОСТ Р ИСО/МЭК 9594-1:

- a) справочник;
 - b) информационная база справочника;
 - c) пользователь (справочника).
- ### 3.2 Определения модели справочника

В настоящем стандарте использованы следующие термины, определенные в ИСО/МЭК 9594-2

* Оригинал стандарты и проектов ИСО/МЭК — во ВНИИКИ Госстандарта России.

- a) агент системы справочника;
- b) агент пользователя справочника.

3.3 Определения информационной базы справочника

В настоящем стандарте использованы следующие термины, определенные в ИСО/МЭК 9594-2:

- a) запись псевдонима;
- b) дерево информации справочника;
- c) запись (справочника);
- d) непосредственный старший;
- e) запись/объект непосредственно старшего;
- f) объект;
- g) класс объекта;
- h) запись объекта;
- i) подчиненный;
- j) старший.

3.4 Определения записи справочника

В настоящем стандарте использованы следующие термины, определенные в ИСО/МЭК 9594-2:

- a) атрибут;
- b) тип атрибута;
- c) значение атрибута;
- d) условие значения атрибута;
- e) операционный атрибут;
- f) атрибут пользователя;
- g) правило сравнения.

3.5 Определения имени

В настоящем стандарте использованы следующие термины, определенные в ИСО/МЭК 9594-2:

- a) псевдоним, имя псевдонима;
- b) различительное имя;
- c) имя (справочника);
- d) предполагаемое имя;
- e) относительное различительное имя.

3.6 Определения распределенных операций

В настоящем стандарте использованы следующие термины, определенные в ИСО/МЭК 9594-4:

- a) сссылание;
- b) обращение.

3.7 Определения абстрактных услуг

В настоящем стандарте определены следующие термины:

- a) **фильтр** — утверждение о наличии или о значении некоторых атрибутов записи с целью ограничения области поиска;
- b) **инициатор** — пользователь, который начинает операцию;
- c) **служебные ограничения** — параметры, передаваемые как часть операции, которые ограничивают различные аспекты ее выполнения.

4 СОКРАЩЕНИЯ

- ВОС — взаимосвязь открытых систем
- ИБС — информационная база справочника
- ДИС — дерево информации справочника
- АСС — агент системы справочника
- АПС — агент пользователя справочника
- РАУС — регион административного управления справочником
- ОРИ — относительное различительное имя

5 СОГЛАШЕНИЯ

В настоящем стандарте под понятием «спецификация справочника» следует понимать ГОСТ Р ИСО/МЭК 9594-3, а под понятием «спецификации справочника» — части 1—9 ГОСТ Р ИСО/МЭК 9594.

Пронумерованные элементы списка (в отличие от элементов с предшествующими знаками дефиса или буквами) должны рассматриваться как шаги процедуры.

Настоящий стандарт определяет операции справочника, используя нотацию удаленных операций, определенную в ГОСТ Р ИСО/МЭК 9072-1.

6 ОБЩЕЕ ОПИСАНИЕ УСЛУГ СПРАВОЧНИКА

В соответствии с ИСО/МЭК 9594-2 услуги справочника предоставляются агентам пользователя справочника (АПС) через пункты доступа, где каждый агент действует от имени пользователя. Этот принцип показан на рисунке 1. Справочник предоставляет услуги своим пользователям через пункт доступа, выполняя набор операций справочника.

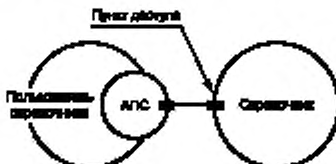


Рисунок 1 — Доступ к справочнику

Существуют три типа операций справочника:

- операции типа «чтение справочника» запрашивают одну запись справочника;
- операции типа «поиск справочника» потенциально запрашивают несколько записей справочника;
- операции типа «модификация справочника».

Эти типы операций определены в разделах 9—11 соответственно. Требования соответствия операциям справочника определены в ГОСТ Р ИСО/МЭК 9594-3.

7 ТИПЫ ИНФОРМАЦИИ И ОБЩИЕ ПРОЦЕДУРЫ

7.1 Введение

В данном разделе идентифицированы, а в некоторых случаях и определены многие типы информации, которые в дальнейшем используются в определении операций справочника. К таким типам информации относятся либо достаточно общие для нескольких операций, которые вероятно могут появиться в будущем, либо достаточно сложные или «самоопределяющиеся» в связи с тем, что их показатели определены отдельно от используемой их операции.

Некоторые типы информации, используемые в определении услуг справочника, фактически определены в других стандартах. В подразделе 7.2 идентифицированы эти типы и указаны источники их определения. В каждом из последующих подразделов (7.3—7.11) идентифицирован и определен тип информации.

В данном разделе также определены некоторые общие элементы процедуры, которая применяется в большей части или во всех операциях справочника.

7.2 Типы информации, определенные в других стандартах

Типы информации, определенные в ИСО/МЭК 9594-2:

- Attribute;
- AttributeType;
- AttributeValue;

- d) AttributeValueAssertion;
- e) DistinguishedName;
- f) Name;
- g) RelativeDistinguishedName.

Тип информации, определенный в ГОСТ Р ИСО/МЭК 9594-6.

- a) PresentationAddress.

Типы информации, определенные в ГОСТ Р ИСО/МЭК 9594-8.

- a) Certificate,
- b) SIGNED;
- c) CertificationPath.

Тип информации, определенный в ГОСТ Р ИСО/МЭК 9072-1.

- a) InvokeID.

Типы информации, определенные в ИСО/МЭК 9594-4.

- a) OperationProgress,
- b) ContinuationReference

7.3 Общие аргументы

Информация CommonArguments может быть представлена для определения вызова каждой операции, которую может выполнить справочник.

CommonArguments	::=	SET	
serviceControls	[30]	ServiceControls DEFAULT {},	
securityParameters	[29]	SecurityParameters OPTIONAL,	
requestor	[28]	DistinguishedName OPTIONAL,	
operationProgress	[27]	OperationProgress DEFAULT { nameResolutionPhase notStarted },	
aliasesRDNs	[26]	INTEGER OPTIONAL,	
criticalExtensions	[25]	BIT STRING OPTIONAL,	
referenceType	[24]	ReferenceType OPTIONAL,	
entryOnly	[23]	BOOLEAN DEFAULT TRUE,	
exclusions	[22]	Exclusions OPTIONAL,	
nameResolveOnMaster	[21]	BOOLEAN DEFAULT FALSE }	

Компонент ServiceControls определен в 7.5. Его отсутствие равноценно наличию пустого набора служебных ограничений.

Компонент SecurityParameters определен в 7.9. Его отсутствие равноценно наличию пустого набора параметров защиты.

Различительное имя requestor идентифицирует инициатора конкретной операции. Оно содержит имя пользователя, которое идентифицировано во время обращения к справочнику. Различительное имя может потребоваться, если запрос должен выдаваться с подписью (см. 7.10), и должно содержать имя пользователя, инициирующего запрос.

Компоненты OperationProgress, referenceType, entryOnly, exclusions и nameResolveOnMaster определены в ИСО/МЭК 9594-4. Они обеспечиваются АПС только при действии непрерывного указателя, выдаваемого АПС в ответ на предыдущую операцию, и их значения копируются АПС из этого непрерывного указателя.

Компонент aliasesRDNs информирует АПС о том, какой компонент object операции был создан путем переименования псевдонима во время предыдущей попытки операции. Целочисленное значение указывает количество относительных различительных имен (ОРИ) в имени, полученном из переименованного псевдонима. (Это значение может быть установлено в ответе на обращение к предыдущей операции.)

Примечание — Этот компонент предусмотрен для обеспечения совместности реализаций справочника издания 1988 г. АПС (и АСС), реализуемые в соответствии с более поздними изданиями спецификации справочника, должны всегда опускать этот параметр из CommonArguments последующего запроса. Таким образом, справочник не будет сигнализировать об ошибке, если одни псевдонимы переименованы в другие.

7.3.1 Критические расширения

Компонент criticalExtensions обеспечивает способ перечисления набора расширений, которые критичны к выполнению операций справочника. Если инициатор расширенной операции желает

указать, что операция должна быть выполнена с одним или несколькими расширениями (т.е., что выполнение операции без этих расширений неприемлемо), он это делает путем установки бита(ов) `criticalExtensions`, который(ые) соответствует(ют) этому(им) расширению(ям). Если справочник (или некоторая его часть) не способен выполнить критическое расширение, он выдает указатель `unavailableCriticalExtension` (в виде `ServiceError` или `PartialOutcomeQualifier`). Если справочник не способен выполнить расширение, которое не критично, он игнорирует наличие расширения.

Настоящий стандарт определяет ряд расширений, которые возможны в реализациях справочника издания 1993 г. Эти расширения представлены в виде дополнительных перечисляемых битов в `BIT STRING` или в виде дополнительных компонентов `SET` или `SEQUENCE` и игнорируются системами версии 1988 г. Каждому такому расширению присваивается целочисленный идентификатор, являющийся номером бита, который может быть установлен в `criticalExtensions`. Если критичность расширения определяется как критическая, АПС должен установить в компоненте `criticalExtensions` соответствующий бит. Если определенная критичность не является критической, АПС может, но не обязательно, установить соответствующий бит в `criticalExtensions`.

Расширения, их идентификаторы, операции, в которых они возможны, рекомендуемая критичность, и разделы, в которых они определены, приведены в таблице 1.

Таблица 1 — Расширения

Расширение	Идентификатор	Операции	Критичность	Определено
<code>subentries</code>	1	Все	Некритично	7.5
<code>copyShallDo</code>	2	Чтение, сравнение, список, поиск	*	7.5
<code>attributeSizeLimit</code>	3	Чтение, поиск	*	7.5
<code>extraAttributes</code>	4	То же	*	7.6
<code>modifyRightsRequest</code>	5	Чтение	*	9.1
<code>pagedResultsRequest</code>	6	Список, поиск	*	10.1
<code>matchedValuesOnly</code>	7	Поиск	*	10.2
<code>extendedFilter</code>	8	*	*	10.2
<code>targetSystem</code>	9	Добавить/Запись	Критично	11.1
<code>useAliasOnUpdate</code>	10	Добавить/Запись, Удалить/Запись, Модифицировать/Запись, Переместить/Запись	*	11.1
<code>newSuperior</code>	11	Модифицировать/Запись	*	11.4

7.4 Общие результаты

Информация `CommonResults` должна быть представлена, чтобы квалифицировать результат каждой операции поиска, которую справочник может выполнить.

`CommonResults` : = SET {

`securityParameters` [30] `SecurityParameters` OPTIONAL,
`performer` [29] `DistinguishedName` OPTIONAL,
`aliasDereferenced` [28] `BOOLEAN` DEFAULT FALSE }

Компонент `SecurityParameters` определен в 7.9. Его отсутствие эквивалентно наличию там пустого набора параметров безопасности.

Различительное имя `Performer` идентифицирует исполнителя конкретной операции. Оно может потребоваться, когда результат должен подписываться (см. 7.10), и должно содержать имя АСС, подписавшего результат.

Компонент `aliasDereferenced` устанавливается в значение «истинно», если предполагаемое имя объекта или базового объекта, являющегося целью операции, включает все разменованные псевдонимы.

7.5 Служебные ограничения

Параметр `ServiceControls` содержит функции управления, с помощью которых, при их наличии, можно направлять или ограничивать предоставляемые услуги.

```

ServiceControls ::= SET {
  options                {0} BIT STRING {
    preferChaining        {0},
    chainingProhibited    {1},
    localScope            {2},
    dontUseCopy           {3},
    dontReferenceAliases  {4},
    subentries            {5},
    copyShallDo           {6} } DEFAULT {5},
  priority               {1} INTEGER {low {0}, medium {1},
                                high {2} } DEFAULT medium,
  timeLimit              {2} INTEGER OPTIONAL,
  sizeLimit              {3} INTEGER OPTIONAL,
  scopeOfReferral        {4} INTEGER {dmd{0}, country{1} } OPTIONAL,
  attributeSizeLimit     {5} INTEGER OPTIONAL }

```

Компонент options содержит ряд указаний, каждое из которых, будучи установлено, утверждает предложенное условие. Таким образом,

- preferChaining указывает, что для обеспечения услуг предпочтительнее использовать сцепление, чем обращение. Справочник не обязан следовать этому правилу предпочтений;
- ChainingProhibited указывает, что сцепление и другие способы распределения запроса по справочнику запрещены;
- LocalScope указывает, что операция должна ограничиваться локальными возможностями. Определение таких возможностей является локальным вопросом, например, в пределах одного АСС или одного региона административного управления справочником (РАУС);
- DontUseCopy указывает, что скопированная информация (согласно ИСО/МЭК 9594-4) не должна использоваться для обеспечения услуг;
- DontReferenceAliases указывает, что любой псевдоним, используемый для идентификации записи, подтвержденной операцией, не должен разменовываться.

Примечание — Это необходимо для того, чтобы избежать ссылки на саму запись псевдонима, а не на обозначенную псевдонимом запись, например, для чтения записи псевдонима;

- subentries указывает, что операциям «поиск» или «список» доступны только подзаписи, нормальные записи оказываются недоступными, т. е. справочник ведет себя так, как если бы нормальных записей не существовало. Если такое служебное ограничение не установлено, то операции доступны только нормальные записи, а подзаписи оказываются недоступными. Это служебное ограничение игнорируется всеми операциями, кроме «поиск» или «список».

Примечания

1 Влияние подзаписей на управление доступом, схема и атрибуты общего пользования все еще сохраняются, даже если подзаписи недоступны.

2 Если такое служебное ограничение установлено, нормальные записи могут по-прежнему определяться в виде базового объекта операции;

- CopyShallDo указывает, что если справочник способен частично, но не полностью удовлетворить запросы на копирование записи, он не должен сцеплять запрос. Это действует только в том случае, если dontUseCopy не установлен. Если copyShallDo не установлен, справочник может использовать теневые данные только в том случае, если они достаточно полные, чтобы операция могла быть удовлетворена копией. Запрос может быть удовлетворен только частично либо потому, что некоторые из требуемых атрибутов отсутствуют в теневой копии, либо потому, что АСС, сохраняющий теневые данные, не поддерживает требуемых правил сравнения на такие данные. Если copyShallDo установлен и справочник не способен полностью удовлетворить запрос, он должен установить incompleteEntry и удаляемой информации записи.

Если этот компонент отсутствует, предполагается следующее: предпочтительность сцеплению не назначается, хотя сцепление не запрещено, ограничения на возможности операции не налагаются, использование копии разрешено, псевдонимы должны быть разменованы (кроме операций модификации, для которых разменовывание псевдонима не обеспечивается), подзаписи недоступны, а операции, не полностью удовлетворяемые теневыми данными, являются объектами дальнейшего сцепления.

Компонент *priority* (низкий, средний или высокий) относится к услуге, которая должна обеспечиваться. Следует заметить, что это не гарантированная услуга справочника в том смысле, что в целом он не организовывает очереди. Не существует каких-либо взаимосвязей, предполагающих использование приоритетов на нижерасположенных уровнях.

TimeLimit указывает в секундах максимальное время, в течение которого должна быть обеспечена услуга. Если ограничения не могут быть обеспечены, сообщается об ошибке. Если этот компонент опущен, никакого временного ограничения не предполагается. В случае превышения временного предела при выполнении операций «список» или «поиск», результатом может быть произвольный набор накопленных результатов.

Примечание — Этот компонент не предполагает больших временных затрат на обработку запроса в течение отведенного времени: в течение этого времени в процесс обработки запроса может быть включено любое число АСС.

SizeLimit применим только при операциях «список» или «поиск». Этот компонент указывает максимальное число объектов, подлежащих выдаче. В случае превышения этого количества, результатом этих операций может быть произвольный набор накопленных результатов, равный ограничению по количеству. Любые последующие результаты должны быть аннулированы.

Компонент *scopeOfReferral* указывает ту область, к которой должно относиться обращение, выданное АСС. В зависимости от выбранного значения — *dmf* или *country* могут быть выданы обращения к другим АСС только в выбранной области. Это относится к обращениям, когда ошибка *Referral* и параметр *unexplored* получены в результатах операций «список» или «поиск».

AttributeSizeLimit указывает наибольший размер любого атрибута (т. е. тип и все его значения), который включен в выдаваемую информацию записи. Если атрибут превышает этот предел, то все его значения исключаются из выдаваемой информации записи, а *IncompleteEntry* устанавливается в ней. Размер атрибута определяется его длиной в октетах в локальном конкретном синтаксисе АСС, содержащем данные. Из-за различных способов хранения данных в прикладных программах это ограничение не является точным. Если этот параметр не определен, то никакой предел не устанавливается.

Примечание — Значения атрибута, выдаваемые как часть различительного имени записей, исключаются от этого ограничения.

Определенные сочетания *priority*, *timeLimit*, и *sizeLimit* могут привести к конфликтам. Например, короткий промежуток времени может войти в противоречие с низким приоритетом; ограничение большого размера может войти в противоречие с коротким промежутком времени, и т. д.

7.6 Выбор информации записи

Параметр *EntryInformationSelection* указывает, какая информация запрошена из записи в получаемой услуге

```
EntryInformationSelection      ::= SET {
    attributes CHOICE {
        allUserAttributes       {0} NULL,
        select                  {1} SET OF AttributeType
        -- пустой набор означает, что атрибуты не требуются --
        DEFAULT allUserAttributes, NULL,
    infoTypes                   {2} INTEGER {
        attributeTypesOnly      {0},
        attributeTypesAndValues {1} DEFAULT attributeTypesAndValues,
    extraAttributes CHOICE {
        allOperationalAttributes {3} NULL,
        select                    {4} SET OF AttributeType | OPTIONAL }
    }
```

Компонент *attributes* определяет пользователи и операционные атрибуты, относительно которых требуется информация

- Если выбран вариант *select*, перечисляются привлекаемые атрибуты. Если список пустой, никакие атрибуты не должны выдаваться. При наличии атрибута должна быть выдана информация о выбранных атрибутах. *AttributeError* с проблемой *noSuchAttributeOrValue* должен выдаваться только в том случае, если в наличии нет ни одного из выбранных атрибутов.

- h) Если выбран факультативный атрибут `allUserAttributes`, запрашивается информация о всех атрибутах пользователя в записи.

Информация атрибута выдается только в том случае, если права доступа достаточны. `SecurityError` (с проблемой `insufficientAccessRights`) должен выдаваться только в случае, если права доступа не допускают чтения всех запрошенных значений атрибутов.

Компонент `infoTypes` определяет, требуется ли информация относительно типа и значения атрибута (по умолчанию) или только о типе атрибута. Если компонент `attributes` не запрашивает никаких атрибутов, он ничего не означает.

Компонент `extraAttributes` определяет набор дополнительных атрибутов пользователя и операционных атрибутов, относительно которых запрошена информация. Если выбрана факультативная возможность `allOperationalAttributes`, то запрашивается информация о всех справочных операционных атрибутах в записи. Если выбрана факультативная возможность `select`, запрашивается информация о перечисленных атрибутах.

П р и м е ч а н и е — Этот компонент может быть использован для запроса информации, например, о конкретных операционных атрибутах, если компонент `attributes` установлен в значение `allUserAttributes`, либо о всех операционных атрибутах. Если один и тот же атрибут содержится или предполагается в списках обоих компонентов `attributes` и `extraAttributes`, он рассматривается как и при однократном запросе.

Запрос конкретного атрибута всегда рассматривается как запрос самого атрибута и всех его подтипов (кроме запросов, обрабатываемых системой версии 1988 г.)

