ГОСТ Р ИСО/МЭК 10165-4—2001

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

ВЗАИМОСВЯЗЬ ОТКРЫТЫХ СИСТЕМ СТРУКТУРА ИНФОРМАЦИИ АДМИНИСТРАТИВНОГО УПРАВЛЕНИЯ

Часть 4

Руководство по определению управляемых объектов

Издание официальное

БЗЗ—2001/47

ГОССТАНДАРТ РОССИИ
Москва

gost.e-ecolog.ru

Предисловие

1 РАЗРАБОТАН Государственным научно-исследовательским и конструкторско-технологическим институтом «ТЕСТ» Министерства Российской Федерации по связи и информатизации

ВНЕСЕН Министерством Российской Федерации по связи и информатизации

2 ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 6 сентября 2001 г. № 376-ст

3 Настоящий стандарт содержит полный аутентичный текст международного стандарта ИСО/МЭК 10165-4:1992 «Информационная технология. Взаимосвязь открытых систем. Структура информации административного управления. Часть 4. Руководство по определению управляемых объектов» с учетом Изменения № 1 (1996 г.) и Дополнений № 1 (1996 г.), № 2 (1998 г.), № 3 (1998 г.)

4 ВВЕДЕН ВПЕРВЫЕ

© ИПК Издательство стандартов, 2001

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

Содержание

1 Область применения 1
2 Нормативные ссылки 2
3 Определения 2
4 Сокращения 4
5 Соглашения 4
6 Общие вопросы 4
7 Общие принципы определения управляемых объектов 11
8 Обозначения для определений управляемых объектов 17
9 Руководство по разработке эквивалентных модулей АСН.1:1994 и АСН.1:1990 38
10 Соглашения для АСН.1 и директив РОУО 43
Приложение А Примеры 48
Приложение В Руководство по применению Z при формализации поведения управляемых объектов 52

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационнаятехнология

ВЗАИМОСВЯЗЬОТКРЫТЫХСИСТЕМ СТРУКТУРАИНФОРМАЦИИАДМИНИСТРАТИВНОГОУПРАВЛЕНИЯ

Часть4

Руководствопоопределениюуправляемыхобъектов

Informationtechnology.OpenSystemsInterconnection.Structureofmanagementinformation. Guidelinesforthedefinitionofmanagedobjects

Датавведения2002—07—01

1Областьприменения

Настоящийстандартявляетсяруководствомдляразработчиковдругихстандартовирекомендаций,содержащихопределенияуправляемыхобъектов,которое:

а)обеспечиваетсогласованностьмеждуопределениямиуправляемыхобъектов;

б)гарантируетразработкуназванныхопределенийспособом,совместимымсостандартамиирекомендациямистандартовпоадминистративномууправлениюВОС;

г)уменьшаетдублированиеусилийразличныхрабочихгрупп,идентифицируяобщиеполезныекомпоновкидокументов,процедурыиопределения.

Внастоящемстандартеопределены:

а)взаимосвязимеждустандартамиирекомендациями,относящимисякадминистративномууправлениюВОС,иопределениямиклассовуправляемыхобъектов,атакжепринципыиспользованияэтихрекомендацийистандартоввопределенияхклассовуправляемыхобъектов;

б)методы,применяемыедляопределенияклассовуправляемыхобъектов,ихатрибутов,сообщений,действийиповедения,включая:

1)сводкувопросов,которыедолжныбытьотраженывопределении,

2)обозначения,которыерекомендуетсяиспользоватьвопределении,

3)руководствапосогласованности,которымимогутследоватьопределения;

г)взаимосвязимеждуопределениямиклассовуправляемыхобъектовипротоколамиадминистративногоуправленияито,чтотребуетсявотносящихсякпротоколуопределениях;

д)рекомендуемаяструктурадокументацииляопределенийклассовуправляемыхобъектов.

Настоящийстандартприменяетсядляразработкилюбыхрекомендацийистандартов,определяющих:

а)информациюадминистративногоуправления,котораядолжнабытьпереданаилиобработанаспомощьюпротоколаадминистративногоуправленияВОС;

б)управляемыеобъекты,ккоторымотноситсяэтаинформация.

Внастоящемстандартенеопределяютсяиненодразумеваются:

а)какие-либоограничениянаразработкуопределенийклассовуправляемыхобъектоввтерминахихфункциональныхвозможностей,настандартыирекомендации,которыенекмотносятся,илинаихиспользованиевконкретнойсредеадминистративногоуправления;

б)руководствопоопределениюресурсов;встандартеприводитсятолькоруководствопоопределениюуправляемыхобъектов,которыеобеспечиваютточкузренияадминистративногоуправлениянаресурсы.

2Нормативныессылки

Внастоящемстандартеиспользованыссылкинаследующиестандарты:

ГОСТРИСО/МЭК7498-1—99Информационнаятехнология.Взаимосвязьоткрытыхсистем. Базоваяэталоннаямодель.Часть1.Базоваяэталоннаямодель

Изданиеофициальное

ГОСТРИСО7498-3—97Информационнаятехнология.Взаимосвязьоткрытыхсистем.Базовая эталоннаямодель.Часть3.Присвоениеименииадресация

ГОСТРИСО/МЭК7498-4—99Информационнаятехнология.Взаимосвязьоткрытыхсистем. Базоваяэталоннаямодель.Часть4.Основыадминистративногоуправления

ГОСТРИСО/МЭК8824—93Информационнаятехнология.Взаимосвязьоткрытыхсистем.Спецификацияабстрактно-синтаксическойнотацияверсии1(АСН.1)

ГОСТ Р ИСО/МЭК 9595—99 Информационная технология. Взаимосвязь открытых систем. Определениеобщихуслугадминистративногоуправления

ГОСТ Р ИСО/МЭК 10040—99 Информационная технология. Взаимосвязь открытых систем. Основныеположенияадминистративногоуправлениясистемы

ГОСТРИСО/МЭК10164-1—99Информационнаятехнология.Взаимосвязьоткрытыхсистем. Административноеуправлениесистемы.Часть1.Функцииадминистративногоуправленияобъектом

ГОСТРИСО/МЭК10164-2—99Информационнаятехнология.Взаимосвязьоткрытыхсистем. Административноеуправлениесистемы.Часть2.Функцииадминистративногоуправлениясостоянием

ГОСТРИСО/МЭК10165-1—2001Информационнаятехнология.Взаимосвязьоткрытыхсистем.Структураинформацииадминистративногоуправления.Часть1.Модельинформацииадминистративногоуправления

ГОСТРИСО/МЭК10165-2—2001Информационнаятехнология.Взаимосвязьоткрытыхсистем.Структураинформацииадминистративногоуправления.Часть2.Определениеинформацииадминистративногоуправления

ИСО/МЭК 8824-1—98* Информационная технология. Абстрактная синтаксическая нотация версии1(АСН.1).Часть1.Спецификацияосновнойнотации

ИСО/МЭК9594-2—98*Информационнаятехнология.Взаимосвязьоткрытыхсистем.Справочник.Часть2.Общееописаниепринципов,моделейиуслуг

ИСО/МЭК9596-1—98Информационнаятехнология.Взаимосвязьоткрытыхсистем.Протокол информацииадминистративногоуправления.Часть1.Спецификацияпротокола

ИСО/МЭК9834-1—93*Информационнаятехнология.Взаимосвязьоткрытыхсистем.Процедурыработыполномочныхоргановрегистрацииво С.Часть1.Общиепроцедуры

ИСО/МЭК10164-3—93*Информационнаятехнология.Взаимосвязьоткрытыхсистем.Административноеуправлениесистемы.Часть3.Функцииадминистративногоуправленияивзаимоотношениями

ИСО/МЭК10164-4—93*Информационнаятехнология.Взаимосвязьоткрытыхсистем.Административноеуправлениесистемы.Часть4.Функцииуведомленияонештатныхситуациях

ИСО/МЭК11587—96Информационнаятехнология.Взаимосвязьоткрытыхсистем.Применениевконтекстесистемуправленияспроцессамитранзакции

3Определения

3.1Определениябазовойэталонноймодели

В настоящем стандарте использованы следующие термины, определенные в ГОСТ РИСО/МЭК7498-1:

-(N)-соединение;

-(N)-категория;

-(N)-уровень;

-(N)-пункт-доступа-к-услуге;

-открытаясистема;

-административноеуправлениесистемы.

*Оригиналы и проекты международных стандартов — во ВНИИКИ Госстандарта России.

3.2Определениянаименованияиадресации

Внастоящемстандартеиспользованследующийтермин,определенныйвГОСТРИСО/МЭК 7498-3:

-(N)-селектор.

3.3Определенияадминистративногоуправления

В настоящем стандарте использованы следующие термины, определенные в ГОСТ РИСО/МЭК7498-4:

-управляемыйобъект;

-операция(N)-уровня.

3.4Определенияадминистративногоуправлениясистемы

В настоящем стандарте использованы следующие термины, определенные в ГОСТ РИСО/МЭК10040:

-агент;

-родовыеопределения;

-классуправляемыхобъектов;

-информацияадминистративногоуправления;

-управляющий;

-протоколадминистративногоуправления(N)-уровня;

-сообщение;

-типсообщения;

-операция(административногоуправлениясистемы);

-(прикладной)протоколадминистративногоуправлениясистемы.