При ответе на запрос информации атрибута справочник обращается со всеми коллективными атрибутами записи как с фактическими атрибутами пользователя записи, т. е. они выбираются подобно другим атрибутам пользователя и включаются в выдаваемую информацию записи. Запрос значения `allUserAttributes` запрашивает все коллективные, а также обычные атрибуты записи. Атрибут является коллективным атрибутом записи, если все перечисленное ниже истинно.

- он расположен в подзаписи, спецификация подзаписи которого включает запись;
- он не исключается наличием в записи значения атрибута `collectiveExclusions`, равного типу коллективного атрибута;
- он разрешен правилом формирования содержимого для структурного класса объекта записи.

7.7 Информация записи

Параметр `EntryInformation` переносит информацию, выбранную из записи

```
EntryInformation = SEQUENCE {
    name                Name,
    fromEntry           BOOLEAN DEFAULT TRUE,
    information         SET OF CHOICE {
        attributeType   Attribute Type,
        attribute       Attribute } OPTIONAL,
    incompleteEntry     [3] BOOLEAN DEFAULT FALSE
    -- система, отличная от версии 1988 г. --
}
```

Параметр `Name` указывает различительное имя записи или имя псевдонима записи. Различительное имя записи выдается всякий раз, когда это разрешено стратегией управления доступом. Если разрешен доступ к атрибутам записи, но не к его различительному имени, справочник может выдать либо ошибку, либо имя действительного псевдонима записи.

П р и м е ч а н и я

1 Если запись была размещена с помощью псевдонима, то этот псевдоним является действительным. В противном случае невозможно гарантировать, что псевдоним действительно вне области распространения таких спецификаций справочника.

2 В тех случаях, когда конкретный компонент справочника выбрал имена псевдонимов, доступные ему для передачи, рекомендуется, чтобы по возможности он выбирал одно и то же имя псевдонима для повторных запросов, выдаваемых одним и тем же запросчиком, с целью для обеспечения непрерывности обслуживания.

Параметр `fromEntry` указывает, откуда получена информация: из записи (ИСТИННО) или из копии записи (ЛОЖНО).

Параметр `information` имеет место, если из записи выдана любая информация атрибута, и содержит набор `attributeTypes` и `attributes`.

Параметр *IncompleteEntry* имеет место и устанавливается в значение «истинно» всякий раз, когда выдаваемая информация записи не полна относительно запроса пользователя, например потому что атрибуты или значения атрибута опущены по причинам управления доступом (и их существование разрешается раскрывать), наличием неполной теневой информации вместе с *copyShallDo* или потому, что предел *attributeSizeLimit* был превышен. Он не устанавливается в значение «истинно», если вместо различительного имени было выдано имя псевдонима.

7.8 Фильтр

7.8.1 Фильтр

Параметр *Filter* означает проверку, которую либо выдерживает или не выдерживает конкретная запись. Фильтр выражается в понятиях утверждений относительно наличия определенных атрибутов записи или их значений и считается положительным только в том случае, если он выдает значение «истинно».

Примечание -- Фильтр может иметь значения «истинно», «ложно», либо «неопределенное».

Filter	==	CHOICE {
item	[0]	FilterItem,
and	[1]	SET OF Filter,
or	[2]	SET OF Filter,
not	[3]	Filter }
FilterItem	::=	CHOICE {
equality	[0]	AttributeValueAssertion,
substrings	[1]	SEQUENCE {
type		AttributeType ({SupportedAttributeTypes}),
strings		SEQUENCE OF CHOICE {
initial	[0]	AttributeValue ({SupportedAttributes}[#type]),
any	[1]	AttributeValue ({SupportedAttributes}[#type]),
final	[2]	AttributeValue ({SupportedAttributes}[#type]) },
greaterOrEqual	[2]	AttributeValueAssertion,
lessOrEqual	[3]	AttributeValueAssertion,
present	[4]	AttributeType,
approximateMatch	[5]	AttributeValueAssertion,
extensibleMatch	[6]	MatchingRuleAssertion }
MatchingRuleAssertion	=	SEQUENCE {
matchingRule	[1]	SET SIZE {1..MAX} OF MATCHING-RULE &ad,
type	[2]	AttributeType OPTIONAL,
matchValue	[3]	MATCHING-RULE &AssertionType (CONSTRAINED BY {
-- <i>matchValue</i> должен представлять собой значение типа, указанное полем <i>&AssertionType</i> одного из объектов информации <i>MATCHING-RULE</i> , идентифицированных правилом <i>matchingRule</i> --),		
dnAttributes	[4]	BOOLEAN DEFAULT FALSE }

Filter может быть представлен *FilterItem* (см. 7.8.2) или выражением, включающим более простые фильтры, объединенные логическими операторами *and*, *or* и *not*.

Если *Filter* представлен *FilterItem*, он принимает значения *FilterItem* (т. е. «истинно», «ложно» или «неопределенное»).

Filter, представляющий логическое *and* (И) набора фильтров, имеет значение «истинно», если набор пустой или если каждый фильтр установлен в значение «истинно»; он имеет значение «ложно», если, по меньшей мере, один фильтр установлен в значение «ложно», в противном случае он имеет значение «неопределенное» (т. е., если, по меньшей мере, один из фильтров имеет значение «неопределенное» и ни один из фильтров не имеет значение «ложно»).

Filter, представляющий логическое *or* (ИЛИ) набора фильтров, принимает значение «ложно», если набор пустой или если каждый фильтр установлен в значение «ложно», он принимает значение «истинно», если, по меньшей мере, один из фильтров установлен в значение «истинно»; в противном случае он имеет значение «неопределенное» (т. е., если, по меньшей мере, один из фильтров имеет значение «неопределенное» и ни один из фильтров не имеет значение «истинно»).

Filter, представляющий логическое по (НЕТ) фильтра, имеет значение «истинно», если фильтр установлен в значение «ложно»; он принимает значение «ложно», если фильтр установлен в значение «истинно»; и значение «неопределенное», если фильтр имеет значение «неопределенное».

7.8.2 Элемент фильтра

FilterItem представляет собой утверждение о наличии или о значении атрибутов в проверяемой записи. Утверждение о конкретном типе атрибута также подтверждается, если запись содержит подтип атрибута и утверждение имеет значение «истинно» для подтипа или если имеется коллективный атрибут записи (см. 7.6), для которого утверждение имеет значение «истинно». Каждое утверждение может иметь значение «истинно», «ложно» или «неопределенное».

Каждый FilterItem включает в себя или предполагает наличие одного или нескольких AttributeTypes, которые идентифицируют конкретные атрибуты.

Любое утверждение о значении такого атрибута определено только в том случае, если с помощью механизма оценки можно узнать AttributeType, если смысловые параметры AttributeValue соответствуют синтаксису атрибута, определенному для данного типа атрибута, если предполагаемое или указанное правило сравнения применимо к такому типу атрибута и если представленный matchValue (при его использовании) соответствует синтаксису, определенному для указанных правил сравнения.

Примечания

1 Если эти условия не удовлетворяются, FilterItem имеет неопределенное значение.

2 Ограничения управления доступом могут повлиять на оценку FilterItem.

Утверждения значений атрибута в элементах фильтра оцениваются путем использования правил сравнения, определенных для данного типа атрибута. Утверждения правил сравнения оцениваются согласно их определениям. Правило сравнения, определенное для конкретного синтаксиса, может быть использовано только для утверждения атрибутов или подтипов данного синтаксиса.

FilterItem может иметь неопределенное значение (как описано выше). В противном случае FilterItem устанавливает:

- equality — имеет значение «истинно» только в том случае, если существует такое значение атрибута или одного из его подтипов, что правило сравнения equality, применимое к этому и представленному значению, выдает значение «истинно»;
- substrings — имеет значение «истинно» только в том случае, если существует такое значение атрибута или одного из его подтипов, что правило сравнения substrings, применимое к этому и представленному значению в strings, выдает значение «истинно». В ГОСТ Р ИСО/МЭК 9594-6 приведено описание семантики представленного значения;
- greaterOrEqual — имеет значение «истинно» только в том случае, если существует такое значение атрибута или одного из его подтипов, что правило сравнения ordering, применимое к этому и представленному значению, выдает значение «ложно». Другими словами, существует значение атрибута, которое больше или равно представленному значению;
- lessOrEqual — имеет значение «истинно» только в том случае, если существует такое значение атрибута или одного из его подтипов, что правило сравнения equality или ordering, применимое к этому и представленному значению, выдает значение «истинно». Другими словами, существует значение атрибута, которое меньше или равно представленному значению;
- present — имеет значение «истинно» только при наличии в записи атрибута или одного из его подтипов;
- approximateMatch — имеет значение «истинно» только в том случае, если существует значение атрибута или одного из его подтипов, для которого локально определяемый аппроксимирующий алгоритм сравнения (например, варианты сравнения по слогам, фонетического сравнения и т.д.) указывает значение «истинно». В данном издании настоящего стандарта не устанавливаются каких-либо конкретных руководящих принципов аппроксимирующего сравнения. Если аппроксимирующее сравнение не обеспечивается, то этот FilterItem должен рассматриваться как сравнение для equality;
- ExtensibleMatch — имеет значение «истинно» только в том случае, если существует значение атрибута указанного типа или одного из его подтипов, для которого применимо правило сравнения, определенное в matchingRule, и представленное значение matchValue указывает «истинно».

Если предлагается несколько правил сравнений, то способ объединения этих правил в новое правило не определяется (это локально определяемый алгоритм, который отражает семантику составных правил сравнения — например, фонетическое сравнение плюс сравнение ключевых слов).

Если `type` отсутствует, то сравниваются все типы атрибута, соответствующие этому правилу сравнения. Если `dnAttributes` имеет значение «истинно», то дополнительно к атрибутам записи в оценке сравнения используются атрибуты различительного имени записи.

Если в фильтре требуется `extensibleMatch` (а не `extendedFilter`), то в параметре `criticalExtensions` в `CommonArguments` должен быть установлен бит `extendedFilter`, указывая, что такое расширение критически важно.

Примечание — `extensibleMatch` не допускается для систем издания 1988 г.

7.9 Страничные результаты

Параметр `PagedResultsRequest` используется АПС для запроса выдачи результатов операций «список» или «поиск» ему в страничном виде: он запрашивает АСС выдать только поднабор — страницу — результатов операции, в частности, следующие подчиненные `pageSize` или записи, а также `queryReference`, который может быть использован для запроса следующего набора результатов после следующего запроса. Он не может использоваться, если результаты должны подписываться, и он не обеспечивается системами издания 1988 г. Хотя АПС может запрашивать `pagedResults`, АСС разрешается игнорировать этот запрос и выдавать свои результаты обычным образом.

```

PagedResultsRequest ::= CHOICE {
  newRequest      SEQUENCE {
    pageSize      INTEGER,
    sortKeys       SEQUENCE OF SortKey OPTIONAL,
    reverse        [1] BOOLEAN DEFAULT FALSE,
    unmerged       [2] BOOLEAN DEFAULT FALSE },
  queryReference  OCTET STRING }

SortKey ::= SEQUENCE {
  type            AttributeType,
  orderingRule    MATCHING-RULE.&id OPTIONAL }

```

Для новых операций «список» или «поиск» `PagedResultsRequest` устанавливается в `newRequest`, который содержит следующие параметры:

- `PageSize` определяет максимальное число подчиненных или записей, выдаваемых в результатах. АСС должен выдавать не более требуемого числа записей. Параметр `SizeLimit` (при его наличии) игнорируется;
- `SortKeys` определяет последовательность типов атрибутов с факультативными правилами упорядоченного сравнения, которые используются в виде ключей сортировки выдаваемых записей по их передаче АПС. Записи сортируются в соответствии со значениями атрибута `type` первого `SortKey` в последовательности, а в случае нескольких записей, имеющих один и тот же вид сортировки — следующего `SortKey` в последовательности, и так далее.

Для конкретного `SortKey` АСС использует правило сравнения `orderingRule`, при его наличии, в противном случае — правило сравнения атрибута, если таковое определено; АСС игнорирует ключ сортировки, если ничего не определено. Если тип атрибута многозначен, используется значение «наименьший»; если тип атрибута отсутствует в выдаваемых результатах, он рассматривается как «наибольший» из всех других сравниваемых значений. АСС разрешается обеспечивать только определенные последовательности ключа сортировки (таким образом, АСС, который хранит и выдает свои данные по внутреннему порядку «фамилии по алфавиту», сможет справиться только с одной последовательностью ключа сортировки). Если он не может обеспечивать требуемую последовательность, он должен использовать последовательность, принятую по умолчанию,

- если параметр `reverse` имеет значение «истинно», АСС будет выдавать результаты сортировки в обратной последовательности (т. е. от «наибольшего» до «наименьшего»; если тип атрибута многозначен, используется «наибольший», если тип атрибута отсутствует в выдаваемых результатах, он рассматривается как «наименьший» из всех других сравниваемых значений). Если он имеет значение «ложно», АСС выдает результаты в возрастающем порядке. Если ни один из параметров `sortKeys` не определен, этот параметр игнорируется;

- d) если параметр `unmerged` имеет значение «истинно» и АСС должен объединить результаты, полученные от многих других АСС, он должен выдать все данные от одного АСС (в порядке сортировки) перед выдачей данных от следующего АСС. Если этот параметр имеет значение «ложно», АСС должен собрать результаты от всех других АСС и отсортировать объединенные данные до их выдачи. Если ни один из параметров `SortKeys` не определен, этот параметр игнорируется.

Для последующего запроса, т. е. для запроса следующего набора страничных результатов, АПС выдает тот же запрос операций «список» или «поиск», что и раньше, но устанавливает `PagedResultsRequest` в значение `queryReference` с тем же значением этого параметра, которое было выдано в параметре `PartialOutcomeQualifier` предыдущих результатов. АПС не воспринимает `queryReference`, который доступен АСС для использования, поскольку он желает записать информацию контекста для запроса. АСС использует эту информацию, чтобы определить, какой из результатов выдать следующим.

Примечания

- 1 Если ИБС изменяется между запросами операции «поиск», АПС могут не обнаружить результатов этих изменений. Это зависит от реализации.
- 2 Указатель запроса может остаться действительным даже в том случае, если АПС имеет новую операцию «список» или «поиск». АПС может запросить постраничные результаты, выдав несколько запросов и затем ответить на предыдущий запрос и запросить следующую страницу результатов, используя обеспеченный для него указатель запроса. Число «активных» указателей запроса, которые может выдать АПС, зависит от реализации локального АСС, а также от срока действия таких указателей запроса.
- 3 Постраничные результаты не обеспечиваются в протоколе системы справочника. Постраничные результаты полностью обеспечиваются АСС, в котором подключен данный АПС.

7.10 Параметры защиты

`SecurityParameters` управляют операцией различных средств защиты относительно операций справочника.

П р и м е ч а н и е. Эти параметры передаются от отправителя к получателю. При наличии параметров в аргументе операции запросчик является отправителем, а исполнитель — получателем. В выдаваемом результате роли меняются.

```
SecurityParameters      := SET {
  certification-path    [0] CertificationPath OPTIONAL,
  name                  [1] DistinguishedName OPTIONAL,
  time                  [2] UTCTime OPTIONAL,
  random                [3] BIT STRING OPTIONAL,
  target                [4] ProtectionRequest OPTIONAL }
ProtectionRequest ::= INTEGER {none(0), signed(1)}
```

Компонент `CertificationPath` состоит из сертификата отправителя и, факультативно, из последовательности пар сертификатов. Сертификат используется для связи ключа общего пользования отправителя и различительного имени, но может быть использован для проверки подписи в аргументе или в результате. Этот параметр должен иметь место, если аргумент или результат подписывается. Последовательность пар сертификатов состоит из пересечений уполномоченных по сертификации с сертификатами. Он используется, чтобы обеспечить проверку правильности сертификата отправителя. Он не требуется, если получатель использует того же уполномоченного по сертификации, что и отправитель. Если получатель запрашивает действительный набор пар сертификатов и этого параметра нет, то вопрос о том, отклоняет ли получатель подпись в аргументе или в результате, или же пытается создать путь сертификации, решается локально.

Параметр `name` — это различительное имя первого назначенного получателя аргумента или результата. Например, если АПС создает подписываемый аргумент, то этот параметр является различительным именем АСС, которому предоставлена операция.

Параметр `time` — это предполагаемое время действия действительности подписи при использовании подписываемых аргументов. Он используется в сочетании со случайным числом для обнаружения повторных угроз защите.

Число `random` должно быть разным для каждого неистекшего полномочия. Оно используется в сочетании с параметром времени для обнаружения повторных угроз, если аргумент или результат подписан.

Параметр `target ProtectionRequest` может иметь место только в запросе на операцию, которую следует выполнить, и указывает предпочтительность запросчика относительно степени защиты, которую должна быть обеспечена для результата. Возможны два уровня обеспечения защиты: `none` (не требуется никакой защиты, по умолчанию) и `signed` (запрашивается справочник для подписания результата). Степень защиты, фактически обеспечиваемая для результата, указывается формой результата и может быть такой же или более низкой, чем запрошенная, в зависимости от ограничений справочника.

7.11 Общие элементы процедуры для управления базовым доступом

В данном разделе определяются элементы процедур, общие для всех операций абстрактных услуг при действии управления базовым доступом.

7.11.1 Размещение псевдонима

Если в процессе размещения целевой записи объекта (идентифицированной в аргументе операции абстрактной услуги) требуется размещение псевдонима, то никаких конкретных разрешений для этого не требуется. Однако, если размещение псевдонима может привести к выдаче `ContinuationReference` (т. е. в `Referral`), следует использовать следующую последовательность управляющих действий доступом, которая может быть применена также к обращению, полученному в ответе от другого АСС. То есть АСС должен рассмотреть все обращения, выработанные как локально, так и удаленно.

1) Разрешение `Read` требуется для записи псевдонима. Если разрешение не предоставляется, операция заканчивается с отрицательным результатом в соответствии с процедурой, описанной в 7.11.3.

2) Разрешение `Read` требуется для атрибута `AliasedObjectName` и единственного значения, которое он содержит. Если разрешение не предоставляется, операция заканчивается с отрицательным результатом и выдается ошибка `NameError` с проблемой `aliasDereferencingProblem`. Элемент `matched` должен содержать имя записи псевдонима.

Примечание — Кроме описанных выше управлений доступом стратегия защиты может предотвратить раскрытие информации о сведениях, которая в противном случае могла передаваться в виде `ContinuationReference` и `Referral`. Если такая стратегия действует и если АПС ограничивает услуги, определяя `ChainingProhibited`, справочник может выдать `ServiceError` с проблемой `chainingRequired`. В противном случае должна быть выдана `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`.

7.11.2 Выдача `NameError`

Если при выполнении операции абстрактной услуги указанный целевой объект (псевдоним или запись), например имя записи, подлежащее чтению или `baseObject` и `Search`, не может быть найден, должна быть выдана ошибка `NameError` с проблемой `noSuchObject`. Элемент `matched` должен содержать либо имя следующей старшей записи, для которой уже имеется разрешение `DiscloseOnError`, либо имя корня дерева информации справочника (ДИС) (т. е. пустой `RDNSequence`).

Примечание — АСС, не имеющий доступа ко всем старшим записям, может осуществить другой выбор.

7.11.3 Нераскрытость существования записи

Если при выполнении операции абстрактной услуги, необходимое разрешение уровня записи не предоставлено для указанной целевой записи объекта, например для записи, подлежащей чтению, операция заканчивается с отрицательным результатом и выдается одна из следующих ошибок: если разрешение `DiscloseOnError` предоставлено целевой записи — `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`, в противном случае `NameError` с проблемой `noSuchObject`. Элемент `matched` должен содержать либо имя следующей старшей записи, для которой предоставлено `DiscloseOnError`, либо имя корня ДИС (т. е. пустой `RDNSequence`).