3.5Определениямоделиинформацияадминистративногоуправления

В настоящем стандарте использованы следующие термины, определенные в ГОСТ РИСО/МЭК10165-1:

-действие;

-фактическийкласс;

-атрибутивнаягруппа;

-идентификаторатрибута;

-типатрибутов;

-множествозначенийатрибута;

-поведение;

-характеристика;

-условныйпакет;

-вмещение;

-наследование;

-иерархиянаследования;

-управляемыйобъектначальныхзначений;

-реализация;

-обязательныйпакет;

-кратноенаследование;

-связываниеимен;

-пакет;

-параметр;

-множестводопустимыхзначений;

-относительноеотличающееимя;

-множествотребуемыхзначений;

-специализация;

-подкласс;

-суперкласс;

-старшийобъект;

-подчиненныйобъект.

3.6ОпределениеУОИУ

В настоящем стандарте используют следующие термины, определенные в ГОСТРИСО/МЭК9595:

а) атрибут;

б) услуги общей информации (административного) управления.

3.7ОпределениеАСН.1ИСО/МЭК8824-1:
а) идентификатор объекта;
б) тип «последовательность»;
в) тип «последовательность—из»;
г) тип множество;
д) тип «множество—из»;
е) подтип;
ж) тип;
и) имя ссылки на тип;
к) имя ссылки на значение.

3.8Дополнительныеопределения

3.8.1**определениеклассауправляемыхобъектов:**Наборопределенийатрибутов,операций,сообщенийиповедения,которомуприсвоеноимяклассауправляемыхобъектовизадокументированныйсиспользованиемшаблонаклассауправляемыхобъектовиодногоилинесколькихдругихшаблоновизопределенныхвнастоящемстандартетипов,накоторыепрямоиликосвенноссылается шаблонклассауправляемыхобъектов.Определениеклассауправляемыхобъектоввключаетвсебя всеэлементыопределения,наследуемыеотсуперкласса(ов)этогоклассауправляемыхобъектов,и всеэлементыопределения,образующиеспецификацию(ии)суперкласса(ов).

3.8.2**шаблон:**Стандартныйформатдлядокументацииопределенияэлементаинформациидминистративногоуправления.

3.8.3**классобъектовсправочника:**Классобъектов,определенныйвИСО/МЭК9594-2.

4Сокращения

Внастоящемстандартеиспользованысследующиесокращения:
АСН.1—абстрактнаясинтаксическаянотацияверсии1;
БДУ—блокданныхуслуги;
ВОС—взаимосвязьоткрытыхсистем;
ЗСУО—заявкаосоответствииуправляемомуобъекту;
КУ—качествоуслуги;
МФО—методыформальногоопределения;
ООИ—относительноеотличающееимя;
ПБД—протокольныйблокданных;
ПДУ—пунктдоступакуслуге;
ПК—подкомитет;
ПОИУ—протоколобщейинформации(административного)управления;
РГ—рабочаягруппа;
РОУО—руководствопоопределениюуправляемыхобъектов;
СИУ—структураинформации(административного)управления;
СТК—совместныйтехническийкомитет;
УО—управляемыйобъект;
УОИУ—услугиобщейинформации(административного)управления;
УОНЗ—управляемыйобъектначальныхзначений;
(N)-ПДУ—(N)-пункт-доступа-к-услуге;

5Соглашения

Внастоящемстандартешрифтом(полужирнымикурсивом)выделентекст,вкоторомиспользовананотацияАСН.1илиопределеннаявразделе8.

6Общиевопросы

6.1Целостностьвзаимосвязей

Приопределенииклассовуправляемыхобъектовважнорассмотретьситуации,когдаимеются требованиясогласованности,которыедолжныприменятьсякэкземплярамэтихклассов,например

ситуации, когда поведение управляемого объекта ограничено правилами, зависящими не только от состояния данного объекта, но и от состояний других управляемых объектов в системе. Любые такие ограничения должны быть выражены как поведение, связанное с рассматриваемыми классами управляемых объектов.

Частным случаем, в котором определения, связанные с экземпляром управляемого объекта, должны явным образом устанавливать правила согласования, является операция удаления delete. Для этой операции такие правила согласования задаются в связывании(иях) имен, ассоциированном(ых) с классом управляемых объектов. Результат операции delete должен быть определен таким образом, чтобы было ясно, при каких обстоятельствах удаление допустимо и какова последовательность удаления. В частности, связывание имен должно устанавливать, допустимо ли удаление экземпляра класса, когда еще существуют содержащиеся в нем управляемые объекты, и какие правила применяются в случаях, когда между удаляемым и прочими управляемыми объектами есть другие (не относящиеся к вмещению) взаимосвязи, как те, которые могут существовать вследствие наличия атрибутов взаимосвязи (см. ИСО/МЭК 10164-3). Применяемые для удаления правила согласованности должны быть такими, чтобы операция удаления не могла привести к несогласованным взаимосвязям. Хотя эти правила согласованности определяются как часть связывания имен, правила, которые применяются для удаления данного управляемого объекта, устанавливаются в момент реализации этого управляемого объекта.

6.2 Наследуемые характеристики

Процесс наследования приводит к включению всех характеристик суперкласса(ов) класса управляемых объектов в определение этого класса. Это правило применяется рекурсивно, завершаясь на вершине иерархии наследования, называемой «высшим классом». Следовательно, данное определение класса управляемых объектов включает в себя все характеристики, которые являются частью определения высшего класса, плюс все характеристики, добавленные в процессе определения тех подклассов высшего класса, которые образуют часть иерархии наследования этого класса управляемых объектов.

6.3 Факультативность

Во общем случае включение факультативных возможностей в определения классов управляемых объектов является нежелательным, так как по мере роста числа таких возможностей взаимодействие становится более трудным. Как установлено в ГОСТ Р ИСО/МЭК 10165-1, определение класса управляемых объектов может содержать условные пакеты, которые присутствуют в экземпляре этого класса, если выполнены заданные условия. Подразумевается, что условия, применяемые для этих пакетов, должны относиться к стандартным свойствам ресурса, который представляет класс управляемых объектов, или к факультативным функциям управления, поддерживаемым управляемой системой.

6.4 Регистрация

Процесс определения классов управляемых объектов требует присвоения глобально однозначных идентификаторов — идентификаторов объектов — различным составляющим класса управляемых объектов, таким как имя класса управляемых объектов, типы атрибутов и пр. Значения этих идентификаторов используются в протоколах административного управления для однозначной идентификации различных сторон управляемых объектов и связанных с ними атрибутов, операций и сообщений. Следовательно, для разработки определения класса управляемых объектов предварительно необходимо, чтобы органы по стандартизации или другие организации идентифицировали или установили подходящий метод регистрации, позволяющий создавать значения идентификаторов объектов. ИСО/МЭК 8824-1 устанавливает структуру идентификатора объекта и значения начальных дуг; дальнейшая информация об установлении и методов и уполномоченных по регистрации приведена в ИСО/МЭК 9834-1.

После того как элементу информации административного управления было присвоено значение идентификатора объекта требуется, чтобы никакой пересмотр определения этого элемента не изменял семантику информации. На практике это означает, что редакционные изменения зарегистрированных определений информации административного управления допускаются, но определения не должны изменяться так, что это будет видно в протоколе.

Все значения идентификаторов объектов, зарегистрированные в международных стандартах по административному управлению системы, размещаются под дугой

{joint-iso-ccittms(9)}

Распределение дуг ниже **{joint-iso-ccittms(9)}** определяется настоящим стандартом. Ниже **{joint-iso-ccittms(9)}** дуги распределяются на основе стандартов по административному управлению системы так, как показано в таблице 1.

Таблица 1 — Распределение дуг ниже

{joint-iso-ccittms(9)}

Дуга	Стандарт
smo(0)	Основные положения административного управления системы, ГОСТ Р ИСО/МЭК 10040
cmip(1)	Протокол общей информации административного управления, ИСО/МЭК 9596-1
function(2)	Функции административного управления системы, ГОСТ Р ИСО/МЭК 10164-1 и последующие части этого стандарта
smi(3)	Структура информации административного управления, ГОСТ Р ИСО/МЭК 10165-1 и последующие части этого стандарта
applicationContext(4)	Прикладные контексты, ИСО/МЭК 11587

Распределение дуг ниже этого уровня не определено 6.4.1—6.4.5. Дуги, которые потребуются для последующих стандартов по административному управлению системы, будут вводиться по мере необходимости в виде дополнения к настоящему стандарту.

Примечание — Схема распределения значений идентификаторов объектов, описанная в настоящем подразделе, применяется только для значений идентификаторов объектов в стандартах по административному управлению системы, разработанных совместно РГ4 ПК21 СТК1 ИСО/МЭК и МСЭ-Т. Если другие органы организациям по стандартизации необходимо в ходе разработки стандартов по административному управлению системы распределять значения идентификаторов объектов, они должны установить свои собственные схемы распределения ниже соответствующего уполномоченного по регистрации. Структура, принятая при разработке таких стандартов, может служить в качестве примера того, как устанавливается подходящая схема распределения значений, но окончательный выбор схемы от вечает соответствующая организация. Для обеспечения читаемости значений идентификаторов объектов рекомендуется использовать именную и числовую формы представления значений идентификаторов объектов, как определено в ИСО/МЭК 8824-1.

6.4.1 Распределение идентификаторов объектов для основных положений административного управления системы

Примечание — Выделение этих дуг определяется основными положениями административного управления системы. Здесь они приводятся только в справочных целях.

Ниже **{joint-iso-ccitt ms (9) smo (0)}** выделены дуги для регистрации идентификаторов прикладных контекстов, абстрактных синтаксисов и модулей АСН.1, приведенные в таблице 2.