Примечание — АСС, не имеющий доступа ко всем старшим записям, может осуществить другой выбор.

Кроме того, всякий раз, когда справочник обнаруживает операционную ошибку (включая `Referral`), он должен гарантировать, что, выдавая такую ошибку, он не нарушает существования упомянутой целевой записи и любой из двух старших записей. Например, прежде чем выдать `ServiceError` с проблемой `timeLimitExceeded` или `UpdateError` с проблемой `notAllowedOnNonLeaf`, справочник проверяет наличие для целевой записи разрешения `discloseOnError`. Если разрешение не предоставлено, должна быть привлечена процедура, описанная в предыдущем абзаце.

7.11.4 Выход различительного имени

При выполнении операции «сравнение», «список» или «поиск» требуется разрешение ReturnDN для записи object (или baseObject), если в результате разменовывания псевдонима должно быть выдано различительное имя объекта в параметре name результата операции (см. 9.2.3). Если такое разрешение не предоставлено, справочник вместо этого должен выдать имя псевдонима, как описано в 7.7, или опустить параметр «имя».

При выполнении операции «чтение» или «поиск» требуется разрешение ReturnDN для записи, чтобы выдать различительное имя в EntryInformation. Если такое разрешение не предоставлено, справочник должен выдать имя псевдонима вместо различительного имени, как описано в 7.7, либо, если никакое имя псевдонима недоступно, операция заканчивается с отрицательным результатом вместе с NameError (в случае операции «чтение») или из результатов записи исключается (в случае операции «поиск»).

Если в результате выдается обеспечиваемое пользователем имя псевдонима, флаг aliasDeferreded в CommonResults не должен устанавливаться в значение «истинно».

7.12 Факультативно подписываемые параметры

Тип информации OPTIONALLY-SIGNED — это такой тип, значения которого по выбору создателя могут сопровождаться их цифровой подписью. Эта возможность определяется с помощью следующего типа:

```
OPTIONALY-SIGNED {type} = CHOICE {
    unsigned          Type,
    signed             SIGNED {type}}
```

Тип SIGNED, который описывает форму подписываемого формата информации, определен в ГОСТ Р ИСО/МЭК 9594-3.

8 ОПЕРАЦИИ «СВЯЗКА» И «РАЗВЯЗКА»

Операции DirectoryBind и DirectoryUnbind, определенные в 8.1 и 8.2 соответственно, используются АПС в начале и конце конкретного периода обращения к справочнику.

8.1 Операция «связка» справочника

8.1.1 Синтаксис операции «связка»

Операция DirectoryBind используется в начале периода обращения к справочнику.

```
directoryBind OPERATION ::= {
    ARGUMENT    directoryBindArgument
    RESULT      directoryBindResult
    ERROR       directoryBindError }

DirectoryBindArgument ::= SET {
    credentials [0] Credentials OPTIONAL,
    versions     [1] Versions DEFAULT {v1}}

Credentials ::= CHOICE {
    simple [0] SimpleCredentials,
    strong [1] StrongCredentials,
    externalProcedure [2] EXTERNAL }

SimpleCredentials ::= SEQUENCE {
    validity [1] SET {
        time1 [0] UTCTime OPTIONAL,
        time2 [1] UTCTime OPTIONAL,
        random1 [2] BIT STRING OPTIONAL,
        random2 [3] BIT STRING OPTIONAL } OPTIONAL,
    password [2] CHOICE {
        unprotected OCTET STRING,
        StrongCredentials ::= SET {
            certification-path [0] CertificationPath OPTIONAL,
            bind-token [1] Token,
            name [2] DistinguishedName OPTIONAL }
    Token ::= SIGNED { SEQUENCE {
        algorithm [0] AlgorithmIdentifier,
```

```

name                [1] DistinguishedName,
time                [2] UTCTime,
random              [3] BIT STRING }}
Versions . . . = BIT STRING {v1(0)}
DirectoryBindResult ::= DirectoryBindArgument
directoryBindError ERROR ::= {
    PARAMETER SET {
        versions      [0] Versions DEFAULT {v1},
        error          CHOICE {
            serviceError [1] ServiceProblem,
            securityError [2] SecurityProblem }}}

```

8.1.2 Аргументы операции «связка» справочника

Аргумент *credentials* в *DirectoryBindArgument* позволяет справочнику устанавливать идентичность пользователя. Удостоверения личности могут быть *simple*, *strong* или внешне определяемыми (*externalProcedure*), как описано в ГОСТ Р ИСО/МЭК 9594-8.

При использовании параметра *simple* он содержит *name* (всегда различительное имя объекта) и, факультативно, параметры *validity* и *password*. Он обеспечивает ограниченную степень защиты. Параметр *password* может быть *unprotected* или *protected* (либо *Защита1*, либо *Защита2*), как описано в разделе 5 ГОСТ Р ИСО/МЭК 9594-8. Параметр *validity* обеспечивает аргументы *time1*, *time2*, *random1* и *random2*, которые принимают свои значения в соответствии с двусторонним соглашением и могут быть использованы для обнаружения повторов. В некоторых случаях защищенный пароль может быть проверен объектом, который узнает пароль только после локальной регенерации защиты своей собственной копии пароля и сравнения результата со значением в аргументе «связка» (*password*). В других случаях возможно непосредственное сравнение.

При использовании параметра *strong* он содержит *bind-token* и, факультативно, *certificate-path* (сертификат и последовательность пересечений сертификатов и уполномоченных по сертификации, как определено в ГОСТ Р ИСО/МЭК 9594-8) и *name* запросчика. Это позволяет справочнику подтвердить идентичность запросчика, устанавливающей ассоциацию, и наоборот.

Аргументы полномочий связи используются следующим образом. *algorithm* — это идентификатор алгоритма, применяемого для подписания такой информации; *name* — это имя предполагаемого получателя. Параметр *time* содержит время существования полномочий. Число *random* должно быть различным для каждого полномочия, и может использоваться получателем для обнаружения повторов.

При использовании *externalProcedure* семантика, используемая схемой аутентификации, не входит в предмет рассмотрения настоящего стандарта.

Аргумент *version* в *DirectoryBindArgument* определяет версии услуг, в предоставлении которых АПС готов участвовать. Для настоящей версии протокола его значение должно быть равно *v1(0)*. Переход к будущим версиям справочника должен быть упрощен следующим:

- любые элементы *DirectoryBindArgument*, отличные от определенных в настоящем стандарте, должны быть приняты и проигнорированы;
- дополнительные факультативные функции для именованных битов не определенного *DirectoryBindArgument* (например, версии) должны быть приняты и проигнорированы.

8.1.3 Результаты операции «связка»

При успешном выполнении запроса операции «связка» должен быть выдан положительный результат.

Аргумент *credentials* в *DirectoryBindResult* позволяет пользователю устанавливать идентичность справочника. Он позволяет передавать АПС информацию, идентифицирующую АСС (который непосредственно обеспечивает услуги справочника). Эта информация должна иметь тот же формат (т. е. *CHOICE*), который обеспечивается пользователем.

Параметр *versions* в *DirectoryBindResult* указывает, какие из версий услуг, запрошенных АПС, фактически готов обеспечить АСС.

8.1.4 Ошибки при выполнении операции «связка»

При безуспешном выполнении запроса операции «связка» должна быть выдана ошибка.

Параметр *versions* в *DirectoryBindError* указывает, какая версия обеспечивается АСС.

SecurityError или serviceError должны быть представлены следующим образом

```
securityError    inappropriateAuthentication
                invalidCredentials
serviceError     unavailable
```

8.2 Операция «развязка»

Операция DirectoryUnbind используется в конце периода обращения к справочнику.

directoryUnbind OPERATION := emptyUnbind

DirectoryUnbind не имеет аргументов.

9 ОПЕРАЦИИ СПРАВОЧНИКА ТИПА «ЧТЕНИЕ»

Существует две операции типа «чтение»: read(чтение) и compare(сравнение), определенные в 9.1 и 9.2 соответственно. Операция «отклонение», определенная в 9.3, представлена вместе с этими операциями для полноты.

9.1 Чтение

9.1.1 Синтаксис операции «чтения»

Операция read используется для получения информации из явно идентифицированной записи. Она может быть использована также для проверки правильности различительного имени. Факультативно аргументы операции могут быть подписаны запросчиком (см. 7.10). При необходимости справочник может обеспечить подпись результата.

```
read OPERATION ::= {
  ARGUMENT      ReadArgument
  RESULT        ReadResult
  ERRORS         { attributeError | nameError | serviceError |
                  referral | abandoned | securityError }
  CODE          id-opcode-read }
ReadArgument    ::= OPTIONALLY-SIGNED { SET {
  object         [0]    Name,
  selection      [1]    EntryInformationSelection DEFAULT {},
  modifyRightsRequest
                    [2]    BOOLEAN DEFAULT FALSE,
  COMPONENTS OF CommonArguments {} }
ReadResult      ::= OPTIONALLY-SIGNED { SET {
  entry          [0]    EntryInformation,
  modifyRights   [1]    ModifyRights OPTIONAL,
  COMPONENTS OF CommonResults {} }
ModifyRights    ::= SET OF SEQUENCE {
  item           CHOICE {
    entry         [0]    NULL,
    attribute     [1]    Attribute Type,
    value         [2]    AttributeValueAssertion },
  permission     [3]    BIT STRING {add (0), remove(1), rename (2), move(3) } }
```

9.1.2 Аргументы операции «чтения»

Аргумент object идентифицирует запись объекта, из которой запрашивается информация. Если имя имеет один или несколько псевдонимов, они размениваются (если это только не запрещено соответствующими служебными ограничениями).

Аргумент selection указывает, какая информация запрашивается из записи (см. 7.6). Однако не следует полагать, что выдаваемые атрибуты те же, что и запрашиваемые или ограничиваемые ими.

CommonArguments (см. 7.3) содержит спецификацию служебных ограничений, относящихся к запросу. К данной операции компонент sizeLimit не имеет отношения и при его наличии он игнорируется.

Аргумент ModifyRightsRequest используется для запроса выдать права запросчика на модификацию записи и ее атрибутов.

9.1.3 Результаты операции «чтения»

При успешном выполнении запроса операции «чтение» должен быть выдан положительный результат.

Параметр результата entry содержит запрашиваемую информацию (см. 7.7)

Параметр `ModifyRightsRequest` имеет место, если он был запрошен посредством аргумента `modifyRightsRequest` и пользователь имеет привилегии на модификацию некоторой части или всей запрошенной информации записи, а выдача этой информации допускается локальной стратегией защиты. При выдаче результата права запросчика на модификацию передаются для данной записи и атрибутов, указанных в аргументе `selection`. Этот параметр содержит следующее.

- Элемент `SET` выдается для каждого параметра `entry`, для каждого запрашиваемого атрибута, пользователь которого имеет право добавлять или удалять, и для каждого выдаваемого значения атрибутов, указанием которого права пользователей добавлять или удалять отличаются от соответствующих прав относительно самого атрибута.
- Выдаваемые `permission` указывает, какие операции или действия пользователя над записью будут успешно выполнены. В случае записи `remove` указывает, что успешной может быть операция `RemoveEntry`; `rename` указывает, что успешной может быть операция `ModifyDN` при отсутствии параметра `newSuperior`; и `move` указывает, что успешной может быть операция `ModifyDN` при наличии параметра `newSuperior` и при неизменном `OPI`.

В случае атрибутов и их значений `add` указывают, что успешной может быть операция `ModifyEntry`, которая добавляет атрибут или значение, и `remove` указывает, что успешной может быть операция `ModifyEntry`, которая удаляет атрибут или значение.

Примечание — Операция перемещения записи к новой старшей может зависеть также от разрешений, связанных с этой старшей (как, например, при базовом управлении доступом). Они игнорируются при определении `permission`.

9.1.4 Ошибки операции «чтение»

При безуспешном выполнении запроса операции «чтение» должна быть выдана одна из указанных ниже ошибок. Если ни один из атрибутов, явно указанных в `selection`, не может быть выдан, тогда должен быть передан параметр `AttributeError` с проблемой `noSuchAttributeOrValue`. Причины, по которым могут быть выданы другие ошибки, определены в разделе 12.

9.1.5 Принятые решения при выполнении операции «чтение» и использовании базового управления доступом

Если базовое управление доступом действует при чтении записи, то применяется следующая последовательность управлений доступом.

- 1) Для читаемой записи требуется разрешение `Read`. Если разрешение не предоставлено, операция заканчивается с отрицательным результатом в соответствии с 7.11.3.
- 2) Если элемент `infoTypes` в `selection` определяет, что должны быть выданы только типы атрибутов, то разрешение `Read` требуется для каждого подлежащего выдаче типа атрибута. Если разрешение не предоставлено, соответствующий тип атрибута не учитывается в `ReadResult`. Если в результате применения этих управлений никакая информация атрибута не выдается, вся операция оказывается безуспешной в соответствии с 9.1.5.1.
- 3) Если элемент `infoTypes` операции `selection` определяет, что должны быть выданы типы атрибутов и их значения, то разрешение `Read` требуется для каждого подлежащего выдаче типа атрибута и для каждого значения. Если не предоставлено разрешение на тип атрибута, соответствующий атрибут в `ReadResult` не учитывается. Если не предоставлено разрешение на значение атрибута, это значение исключается из соответствующего атрибута. Если разрешение не предоставлено ни на одно из значений атрибута, выдается элемент `Attribute`, содержащий пустой `SET OF AttributeValue`. Если в результате применения этих управлений никакая информация атрибута не выдается, вся операция заканчивается безуспешно в соответствии с 9.1.5.1.

9.1.5.1 Выдаваемые ошибки

Если операция заканчивается безуспешно, как определено в перечислениях 2 и 3 пункта 9.1.5, может быть выдана одна из следующих ошибок:

- a) если была выбрана факультативная возможность с открытым окончанием (т.е. `allUserAttributes` или `allOperationalAttributes`), должна быть выдана ошибка `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`,
- b) в противном случае, т.е. при определении выбора `select {n attributes и/или в extraAttributes}`, если разрешение `DiscloseOp` предоставлено любому из выбранных атрибутов, должна быть выдана ошибка `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`, а иначе должна выдаваться та же ошибка `AttributeError` с проблемой `noSuchAttributeOrValue`.

9.1.5.2 *Нераскрываемость мелких результатов*

При выдаче неполного результата в EntryInformation, т. е. если некоторые из атрибутов или значений атрибута будут опущены из-за используемых управлений доступом, элемент isCompareEntry должен быть установлен в значение «истинно», если разрешение DisclosureOnError предопределено, по меньшей мере, одному типу атрибута, который не указывается в результате, или, по меньшей мере, одному значению атрибута, которое не указывается в результате (для такого типа атрибута разрешение «чтение» было предоставлено).

9.2 Сравнение

9.2.1 Синтаксис операции «сравнение»

Операция compare используется для сравнения значения (которое выдается в виде аргумента запроса) со значением(ями) конкретного типа атрибута в конкретной записи объекта. Аргументы операции факультативно могут быть подписаны запросчиком (см. 7.10). При необходимости справочник может обеспечить подпись результата

```
compare OPERATION ::= {
  ARGUMENT          CompareArgument
  RESULT             CompareResult
  ERRORS             { attributeError : nameError : serviceError
                      [ referral | abandoned | securityError ]
  CODE               id-opcode-compare }

CompareArgument ::= OPTIONALY-SIGNED { SET {
  object           [0] Name
  purported        [1] AttributeValueAssertion
  COMMONS OF
CompareResult     OPTIONALY-SIGNED { SET {
  name             Name OPTIONAL
  matched          [0] BOOLEAN,
  fromEntry        [1] BOOLEAN DEFAULT TRUE,
  matchedSubtype   [2] AttributeType OPTIONAL,
  COMMONS OF
                  CommonResults } }
```

9.2.2 Аргументы операции «сравнение»

Аргумент object — это имя конкретной записи объекта. Если Name использует один или несколько псевдонимов, они размыкаются (если это не запрещено соответствующими служебными ограничениями).

Аргумент purported определяет тип и значение атрибута, которые должны сравниваться с содержащимися в записи. Результат сравнения устанавливается в значение «истинно», если запись содержит предполагаемый тип атрибута или один из его подтипов, или если существует коллективный атрибут записи, который является предполагаемым типом атрибута или одним из подтипов (см. 7.6), и если имеется значение этого атрибута, которое сравнивается с предполагаемым значением путем использования правила сравнения equality данного атрибута.

CommonArguments (см. 7.3) определяет служебные ограничения, используемые в запросе. Для целей этой операции компонент sizeLimit не уместен, а при его наличии он игнорируется.

9.2.3 Результаты операции «сравнение»

При успешном выполнении запроса операция «сравнение» (т. е. сравнение фактически выполнено) должен быть выдан результат.

Параметр name — это различительное имя записи или имя псевдонима записи, как указано в 7.7. Он имеет место только в том случае, если псевдоним был размыкаем, а имя, подлежащее выдаче, отличается от имени object, обеспечиваемого в аргументе операции.

Параметр результата matched содержит результат сравнения. Он принимает значение «истинно», если значения совпадают, и «ложно» в противном случае.

Если fromEntry установлено в значение «истинно», информация сравнивалась относительно записи; если «ложно» — информация сравнивалась относительно копии.

Параметр MatchedSubtype имеет место только в том случае, если результат сравнения имеет значение «истинно» и если сравнение было успешным, т. е. подтип предполагаемого атрибута совпал со сравниваемым значением. Он содержит совпадающий подтип. При наличии нескольких таких подтипов выдается самый высокий по иерархии.

9.2.4 Ошибки операции «сравнение»

При успешном выполнении запроса операции «сравнение» должна быть выдана хотя бы одна из перечисленных ошибок. Условия выдачи конкретных ошибок определены в разделе 12.

9.2.5 Принятие решений при выполнении операции «сравнение» и использовании базового управления доступом

Если базовое управление доступом действует при сравнении записи, используется следующая последовательность управлений доступом.

- 1) Для сравниваемой записи требуется разрешение Read. Если разрешение не предоставлено, операция заканчивается с отрицательным результатом в соответствии с 7.11.3.
- 2) Для сравниваемых атрибутов требуется разрешение Compare. Если разрешение не предоставлено, операция заканчивается с отрицательным результатом в соответствии с 9.2.5.1
- 3) Если для сравниваемого атрибута имеется значение, которое соответствует аргументу reported и для которого предоставлено разрешение Compare, операция выдает в параметре matched результата CompareResult значение «истинно». В противном случае выдается значение «ложно».

9.2.5.1 Выдаваемые ошибки

Если операция заканчивается безуспешно, как определено в перечислении 2 пункта 9.2.5, может быть выдана одна из следующих ошибок: если разрешение DiscloseOnError предоставлено сравниваемому атрибуту, должна быть выдана SecurityError с проблемой insufficientAccessRights или noInformation, в противном случае должна быть выдана AttributeError с проблемой noSuchAttributeOrValue.

9.3 Отклонение

Операции, запрашивающие справочник, могут быть отклонены с помощью операции Abandon, если пользователь больше не заинтересован в результате.

```

abandon OPERATION      := {
  ARGUMENT              AbandonArgument
  RESULT                AbandonResult
  ERRORS                {abandonFailed}
  CODE                  id-opcode-abandon |
  AbandonArgument      := SEQUENCE |
  invokeID              [0] InvokeID |
  AbandonResult         = NULL

```

Существует единственный аргумент invokeID, определяющий операцию, которая должна быть отклонена. Значение invokeID — это то же самое значение, которое использовалось для вызова подлежащей отклонению операции.

При успешном выполнении запроса должен быть выдан положительный результат, хотя в нем не должна передаваться информация. Первоначальная операция должна закончиться безуспешно с указанием ошибки Abandoned.