Таблица 2 — Распределение дуг ниже

{joint-iso-ccittms(9)smo(0)}

Дуга	Назначение
applicationContext(0)	Распределение идентификаторов прикладных контекстов
negotiationAbstractSyntax(1)	Распределение идентификаторов версий договорных абстрактных синтаксисов
asn1Modules(2)	Распределение идентификаторов модулей АСН.1

Ниже **{joint-iso-ccittms(9)smo(0)applicationContext(0)}**, как установлено в ГОСТ Р ИСО/МЭК 10040, выделены дуги для регистрации идентификаторов конкретных прикладных контекстов, приведенные в таблице 3.

Таблица 3 — Распределение дуг ниже

{joint-iso-ccittms(9)smo(0)applicationContext(0)}

Дуга	Назначение
systems-management(2)	Идентификация прикладных контекстов административного управления системы

Ниже **{joint-iso-ccittms(9)smo(0)negotiationAbstractSyntax(1)}**, как установлено в ГОСТРИСО/МЭК10040, выделены дуги для регистрации конкретных версий договорных абстрактных синтаксисов, приведенные в таблице 4.

Таблица 4 — Распределение дуг ниже

{joint-iso-ccittms(9)smo(0)negotiationAbstractSyntax(1)}

Дуга	Назначение
version(1)	Идентифицирует версию 1 договорного абстрактного синтаксиса

Ниже **{joint-iso-ccittms(9)smo(0)asn1Modules(2)}**, как установлено в ГОСТРИСО/МЭК 10040, выделены дуги для регистрации идентификаторов конкретных модулей АСН.1, приведенные в таблице 5.

Таблица 5 — Распределение дуг ниже

{joint-iso-ccittms(9)smo(0)asn1Modules(2)}

Дуга	Назначение
negotiationefinitions(0)	Распределение идентификаторов версий для модулей АСН.1, которые содержат определения, связанные с договорным абстрактным синтаксисом

Ниже **{joint-iso-ccittms(9)smo(0)asn1Modules(2)negotiationefinitions(0)}**, как установлено в ГОСТРИСО/МЭК10040, выделены дуги для регистрации конкретных версий модулей АСН.1, приведенные в таблице 6.

Таблица 6 — Распределение дуг ниже

{joint-iso-ccittms(9)smo(0)asn1Modules(2)negotiationefinitions(0)}

Дуга	Назначение
version1(1)	Идентифицирует версию 1 модуля АСН.1, который содержит определения, связанные с договорным абстрактным синтаксисом

6.4.2 Распределение идентификаторов объектов для ПОИУ

Примечание — Выделение этих дуг определяется ПОИУ. Здесь они приводятся только в справочных целях. Версия 1 ПОИУ устарела и заменена версией 2. Версия 1 была определена ИСО/МЭК 9596-1 и имела аналогичной рекомендации МККТТ.

Ниже **{joint-iso-ccittms(9)cmip(1)}** выделены дуги для каждой версии ПОИУ так, как описано в 6.4.2.1 и 6.4.2.2.

6.4.2.1 ПОИУ версии 1

Ниже **{joint-iso-ccittms(9)cmip(1)}** выделены дуги для версии 1 ПОИУ так, как показано в таблице 7.

Таблица 7 — Распределение дуг ниже

{joint-iso-ccittms(9)cmip(1)} для ПОИУ версии 1

Дуга	Назначение
version1(1)	Распределение идентификаторов объектов для ПОИУ версии 1

Ниже **{joint-iso-ccittms(9)cmip(1)version1(1)}** для целей, описанных в ИСО/МЭК 9596-1, выделены дуги так, как показано в таблице 8.

Таблица8 — Распределениедугниже

{joint-iso-ccittms(9)cmip(1)version1(1)}

Дуга
aAssociateUserInfo(1)
aAbortUserInfo(2)
protocol(3)
abstractSyntax(4)

6.4.2.2ПОИУверсии2

Ниже**{joint-iso-ccittms(9)cmip(1)}**выделеныдугидляверсии2ПОИУтак,какпоказанов таблице9.

Таблица9— Распределениедугниже

{joint-iso-ccittms(9)cmip(1)}дляПОИУверсии2

Дуга	Назначение
modules(0)	Распределениеидентификаторовобъектовдлямодулей АСН.1ПОИУ
cmip-pci(1)	Распределение идентификаторов объектов для управляющей информации ПОИУ

Ниже**{joint-iso-ccittms(9)cmip(1)modules(0)}**дляцелей,описанныхвИСО/МЭК9596-1, выделеныдугитак,какпоказановтаблице10.

Таблица10 — Распределениедугниже

{joint-iso-ccittms(9)cmip(1)modules(0)}

Дуга
aAssociateUserInfo(1)
aAbortUserInfo(2)
protocol(3)

Ниже**{joint-iso-ccittms(9)cmip(1)cmip-pci(1)}**дляцелей,описанныхвИСО/МЭК9596-1, выделеныдугитак,какпоказановтаблице11.

Таблица11 — Распределениедугниже

{joint-iso-ccittms(9)cmip(1)cmip-pci(1)}

Дуга
reserved1(1)
reserved2(2)
reserved3(3)
abstractSyntax(4)

6.4.3Распределениеидентификаторовобъектовдлястандартовфункций

Ниже**{joint-iso-ccittms(9)function(2)}**выделеныдугидляидентификациистандартовфункцийтак,какпоказановтаблице12.

Таблица 12 — Распределение дуг ниже

{joint-iso-ccittms(9)function(2)}

Дуга	Стандарт
partX(X)	(ГОСТ Р) ИСО/МЭК 10164-X, где X — номер части

Дуги ниже **{joint-iso-ccittms(9)function(2)partX(X)}** показаны в таблице 13.

Таблица 13 — Распределение дуг ниже

{joint-iso-ccittms(9)function(2)partX(X)}

Дуга	Назначение
standardSpecificExtension(0)	Специфические для стандарта расширения схемы распределения
functionalUnitPackage(1)	Распределение идентификаторов пакетов функциональных блоков
asn1Modules(2)	Распределение идентификаторов модулей АСН.1
managedbjectClass(3)	Распределение идентификаторов классов управляемых объектов
package(4)	Распределение идентификаторов пакетов
parameter(5)	Распределение идентификаторов параметров
nameBinding(6)	Распределение идентификаторов связываний имен
attribute(7)	Распределение идентификаторов атрибутов
attributeGroup(8)	Распределение идентификаторов атрибутивных групп
action(9)	Распределение типов действий
notification(10)	Распределение типов сообщений
relationshipClass(11)	Распределение идентификаторов классов управляемых взаимосвязей
relationshipMapping(12)	Распределение идентификаторов отображений взаимосвязей
relationshipRole(13)	Распределение идентификаторов ролей взаимосвязей

В любом стандарте функций могут быть выделены дополнительные дуги ниже этого уровня (например для распределения идентификаторов конкретных атрибутов).

6.4.4 Распределение идентификаторов объектов для стандартов СИУ

Ниже **{joint-iso-ccittms(9)smi(3)}** выделены дуги для идентификации стандартов СИУ так, как показано в таблице 14.

Таблица 14 — Распределение дуг ниже

{joint-iso-ccittms(9)smi(3)}

Дуга	Стандарт
partX(X)	(ГОСТ Р) ИСО/МЭК 10165-X, где X — номер части

Дуги ниже **{joint-iso-ccittms(9)smi(3)partX(X)}** показаны в таблице 15.

В любом стандарте функций могут быть выделены дополнительные дуги ниже этого уровня (например для распределения идентификаторов конкретных атрибутов).

Таблица 15 — Распределение дуг ниже

{joint-iso-ccittms(9)smi(3)partX(X)}

Дуга	Назначение
standardSpecificExtension(0)	Специфические для стандарта расширения схемы распределения
asn1Modules(2)	Распределение идентификаторов модулей АСН.1
managedbjectClass(3)	Распределение идентификаторов классов управляемых объектов
package(4)	Распределение идентификаторов пакетов
parameter(5)	Распределение идентификаторов параметров
nameBinding(6)	Распределение идентификаторов связываний имен
attribute(7)	Распределение идентификаторов атрибутов
attributeGroup(8)	Распределение идентификаторов атрибутивных групп
action(9)	Распределение типов действий
notification(10)	Распределение типов сообщений
relationshipClass(11)	Распределение идентификаторов классов управляемых взаимосвязей
relationshipMapping(12)	Распределение идентификаторов отображений взаимосвязей
relationshipRole(13)	Распределение идентификаторов ролей взаимосвязей

6.4.5 Идентификатор объекта для фактического класса

Значение идентификатора объекта

{joint-iso-ccittms(9)smi(3)part4(4)managedbjectClass(3)actualClass(42)}

присвоено настоящим стандартом для выражения семантики «фактический класс», определенной в ГОСТРИСО/МЭК10165-1. Когда это значение использовано для спецификации базового класса управляемых объектов в запросе операции УОИУ, оно указывает, что получатель операции административного управления системы должен ответить как к какчлену своего фактического класса. Управляемый объект идентифицирует свой фактический класс (см. 7.4.3) с помощью значения своего атрибута «класс управляемого объекта».

6.5 Соответствие

Общие требования соответствия, относящиеся к стандартам информации административного управления, установлены в ГОСТРИСО/МЭК10040.

6.6 Сложность определений управляемых объектов

В процессе моделирования должна быть минимизирована сложность определений управляемых объектов. В любом случае операции административного управления не должны быть более сложными, чем соответствующие свойства участвующей в операции среды ВОС.

6.7 Создание и удаление управляемого объекта

Создание и удаление экземпляров управляемых объектов может происходить следующими методами:

- управляемые объекты могут быть созданы и удалены в результате взаимодействий протокола административного управления. Для этих целей определены операции создания и удаления с соответствующей семантикой;

- управляемые объекты могут быть созданы и удалены в результате операций ресурса, к которому они относятся, обычно — протокольного автомата. В этом случае операции создания и удаления не могут быть определены. Примером является представление соединений для целей административного управления;

- управляемые объекты могут быть созданы и удалены другими способами. В этом случае не определены операции создания и удаления. Примером является управляемый объект, который всегда автоматически создается при инициализации части оборудования и который не может быть удален с помощью административного управления.

Выбор одного из трех перечисленных методов создания управляемого объекта может отличаться от выбора метода удаления.

В одних случаях может существовать единственный метод, с помощью которого управляемый объект конкретного класса может быть создан и удален. В других случаях может оказаться возможным создание и удаление управляемых объектов конкретного класса разными методами.

6.7.1 Управляемые объекты начальных значений

При создании управляемого объекта может оказаться желательной возможность присвоить значения по умолчанию, которые сами подвержены изменениям в результате операций административного управления. Это может быть достигнуто путем спецификации управляемого объекта начальных значений (УОНЗ), атрибуты которого могут изменяться операциями административного управления и который может обеспечивать значения по умолчанию для соответствующих атрибутов создаваемых экземпляров других классов управляемых объектов.

При создании нового управляемого объекта с использованием УОНЗ, значения атрибутов в УОНЗ используются в качестве начальных значений соответствующих атрибутов в новом управляемом объекте. Определение класса управляемых объектов может устанавливать, как выбирается УОНЗ. Спецификация УОНЗ должна определять обстоятельства, при которых он предоставляет начальные значения, как он предоставляет эти начальные значения и к каким атрибутам применимы предоставляемые им начальные значения.

Когда для изменения атрибутов УОНЗ используются операции административного управления, атрибуты созданных ранее с использованием этого УОНЗ управляемых объектов не изменяются. Аналогично операции административного управления, осуществляемые над атрибутами управляемых объектов, созданных с использованием УОНЗ, не влияют на атрибуты УОНЗ.

6.7.2 Источники начальных значений атрибутов

Начальные значения атрибутов управляемых объектов, используемые во время создания, получаются из нескольких источников, как определено в ГОСТРИСО/МЭК 10165-1. Когда атрибут представляет конкретное значение, которое должно быть согласованным с нижележащим ресурсом, это значение является обязательным.

7 Общие принципы определения управляемых объектов

Описанные ниже общие принципы являются руководством для авторов определений управляемых объектов и должны способствовать согласованности между этими определениями; поэтому авторам определений управляемых объектов рекомендуется по мере возможности следовать данному руководству.

7.1 Общность

Авторы определений управляемых объектов должны стараться идентифицировать и использовать в качестве основы:

- общие классы управляемых объектов, определенные в международных стандартах;
- общие классы управляемых объектов и другие свойства, определенные в ГОСТРИСО/МЭК 10165-2.

Авторы определений управляемых объектов должны также стараться рассматривать и повторно использовать определения, разработанные в других рабочих группах, для увеличения общности определений. Эта цель может быть достигнута путем разработки моделей управления, являющихся общими для нескольких групп определений управляемых объектов.

7.2 Задачи управления

Определения классов управляемых объектов и их компонентов должны полностью удовлетворять требованиям, относящимся к конкретным целям административного управления. Такие требования, вероятно, включают в себя административное управление парными аспектами протокола операций уровня и/или по уровня и различные проблемы на границе услуги, не оговоренные специально поставщиком нижележащих услуг (например качество нижележащей услуги может не соответствовать приемлемому уровню). В ходе разработки определений классов управляемых объектов важно сохранять техническую обоснованность для каждой цели административного управления. Для

объяснениятого,какаждыйкомпонентопределенияинформациидминистративногоуправления (например классы управляемых объектов, атрибуты, операции, сообщения и пр.) соотносится с этойтехническойобоснованностью,следуетиспользоватькомментарии.

Вопросы,существенныедляадминистративногоуправления,должныбытьзафиксированыв управляемых объектах, представляющих ресурсы, к которым относятся эти вопросы, т. е., если должен быть определен управляемый объект, представляющий конкретный ресурс (например соединение), то информация, относящаяся к этому ресурсу, должна быть отражена в соответствующем(их) управляемом(ых) объекте(ах) и нигде более.

7.3Структурирование

Дляпредставленияструктурыуправляемыхобъектовимеетсярядспособов,позволяющихотразитьгруппированиеданныхилифункциональныхвозможностей.Каждаяизэтихспособовимеет своипреимуществаинедостатки;выборнаиболееподходящихспособовдляконкретнойспецификациизависитотрядаописанныхнижекритериев.

Способыструктурирования,описанныевГОСТРИСО/МЭК10165-1,включаютвсебя:

-атрибутивныегруппы;

-подклассы(специализацию);

-кратноенаследование;

-вмещаемыеуправляемыеобъекты;

-пакеты.

Могут быть определены группы атрибутов, операций и сообщений, которые могут присутствоватьилиотсутствоватьвзависимостиотстандартизованныхусловий,такихкаквыборконкретныхоперацийвбазовомстандарте.Такиегруппыфункциональныхвозможностейприсутствуютили отсутствуюткакединоецелое.Группыфункциональныхвозможностеймогутвозникатьвследствие факультативноговыборадляресурсаизстандартаслоя(напримеробеспечениетранспортныхуслуг класса4),приводящегокдополнительнымтребованиямадминистративногоуправленияидополнительнымвозможностям,иливследствиеобеспеченияопределенныхфункцийадминистративногоуправления(напримеручета).Этигруппыфункциональныхвозможностейопределяютсяспомощьюусловныхпакетов,предоставляемыхшаблономклассауправляемыхобъектов.

Однимизважныхкритериеввыбораспособаструктурированияявляетсястатическоеилидинамическоеприсутствиегруппы.Еслиприсутствиегруппыфиксированонамоментспецификации, топодходящимспособомможетоказатьсяиспользованиеатрибутивныхгрупп,подклассов,кратногонаследованияиливмещаемыхуправляемыхобъектов.Еслиприсутствиефиксированонамомент реализацииилиустановки,томожетоказатьсяподходящимиспользованиевмещаемыхуправляемыхобъектовилиусловныхпакетов.Еслиприсутствиегруппыможетизменятьсязавремяжизни вмещающих(илиинкапсулирующих)управляемыхобъектов,томожетоказатьсяподходящимиспользованиевмещаемыхуправляемыхобъектов,которыединамическисоздаютсяиуничтожаются.

Другимкритериемявляетсяналичиенесколькихэкземпляровгруппывуправляемомобъекте.В этомслучаеподходитиспользованиевмещаемыхуправляемыхобъектов;впротивномслучаеможет использоватьсялюбойизпятиперечисленныхметодов.

7.4Управляемыеобъекты

7.4.1 Реализация суперклассов

Дляобеспеченияобщейосновы,накоторойспециализируютсяподклассы,могутбытьопределеныклассыуправляемыхобъектов,которыеникогданереализуются.Напримерможетбытьопределен родовой класс управляемых объектов «виртуальная цепь», подклассами которого могут бытьпостоянныеипереключаемыевиртуальныецепи.

Внекоторыхслучаях,вчастности,когдаподклассыопределяютсядляпересмотрастандарта, могутсуществоватьсуперклассы,экземплярыкоторыхмогутбытьреализованы.

7.4.2 Неограниченные суперклассы

Правила наследования ограничивают способы, которыми могут быть изменены множества допустимыхитребуемыхзначенийатрибутовклассауправляемыхобъектовприопределенииподкласса этого класса. Точно также правила ограничивают возможности добавления параметров к действиямисообщениям.Этиограничениягарантируютсовместимостьподклассассуперклассом.

По этой причине при определении класса управляемых объектов, который, как ожидается, можетбытьсуперклассомпоследующихклассовуправляемыхобъектов,полезнообеспечитьподобногородарасширения.Хотяневсерасширенияможнопредвидетьиобеспечить,следующиеприемыоставляютширокиевозможностидлярасширенийприсохраненииисовместимости:

- синтаксис (тип) каждого атрибута следует определять таким образом, чтобы включить в него все значения, которые разумно могут быть согласованы с семантикой атрибута, даже если некоторые из этих значений не являются немедленно необходимыми или желательными;

- следует обеспечивать возможности расширения в каждом определении действия или сообщения;

- следует определять «неограниченный суперкласс», который включает в себя все элементы без каких-либо ограничений, в качестве основы для определения более ограниченных подклассов. Для атрибутов это означает пустое множество требуемых значений и множество допустимых значений, равное синтаксису атрибута;

- следует определять конкретные подклассы этого неограниченного суперкласса, которые устанавливают требуемые ограничения на атрибуты, действия и сообщения.

Авторы определений управляемых объектов могут обеспечивать возможности расширения только для некоторых из атрибутов, действий и сообщений неограниченного суперкласса.

7.4.3 Фактический класс

Определение класса управляемых объектов состоит из шаблона **MANAGE BJECT CLASS** (см. 8.3), зарегистрированного со значением идентификатора объекта для этого класса, набора шаблонов, на которые ссылается данный шаблон, и всех шаблонов, на которые ссылаются шаблоны этого набора.

Управляемый объект идентифицирует свой фактический класс с помощью значения атрибута «класс управляемого объекта», которое является значением идентификатора объекта, использованного для регистрации его шаблона **MANAGE BJECT CLASS**. Каждый управляемый объект:

- обеспечивает все характеристики, определенные для его фактического класса, в соответствии с присутствующими пакетами;

обеспечивает только те операции, которые определены в его фактическом классе для присутствующих пакетов;

- создает сообщения только тогда, когда поведение, определенное для переключателей этих сообщений в фактическом классе, применяется к присутствующим пакетам.