Если запрос заканчивается безуспешно, должна быть выдана ошибка AbandonFailed. Как локальное решение АСС может предпочесть не отклонять операцию и выдать ошибку AbandonFailed. Эта ошибка описана в 12.3.

Отклонение применимо только к запрашивающим операциям, т. е. «чтение», «сравнение», «список» и «поиск».

АСС может отклонить операцию локально. Если АСС сдает операцию или распространяет ее на другие АСС, то он, в свою очередь, может запросить их отклонить операцию.

10 ОПЕРАЦИЯ СПРАВОЧНИКА ТИПА «ПОИСК»

Существуют две операции типа «поиск»: list(список) и match(поиск), определенные в 10.1 и 10.2 соответственно.

10.1 Список

10.1.1 Синтаксис операции «список»

Операция list используется для получения списка непосредственных подчиненных явно идентифицированной записи. В некоторых ситуациях выдаваемый список может оказаться неполным. Аргументы операции могут быть факультативно подписаны запросчиком (см. 7.10). При необходимости справочник может обеспечить подпись результата.

```

List OPERATION      := {
    ARGUMENT         ListArgument
    RESULT            ListResult
    ERRORS            [ nameError | serviceError | referral
    CODE              id-opcode-list ]
ListArgument        := OPTIONAL-SIGNED | SET {
    object            [0] Name,
    pagedResults      [1] PagedResultsRequest OPTIONAL,
    COMPONENTS OF    CommonArguments {}
ListResult           := OPTIONAL-SIGNED | CHOICE {
    listInfo          SET {
        name          Name OPTIONAL,
        subordinates  [1] SET OF SEQUENCE {
            rdn        RelativeDistinguishedName,
            aliasEntry  [0] BOOLEAN DEFAULT FALSE,
            fromEntry   [1] BOOLEAN DEFAULT TRUE },
            partialOutcomeQualifier [2] PartialOutcomeQualifier OPTIONAL,
        COMPONENTS OF CommonResults },
    uncorrelatedListInfo [0] SET OF ListResult {}
PartialOutcomeQualifier := SET {
    limitProblem        [0] LimitProblem OPTIONAL,
    unexplored          [1] SET OF ContinuationReference OPTIONAL,
    unavailableCriticalExtensions [2] BOOLEAN DEFAULT FALSE,
    unknownErrors       [3] SET OF ANY OPTIONAL,
    queryReference      [4] OCTET STRING OPTIONAL }
LimitProblem          := INTEGER {
    timeLimitExceeded (0), sizeLimitExceeded (1), administrativeLimitExceeded (2) }

```

10.1.2 Аргументы операции «список»

Аргумент **object** идентифицирует ту запись объекта (или, возможно, корня), чьи непосредственно подчиненные записи должны быть внесены в список. Если **Name** привлекает несколько псевдонимов, они разменовываются (если это не запрещено соответствующими служебными от-
ражениями).

Аргумент **PagedResults** используется при запросе постраничной выдачи результатов операции, как описано в 7.9.

10.1.3 Результаты операции «список»

Запрос будет успешным, если объект найден, независимо от наличия какой-либо подчиненной информации для выдачи.

Параметр **name** — это различительное имя записи или имя псевдонима записи, как описано в 7.7. Он имеет место только в том случае, если псевдоним был разменован, а подлежащее выдаче имя отличается от имени **baseObject**, обеспечиваемого в аргументе операции.

Параметр **subordinates** содержит информацию относительно непосредственно подчиненных (при их наличии) поименованной записи. Если любая подчиненная запись является псевдонимом, она не должна разменовываться.

Параметр **rdn** — это относительное различительное имя подчиненной записи

Параметр **fromEntry** указывает, откуда была получена информация: при значении «истинно» — из записи, а при значении «ложно» — из ее копии.

Параметр **aliasEntry** в значении «истинно», указывает, что подчиненная запись является записью псевдонима, а в значении «ложно» — что она не является таковой.

Параметр **partialOutcomeQualifier** состоит из пяти описываемых ниже подкомпонентов. Этот параметр должен иметь место всякий раз, когда результат неполный либо из-за того, что вследствие

ограничений времени, размера или проблем административного управления регионами ДИС не исследованы, либо из-за недоступности некоторых критических расширений, либо из-за получения неизвестной ошибки либо из-за того, что выданы постраничные результаты.

- Параметр `LimitProblem` указывает, какое из ограничений превышено: временное, размерное, или административное управление. Выдаются те результаты, которые были получены при достижении пределов.
- Параметр `unexplored` должен иметь место, если регионы ДИС не исследовались. Его информация позволяет АПС продолжать обработку операции «список», взаимодействуя при необходимости с другими пунктами доступа. Этот параметр состоит из набора (возможно пустого) указателей `ContinuationReferences`, содержащих имя базового объекта, из которого может быть развита далее операция, соответствующее значение `OperationProgress` и набор пунктов доступа, из которых можно далее распространить запрос. Выдаваемые указатели `ContinuationReferences` должны находиться в области распространения обращения, запрошенного в служебном ограничении операции.
- Параметр `UnavailableCriticalExtensions`, при его наличии, указывает, что одно или несколько критических расширений оказались недоступны в некоторой части справочника.
- Параметр `UnknownErrors` используется для выдачи неизвестных типов ошибок или параметров, полученных от других АСС при выполнении операции. Каждый член набора `SET` содержит одну такую неизвестную ошибку (см. 7.5.2.4 в ГОСТ Р ИСО/МЭК 9594-5).
- Параметр `QueryReference` должен иметь место, если АПС запросил постраничные результаты, а АСС не выдал всех доступных результатов (см. 7.9).

Если АПС запросил защищенный запрос подписанного, то параметр `uncorrelatedListInfo` может включать в себя многие наборы параметров результата, полученных из различных компонентов справочника и подписанных имен. Если ни один из АСС в цепочке не может установить соотношение между всеми результатами, то АПС должен скомпоновать фактический результат из различных частей.

10.1.4 Ошибки операции «список»

Если запрос на выполнение операции «список» оказался успешным, должна быть выдана хотя бы одна из перечисленных ошибок. Ситуации, в которых должны выдаваться конкретные ошибки, описаны в разделе 12.

10.1.5 Принятие решений при выполнении операции «список» в использовании базового управления доступом

Если базовое управление доступом действует относительно части операции «список», то применяется следующая последовательность управлений доступом.

- Для записи, идентифицированной аргументом `object`, не требуется никакого конкретного разрешения.
- Для каждой непосредственной подчиненной записи, для которой должно быть выдано `RelativeDistinguishedName` и `subordinates`, требуются разрешения `Browse` и `ReturnOn`. Записи, для которых эти разрешения не предоставляются, игнорируются. Если в результате принятия таких управлений никакая подчиненная информация (кроме любой `ContinuationReferences` в `PartialOutcomeQualifier`) не выдается и если разрешение `DiscloseOnError` не предоставлено этой записи, идентифицированной аргументом `object`, то операция заканчивается безуспешно с выдачей параметра `NameError` с проблемой `noSuchObject`. Элемент `matched` должен либо содержать имя следующей старшей записи, для которой предоставлено разрешение `DiscloseOnError`, либо имя корня ДИС (т.е. пустой `RDNSequence`). В противном случае операция продолжается, но никакая подчиненная информация (кроме любого `ContinuationReferences` в `PartialOutcomeQualifier`) с ней не передается.

Примечание — В случае выдачи `NameError` пустой `RDNSequence` может быть использован АСС, не имеющий доступа ко всем старшим записям.

Примечания

1 Стратегия защиты может предотвратить раскрытие подчиненной информации, которая в противном случае была бы передана как `ContinuationReferences` в `PartialOutcomeQualifier`. Если такая стратегия действует и если АПС ограничивает услуги, определяя `chainingProhibited`, справочник может выдать `ServiceError` с проблемой `chainingRequired`. В противном случае выполняется процедура, описанная выше в перечислении 2.

2 Стратегия защиты может предостеречь от ошибочного указания того, что перечисленная подчиненная запись является записью псевдонима. Например, если для АПС не предоставлен доступ к чтению записи псевдонима, его атрибута ObjectClass и значения alias, которые она содержат, то справочник может опустить компонент aliasEntry в subordinates из ListResult или установить его в значение «ложно».

3 Если разрешение DisallowOnError не предоставлено записи, идентифицированной аргументом object, то PartialOutcomeQualifier, указывающий limitProblem или unavailableCriticalExtensions, не должен вызываться, поскольку он может нарушить защиту этой записи.

10.2 Поиск

10.2.1 Синтаксис операции «поиск»

Операция search используется для осуществления поиска части ДИС для соответствующих записей и выдачи выбранной информации из этих записей. Аргументы операции могут быть факультативно подписаны запросчиком (см. 7.10). При необходимости справочник может обеспечить подпись результата

```

search OPERATION      ::= [
    ARGUMENT          SearchArgument
  RESULT             SearchResult
  ERRORS              [ attributeError | nameError | serviceError
                       | referral | abandoned | securityError ]
  CODE                id-opcode-search ]
SearchArgument ::= OPTIONAL-SIGNED { SET {
  baseObject          [0] Name,
  subset              [1] INTEGER {
                       baseObject(0), oneLevel(1), wholeSubtree(2)} DEFAULT baseObject,
  filter              [2] Filter DEFAULT and: {},
  searchAliases       [3] BOOLEAN DEFAULT TRUE,
  selection           [4] EntryInformationSelection DEFAULT {},
  pagedResults        [5] PagedResultsRequest OPTIONAL,
  matchedValuesOnly   [6] BOOLEAN DEFAULT FALSE,
  extendedFilter      [7] Filter OPTIONAL,
  COMPONENTS OF      CommonArguments }
SearchResult ::= OPTIONAL-SIGNED { CHOICE {
  searchInfo          SET {
    name              Name OPTIONAL,
    entries            [0] SET OF EntryInformation,
    partialOutcomeQualifier [2] PartialOutcomeQualifier OPTIONAL,
    COMPONENTS OF    CommonResults }
  uncorrelatedSearchInfo [0] SET OF SearchResult }

```

10.2.2 Аргументы операции «поиск»

Аргумент BaseObject определяет запись объекта (или, возможно, корень), относительно которого выполняется поиск.

Аргумент subset указывает, к чему должен относиться поиск:

- только к BaseObject;
- только к непосредственным подчиненным базового объекта (oneLevel);
- к базовому объекту и ко всем его подчиненным (wholeSubtree).

Аргумент filter используется для исключения тех записей из области поиска записей, которые не представляют интереса. Информация должна выдаваться только относительно тех записей, которые удовлетворяют фильтру (см. 7.8.)

П р и м е ч а н и е — Если фильтр определен с завышенной оценкой, он может устранить все записи из результата поиска, даже если существуют записи кандидаты, соответствующие частям фильтра. Пользователь должен упростить фильтр и повторить попытку. Справочник не обеспечивает поддержки для идентификации этих записей и для тех изменений, которые должны быть внесены в фильтр.

Псевдонимы должны быть разыменованы при определении местоположения базового объекта в зависимости от установки параметра служебного ограничения `dontDereferenceAliases`. Псевдонимы подчиненных базового объекта должны разыменовываться во время поиска в зависимости от установки параметра `searchAliases`. Если параметр `searchAliases` установлен в значение «истинно», псевдонимы должны быть разыменованы. Если параметр установлен в значение «ложно», псевдонимы не должны разыменовываться. Если параметр `searchAliases` установлен в значение «истинно», в поддереже записи псевдонимов должен продолжаться поиск.

Аргумент `selection` указывает, какая информация запрошена из записей (см. 7.6). Однако не следует считать, что выдаваемые атрибуты те же, что и запрошенные или ограниченные запросом. Аргумент `PagedResults` используется, чтобы запросить, какие из результатов операции выданы постранично, как описано в 7.9.

Аргумент `MatchedValuesOnly` указывает, что некоторые значения атрибута должны быть опущены из выдаваемой информации записи. В частности, если подлежащий выдаче атрибут многозначен и некоторые, но не все его значения, внесенные в фильтр поиска, выдают значение «истинно» через элементы фильтра, отличные от `equality` или `present`, то остальные значения также опускаются из выдаваемой информации записи.

Аргумент `ExtendedFilter` используется в средах смешанных версий для определения фильтра, альтернативного описанному выше. При наличии этого аргумента, аргумент `filter` (если таковой имеется) должен игнорироваться системами издания 1993 г. `ExtendedFilter` всегда игнорируется системами издания 1988 г.

Примечание — Путем включения обоих фильтров АПС может указать один фильтр для использования системами издания 1988 г., а другой — системами 1993 г. в распределенной обработке запроса поиска. Системы издания 1988 г. не обеспечивают атрибут `polyatomicity` и не дают оценок правилам сравнения.

10.2.3 Результаты операции «поиск»

Запрос будет успешным, если будет определено местоположение объекта `baseObject` независимо от наличия каких-либо подчиненных объектов для выдачи.

Примечание — Как следствие этого, результат нефильтрованного поиска, примененного к единственной записи, может быть не идентичен результату операции «чтение», для которой при поиске используется тот же самый набор атрибутов записи. Это объясняется тем, что последний должен выдать ошибку `attributeError`, если ни одного из выбранных атрибутов в записи нет.

Параметр `name` — это различительное имя записи или имя псевдонима записи, как описано в 7.7. Он указывается только в том случае, если псевдоним был разыменован и имя, которое должно быть выдано, отличается от имени `baseObject`, обеспечиваемого в аргументе операции.

Параметр `entries` содержит запрашиваемую информацию из каждой записи (от нуля до нескольких), которая удовлетворяет фильтру (см. 7.5).

Параметр `partialOutcomeQualifier` описан в 10.1.3.

Примечание — Если выдаваемая информация записи некомплектна для конкретной записи, это указывается через параметр `incompleteEntry` в выдаваемой информации записи.

Параметр `UncorrelatedSearchInfo` аналогичен описанному в 10.1.3 параметру для `uncorrelatedListInfo`.

10.2.4 Ошибки операции «поиск»

Если запрос на выполнение операции «поиск» оказался безуспешным, должна быть выдана хотя бы одна из перечисленных ошибок. Ситуация, в которых должны выдаваться конкретные ошибки, определены в разделе 12.

10.2.5 Принятие решений при выполнении операции «поиск» в использовании базового управления доступом

Если базовое управление доступом действует для подлежащих поиску частей ДИС, то используется следующая последовательность управлений доступом.

- 1) Для записи, идентифицированной аргументом `object` не требуется никакого конкретного разрешения.

Примечание — Если `baseObject` указывается в `SearchArgument` (т.е. если аргумент `subset` определяет `baseObject` или `wholeSubtree`), используются управления доступом, указанные в перечислениях 2—4.

- 2) Для каждой записи в области действия аргумента `SearchArgument`, являющегося кандидатом на рассмотрение, требуется разрешение `Browse`. Записи, для которых это разрешение не предоставлено, игнорируются.
- 3) Аргумент `filter` применим к каждой записи, оставленной на рассмотрение с учетом перечисления 2, и соответствии со следующим:

- а) для каждого `FilterItem`, определяющего атрибут, требуется разрешение `FilterMatch` для типа атрибута, прежде чем `FilterItem` может быть оценен как «истинный» или «ложный». `FilterItem`, для которого такое разрешение не предоставлено, оценивается как «неопределенный»;
 - б) для каждого `FilterItem`, который дополнительно определяет значение атрибута, требуется разрешение `FilterMatch` для каждого хранимого значения атрибута, которое подлежит рассмотрению с целью сравнения. При наличии значения, которое соответствует `FilterItem` и для которого предоставлено разрешение, `FilterItem` оценивается как «истинный», и в противном случае как «ложный».
- 4) Если применены процедуры, определенные в перечислениях 2 и 3, запись выбирается или отклоняется. Если в результате применения этих управлений ко всему рассматриваемому поддереву никаких записей не было выбрано (за исключением любой `ContinuationReference` и `PartialOutcomeQualifier`) и если разрешения `DiscloseOnError` не предоставлено для записи, идентифицированной аргументом `baseObject`, то операция заканчивается безуспешно и должен быть выдан параметр `NameError` с проблемой `noSuchObject`. Элемент `matched` должен содержать либо имя следующей старшей записи, для которой предоставлено разрешение `DiscloseOnError`, либо имя корня ДИС (т. е. пустой `RDNSequence`). В противном случае операция продолжается, но никакой подчиненной информации с ней не передается.

П р и м е ч а н и я

- 1 В случае выдачи `NameError ACC`, который не имел доступа ко всем старшим записям, может использоваться пустая `RDNSequence`.
- 2 Стратегия защиты может предотвратить раскрытие информации о сведениях, которая в противном случае была бы передана как `ContinuationReference` в `PartialOutcomeQualifier`. Если такая стратегия дефакто и если АПС ограничивает услуги, определяя `chainingProhibited`, справочник может выдать `ServiceError` с проблемой `chainingRequired`. В противном случае `ContinuationReference` опускается из `PartialOutcomeQualifier`.
- 5) В противном случае для каждой выбранной записи выдается следующая информация:
 - а) если элемент `infoTypes` в `selection` определяет, что должны быть выданы только типы атрибутов, то для каждого подлежащего выдаче типа атрибута, требуется разрешение `Read`. Если разрешение не предоставлено, тип атрибута опускается из `EntryInformation`. Если в результате применения этих управлений никакой информации о типе атрибута не выбирается, выдается элемент `EntryInformation`, но никакой информации о типе атрибута с ним не передается (т. е. элемент `SET OF CHOICE` опускается или он пустой);
 - б) если элемент `infoTypes` в `selection` определяет, что должны быть выданы только типы атрибутов и их значения, то для каждого типа атрибута и каждого значения требуется разрешение `Read`. Если разрешение для типа атрибута не предоставлено, атрибут опускается в `EntryInformation`. Если не предоставлено разрешение для значения атрибута, это значение опускается в соответствующем атрибуте. В случае, если разрешение не предоставлено ни на одно из значений атрибута, выдается элемент `Attribute`, содержащий пустой `SET OF AttributeValue`. Если в результате применения этих управлений никакой информации атрибута не выбирается, выдается элемент `EntryInformation`, но никакой информации атрибута с ним не передается (т. е. элемент `SET OF CHOICE` опущен или он пустой).

П р и м е ч а н и е 1 — Если разрешение `DiscloseOnError` не предоставлено для записи, идентифицированной аргументом `baseObject`, то `PartialOutcomeQualifier`, указывающий `limitProblem` или `unavailableCriticalExtension`, не должен выдаваться, поскольку он может нарушить защиту этой записи.

10.2.5.1 Размещение псевдонима во время операции «поиск»

Для размещения псевдонима во время операции «поиск» никаких конкретных разрешений не требуется (определяется установкой параметра `searchAliases` в значение «истинно»). Однако, если для каждой встретившейся записи псевдонима размещение псевдонима обусловило выдачу `ContinuationReference` и `partialOutcomeQualifier`, то применимые следующие управления доступом требуют разрешения `Read` для записи псевдонима, для атрибута `AliasedObjectName` и для единственного его значения. Если какое-либо из этих разрешений не предоставлено, `ContinuationReference` должен быть опущен в `partialOutcomeQualifier`. Эти управления доступом должны также применяться к `continuationReference`, который поступил в ответе из другого АСС. То есть АСС должен контролировать все указатели `continuationReferences` независимо от того, созданы они локально или нет.

Примечание — Кроме описанных выше управлений доступом стратегия защиты может предотвращать раскрытие информации о сведениях, которая в противном случае была бы передана как ContinuationReferences в PartialOutcomeQualifier. Если такая стратегия действует и если АПС ограничивает услуги, определяя chainingProhibited, справочник может выдать ServiceError с проблемой chainingRequired. В противном случае ContinuationReference опускается из partialOutcomeQualifier.