Отсутствие каких-либо конструкций РОУО для характеристик в определении класса управляемых объектов исключает эти характеристики из определения класса. Подкласс может добавить исключенную конструкцию явным определением. Каждый подкласс имеет свое собственное зарегистрированное значение идентификатора объекта. Например, если свойство REPLACE не задано для однозначного атрибута, то этот атрибут в экземпляре данного класса должен рассматриваться как доступный только для чтения; определение подкласса может расширить исходный класс, добавив конструкцию REPLACE для спецификации того, что атрибут может быть заменен в экземплярах подкласса и в экземплярах, совместимых с этим подклассом.

7.5 Атрибуты

7.5.1 Множества значений атрибутов

В некоторых случаях факультативные возможности базового стандарта позволяют изменять множество значений атрибута в соответствии с выбором реализации. Типичным примером является случай, когда базовый стандарт допускает широкий диапазон размеров пакетов, но соответствующая ему реализация может поддерживать более ограниченный диапазон. В такой ситуации определение поведения атрибута должно идентифицировать имеющиеся возможности.

Может оказаться необходимым определить вырожденные значения (null) как допустимые значения атрибута или, в случае атрибутов УОНЗ, определить значения атрибута с присвоенной им конкретной семантикой, такой как «создать управляемый объект со значением null для соответствующего атрибута» или «игнорировать этот атрибут как источник начального значения». Методы определения таких значений включают в себя определение абстрактного синтаксиса с выборочного типа, когда один выбор определяет нормальное множество значений атрибута, а другой — значения с присвоенной конкретной семантикой.

Определение множества допустимых значений атрибута может быть получено разными способами, включая:

- статическое определение множества значений атрибута, являющееся частью определения класса управляемых объектов;

- определение второго атрибута, значение которого указывает множество значений, которые может принимать рассматриваемый атрибут.

Первый из этих методов минимизирует число определений атрибутов, связанных с классом управляемых объектов; однако если требуется несколько вариантов атрибута, то второй метод позволяет избежать определения нескольких подклассов для обработки каждого возможного варианта множества значений.

7.5.2 Типы атрибутов

Структурированные атрибуты, в определении синтаксиса которых в качестве базового используют тип «последовательность», «последовательность-из» или «множество», должны использоваться только тогда, когда не требуется изменений отдельных элементов атрибута, т.к. эти типы АСН.1 соответствуют типам атрибутов с единственным значением. Когда необходимо обращаться к нескольким атрибутам вместе, обеспечивая при этом возможность работать с каждым из них по отдельности, могут быть определены атрибутивные группы и, при необходимости, могут быть использованы определения действий и поведения для установления любых зависимостей между членами группы.

Примечание — Не подразумевается, что есть особая спецификация поведения для самой атрибутивной группы, которая не применяется к атрибутам, рассматриваемым по отдельности.

7.6 Взаимосвязи значений атрибутов

Значение атрибута может быть ограничено некоторыми функциями других значений атрибутов. Все взаимосвязи такого рода должны быть идентифицированы.

Когда значение атрибута ограничено другими атрибутами в том же самом управляемом объекте, могут существовать требования синхронизации для операции управления, в соответствии с которыми изменения значений одного или нескольких связанных атрибутов, приводящие к недопустимым значениям в связанных атрибутах, вызывают отказ. Если такие требования существуют, то они должны быть задокументированы как часть определения поведения класса управляемых объектов.

Когда значение атрибута ограничено другими атрибутами в разных управляемых объектах (если требования синхронизации также существуют), это должно быть задокументировано в поведении, связанном с классом управляемых объектов. В этом случае, когда все управляемые объекты находятся в одной и той же управляемой системе и один атрибут может изменять одна операция управления, требования реализуются через возможность элементарной межобъектной синхронизации УОИУ.

Общие проблемы синхронизации между несколькими операциями управления, между разными атрибутами в разных управляемых объектах или между несколькими управляемыми системами не могут быть решены с помощью современных протоколов административного управления систем.

7.7 Моделирование ПДУ

Существует общее требование — представлять как часть связанной с слоем структуры управляемого объекта, взаимосвязи между (N)-категориями, (N)-селекторами и (N+1)-категориями. Возможны различные решения, например:

- моделирование взаимосвязи как информации, содержащейся в управляемых объектах (N+1)-уровня;
- моделирование взаимосвязи как информации, содержащейся в управляемых объектах (N)-уровня;
- моделирование взаимосвязи как информации, содержащейся в управляемых объектах, которые не относятся ни к какому уровню, например в управляемых объектах, общих для всех уровней.

Настоящим стандартом рекомендуется второй из перечисленных подходов. В частности, (N)-ПДУ должны быть представлены отдельными управляемыми объектами, которые имеют в качестве атрибутов адреса (и другую информацию), вместе с взаимозаменяемыми атрибутами, указывающими на управляемые объекты (N)- и (N+1)-категории, ассоциированные с (N)-ПДУ. Для реализации требования согласованности селекторов, необходимых для недвусмысленной адресации ВОС, рекомендуется, чтобы управляемые объекты (N)-ПДУ содержались в управляемых объектах, соответствующих (N)-категориям, с которыми они связаны.

Примечание — Названное выше требование согласованности состоит в том, что адрес (N)-категории вместе с (N)-селектором должен однозначно идентифицировать (N+1)-категорию (или набор (N+1)-категорий одного и того же типа). Принимая, что это требование равносильно требованию однозначности для присвоения значений (N)-селекторов в контексте данной (N)-категории, обеспечение требования согласованности может быть более просто достигнуто, если информация селектора обеспечивается (N)-, а не (N+1)-категорией.

7.8Статистика

7.8.1 Согласованность

Авторыопределенийуправляемыхобъектовдолжныстаратьсядостичьсогласованностисостатистикойуровней, принимаяпринципы ГОСТРИСО/МЭК7498-1; вчастности, понятие«регистрируемаяинформация»относитсякинформации,рассматриваемойприадминистративномуправлениичерезуправляемыеобъекты, представляющиересурсы, ккоторымотносится информация.

Кандидатамивхарактеристики(N)-слоя,статистикакоторыхможетрегистрироваться,относятся:

-локальныеошибки;

-успешныйравноправныйобмен;

-взаимныеотказы,

-отказывуслуге.

Например,применениеопределенныхвышепринциповксоединению,определенномувГОСТРИСО/МЭК7498-1,приводиткидентификацииследующихосновныххарактеристик:

-числоустановленныхсоединений(N)-категорийсдругимипарнымикатегориями(N)-слоя;

-числолокальныхотказовприустановленныхсоединениях(N)-категорий;

-числоотказовпривзаимныхсогласованияхдлясоединений(N)-категорий,

-числоотвергнутых(N—1)-соединенийспоставщикаминижележащихуслуг.

Этотнаборстатистическихxарактеристикобеспечиваетсогласованныйвзгляднато,чтопроисходитнакаждомуровне(вслучае, ориентированномнасоединении)бездублированияcчетчиков.

Примечание—Аналогичныемоделитребуютсядляошибок, разрывовсоединенийипр.

7.8.2 Счетчики ПБД

АвторыопределенийуправляемыхобъектовдолжныспецифицироватьсчетчикиПВД(N)-уровня(иоктетовПБД), анеБДУ(иоктетовБДУ).

Примечание—ВероятнотребуетсяподсчитыватьтолькочислооктетовПБДнаограниченномчисле(N)-уровней.

7.8.3 Перекрытия

Авторыопределенийуправляемыхобъектовдолжныстаратьсядостигнутьсогласованностиизбегатьизлишнегодублированияилиперекрытиястатистическойинформации.Например,обеспечиваяподсчетзапросовотправкипервогоПБДизапросовотправкиответногоПБД,необязательноувеличиватьобасчетчикаодновременно.Суммаэтихсчетчиковвсегдаравнаобщейотправке ПБД.

7.8.4 Непереустанавливаемые счетчики

Рекомендуются непереустанавливаемые счетчики, так как они допускают многократное наблюдение безнеобходимости сложныхмеханизмоввзаимнойблокировки, связанныхскоординациейпереустановки.

7.8.5 Счетчики событий

Должен быть обеспечен подсчет событий административно управляемого ресурса, которые приводятксозданиюсообщений,т.к.созданиеУОИУМ—EVENT—REPORTможетбытьподавлено опережающимидискриминаторамисобытий.

7.9Счетчики

Дляадминистративногоуправлениясчетчикамидолжныбытьизвестныихмодули,т.к.впротивном случае управляющий не сможет определить, когда сбрасывается счетчик. Следовательно, имеются, покрайнеймере, четыре возможности определения счетчиков:

-счетчикиникогданесбрасываются;

-модулификсированыкакчастьопределенияклассауправляемыхобъектов;

-модулиопределяютсясоответствующимиатрибутами;

-модулиопределяютсяреализациейиспецифицированывЗСУО.

Примечание—Дляклассовуправляемыхобъектов, определенныхСТК1ИСО/МЭКдляуровней 1—4, принятподход, прикоторомсчетчикиникогданесбрасываются.

7.10Таймеры

Преимуществамогутбытьполученыпринаследованииобщейспецификацииточности,скоторойсистемыдолжнысохранятьзначенияатрибутовтаймеров,используемыхпривзаимодействи-

яхадминистративногоуправления.Взаимосвязимеждузначениямиэтихатрибутовифактическими операциямитаймеровввпротоколахдокументируютсявописанииповедения.

Примечание—Дляклассовуправляемыхобъектов,определенныхСТК1ИСО/МЭКдляуровней 1—4,сцельюобеспечениядостаточнобольшогодиапазонабезэкстраординарнойточностидлявыражения значенийтаймеровиспользованопредставлениесплавающейточкойсмантиссойдлиной16битиэкспонентойдлиной до16бит.(Этонеподразумевает,чтодолжнаиспользоватьсяарифметикасплавающейточкой). Системыдолжныбытьвсостоянииисохранятьзначенияэтойточностью.Допускаядругиеограничения,должнобытьпринятотребованиеустанавливатьатрибуттаймерасэтойточностью.

7.11Обновлениеатрибутов

Авторыопределенийклассовуправляемыхобъектовдолжныгарантировать,чтоприопределении атрибутов,которыемогутобновлятьсяоперациямиуправленияиобычнымиоперациямиресурса,определенырезультатыконкурирующихобновлений.Вчастности,результатоперациизамены значенияатрибутаможетбытьпотерян,еслиресурсобновляеттотжесамыйатрибут.

7.12Точностьатрибутов

Управляющаясистемаможетпопытатьсяустановитьзначениеатрибутасбольшейточностью, чемобеспечиваетсяуправляемойсистемой.Такиевысокоточныезначениямогутбытьаппроксимированыближайшимзначениемустановленнойточности.

7.13Идентификацияуправляемогообъекта

Каждоеопределениеклассауправляемыхобъектов,экземплярыкоторогомогутсуществовать, должновключатьвсебя,покрайнеймере,одинатрибут,пригодныйдляиспользованиявкачестве именующегоатрибутауправляемогообъекта.Подходящийатрибутявляетсяобязательнымиможет бытьпроверенноравенство;егосемантикадолжнадопускатьсохранениефиксированногозначения напротяжениивременижизникаждогоуправляемогообъекта,использующегоатрибутдлянаименования;егоидентификаторизначениедолжныоднозначноидентифицироватьуправляемыйобъект средивсехдругих,поименованныхтемжесамымстаршимобъектом.

Приудаленииуправляемогообъектазначение,присвоенноеегоименующемуатрибуту,становитсядоступнымдляповторногоиспользованиясцельюидентификацииуправляемыхобъектов, создаваемыхвдальнейшемвтомжесамомстаршемобъекте.

Еслинеобходимогарантировать,чтоэкземплярклассауправляемыхобъектовпослеудаления остаетсяотличимымотвсехдругихэкземпляровэтогокласса,тоследуетопределитьдополнительныйатрибут,входящийвопределениеклассауправляемыхобъектов,—атрибутоднозначнойидентификации,—семантикакоторогообеспечиваетоднозначнуюидентификациюовремени.Другие классыуправляемыхобъектовнеобязанысодержатьатрибутоднозначнойидентификации.

Атрибутоднозначнойидентификациидолженбытьдоступентолькодлячтенияи,когдаон входитвуправляемыйобъект,долженвключатьсявсообщения,создаваемыеэтимобъектом.

7.14Сообщения

7.14.1 Отказуслуги

Недолжнысоздаватьсясообщения,относящиесякотказамнижележащейуслуги,такккакза сообщенияолюбыхпричинахтакогоненормальногозавершениядолженнестиответственность управляемыйобъект,представляющийнижележащуюуслугу.Этоусловиедолжнопредотвратить распространениевверхпоуровнямненормальногозавершенияигенерациюложныхсообщений.

7.14.2 Сохранениеинформации

Сообщениясодержатинформациюособытии,которуюиначемоглабытьпотерена.Например:

- заголовокполяполученногоПБД,длякоторогобылаобнаруженаошибкапротокола;
- статистическиеданныеосоединении,котороедолжнобытьзавершено;
- время,втечениекоторогопроисходитпоследовательностьконкретныхсобытий.

7.15Использованиеопераций

Определениеклассауправляемыхобъектовдолжновключатьвсебясоответствующиеоперации.Длявызововуправляемойсистемойдолжныбытьопределенысообщения.Длявызововуправляющейсистемойоперацииспецифицируютсявсоответствиисихнепосредственнымвлияниемна управляемыеобъектывуправляемойсистемеследующимобразом:

а)еслинепосредственнымрезультатомявляетсясозданиеэкземпляраклассауправляемыхобъектов,тоиспользуетсяоперацияCreate.ОперацияCreateнеиспользуется:длясложныхдействий, которыетребуютскоординированногосозданиянесколькихуправляемыхобъектов;когдауправляемыйобъектсоздаетсякакпобочныйрезультатизмененияругогоуправляемогообъекта;когдауправляемыеобъектысоздаютсяврезультатеизмененияcостоянияругогоуправляемогообъекта;

б)еслинепосредственнымрезультатомявляетсяудалениеуправляемогообъекта,тоиспользуется операция elete;

в)еслинепосредственнымрезультатомявляетсяустановлениезначенийатрибутовуправляемогообъектаравнымизаданнымзначениям,тоиспользуетсяоперацияReplace—attribute—value;

г)еслинепосредственнымрезультатомявляетсяустановлениезначенийатрибутовуправляемогообъектаравнымизначениямпоумолчанию(приусловии,чтотакиезначениябылиопределены),тоиспользуетсяоперацияReplace—with—default—value;

д)еслинепосредственнымрезультатомявляетсядобавлениеилиудалениечленовмногозначныхатрибутовуправляемогообъекта,тоиспользуетсяоперацияAdd—memberилиRemove—member;

е)еслинепосредственнымрезультатомявляетсяполучениеотуправляемогообъектазначений атрибутов,тоиспользуетсяоперацияGet—attribute—value;

ж)вовсехдругихслучаях,напримеркогданетнепосредственногорезультатаилинепосредственныйрезультатявляетсякомбинациейперечисленныхвыше,илиимеетсякакое-либовлияние наобъектвцелом,используетсяоперацияAction.Примерамииспользованияэтойоперацииявляются случаи,когда:

1)невозможноопределитьтребуемуюоперациюнадмножествомуправляемыхобъектов с использованием области действия и фильтров вместе с операциями Get—attribute—value, Replace—attribute—value, Replace—with—default—value, Greate, elete, Add—member или Remove—member;

2)требуетсявкачествеэлементарнойоперациисозданиенесколькихуправляемыхобъектов;

3)влияниеоказываетсянанесколькообъектовбезобщихатрибутов;

4)запросилиответоперациисодержитинформацию,котораянеможетмоделироватьсяатрибутамиуправляемыхобъектов.

Понятия непосредственного и побочного результатов рассмотрены в ГОСТ Р ИСО/МЭК 10165-1.

8 Обозначения для определений управляемых объектов

8.1 Обзоробозначений

Определенные в настоящем разделе шаблоны обеспечивают общий набор обозначений для представленияразличныхаспектовопределенийклассовуправляемыхобъектовисвязанныхсними структурнаименования.Формальныеопределенияшаблоновсодержатсяв8.3—8.11;использованныевэтихформальныхопределенияхсинтаксическиесоглашенияописаныв8.2.Этиформальные определенияустанавливаютконструкции,которыеможетилидолженсодержатькаждыйшаблон,и порядок,вкоторомконструкциидолжныпоявлятьсявшаблоне.Примерыиспользованияэтихобозначений приведены в приложении А.

Структураиповедениеклассауправляемыхобъектовопределяются,восновном,спомощью шаблонаклассауправляемыхобъектов.Шаблонидентифицируетвзаимосвязинаследования,которыесуществуютмеждуопределяемымидругимиклассамиуправляемыхобъектов,ипакетыповедения,атрибутов,сообщенийиопераций,которыевключаютсявопределениеклассауправляемых объектов.Дляповторногоиспользованиячастейданнойспецификации,вспецификацияхдругих классовуправляемыхобъектовопределеныдополнительныешаблоны,обеспечивающиеспецификациюатрибутов(отдельныхивгруппах),поведения,действий,сообщений,параметровипакетов. Этидополнительныешаблоныявляются«вызываемыми»другимишаблонамиспомощьюметода ссылок,определенноговв8.2.Этотметодпозволяетссылатьсяилюбогостандартанаспецификации,содержащиесявдругихстандартах,допуская,такимобразом,использованиеродовыхопределенийдляопределенийклассовуправляемыхобъектов.Этидополнительныешаблоны,прижелании,могутбытьвключенывтелоопределения.

Наименованиеклассауправляемыхобъектовопределяетсяспомощьюшаблонасвязывания имен.Этотшаблонидентифицируетименуемыйклассуправляемыхобъектовиопределяетотносительное отличающее имя, которое может быть использовано для присвоения имен экземплярам классавконтекстеконкретногостаршегокласса.Этотшаблонобеспечиваетспецификациювзаимосвязей,существующихмеждудвумяклассамиобъектовврезультатесвязыванияимен.

8.2 Соглашения,использованныевопределенияхшаблонов

Шаблонначинаетсясметки-шаблонаиименишаблона**TEMPLATE—NAME.** Шаблон содержитодинилинесколькоконструкций,каждаяизкоторыхимеетимя**CNSTRUCT—NAME** и

можетиметьаргумент-конструкции.Аргумент-конструкцииможетсостоятьизнесколькихэлементовдлявызоваизопределенияконкретнойконструкции.Длякаждогоиспользованияшаблонадекларируетсяуникальнаяметка-шаблона,спомощьюкоторойможноссылатьсяиздругихшаблонов наданныйэкземпляршаблона,и,еслиприсутствуетконструкция**REGISTERE AS,** присваиваетсязначениеидентификатораобъектаАСН.1,подкоторымзарегистрированданныйэкземпляриспользования шаблона. Символ «;» используется в качестве признака конца каждой конструкции (кроме**REGISTERE AS и EFINE AS)** и конца шаблона.