10.2.5.2 Нераскрываемость неполных результатов

При выдаче неполного результата в EntryInformation, т. е. если некоторые из атрибутов или значений атрибута будут опущены из-за используемых управлений доступом, элемент incompleteEntry должен быть установлен в значение «истинно», если разрешение DiscloseOnError предоставлено, по меньшей мере, одному типу атрибута, который не указывается в результате, или, по меньшей мере, одному значению атрибута, которое не указывается в результате (для такого типа атрибута разрешение «чтение» было предоставлено).

11 ОПЕРАЦИИ СПРАВОЧНИКА ТИПА «МОДИФИКАЦИЯ»

Существуют четыре операции справочника типа «модификация»: addEntry (добавление записи), removeEntry (удаление записи), modifyEntry (модификация записи) и modifyDN (модификация РИ), определенные в 11.1–11.4 соответственно.

Примечания

- 1 Каждая из этих операций идентифицирует целевую запись по ее различительному имени.
- 2 Успешность выполнения операций AddEntry, RemoveEntry и ModifyDN может зависеть от физического расположения ИБС в справочнике. О безуспешности выполнения операции должно быть сообщено вместе с ошибкой UpdateError и проблемой affectsMultipleDSAs (см. ИСО/МЭК 9594-4).
- 3 В случае отказа нижерасположенного механизма связи, результат операции будет неопределенным. Пользователь должен использовать операции опроса справочника для проверки успешности выполнения предпринимаемой операции модификации.

11.1 Добавление записи

11.1.1 Синтаксис операции «добавление записи»

Операция addEntry используется для добавления **лиственной** записи (записи объекта или псевдонима) в ДИС. Аргументы операции факультативно могут быть подписаны запросчиком (см. 7.10).

```
addEntry OPERATION ::= {
  ARGUMENT          AddEntryArgument
  RESULT             AddEntryResult
  ERRORS             | attributeError | nameError | serviceError |
                     | referral | securityError | updateError |
  CODE               id-opcode-addEntry |
  AddEntryArgument  ::= OPTIONALLY-SIGNED { SET {
    object            [0] Name,
    entry             [1] SET OF Attribute,
    targetSystem      [2] AccessPoint OPTIONAL,
    COMPONENTS OF    CommonArguments } }
  AddEntryResult     ::= NULL
```

11.1.2 Аргументы операции «добавление записи»

Аргумент object идентифицирует запись, подлежащую добавлению. Ее непосредственная старшая запись, которая уже должна существовать для успешного продолжения операции, определяется удалением последнего компонента ОРИ (относящегося к создаваемой записи).

Аргумент entry содержит информацию атрибута, которая вместе с информацией атрибута из ОРИ образует создаваемую запись. Справочник должен гарантировать, что запись соответствует схеме справочника. Если создаваемая запись является псевдонимом, то для подтверждения того, что атрибут aliasedObjectName указывает на действительную запись, никакой проверки не требуется.

Аргумент TargetSystem указывает АСС, который должен хранить новую запись. Если этот аргумент отсутствует, то в качестве АСС следует предполагать тот же АСС, который содержит старшую запись нового объекта. При наличии этого аргумента он должен указывать АСС со специфицированным AccessPoint. Этот параметр должен отсутствовать при добавлении подписи.

При наличии аргумента в параметре `criticalExtensions` аргументов `CommonArguments` должен быть установлен бит `targetSystem`, указывая, что данное расширение критическое.

Примечание — Если выбор указанного или предполагаемого АСС противоречит локальной административной стратегии, операция не выполняется и выдается ошибка.

`CommonArguments` (см. 7.3) содержит спецификации служебных ограничений, относящихся к запросу. Факультативная возможность `dontDereferenceAlias` игнорируется (и рассматривается как установленная), если только бит критического расширения `useAliasOnUpdate` не установлен в значении `criticalExtensions`. Таким образом, псевдонимы разменовываются этими операциями только в том случае, если факультативная возможность `dontDereferenceAlias` не установлена, а бит `useAliasOnUpdate` установлен. Компонент `sizeLimit`, при его наличии, игнорируется.

Примечание — Операции корректировки, которые предполагают разменовывание имени псевдонима, всегда будут заканчиваться успешно, если они используют АСС издания 1988 г.

11.1.3 Результаты операции «добавление записи»

При успешном выполнении запроса должен быть выдан положительный результат, хотя вместе с ним никакой информации не должно выдаваться.

11.1.4 Ошибки операции «добавление записи»

Если запрос на выполнение операции «добавление записи» оказался безуспешным, должна быть выдана хотя бы одна из перечисленных ошибок. Ситуации, в которых должны выдаваться конкретные ошибки, описаны в разделе 12.

11.1.5 Принятие решений при выполнении операции «добавление» и использовании базового управления доступом

Если базовое управление доступом действует при добавлении записи, применяется следующая последовательность управлений доступом:

- 1) Для непосредственной старшей записи, идентифицированной аргументом `object`, не требуется никакого конкретного разрешения.

Примечание — Стратегия защиты может воспрепятствовать пользователям справочника добавлять записи через группы АСС (например, с использованием аргумента `targetSystem`). В этом случае может быть выдана соответствующая ошибка `NameError`, `ServiceError`, `SecurityError` или `UpdateError` при условии, что это не противоречит наличию непосредственной старшей записи. Если это происходит (т.е. `DiscloseOnError` не выдается для старшей записи), то относительно старшей записи должна быть выполнена процедура, определенная в 7.11.3.

- 2) Если запись уже существует с различительным именем, равным аргументу `object`, операция заканчивается успешно в соответствии с 11.1.5.1a.

- 3) При добавлении новой записи требуется разрешение `Add`. Если это разрешение не предоставлено, операция заканчивается безуспешно согласно 11.1.5.1b.

- 4) Для каждого типа добавляемого атрибута и его значения требуется разрешение `Add`. Если никакого разрешения не предоставлено, операция заканчивается безуспешно согласно 11.1.5.1c.

11.1.5.1 Выдаваемые ошибки

Если операция заканчивается безуспешно согласно 11.1.5, применяют следующую процедуру:

- a) если операция заканчивается безуспешно согласно перечислению 2 пункта 11.1.5, то, если разрешение `DiscloseOnError` или `Add` предоставлено существующей записи, должна быть выдана `UpdateError` с проблемой `entryAlreadyExists`. В противном случае, относительно добавленной записи выполняется процедура по 7.11.3;
- b) если операция заканчивается безуспешно согласно перечислению 3 пункта 11.1.5, то относительно добавленной записи выполняется процедура, описанная в 7.11.3;
- c) если операция заканчивается безуспешно согласно перечислению 4 пункта 11.1.5, выдается ошибка `SecurityError` с указанием проблемы `insufficientAccessRights` или `noInformation`.

11.2 Удаление записей

11.2.1 Синтаксис операции «удаление записи»

Операция `RemoveEntry` используется для удаления из ДИС листовой записи (записи объекта или записи псевдонима). Факультативные аргументы операции могут быть подписаны запросчиком (см. 7.10).

```

removeEntry OPERATION      := {
  ARGUMENT                  RemoveEntryArgument
  RESULT                    RemoveEntryResult
  ERRORS                    { nameError : serviceError | referral
                             | securityError | updateError }
  CODE                      id-opcode-removeEntry
  RemoveEntryArgument ::= OPTIONALLY-SIGNED { SET {
    object                   [0] Name,
    COMPONENTS OF          CommonArguments ||
  RemoveEntryResult       ::= NULL

```

11.2.2 Аргументы операции «удаление записи»

Аргумент object идентифицирует запись, подлежащую удалению.

Аргументы CommonArguments (см. 7.3) содержат спецификацию служебных ограничений, относящихся к запросу. Факультативная возможность dontDereferenceAlias игнорируется (и рассматривается как установленная), если только бит критического расширения useAliasOnUpdate не установлен в значении criticalExtensions. Таким образом псевдонимы переименовываются этой операцией только в том случае, если факультативная возможность dontDereferenceAlias не установлена, и бит useAliasOnUpdate установлен. Компонент sizeLimit, при его наличии, игнорируется.

Примечание — Операции обновления, предполагающие переименование имени псевдонима, всегда будут заканчиваться безуспешно, если они используют АСС издания 1988 г.

11.2.3 Результаты операции «удаление записи»

При успешном выполнении запроса должен быть выдан положительный результат, хотя никакой информации с ним не должно выдаваться.

11.2.4 Ошибки операций «удаление записи»

Если запрос на выполнение операции «удаление записи» оказался безуспешным, должна быть выдана хотя бы одна из перечисленных ошибок. Ситуации, в которых должны выдаваться конкретные ошибки, описаны в разделе 12.

11.2.5 Принятие решений при выполнении операции «удаление записи» и использования базового управления доступом

Если при удалении записи действует базовое управление доступом, применяется следующая последовательность управлений доступом.

Для записи, подлежащей удалению, требуется разрешение Remove. Если такое разрешение не предоставлено, операция заканчивается безуспешно согласно 7.11.3.

Примечание — Для любых атрибутов и значений атрибутов, входящих в удаляемую запись, никаких специальных разрешений не требуется.

11.3 Модификация записей

11.3.1 Синтаксис операции «модификация записи»

Операция ModifyEntry используется для выполнения одной или нескольких последовательностей перечисленных ниже модификаций отдельной записи:

- добавление нового атрибута;
- удаление атрибута;
- добавление значений атрибутов;
- удаление значений атрибутов;
- замена значений атрибутов;
- модификация псевдонима.

Факультативно аргументы операции могут быть подписаны запросчиком (см. 7.10).

```

modifyEntry OPERATION      ::= {
  ARGUMENT                  ModifyEntryArgument
  RESULT                    ModifyEntryResult
  ERRORS                    { attributeError : nameError | serviceError
                             | referral | securityError | updateError }
  CODE                      id-opcode-modifyEntry
  ModifyEntryArgument ::= OPTIONALLY-SIGNED { SET {
    object                   [0] Name,
    changes                  [1] SEQUENCE OF EntryModification,

```

COMPONENTS OF	CommonArguments 31
ModifyEntryResult	::= NULL
EntryModification	::= CHOICE {
addAttribute	{0} Attribute,
removeAttribute	{1} AttributeType,
addValues	{2} Attribute,
removeValues	{3} Attribute }

11.3.2 А р г у м е н т ы о п е р а ц и и «м о д и ф и к а ц и я з а п и с и»

Аргумент object идентифицирует запись, к которой относится модификация.

Аргумент changes определяет последовательность модификаций, применяемых в заданном порядке. Если какие-либо отдельные модификации заканчиваются безуспешно, генерируется ошибка AttributeError и запись остается в том же состоянии, в котором она находилась до выполнения операции. То есть операция носит атомарный характер. Конечный результат последовательности модификаций не должен нарушать схему справочника. Однако это возможно, а иногда и необходимо при появлении индивидуальных изменений EntryModification. Могут быть выполнены следующие типы модификации:

- AddAttribute** — идентифицирует новый атрибут, который должен быть добавлен к записи, полностью идентифицированной данным аргументом. Любая попытка добавить уже существующий атрибут приводит к ошибке AttributeError.
- RemoveAttribute** — идентифицирует (своим типом) атрибут, подлежащий удалению из записи. Любая попытка удалить несуществующий атрибут приводит к ошибке AttributeError.
- AddValues** — идентифицирует атрибут указанием его типа в аргументе и определяет одно или несколько его значений, которые должны быть добавлены к атрибуту. Попытка добавить уже существующее значение приводит к ошибке. Попытка добавить значение к несуществующему типу также приводит к ошибке.
- RemoveValues** — идентифицирует атрибут указанием его типа в аргументе и определяет одно или несколько его значений, которые должны быть удалены из атрибута. Если значения не представлены в атрибуте, это приводит к ошибке AttributeError.

П р и м е ч а н и е — Эта операция не допускается при наличии любого значения в ОРИ.

Значения могут быть изменены комбинацией addValues и removeValues в одной операции ModifyEntry.

Аргументы CommonArguments (см. 7.3) содержат спецификации служебных ограничений, относящихся к запросу. Факультативная возможность dontDereferenceAlias игнорируется (и рассматривается как установленная), если только бит критического расширения useAliasOnUpdate не установлен в значении criticalExtensions. Таким образом псевдонимы размещаются этой операцией, если только факультативная возможность dontDereferenceAlias не установлена, а бит useAliasOnUpdate установлен. Компонент sizeLimit, при его наличии, игнорируется.

П р и м е ч а н и е — Операции обновления, осуществляющиеся разномыслием имени псевдонима, всегда будут заканчиваться безуспешно, если они используют АСС издания 1988 г.

Операция может быть использована для модификации операционных атрибутов справочника. Могут быть изменены только те операционные атрибуты, которые не классифицируются как noUserModification (и к которым пользователь имеет действующие права доступа для модификации).

П р и м е ч а н и е — Независимо от наличия разрешения пользователя на модификацию, справочник может изменять значения операционных атрибутов справочника в виде побочного результата выполнения других операций справочника.

Операция может быть использована для модификации коллективных атрибутов только в том случае, если подзапись subentries служебных ограничений установлена в значении «истинно» и если object является подзаписью, фактически содержащей коллективный(ие) атрибут(ы), подлежащий(ие) модификации.

П р и м е ч а н и е — Следует быть внимательным при модификации информации, выдаваемой для чтения записи: некоторая информация может быть получена из коллективных атрибутов, и не может быть модифицирована при выполнении операции, ориентированной на саму запись. Например, невозможно удалить коллективный атрибут из (обычной) записи путем модификации записи removeAttribute (может быть выдана ошибка attributeError с проблемой noSuchAttributeOrValue).

Операция может быть использована для модификации значения атрибута класса «объект» записи, если значения определяют вспомогательные классы объекта. Однако попытка изменить значение класса «объект», которое определяет структурный класс объекта записи, может привести к выдаче ошибки `updateError` с проблемой `objectClassModificationProhibited`. Любая модификация во вспомогательных классах объекта должна сохранять корректность и согласованность суперклассовых ссылок с результатом определения класса объекта.

11.3.3 Результаты операции «модификация записи»

При успешном выполнении запроса должен быть выдан положительный результат, хотя вместе с ним никакой информации не должно выдаваться.

11.3.4 Ошибки при выполнении операции «модификация записи»

Если запрос на выполнение операции «модификация записи» оказался безуспешным, должна быть выдана хотя бы одна из перечисленных ошибок. Ситуации, в которых должны выдаваться конкретные ошибки, определены в разделе 12.

11.3.5 Принятие решений при выполнении операции «модификация записи» и использовании базового управления доступом

Если базовое управление доступом действует при модификации записи, применяется следующая последовательность управлений доступом.

- 1) Для записи, подлежащей модификации, требуется разрешение `Modify`. Если разрешение не предоставлено, операция заканчивается безуспешно согласно 7.11.3.
- 2) Для каждого специфицированного аргумента `EntryModification`, используемого в последовательности, требуются следующие разрешения:
 - a) `Add` для типа атрибута и каждого его значения, указанных в параметре `addAttribute`. Если эти разрешения не предоставлены или атрибут уже существует, операция заканчивается безуспешно согласно 11.3.5.1a.
 - b) `Remove` для типа атрибута, указанного в параметре `removeAttribute`. Если это разрешение не предоставлено, операция заканчивается безуспешно согласно 11.3.5.1b.

Примечание — Для любого значения атрибута, входящего в удаляемый атрибут, никаких специальных разрешений не требуется.

- c) `Add` для каждого значения атрибута, указанного в параметре `addValues`. Если эти разрешения не предоставлены или какое-либо значение атрибута уже существует, операция заканчивается безуспешно согласно 11.3.5.1c.
 - d) `Remove` для каждого значения, указанного в параметре `removeValues`. Если эти разрешения не предоставлены, операция заканчивается безуспешно согласно 11.3.5.1d.

Примечание — Если конечный результат модификации `removeValues` состоит в удалении последнего значения атрибута (что приводит к удалению самого атрибута), то для данного типа атрибута требуется также разрешение `Remove`.

11.3.5.1 Выдаваемые ошибки

Если операция заканчивается безуспешно согласно 11.3.5, применяют следующую процедуру.

- a) Если операция заканчивается безуспешно согласно 11.3.5.2i), может быть выдана одна из следующих ошибок: если атрибут уже существует и разрешение `discloseOnError` или `add` предоставлены этому атрибуту, должна быть выдана ошибка `AttributeError` с проблемой `attributeOrValueAlreadyExists`; в противном случае должна быть выдана ошибка `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`.
- b) Если операция заканчивается безуспешно согласно 11.3.5.2ii), может быть выдана одна из следующих ошибок: если разрешение `DiscloseOnError` предоставлено удаляемому атрибуту и атрибут существует, должна быть выдана ошибка `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`; в противном случае должна быть выдана ошибка `AttributeError` с проблемой `noSuchAttributeOrValue`.
- c) Если операция заканчивается безуспешно согласно 11.3.5.2iii), может быть выдана одна из следующих ошибок: если значение атрибута уже существует и разрешение `discloseOnError` или `add` предоставлено этому значению атрибута, должна быть выдана ошибка `AttributeError` с проблемой `attributeOrValueAlreadyExists`; в противном случае должно быть проверено разрешение `discloseOnError` на уровне атрибутов. Если атрибуту предоставлено разрешение `discloseOnError`, должна быть выдана ошибка `SecurityError` с проблемой `insufficientAccessRights` или `noInformation`, в противном случае должна быть выдана ошибка `AttributeError` с проблемой `noSuchAttributeOrValue`.

- d) Если операция заканчивается безуспешно согласно 11.3.5.2iv), может быть выдана одна из следующих ошибок: если разрешение DiscloseOnError предоставлено для любых значений атрибута, должна быть выдана ошибка SecurityError с проблемой insufficientAccessRights или noInformation, в противном случае должна быть выдана ошибка AttributeError с проблемой noSuchAttributeOrValue.

11.4 Модификация РИ

11.4.1 Синтаксис операции «модификация РИ»

Операция ModifyDN используется для изменения относительного различительного имени записи, для перемещения записи в новую старшую запись в ДИС либо для того и другого. Эта операция может использоваться над записями объекта или псевдонима. Если запись имеет подчиненные записи, то все подчиненные записи переименовываются или перемещаются соответственно (т. е. поддерево остается неизменным). Аргументы операции факультативно могут быть подписаны запятой (см. 7.10).

Примечания

1 Системы издания 1988 г. могут использовать эту операцию только для изменения относительного различительного имени листовой записи.

2 Системы издания 1993 г. могут использовать эту операцию при замене записи на новую старшую, если только прежняя старшая запись, новая старшая запись и все ее подчиненные записи находятся в одном АСС.

3 В результате этой операции записи не пересылаются к новому АСС; все они остаются у персонального АСС.

4 Операция либо заканчивается успешно, либо она не выполняется вообще; она не должна заканчиваться безуспешно, если некоторые записи переименны, а другие нет. Никакие промежуточные состояния этой операции не должны быть наблюдаемыми для пользователей справочника.

5 После выполнения этой операции может потребоваться некоторая автономная активность для сохранения согласованности, например, обновление атрибутов в любых записях, содержащих значения различительных имен, которые относятся к переименованной(ым) или замененной(ым) записи(ям).

6 Атрибут modifyTimeStamp не корректируется для записей, подчиненных переименованной или замененной записи.

```

modifyEntry OPERATION      ::= |
ARGUMENT                   ModifyEntryArgument
RESULT                     ModifyEntryResult
ERRORS                     { attributeError | nameError | serviceError
                             | referral | securityError | updateError |
                             id-opcode-modifyEntry }
CODE                       id-opcode-modifyEntry
ModifyEntryArgument        ::= OPTIONALY-SIGNED { SET {
    object                  [0] Name,
    changes                 [1] SEQUENCE OF EntryModification,
    COMPONENTS OF          CommonArguments } }
ModifyEntryResult          ::= NULL

```

11.4.2 Аргументы операции «модификация РИ»

Аргумент object идентифицирует запись, различительное имя которой должно модифицироваться. Псевдонимы в имени не должны переименовываться.

Аргумент NewRDN определяет новое ОРИ записи. Если операция изменяет запись на новую старшую без изменения ее ОРИ, в этом параметре используется прежнее ОРИ.

Если значение атрибута нового ОРИ больше не существует в записи (либо как часть прежнего ОРИ, либо как неразличительное значение), оно добавляется. Если оно не может быть добавлено, выдается ошибка.

Если признак deleteOldRDN установлен, все значения атрибута в прежнем ОРИ, которые не вошли в новое ОРИ, удаляются. Если этот признак не установлен, прежние значения должны оставаться в записи (но не как часть ОРИ). Признак должен быть установлен, если атрибут одного значения в ОРИ изменяет свое значение в результате этой операции. Если эта операция удаляет последнее значение атрибута, этот атрибут должен быть удален.

Аргумент NewSuperior, при его наличии, определяет запись, которая должна быть заменена на новую старшую запись в ДИС. Запись становится непосредственно подчиненной записи с указанным различительным именем, которая уже должна быть существующей записью объекта. Новая

старшая запись не должна быть самой записью или любой из ее подчиненных записей, или псевдонимом, или такой, которая при замене нарушает какие-либо правила структурирования ДИС. Возможно, что записи, подчиненные замененной, могут иногда нарушить действующую подсхему, в этом случае ответственность за последующее урегулирование этих записей в соответствии с подсхемой несут административные уполномоченные подсхемы, как описано в разделе 13 ИСО/МЭК 9594-1.

При наличии аргумента `newSuperior` в параметре `criticalExtensions` аргумента `CommonArguments` должен быть установлен бит `newSuperior`, указывающий, что это расширение является критическим.

Аргументы `CommonArguments` (см. 7.3) включают спецификацию служебных ограничений, применяемых в запросе. Для этой операции факультативная возможность `dontDereferenceAlias` и компонент `sizeLimit` не требуются, а при их наличии они игнорируются. Эта операция никогда не переименовывает псевдонимов.

11.4.3 Результаты операции «модификация РИ»

При успешном выполнении запроса должен быть выдан положительный результат, хотя никакая информация с ним не должна передаваться.

11.4.4 Ошибки при выполнении операции «модификация РИ»

При безуспешном выполнении запроса операции «модификация РИ» должна быть выдана хотя бы одна из перечисленных ошибок. Условия выдачи конкретных ошибок определены в разделе 12.

11.4.5 Принятие решений при выполнении операции «модификация РИ» и использовании базового управления доступом

Если базовое управление доступом действует при переименовании записей, то применяется следующая последовательность управлений доступом:

- если в результате операции должно измениться ОРИ записи, то для переименовываемой записи требуется разрешение `Rename` (с учетом ее первоначального имени). Если это разрешение не предоставлено, операция заканчивается безуспешно согласно 11.4.5.1;
- если в результате операции запись должна быть перемещена в новую старшую запись в ДИС, требуется разрешение `Export` для записи, рассматриваемой с первоначальным именем, и разрешение `Import` для записи, рассматриваемой с ее новым именем. Если ни одного из этих разрешений не предоставлено, операция заканчивается безуспешно согласно 11.4.5.1.

Примечания

1 Разрешение `Import` должно быть обеспечено согласно АС1.

2 Никаких дополнительных разрешений не требуется даже в том случае, если в результате модификации последнего ОРИ имени должно быть добавлено новое различительное значение или удалено старое.

11.4.5.1 Выдаваемые ошибки

Если операция заканчивается безуспешно согласно 11.4.5, то выполняется процедура, описанная в 7.11.3, относительно записи, подлежащей переименованию (рассматриваемой с ее первоначальным именем).

12 ОШИБКИ

12.1 Предпочтительность ошибок

Справочник не должен продолжать выполнение операции после того, как он определил, что должна быть выдана ошибка.

Примечания

1 Логика этого правила состоит в том, что первая возникшая ошибка может отличаться от повторений одного и того же запроса, поскольку не установлено определенной логической последовательности обработки конкретного запроса. Например, АСС могут отсыкаться в различном порядке.

2 Определенные здесь правила предпочтительности ошибок применимы только к абстрактным услугам, обеспечиваемых справочником в целом. В зависимости от внутренней структуры справочника применимы различные правила.

При одновременном обнаружении нескольких ошибок, справочник определяет из приводимого ниже перечня, какую ошибку следует выдать. Ошибка, расположенная в списке выше, логически предпочтительнее расположенной ниже, и именно она выдается.

- a) `NameError`
- b) `UpdateError`
- c) `AttributeError`

- d) SecurityError
- e) ServiceError

Следующие ошибки не создают никаких конфликтов предпочтительности.

- a) AbandonFailed, поскольку она определена только в операции Abandon, и не может конфликтовать ни с какой другой ошибкой;
- b) Abandoned, которая не выдается, если операция Abandon получена одновременно с обнаружением ошибки. В этом случае ошибка AbandonFailed, сообщающая проблему tooLate, выдается вместе с уведомлением о фактической ошибке;
- c) Referral, которая не является «реальной» ошибкой, а только указанием, обнаруженным справочником, на то, что АПС должен вызвать свой запрос в другой пункт доступа.

12.2 Отклонение

Такой результат может быть выдан для любой ожидающей обработки операции запроса справочника («чтение», «поиск», «сравнение», «список»), если АПС вызывает операцию «отклонение» с соответствующим invokeID.

abandoned ERROR ::= { -- в буквальном смысле это не «ошибка»

CODE id-errcode-abandoned }

С этой ошибкой не выдается никаких параметров

12.3 Безуспешное выполнение операции «отклонение»

Ошибка AbandonFailed сообщает о проблеме, появившейся при попытке выполнения операции «отклонение».

abandonFailed ERROR ::= {

PARAMETER SET {

problem {0} AbandonProblem,

operation {1} InvokeID

CODE id-errcode-abandonFailed }

AbandonProblem ::= INTEGER {noSuchOperation (1), tooLate (2), cannotAbandon (3) }

Параметры этой ошибки имеют следующие значения.

Problem указывает конкретную проблему. Может быть указана любая из следующих проблем:

- a) noSuchOperation, если справочник не имеет никаких сведений об операции, которая должна быть отклонена (это могло произойти из-за того, что никакого приращения операции не было вообще, или из-за того, что справочник забыл о нем);
- b) tooLate, если справочник уже ответил на операцию;
- c) cannotAbandon, если была предпринята попытка отклонить операцию, для которой отклонение запрещено (например, модификация), или если отклонение не может быть выполнено.

Параметр operation указывает конкретную операцию (прилагаемую), которая должна быть отклонена.

12.4 Ошибка атрибута

AttributeError сообщает о проблеме, связанной с атрибутом.

attributeError ERROR ::= {

PARAMETER SET {

object {0} Name,

problems {1} SET OF SEQUENCE {

problem {0} AttributeProblem,

type {1} AttributeType,

value {2} AttributeValue OPTIONAL }

CODE id-errcode-attributeError }

AttributeProblem ::= INTEGER {

invalidAttributeSyntax (2)

undefinedAttributeType (3)

inappropriateMatching (4)

constraintViolation (5)

attributeOrValueAlreadyExists (6)}

Параметры этой ошибки имеют следующие значения

Параметр *object* идентифицирует запись, к которой применялась операция во время обнаружения ошибки.

Может быть указана одна или несколько проблем. Каждая из определенных ниже проблем сопровождается указанием типа атрибута, а при необходимости для исключения двусмысленности — значением объекта, обусловившего проблему:

- a) *NoSuchAttributeOrValue* — в именованной записи отсутствует один из атрибутов или одно из значений атрибута, указанных в виде аргумента операции;
- b) *InvalidAttributeSyntax* — предполагаемое значение атрибута, указанное в виде аргумента операции, не соответствует синтаксису типа атрибута;
- c) *UndefinedAttributeType* — неопределенный тип атрибута был предоставлен в виде аргумента операции. Эта ошибка может иметь место только при выполнении операций *AddEntry* или *ModifyEntry*;
- d) *InappropriateMatching* — предпринята попытка, например, в фильтре использовать правило сравнения, неопределенное для соответствующего типа атрибута;
- e) *ConstraintViolation* — значение атрибута, представленное в аргументе операции, не соответствует ограничениям, налагаемым ИСО/МЭК 9594-2, или определению атрибута (например, оно превышает максимально допустимый размер);
- f) *AttributeOrValueAlreadyExists* — предпринята попытка добавить атрибут, который уже существует в записи, или добавить значение, которое уже существует в атрибуте.

12.5 Ошибка имени

NameError сообщает о проблеме относительно имени, представленного в виде аргумента операции.

```

nameError      ERROR ::= {
    PARAMETER  SET {
        problem      {0} NameProblem,
        matched      {1} Name }
    CODE        id-errcode-nameError }
NameProblem    ::= INTEGER {
    noSuchObject      (1),
    aliasProblem      (2),
    invalidAttributeSyntax (3),
    aliasDereferencingProblem (4) }

```

Параметры этой ошибки имеют следующие значения.

Problem — конкретная встретившаяся проблема. Может быть указана любая из следующих проблем:

- a) *NoSuchObject* — предоставляемое имя не соответствует имени ни одного объекта;
- b) *aliasProblem* — псевдоним был переименован, но он не изменяет никакой объект;
- c) *invalidAttributeSyntax* — тип атрибута несовместим с сопровождающим его значением атрибута в АВА в имени;
- d) *AliasDereferencingProblem* — псевдоним появился в ситуации, когда он не был разрешен или когда доступ был отклонен.

Параметр *matched* содержит имя самой младшей записи (объект или псевдоним) в ДНС, которое было согласовано и является укороченной формой представляемого имени или результирующего имени, если псевдоним был переименован.

Примечание — При появлении проблемы с типами атрибута и/или значениями в имени, указанным в аргументе операции спрашивающего, то она сообщается через ошибку *NameError* (с проблемой *invalidAttributeSyntax*), а не *AttributeError* или *UpdateError*.

12.6 Обращение

Referral переадресует пользователя услуг к одному или нескольким пунктам доступа, которые лучше готовы к выполнению запрашиваемой операции

```

referral      ERROR ::= { -- в буквальном смысле это не «ошибка»
    PARAMETER  SET {
        candidate      {0} ContinuationReference }
    CODE        id-errcode-referral }

```

Эта ошибка имеет один единственный параметр, содержащий ContinuationReference, которая может быть использована для продолжения операции (см. ИСО/МЭК 9594-4)

12.7 Ошибка защиты

SecurityError сообщает о возникшей при выполнении операции проблеме, связанной с защитой.

securityError ERROR	:: =	{
PARAMETER	SET	{
problem		{0} SecurityProblem }
CODE		id-errcode-securityError }
SecurityProblem	:: =	INTEGER }
inappropriateAuthentication		(1),
invalidCredentials		(2),
insufficientAccessRights		(3),
invalidSignature		(4),
protectionRequired		(5),
noInformation		(6)

Эта ошибка содержит единственный параметр, сообщающий о конкретной возникшей проблеме. Могут быть указаны следующие проблемы:

- InappropriateAuthentication — уровень защиты, связанный с удостоверениями личности запросчика, несовместим с уровнем требуемой защиты, например, были обеспечены простые удостоверения личности в то время, как требовались строгие;
- InvalidCredentials — предоставленные удостоверения личности оказались недействительными;
- InsufficientAccessRights — запросчик не имеет права выполнить запрашиваемую операцию;
- InvalidSignature — подпись запроса признана недействительной;
- ProtectionRequired — справочник не желает выполнять запрашиваемую операцию, поскольку аргумент не подписан;
- NoInformation — запрошенная операция вызвала ошибку защиты, при которой информация недоступна.

12.8 Ошибка услуги

ServiceError сообщает о проблемах относительно предоставляемых услуг.

serviceError ERROR	:: =	{
PARAMETER	SET	{
problem		{0} ServiceProblem }
CODE		id-errcode-serviceError }
ServiceProblem	:: =	INTEGER }
busy		(1)
unavailable		(2)
unwilling ToPerform		(3),
chaining Required		(4)
unableToProceed		(5)
invalidReference		(6),
timeLimitExceeded		(7),
administrativeLimitExceeded		(8),
loopDetected		(9)
unavailableCriticalExtension		(10),
outOfScope		(11),
diffError		(12),
invalidQueryReference		(13) }

Эта ошибка содержит единственный параметр, сообщающий о конкретной возникшей проблеме. Могут быть указаны следующие проблемы:

- busy — справочник или некоторая его часть слишком занята, чтобы выполнить в данный момент запрашиваемую операцию, но она может быть выполнена позже через короткий промежуток времени;

- b) unavailable — справочник или некоторая его часть в настоящее время недоступны;
- c) unwilling To Perform — справочник или некоторая его часть не подготовлены к выполнению этого запроса, например, потому, что это приведет к чрезмерному потреблению ресурсов или нарушит стратегию участвующего административного уполномоченного;
- d) chaining Required — справочник не способен выполнить запрос иначе, чем способом сцепления, однако сцепление запрещено факультативной возможностью служебного ограничения chainingProhibited;
- e) unable To Proceed — АСС, вызвавший эту ошибку, не имел административного уполномоченного для соответствующего поименованного контекста и, следовательно, не способен участвовать в процессе присвоения имени;
- f) invalidReference — АСС не способен выполнить запрос, направленный АПС (с помощью OperationProgresses). Это может произойти по причине использования недействительного обращения;
- g) timeLimitExceeded — справочник достиг временного ограничения, установленного пользователем в служебном ограничении. Никаких частичных результатов нет для выдачи пользователю.
- h) administrativeLimitExceeded — справочник достиг некоторого предела, установленного административным уполномоченным, и никаких частичных результатов нет для выдачи пользователю;
- i) loopDetected — справочник не способен выполнить этот запрос из-за внутреннего заикливания;
- j) unavailableCriticalExtension — справочник оказался не способен удовлетворить запрос из-за недоступности одного или нескольких критических расширений;
- k) outOfScope — в запрошенной области ни одно из обращений оказалось недоступным;
- l) dleError — справочник не способен выполнить запрос из-за существующей проблемы совместимости ДИС;
- m) invalidQueryReference — параметры запрошенной операции недействительны. Эта проблема сообщается, если ссылка queryReference в постраничных результатах недействительна.

Примечание — Эта проблема не поддерживается системами издания 1988 г.

12.9 Ошибки обновления

UpdateError сообщает о проблемах, возникших при попытках выполнить операции добавления, удаления или модификации информации в ИБС.

```
updateError ERROR ::= {
  PARAMETER SET {
    problem {0} UpdateProblem }
  CODE id-errcode-updateError }
UpdateProblem ::= INTEGER {
```

- namingViolation (1)
- objectClassViolation (2)
- notAllowedOnNonLeaf (3)
- notAllowedOnRDN (4)
- entryAlreadyExists (5)
- affectsMultipleDSAs (6)
- objectClassModificationProhibited (7)

Эта ошибка содержит один единственный параметр, сообщающий о конкретной обнаруженной проблеме. Могут быть указаны следующие проблемы:

- a) NamingViolation — попытка добавления или модификации могла бы нарушить правила структурирования ДИС, определенные в схеме справочника и в ИСО/МЭК 9594-2. То есть, это могло бы привести к размещению записи в виде подчиненной записи псевдонима или в регионе ДИС, недопустимого для членов его класса объектов, или могло бы определить ОРИ так, что в запись включался бы запрещенный тип атрибута;
- b) ObjectClassViolation — попытка обновления могла бы привести к созданию записи, противоречащей правилам формирования содержимого записи; например, определению его класса объектов, правилам формирования содержимого ДИС или определением в ИСО/МЭК 9594-2 относительно их классов объектов.

- c) *NotAllowedOnNonLeaf* — предпринимаемая операция допускается только для листевых записей ДИС;
- d) *NotAllowedOnRDN* — предпринимаемая операция может повлиять на ОРИ (например, удаление атрибута, который является частью ОРИ);
- e) *EntryAlreadyExists* — при операциях *AddEntry* или *ModifyDN* предпринималось присвоение имени уже существующей записи;
- f) *AffectsMultipleDSAs* — предпринимаемое обновление может потребовать работы с группой АСС, когда эта операция не допускается;
- g) *ObjectClassModificationProhibited* — предпринята попытка модифицировать класс структурированных объектов.

Примечание — Ошибка *UpdateError* не используется для уведомлений о проблемах, связанных с типами атрибута, значениями или нарушениями ограничений, которые возникают при выполнении операций *addEntry*, *removeEntry*, *modifyEntry* или *modifyDN*. О таких проблемах сообщается посредством атрибута *AttributeError*.