Дляупрощенияструктурышаблонов,например,когдаоднаитажесинтаксическаяструктура неоднократноиспользуетсявопределенииишаблона,могутбытьвведеныопределенияобеспечивающихсинтаксисов.Еслитакиеопределениянужны,тоонивводятсяспомощьюключевыхслов

supportingproductions

вконцеопределенияшаблонаисостоятизпродукцийвида

kметка-определенияj -j kсинтаксическое-определениеj

Метка-определенияпозволяетссылатьсянаопределениеизопределенияшаблонаилииздругихобеспечивающихсинтаксическихпродукций,асинтаксическое-определениедаетраскрытиеопределения,используяустановленныенижесинтаксическиесоглашения.Вслучаесинтаксического—определения,специфицирующегонесколькоальтернативныхстрок,принято,чтоссылкинасодержащуюегосинтаксическуюпродукциюдолжнывычислятьсядоединственнойстроки,выбранной изэтихальтернатив.

Определенияшаблоновоснованынаследующихсинтаксическихсоглашениях:

а) все терминальные символы и ключевые слова, образующие часть определения шаблона, зависятотрегистра;

б)когданеобходимодлянедвусмысленнойпередачисинтаксисашаблона,элементышаблона должныотделятьсяотсоседниходнимилинесколькимиразделителями.Допустимымиразделителямиявляютсяпробел,конецстроки,пустаястрокаикомментарий.Одинилинесколькоразделителейдолжныприсутствоватьмежду:

1) меткой-шаблона и**TEMPLATE-NAME;**

2) **TEMPLATE-NAME и CNSTRUCT-NAME;**

3) **CNSTRUCT-NAME** иаргументом-конструкции.

Междулюбойпаройэлементоввшаблонеможетбытьвставленодинилинесколькоразделителей,акогдааргумент—конструкцииисостоитизнесколькихразличныхэлементов—разделители могутбытьвставленыимеждуними.Разделителимогутбытьвставленывнутрьэлементовшаблона, еслитолькоопределениешаблонаявнопозволяетэтосделать;

в)пробелы,пустыестроки,комментариииконцыстрокимеютсмыслтолькокакразделители;

г)комментарийначинаетсяизавершаетсяпаройсимволовиликонцомтойстроки,вкоторой встретиласьперваяпара.Приинтерпретацииишаблонов,определенныхвнастоящемстандарте,комментарийэквивалентенпробелу.Комментариинеимеютнормативногозначения;

д) символ

;

должен отмечать конец каждой конструкции в шаблоне (кроме**REGISTERE AS и EFINE AS)** и конец самого шаблона;

е)дляпредставленияидентификаторовобъектовдолжнаиспользоватьсянотация,определенная в ИСО/МЭК 8824-1, например продукция

идентификатор-объекта-j k**bjectIdentifierValue**j

представляет продукции для всех определений шаблонов настоящего стандарта, а **bjectIdentifierValue** указываетнасоответствующуюнотацию,определеннуювИСО/МЭК8824-1;

ж) строки, ограниченные парой

[ ]

выделяютвпределенииишаблоначасти,которыемогутприсутствоватьилиотсутствоватьвконкретныхслучаяхиспользованияшаблона.Еслизазакрывающейскобкойследуетзвездочка

[ ]★

то содержимое скобок может появляться нуль или несколько раз. Обстоятельства, при которых данныечастиопределениямогутбытьопущеныилиповторены,зависятотопределениятипашаблона;

з) строки, ограниченные парой

k j

выделяют в определении шаблона строки, которые должны быть заменены в конкретных случаях использования шаблона. Структура и смысл подставляемой строки зависят от ее типа;

и) прописные строки обозначают ключевые слова, которые обязательно должны присутствовать в каждом случае использования шаблона, если они не заключены в квадратные скобки

[ ]

для указания их факультативности;

к) символ

|

используется как разделитель альтернативных строк в синтаксических-определениях обеспечивающих продукций **supportingproductions**. Когда обеспечивающая продукция используется для определения альтернативных строк, открывающим разделителем первой альтернативы является -j, символ | является закрывающим и открывающим символом для последующих альтернатив, а закрывающим разделителем последней альтернативы является первый конец строки после ее открывающего разделителя;

л) метка-шаблона должна быть уникальной в пределах стандарта или документа, в котором она объявлена. В стандарте или документе, состоящем из нескольких частей, которые сопровождаются и распространяются по отдельности, метка-шаблона должна быть уникальна в пределах той части, где она объявлена.

Требование уникальности метки-шаблона не зависит от типа помечаемого шаблона. Например, если метка **label** использована в документе для экземпляра использования некоторого шаблона, то не допускается помечать меткой **label** экземпляр использования шаблона того же самого или другого типа.

Когда метка-шаблона объявлена в документе А и указывается из документа В, то ссылка в документе В должна иметь в качестве префикса глобально однозначное имя документа А. В случае документов, названных международно признанными и уполномоченными по наименованиям, такими как МККТТ или ИСО/МЭК, должны использоваться в качестве идентификаторов зарегистрированные обозначения документов, такие как **«CCITTRec.X.722(1992)|IS/IEC10165-4:1992»**. Формат этой строки должен быть установлен уполномоченным по наименованиям для рассматриваемого документа. Когда рассматриваемый документ совместно разработан и опубликован МККТТ и ИСО/МЭК, обозначение документа должно содержать оба обозначения, разделенные знаком «|», как показано в приведенном примере. Если глобально однозначное имя не существует, допускается присвоение указываемому документу значения идентификатора объекта и использование этого значения в качестве глобально однозначного имени документа. Определенный выше синтаксис метки-шаблона выглядит следующим образом:

[идентификатор-документа:]|строка-метки|

идентификатор-документа =

«<имя-стандарта»|идентификатор-объекта

Строка-метки может включать в себя любое число следующих символов:

1) прописные и строчные алфавитные символы;

2) цифры 0—9;

3) символ

-

4) символ

/

в любом порядке, начиная со строчного алфавитного символа, за исключением того, что пара символов

- -

не должна появляться в строке-метке. Например, следующая метка-шаблона

«CCITTRec.X.722(1992)|IS/IEC10165-4:1992»:examplebjectClass

является глобально однозначной меткой для определения **examplebjectClass** в приложении А.

Ссылка на метку без префикса идентификатор-документа указывает на метку, объявленную в том же документе, что и ссылка.

м) в любом месте шаблона, где метка-шаблона присутствует как указание на другой шаблон, она может быть заменена полным текстом указанного шаблона (включая метку-шаблона). Это по-

зволяет включать в тело шаблона другие шаблоны, на которые он ссылается (подшаблоны), при сохранении возможности для этих указываемых шаблонов, в свою очередь, ссылаться на подшаблоны. В результате обеспечивающая продукция **supporting production**

метка-шаблона-определение-шаблона

используется для всех экземпляров метки-шаблона и определения-шаблона;

н) когда необходимо сослаться из шаблона на определение значения или типа АСН.1, имя типа или значения АСН.1 имеет вкачестве префикса имя модуля АСН.1, содержащего определение этого типа или значения. При этом принимается, что имя модуля относится к модулю АСН.1, находящемуся в том же самом документе, что и шаблон, из которого дается ссылка на тип или значение. Следовательно, обеспечивающие продукции

указание-типа- <имя-модуля.<имя-типа

указание-значения- <имя-модуля.<имя-значения

используются для всех определений шаблонов, которые ссылаются на типы или значения АСН.1, где имя-модуля — имя, присвоенное модулю АСН.1 в документе, содержащем ссылку, а имя-типа и имя-значения — имена, присвоенные определениям типов или значений АСН.1, содержащимся в этом модуле. Если необходимо сослаться на определения типов или значений, содержащиеся в других документах, то это можно сделать с помощью локального модуля АСН.1, который использует утверждение **IMPORT** для импорта соответствующих определений типов или значений. (См. раздел 9.)

о) когда в шаблон необходимо включить текст, он включается в виде строки символов, факультативно начинающейся и заканчивающейся символом разделитель-текста, выбранного из числа следующих символов:

! " # $ % ^ & * ' ` ? @ \

Если используется символ разделитель-текста, то один и тот же символ должен использоваться в начале и в конце строки, а если этот разделитель-текста встречается в теле текстовой строки, то он должен быть заменен двойным вхождением символа. Если разделитель-текста не используется, то строка не должна содержать никаких знаков пунктуации, которые являются допустимыми для текстовых строк в этом определении шаблона.

Следовательно, обеспечивающие продукции

выделенная-строка-

разделитель-текста<текстовая-строка разделитель-текста|<текстовая-строка

разделитель-текста- ! | " | $ | S | % | ^ | & | * | ' | ` | ? | @ | \

используются для всех шаблонов, которые допускают выделенные-строки, где текстовая-строка — произвольная последовательность символов; если используется разделитель-текста, то все появления этого символа в текстовой-строке должны быть заменены парой символов разделитель-текста.

За исключением правил, относящихся к использованию разделителей, внутренняя структура текстовой-строки, в частности, использование определенной в настоящем стандарте структуры комментария, не имеет отношения к положениям настоящего стандарта.

8.3 Шаблон класса управляемых объектов

8.3.1 Обзор

Шаблон класса управляемых объектов является основой для формального определения управляемого объекта. Элементы шаблона позволяют разместить класс в подходящем узле дерева наследования, специфицировать различные характеристики класса и определить поведение класса. Ниже определено большинство этих элементов.