ПРИЛОЖЕНИЕ А

(обязательное)

АБСТРАКТНЫЕ УСЛУГИ В АСН.1

В данном приложении приведены определения всех типов и значений АСН.1, а также информационных объектов, содержащихся в настоящем стандарте в виде модуля АСН.1 «DirectoryAbstractService».

```
DirectoryAbstractService {joint-iso-ccitt ds(5) module(1) directoryAbstractService(2) 2}
DEFINITIONS ::=
BEGIN
-- EXPORTS ALL --
-- Определенные в этом модуле типы и значения экспортируются для использования в других модулях АСН.1,
-- содержащихся в спецификациях справочника, и в других прикладных программах, которые, в свою очередь,
-- будут использовать их для доступа к услугам справочника. Другие прикладные программы могут использовать
-- эти типы и значения для своих собственных целей, но это не должно препятствовать расширениям и
-- модификациям, необходимым при обслуживании или усовершенствовании услуг справочника.
IMPORTS
InformationFramework, distributedOperations, authenticationFramework, dap
FROM UsefulDefinitions joint-iso-ccitt ds(5) module(1)
usefulDefinitions(0) 2;
Attribute, AttributeType, AttributeValue, AttributeValueAssertion,
DistinguishedName, Name, RelativeDistinguishedName,
MATCHING-RULE
FROM InformationFramework InformationFramework
OperationProgress, ReferenceType, Exclusions, AccessPoint,
ContinuationReference
FROM DistributedOperations distributedOperations
CertificationPath, SIGNER {}, SIGNATURE {}, AlgorithmIdentifier
FROM AuthenticationFramework authenticationFramework
id-opcode-read, id-opcode-compare, id-opcode-abandon, id-opcode-list,
id-opcode-search, id-opcode-addEntry, id-opcode-removeEntry,
id-opcode-modifyEntry, id-opcode-modifyDN, id-errcode-abandoned,
id-errcode-abandonFailed, id-errcode-attributeError,
id-errcode-nameError, id-errcode-referral, id-errcode-securityError,
id-errcode-serviceError, id-errcode-updateError
FROM DirectoryAccessProtocol dap
OPERATION, ERROR
FROM Remote-Operations-Information-Objects {joint-iso-ccitt
remote-operations(4) informationObjects(5) version1(0) }
```

```

emptyUnbind
FROM Remote-Operations-Useful-Definitions {joint-iso-ccitt
    remote-operations(4) useful-definitions(7) version1 (0)}

InvokeID
FROM Remote-Operations-Generic-ROC-PDUs {joint-iso-ccitt
    remote-operations(5) generic-ROS-PDUs(6) version1(0)} ;
-- Параметризованный тип для представления факультативной
подписи --
OPTIONALLY-SIGNED [Type] ::= CHOICE {
    unsigned Type,
    signed SIGNED [Type]}
-- Общие типы данных --
CommonArguments SET {
    serviceControls [30] ServiceControls DEFAULT {},
    securityParameters [29] SecurityParameters OPTIONAL,
    requestor [28] DistinguishedName OPTIONAL,
    operationProgress [27] OperationProgress DEFAULT { nameResolutionPhaseNotStarted },
    aliasedRIDs [26] INTEGER OPTIONAL,
    criticalExtensions [25] BIT STRING OPTIONAL,
    referenceType [24] ReferenceType OPTIONAL,
    entryOnly [23] BOOLEAN DEFAULT TRUE,
    exclusions [22] Exclusions OPTIONAL,
    nameResolveOMaster [21] BOOLEAN DEFAULT FALSE }
CommonResults ::= SET {
    securityParameters [30] SecurityParameters OPTIONAL,
    performer [29] DistinguishedName OPTIONAL,
    aliasDereferenced [28] BOOLEAN DEFAULT FALSE }
ServiceControls ::= SET {
    options [0] BIT STRING {
        preferChaining (0),
        chainingProhibited (1),
        localScope (2),
        dontUseCopy (3),
        dontDereferenceAliases (4),
        subentries (5),
        copyShallDo (6) | DEFAULT {},
        priority [1] INTEGER {low (0), medium (1), high (2)} DEFAULT medium,
        timeLimit [2] INTEGER OPTIONAL,
        sizeLimit [3] INTEGER OPTIONAL,
        scopeOfReferral [4] INTEGER {dmd(0), country(1)} | OPTIONAL,
        attributeSizeLimit [5] INTEGER OPTIONAL }
EntryInformationSelection ::= SET {
    attributesCHOICE {
        allUserAttributes [0] NULL,
        select [1] SET OF AttributeType
        -- пустой набор означает, что атрибуты не запрошены --
        DEFAULT allUserAttributes: NULL,
    }
    infoTypes [2] INTEGER {
        attributeTypesOnly (0),
        attributeTypesAndValues (1) DEFAULT attributeTypesAndValues,
    }
    extraAttributes CHOICE {
        allOperationalAttributes [3] NULL,
        select [4] SET OF AttributeType | OPTIONAL }
EntryInformation : = SEQUENCE {
    name Name,
    fromEntry BOOLEAN DEFAULT TRUE,
    information SET OF CHOICE {
        attributeType AttributeType,
        attribute Attribute | OPTIONAL,
        incompleteEntry [3] BOOLEAN DEFAULT FALSE
        -- система издания не 1988 г. -- }
}

```

```

Filter      ::= CHOICE {
    item      [0] FilterItem,
    and       [1] SET OF Filter,
    or        [2] SET OF Filter,
    not       [3] Filter }

FilterItem  ::= CHOICE {
    equality   [0] AttributeValueAssertion,
    subranges [1] SEQUENCE {
        AttributeType(SupportedAttributeTypes),
        SEQUENCE OF CHOICE {
            initial [0] AttributeValue(SupportedAttributes){#type},
            any      [1] AttributeValue(SupportedAttributes){#type},
            final    [2] AttributeValue(SupportedAttributes){#type}}},
    greaterOrEqual [2] AttributeValueAssertion,
    lessOrEqual    [3] AttributeValueAssertion,
    present        [4] AttributeType,
    approximateMatch [5] AttributeValueAssertion,
    extensibleMatch  [6] MatchingRuleAssertion }

MatchingRuleAssertion ::= SEQUENCE {
    matchingRule [1] SET SIZE (1 .. MAX) OF MATCHING-RULE.&id,
    type         [2] AttributeType OPTIONAL,
    matchValue    [3] MATCHING-RULE.&AssertionType
                    (CONSTRAINED BY
-- matchValue должно представлять значение типа, указанное полем &AssertionType одного из инфор-
-- мационных объектов MATCHING-RULE, идентифицированных правилом matchingRule --) },
    dnAttributes [4] BOOLEAN DEFAULT FALSE }

PagedResultsRequest ::= CHOICE {
    newRequest      SEQUENCE {
        pageSize    INTEGER,
        sortKeys     SEQUENCE OF SortKey OPTIONAL,
        reverse      [1] BOOLEAN DEFAULT FALSE,
        unmerged     [2] BOOLEAN DEFAULT FALSE },
    queryReference  OCTET STRING }

SortKey      ::= SEQUENCE {
    type         AttributeType,
    orderingRule MATCHING-RULE.&id OPTIONAL }

SecurityParameters ::= SET {
    certification-path [0] CertificationPath OPTIONAL,
    name               [1] DistinguishedName OPTIONAL,
    time               [2] UTCTime OPTIONAL,
    random             [3] BIT STRING OPTIONAL,
    target             [4] ProtectionRequest OPTIONAL }

ProtectionRequest ::= INTEGER {none(0), signed(1)}

-- Операции «связка» и «развязка» --
directoryBind OPERATION ::= {
    ARGUMENT DirectoryBindArgument
    RESULT   DirectoryBindResult
    ERROR     directoryBindError }

DirectoryBindArgument ::= SET {
    credentials [0] Credentials OPTIONAL,
    versions    [1] Versions DEFAULT {v1}}

Credentials ::= CHOICE {
    simple      [0] SimpleCredentials,
    strong      [1] StrongCredentials,
    externalProcedure [2] EXTERNAL }

SampleCredentials ::= SEQUENCE {
    validity [1] SET {
        time1 [0] UTCTime OPTIONAL,
        time2 [1] UTCTime OPTIONAL,
        random1 [2] BIT STRING OPTIONAL,
        random2 [3] BIT STRING OPTIONAL} OPTIONAL,

```

```

password          [2] CHOICE {
    unprotected   OCTET STRING,
    protected     SIGNATURE {OCTET STRING} | OPTIONAL
}
StrongCredentials ::= SET {
    certification-path [0] CertificationPath OPTIONAL,
    bind-token         [1] Token,
    name               [2] DistinguishedName OPTIONAL
}
Token ::= SIGNED | SEQUENCE {
    algorithm [0] AlgorithmIdentifier,
    name      [1] DistinguishedName,
    time      [2] UTCTime,
    random    [3] BIT STRING {}
}
Versions ::= BIT STRING {v(0)}
DirectoryBindResult ::= DirectoryBindArgument
directoryBindError ERROR ::= {
    PARAMETER SET {
        versions [0] Versions DEFAULT {v(0)},
        error     CHOICE {
            serviceError [1] ServiceProblem,
            securityError [2] SecurityProblem {}
        }
    }
}
directoryUnbind OPERATION ::= emptyUnbind
-- Oneπατηα, απρωςμεα α περυσιαα --
read OPERATION ::= {
    ARGUMENT ReadArgument
    RESULT   ReadResult
    ERRORS   {attributeError | nameError | serviceError |
               referral | abandoned | securityError}
    CODE     {id-opcode-read}
}
ReadArgument ::= OPTIONALLY-SIGNED | SET {
    object [0] Name,
    selection [1] EntryInformationSelection DEFAULT {},
    modifyRightsRequest [2] BOOLEAN DEFAULT FALSE,
    COMPONENTS OF CommonArguments {}
}
ReadResult ::= OPTIONALLY-SIGNED | SET {
    entry [0] EntryInformation,
    modifyRights [1] ModifyRights OPTIONAL,
    COMPONENTS OF CommonResults {}
}
ModifyRights ::= SET OF SEQUENCE {
    item CHOICE {
        entry [0] NULL,
        attribute [1] Attribute Type,
        value [2] AttributeValueAssertion,
        permission [3] BIT STRING {add (0), remove(1), rename (2), move(3) }
    }
}
compare OPERATION ::= {
    ARGUMENT CompareArgument
    RESULT   CompareResult
    ERRORS   {attributeError | nameError | serviceError |
               referral | abandoned | securityError}
    CODE     {id-opcode-compare}
}
CompareArgument ::= OPTIONALLY-SIGNED | SET {
    object [0] Name,
    purported [1] AttributeValueAssertion,
    COMPONENTS OF CommonArguments {}
}
CompareResult ::= OPTIONALLY-SIGNED | SET {
    name Name OPTIONAL,
    matched [0] BOOLEAN,
    fromEntry [1] BOOLEAN DEFAULT TRUE,
    matchedSubtype [2] AttributeType OPTIONAL,
    COMPONENTS OF CommonResults {}
}

```

```

abandon OPERATION ::= {
    ARGUMENT      AbandonArgument
    RESULT        AbandonResult
    ERRORS        {abandonFailed}
    CODE          id-opcode-abandon}
AbandonArgument ::= SEQUENCE {
    invokeID      [0] InvokeID}
AbandonResult   ::= NULL
list OPERATION  ::= {
    ARGUMENT      ListArgument
    RESULT        ListResult
    ERRORS        {nameError | serviceError | referral | abandoned | securityError}
    CODE          id-opcode-list}
ListArgument ::= OPTIONALLY-SIGNED { SET {
    object      [0] Name,
    pagedResults [1] PagedResultsRequest OPTIONAL,
    COMPONENTS OF CommonArguments {} } }
ListResult ::= OPTIONALLY-SIGNED { CHOICE {
    listInfo      SET {
        name      Name OPTIONAL,
        subordinates [1] SET OF SEQUENCE {
            rdn      RelativeDistinguishedName,
            aliasEntry [0] BOOLEAN DEFAULT FALSE,
            fromEntry [1] BOOLEAN DEFAULT TRUE },
            partialOutcomeQualifier [2] PartialOutcomeQualifier OPTIONAL,
            COMPONENTS OF CommonResults {} }
        uncorrelatedListInfo [0] SET OF ListResult {} }
    PartialOutcomeQualifier ::= SET {
        limitProblem [0] LimitProblem OPTIONAL,
        unexplored [1] SET OF ContinuationReference OPTIONAL,
        unavailableCriticalExtensions [2] BOOLEAN DEFAULT FALSE,
        unknownErrors [3] SET OF ANY OPTIONAL,
        queryReference [4] OCTET STRING OPTIONAL } }
LimitProblem ::= INTEGER {
    timeLimitExceeded (0), sizeLimitExceeded (1), administrativeLimitExceeded (2) }
search OPERATION ::= {
    ARGUMENT      SearchArgument
    RESULT        SearchResult
    ERRORS        {attributeError | nameError | serviceError | referral | abandoned | securityError}
    CODE          id-opcode-search}
SearchArgument ::= OPTIONALLY-SIGNED { SET {
    baseObject      [0] Name,
    subset          [1] INTEGER { baseObject(0), oneLevel(1), wholeSubtree(2) } DEFAULT
        baseObject,
    filter           [2] Filter OPTIONAL and: {},
    searchAliases   [3] BOOLEAN DEFAULT TRUE,
    selection        [4] EntryInformationSelection DEFAULT {},
    pagedResults     [5] PagedResultsRequest OPTIONAL,
    matchedValuesOnly [6] BOOLEAN DEFAULT FALSE,
    extendedFilter   [7] Filter OPTIONAL,
    COMPONENTS OF CommonArguments {} } }
SearchResult ::= OPTIONALLY-SIGNED { CHOICE {
    searchInfo      SET {
        name      Name OPTIONAL,
        entries     [0] SET OF EntryInformation,
        partialOutcomeQualifier [2] PartialOutcomeQualifier OPTIONAL,
        COMPONENTS OF CommonResults {} }
        uncorrelatedSearchInfo [0] SET OF SearchResult {} } }

```

```

addEntry OPERATION ::= {
  ARGUMENT      AddEntryArgument
  RESULT        AddEntryResult
  ERRORS        {attributeError | nameError | serviceError | referral | securityError | updateError |}
  CODE          id-opcode-addEntry |}
AddEntryArgument ::= {
  object          [0] Name,
  entry           [1] SET OF Attribute,
  targetSystem    [2] AccessPoint OPTIONAL,
  COMPONENTS OF  CommonArguments |}
AddEntryResult ::= NULL
removeEntry OPERATION ::= {
  ARGUMENT      RemoveEntryArgument
  RESULT        RemoveEntryResult
  ERRORS        {nameError | serviceError | referral | securityError | updateError |}
  CODE          id-opcode-removeEntry |}
RemoveEntryArgument ::= {
  object          [0] Name,
  COMPONENTS OF  CommonArguments |}
RemoveEntryResult ::= NULL
modifyEntry OPERATION ::= {
  ARGUMENT      ModifyEntryArgument
  RESULT        ModifyEntryResult
  ERRORS        {attributeError | nameError | serviceError | referral | securityError | updateError |}
  CODE          id-opcode-modifyEntry |}
ModifyEntryArgument ::= {
  object          [0] Name,
  changes         [1] SEQUENCE OF EntryModification,
  COMPONENTS OF  CommonArguments |}
ModifyEntryResult ::= NULL
EntryModification ::= CHOICE {
  addAttribute     [0] Attribute,
  removeAttribute  [1] AttributeType,
  addValues        [2] Attribute,
  removeValues     [3] Attribute |}
modifyDN OPERATION ::= {
  ARGUMENT      ModifyDNArgument
  RESULT        ModifyDNResult
  ERRORS        {nameError | serviceError | referral | securityError | updateError |}
  CODE          id-opcode-modifyDN |}
ModifyDNArgument ::= {
  object          [0] DistinguishedName,
  newRDN          [1] RelativeDistinguishedName,
  deleteOldRDN    [2] BOOLEAN DEFAULT FALSE,
  newSuperior     [3] DistinguishedName OPTIONAL,
  COMPONENTS OF  CommonArguments |}
ModifyDNResult ::= NULL
-- Ошибка в параметре --
abandoned ERROR ::= -- в буквальном смысле это не «ошибка»
CODE          id-ecode-abandoned |}
abandonFailed ERROR ::= {
  PARAMETER    SET {
    problem     [0] AbandonProblem,
    operation    [1] InvokeID |}
  CODE          id-ecode-abandonFailed |}
AbandonProblem ::= INTEGER {noSuchOperation (1), tooLate (2), cannotAbandon (3) |}
attributeError ERROR ::= {
  PARAMETER    SET {
    object      [0] Name,
    problems    [1] SET OF SEQUENCE {
      problem   [0] AttributeProblem,
      type      [1] AttributeType,
      value     [2] AttributeValue OPTIONAL |}

```

```

CODE      id-errcode-attributeError }
AttributeProblem ::= INTEGER {
    noSuchAttributeOrValue          (1)
    invalidAttributeSyntax          (2)
    undefinedAttributeType          (3)
    inappropriateMatching           (4)
    constraintViolation             (5),
    attributeOrValueAlreadyExists   (6)
}

nameError ERROR ::= {
    PARAMETER      SET {
        problem      {0} NameProblem,
        matched      {1} Name }
    CODE      id-errcode-nameError
}
NameProblem ::= INTEGER {
    noSuchObject          (1),
    aliasProblem          (2),
    invalidAttributeSyntax (3),
    aliasDereferencingProblem (4) }

referral ERROR ::= { -- в буквальном смысле это не «ошибка»
    PARAMETER      SET {
        candidate      {0} ContinuationReference }
    CODE      id-errcode-referral }

securityError ERROR ::= {
    PARAMETER      SET {
        problem      {0} SecurityProblem }
    CODE      id-errcode-securityError
}
SecurityProblem ::= INTEGER {
    inappropriateAuthentication (1),
    invalidCredentials          (2),
    insufficientAccessRights     (3),
    invalidSignature            (4),
    protectionRequired          (5),
    noInformation               (6)
}

serviceError ERROR ::= {
    PARAMETER      SET {
        problem      {0} ServiceProblem }
    CODE      id-errcode-serviceError }
ServiceProblem ::= INTEGER {
    busy                    (1),
    unavailable             (2),
    unwillingToPerform      (3),
    chainingRequired        (4),
    unableToProceed         (5),
    invalidReference         (6),
    timeLimitExceeded       (7),
    administrativeLimitExceeded (8),
    loopDetected            (9),
    unavailableCriticalExtension (10),
    outOfScope              (11),
    dirError                 (12),
    invalidQueryReference    (13) }

updateError ERROR ::= {
    PARAMETER      SET {
        problem      {0} UpdateProblem }
    CODE      id-errcode-updateError }
UpdateProblem ::= INTEGER {
    namingViolation          (1),
    objectClassViolation     (2),
    notAllowedOnNonLeaf      (3),
    notAllowedOnRDN          (4),
    entryAlreadyExists       (5),
    affectsMultipleDSAs      (6),
    objectClassModificationProhibited (7)
}
END

```

ПРИЛОЖЕНИЕ В
(справочное)

ОПЕРАЦИОННАЯ СЕМАНТИКА ДЛЯ УПРАВЛЕНИЯ БАЗОВЫМ ДОСТУПОМ

Данное приложение содержит набор диаграмм, которые описывают семантику, связанную с управлением базовым доступом, в том виде, как она используется при выполнении операций справочника.

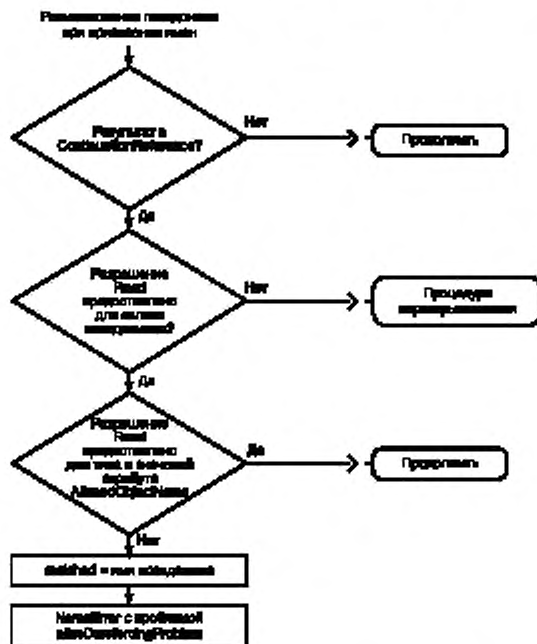


Рисунок В-1 – Разыменование псевдонима при присвоении имени

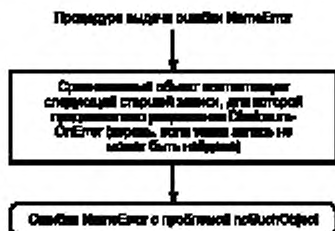


Рисунок В-2 — Уведомление об ошибке имени

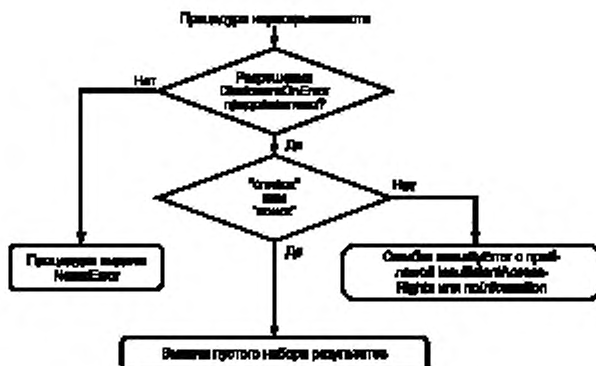


Рисунок В-3 — Нераскрываемость наличия записи

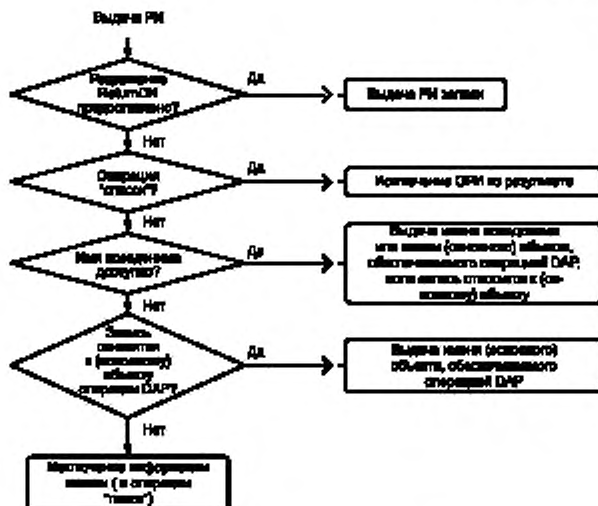


Рисунок В-4 — Выдача различительного имени

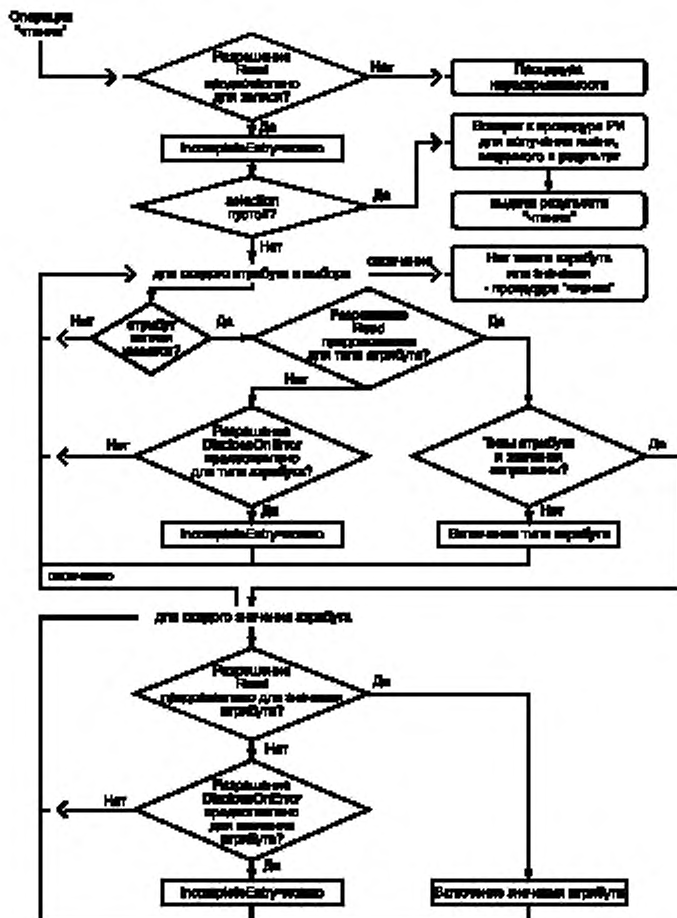


Рисунок В-5 — Операция «чтение»

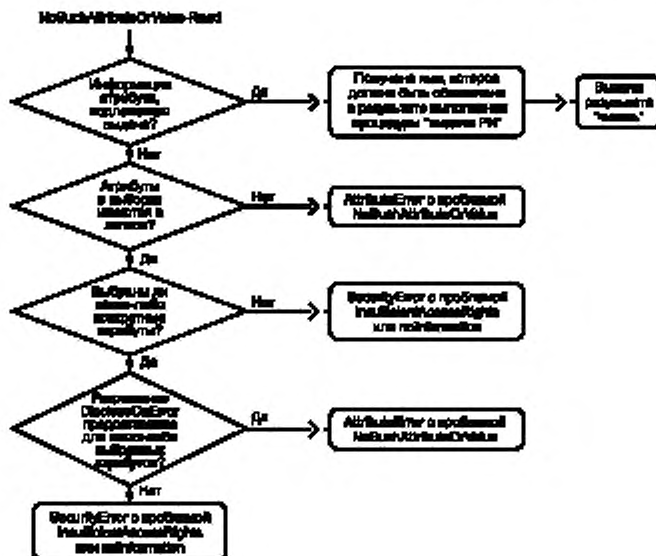


Рисунок В-6 – Отсутствие соответствующего атрибута или значения при выполнении операции «чтение»

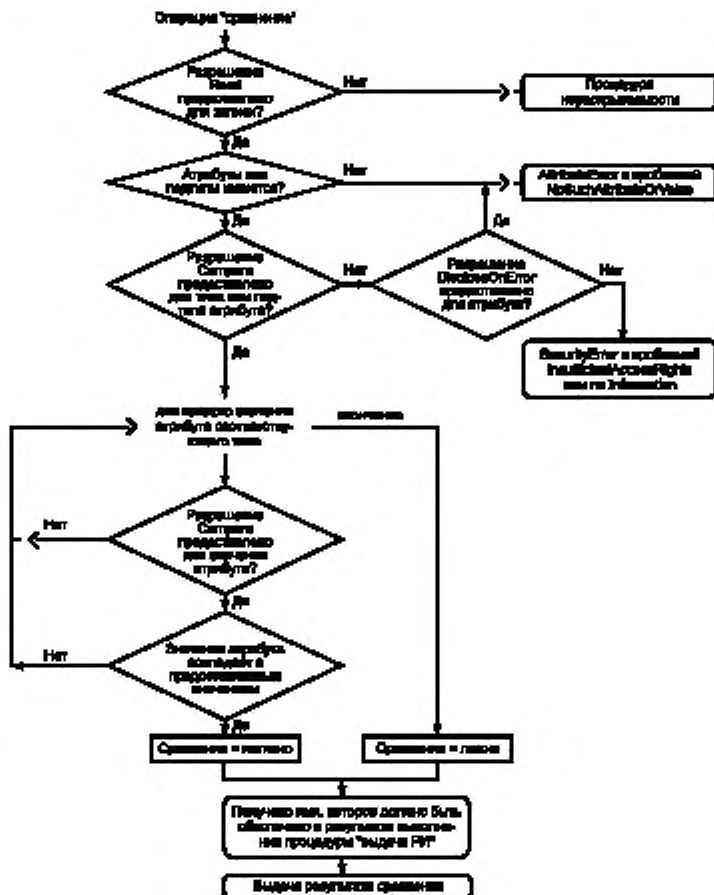


Рисунок В-7 — Операция «сравнение»

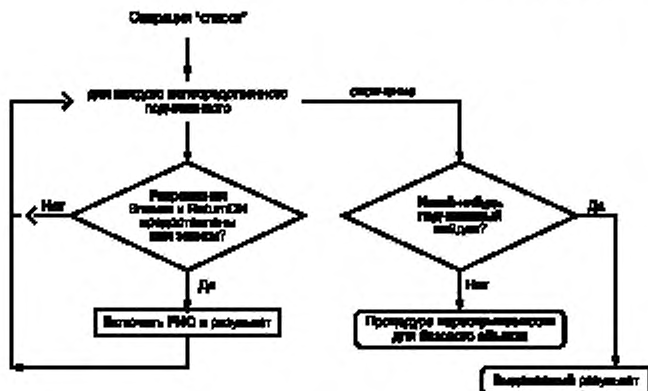


Рисунок В-5 — Операция «список»

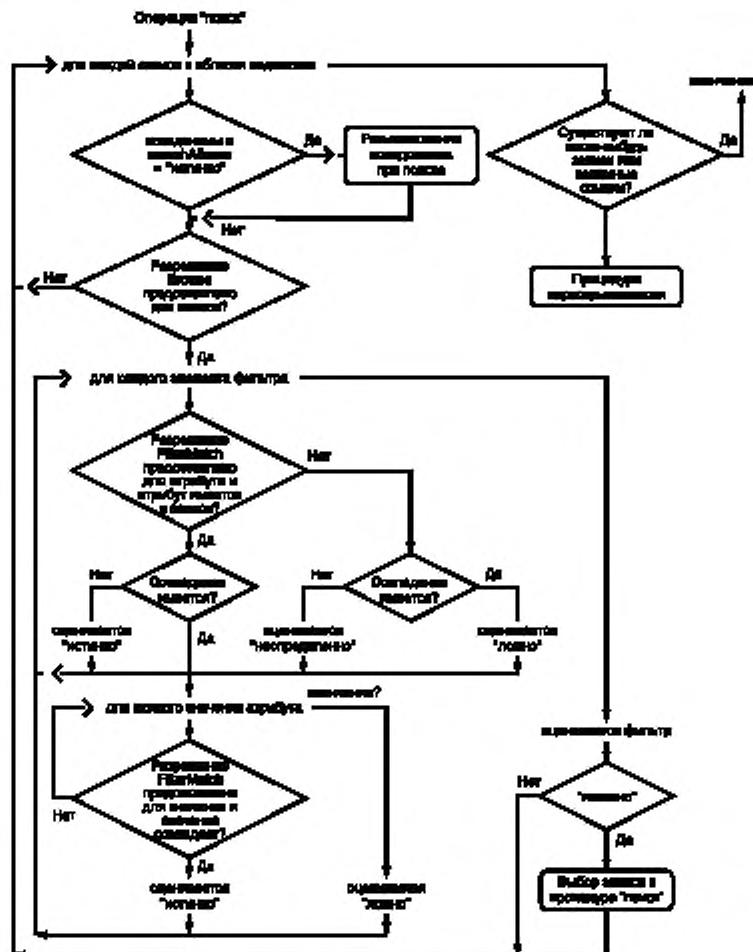


Рисунок В-9 — Операция «поиск»

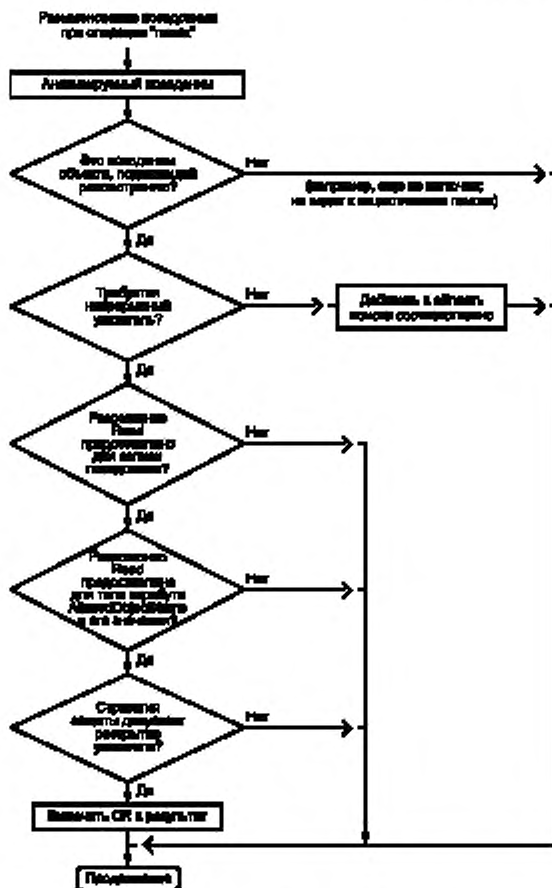


Рисунок В-10 — Размещение псевдонима в операции «поиск»

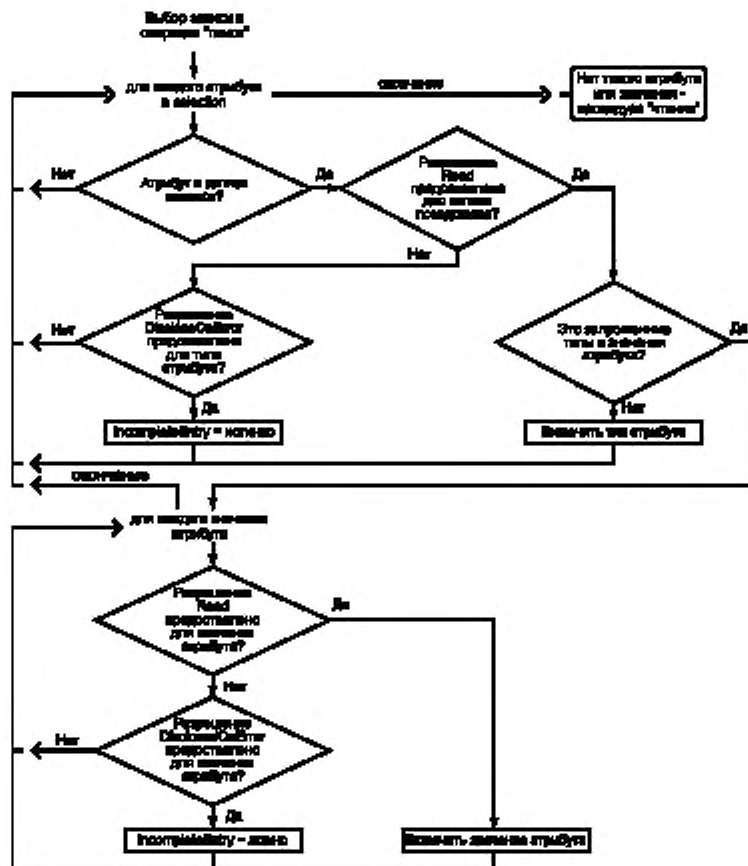


Рисунок В-11 — Выбор записи при выполнении операции «поиск»

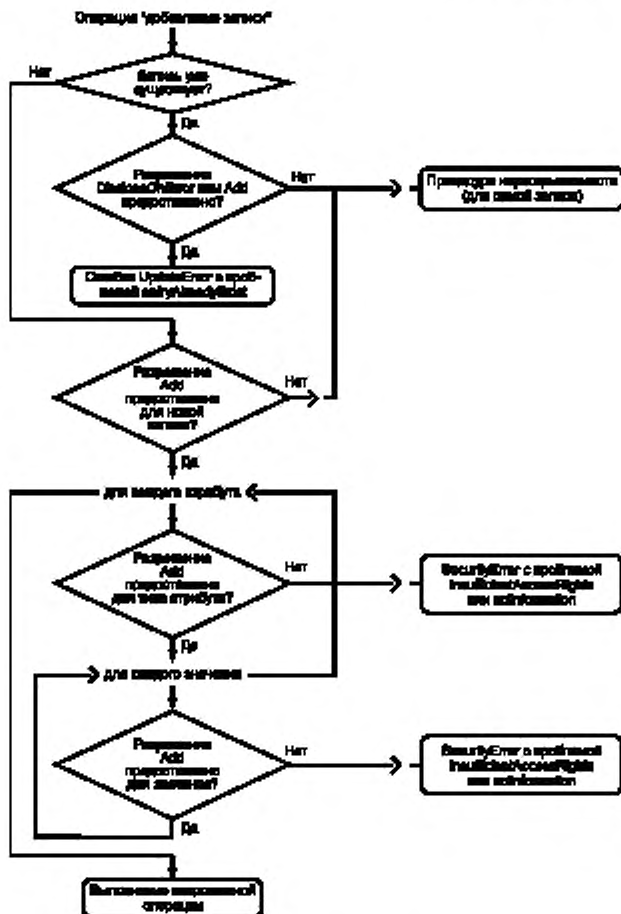


Рисунок В-12 – Операция «добавление записи»

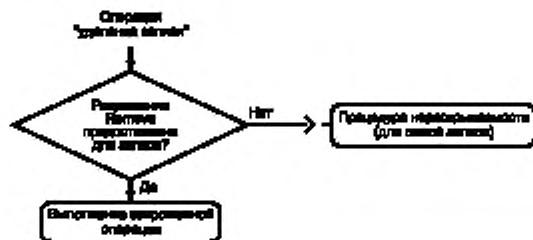


Рисунок В-13 — Операция «удаление записи»

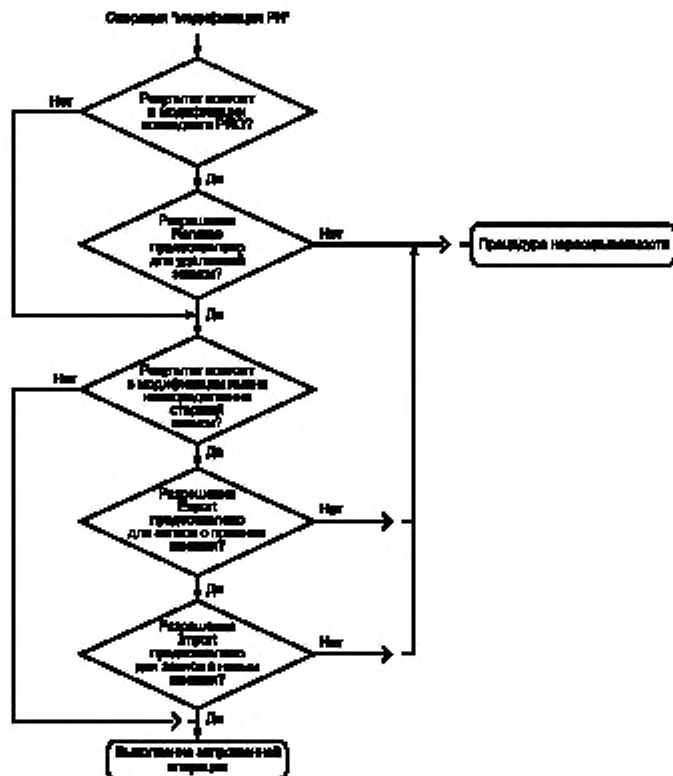


Рисунок В-15 — Операция «модификация РИ»

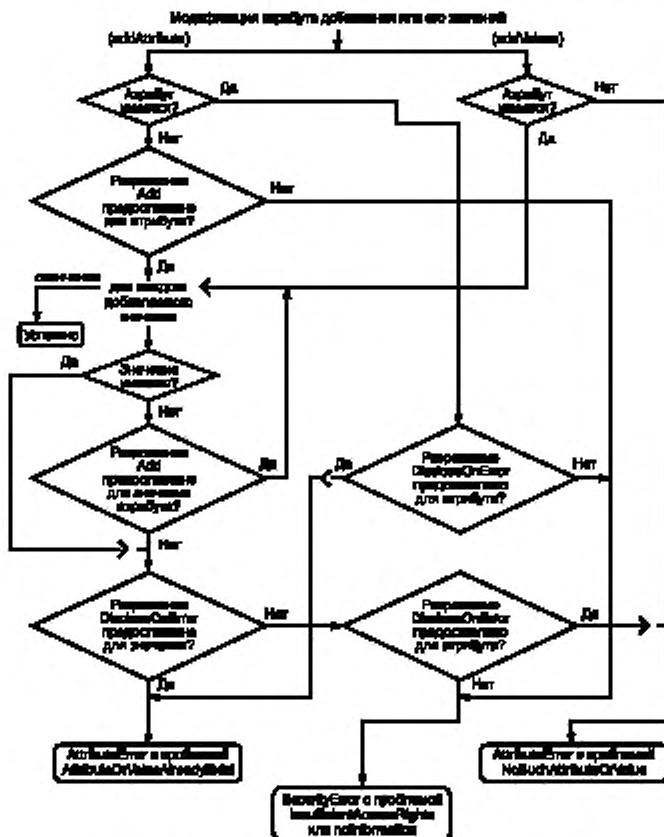


Рисунок В-16 — Модификация добавления атрибута или его значения

Ключевые слова: обработка данных, обмен информацией, взаимосвязь сетей, взаимосвязь открытых систем, справочники

Редактор В. П. Огурцов
Технический редактор Н. С. Громова
Корректор С. И. Фурсова
Компьютерная верстка А. Г. Хоменко

Илл. лист № 021007 от 10.08.95. Сдано в набор 20.05.98. Подписано в печать 10.08.98. Усл. печ. л. 8,98. Уч.-изд. л. 6,95.
Тираж 228 экз. С./Д. 5341. Зак. 420

ИПК Издательство стандартов, 107036, Москва, Колодезный пер., 14.
Набрано в Калужской типографии стандартов на ПЭВМ.
Калужская типография стандартов, ул. Московская, 256.
П.ЗР № 040138