8.3.1.1 Наследование

Каждый класс управляемых объектов определяет суперкласс(ы), из которого(ых) он выводится. Характеристики суперкласса(ов) наследуются подклассом; определение подкласса может добавлять характеристики (специализация подкласса), но не может исключать характеристики суперкласса. Все классы являются подклассами высшего класса.

8.3.1.2 Обязательные пакеты

Определение класса управляемых объектов включает в себя пакеты поведения, атрибутов, операций и сообщений, которые обеспечивают полную спецификацию поведения, характеризующего все экземпляры класса.

8.3.1.3 Условные пакеты

Определение класса управляемых объектов включает в себя пакеты поведения, атрибутов, операций и сообщений, которые присутствуют в экземплярах этого класса вследствие заданного условия.

8.3.1.4Наименованиекласса

Определение класса управляемых объектов включает в себя имя класса, которое может использоваться в протоколе административного управления для ссылок на класс. Это достигается регистрацией значения идентификатора объекта, присвоенного определению класса управляемых объектов.

8.3.2 Структура шаблона

<метка-класса **MANAGE BJECTCLASS**
[ERIVE FROM <метка-класса
[,<метка-класса]*;
]
[**CHARACTERIZE BY** <метка-пакета
[,<метка-пакета]*;
]
[**C N ITI NAL PACKAGES** <метка-пакета
PRESENTI F определение-условия
[,<метка-пакета
PRESENTI F определение-условия]*;
]
REGISTERE AS идентификатор-объекта;
supportingproductions
определение-условия-выделенная-строка

8.3.3 Обеспечивающие определения

8.3.3.1 **ERIVE FR M** <метка-класса[,<метка-класса]*

Конструкция **ERIVE FR M** должна присутствовать во всех определениях классов управляемых объектов, кроме высшего. Следовательно, высший является суперклассом для всех классов управляемых объектов, кроме самого себя.

Метка-класса идентифицирует класс управляемых объектов, из которого выводится определяемый класс, т. е. тот класс управляемых объектов, который является одним из непосредственных суперклассов определяемого класса. Так как допустимо кратное наследование, класс управляемых объектов может иметь несколько непосредственных суперклассов.

Процесс наследования (специализации) требует, чтобы все характеристики суперкласса(ов) были включены в определение подкласса.

Характеристики, которые наследуются от суперкласса, не должны повторяться в документации подкласса, если не используется ни один из описанных в настоящем стандарте методов расширения или изменения определения, наследуемого из суперкласса. Следовательно, конструкция **ERIVE FR M** подразумевает автоматический импорт всех характеристик из определения(ий) суперкласса(ов). Эти характеристики могут быть расширены или изменены элементами, определенными в конструкциях **CHARACTERIZE BY** и **C N ITI NAL PACKAGES**.

Примечание 1 — Перечень всех классов управляемых объектов, характеристики которых наследует определяемый класс, рекомендуется в виде комментария включать в документацию о определении класса управляемых объектов.

Когда в результате наследования несколько раз импортируется определение одного и того же элемента (что может произойти, например, если два определения суперклассов содержат один и тот же атрибут), принимают, что подкласс содержит единственную копию рассматриваемого определения.

С точки зрения разрешения конфликтов, которые могут существовать между элементами, определенными в пакетах, и условными пакетами, наследуемыми или включаемыми в определение класса управляемых объектов при специализации, все пакеты, которые должны быть включены в конкретный класс управляемых объектов, рассматриваются как идентичные. Каждый пакет определяет несколько элементов, которые трактуются следующим образом:

а) **BEHAVI UR.** Пакеты, включаемые в подкласс, расширяют наследуемое поведение. Поведение класса управляемых объектов должно быть выражено таким образом, чтобы учитывать возможное присутствие или отсутствие условных пакетов. Когда поведение расширяется, то установленные или подразумеваемые предусловия могут быть только ослаблены (обязательные предусловия должны оставаться таковыми же или их число может быть уменьшено), установленные или подразумева-

емые постусловия могут быть только усилены (должны удовлетворяться те же постусловия и могут удовлетворяться дополнительные), а установленные или подразумеваемые инварианты остаются неизменными, но могут быть добавлены новые (см. ГОСТРИСО/МЭК 10165-1, 5.2.2.6);

Примечание 2 — При некоторых обстоятельствах могут быть определены подклассы, которые не требуют определения дополнительного поведения сверх того, которое наследуется от суперкласса(ов);

б) **ATTRIBUTES.** В пакетах, включаемых в определение подкласса, могут быть специфицированы пакеты. Когда конструкция **ATTRIBUTES** в пакете идентифицирует атрибут, который неоднократно определен в классе управляемых объектов, применяются следующие правила:

1) в реализованный управляемый объект должен быть включен единственный атрибут этого типа,

2) результирующий список-свойств есть логическое или (**OR**) включенных в подкласс и наследуемых списков-свойств, за исключением свойств **PERMITTE VALUES**, для которого реализуемое множество допустимых значений является пересечением всех спецификаций допустимых значений для этого типа атрибута, и **REQUIRE VALUES**, для которого реализуемое множество обязательных значений является пересечением реализуемого множества допустимых значений с объединением всех спецификаций обязательных значений для этого типа атрибута. Если для свойства атрибута **DEFAULT VALUE** или и **INITIAL VALUE** в совокупности определений заданы противоречивые значения, то включаемый в подкласс пакет должен разрешать этот конфликт,

3) параметры, связанные с данным атрибутом, являются объединением всех параметров, связанных с шаблоном атрибута, и всех параметров, связанных с атрибутом во всех тех пакетах, которые реализуются.

Если класс управляемых объектов предназначен для реализации, то должен быть определен, по крайней мере, один атрибут как часть определения класса, так как необходимо идентифицировать атрибут, который может быть использован для именования экземпляров управляемых объектов,

Примечание 3 — Атрибуты, используемые для наименования, могут быть выбраны из числа любых атрибутов, которые являются частью определения класса. В их число входят все атрибуты, наследуемые от суперклассов и добавленные к классу в результате специализации;

в) **ATTRIBUTE GRUPS.** Для расширяемой атрибутивной группы множество ее членов в экземпляре подкласса является объединением всех атрибутов, определенных в шаблоне атрибутивной группы и добавленных к этой группе в суперклассе(ах) или в подклассе;

г) **ACTIONS.** В определение подкласса могут быть включены действия; это могут быть действия в дополнение к наследуемым от суперклассов, или они могут включать дополнительные параметры для наследуемых действий. Множество параметров, связанных с данным действием, является объединением всех параметров, связанных с шаблоном действия и с действием во всех тех пакетах, которые реализуются;

д) **NOTIFICATIONS.** В определение подкласса могут быть включены сообщения; это могут быть сообщения в дополнение к наследуемым от суперклассов или они могут включать дополнительные параметры для наследуемых сообщений. Множество параметров, связанных с данным сообщением, является объединением всех параметров, связанных с шаблоном сообщения и с сообщением во всех тех пакетах, которые реализуются.

Если пакет включен в определение класса управляемых объектов несколько раз, путем наследования и (или) неоднократным включением в шаблон класса управляемых объектов, то результирующее определение-условия, связанное с пакетом, является логическим ИЛИ всех определений-условий в совокупном множестве определений. Для этой цели пакеты, входящие в конструкции **CHARACTERIZE BY** (обязательные пакеты), рассматриваются как входящие в конструкцию **CONDITIONAL PACKAGES** с определением условия **PRESENT IF TRUE**.

Характеристики в (обязательном или условном) пакете могут зависеть от характеристик других условных пакетов, если только связанные с этими пакетами условия гарантируют, что требуемые характеристики будут присутствовать во всех управляемых объектах, в которых присутствует первый пакет.

8.3.3.2 **CHARACTERIZE BY** <метка-пакета [, <метка-пакета]*

Данная конструкция, если она не присутствует, позволяет включить в определение класса управляемых объектов один или несколько обязательных пакетов поведения, атрибутов, операций и

сообщений в дополнение к тем, которые являются частью определения в результате конструкции **ERIVE FR M**. Метка-пакета идентифицирует определение того пакета, который должен быть включен. Спецификация метки пакета, который включен в определение класса управляемых объектов и как условный пакет, делает этот пакет обязательным для данного класса управляемых объектов и всех его подклассов.

8.3.3.3 **C N ITI NAL PACKAGES** <метка-пакета **PRESENTIF** определение-условия [, <метка-пакета **PRESENTIF** определение-условия]*

Данная конструкция присутствует, если в класс должны быть включены один или несколько условных пакетов. Метка пакета идентифицирует определение того пакета, который используется. Определение-условия является описанием условия, которое, если оно истинное, требует, чтобы пакет был включен в экземпляр класса. Условие должно удовлетворять требованиям к условным пакетам, установленным в ГОСТРИСО/МЭК10165-1. Например,

CN ITI NAL PACKAGES class-4-attributes PRESENTIF соответствующая категория протокола поддерживает операцию класса 4, определенную в ИСО/МЭКХХХХ';

образует допустимую декларацию пакета при условии, что операция класса 4, определенная в стандарте ИСО/МЭКХХХХ, является допустимой факультативной характеристикой ресурса.

Когда имеются специфичные условия, которые препятствуют реализации условного пакета, они должны быть установлены в шаблоне **BEHAVI UR**. Используемый для этого шаблон **BEHAVI UR** может быть в самом условном пакете или в обязательном пакете класса. Если такие спецификации присутствуют в текстовых определениях поведения, то рекомендуется, чтобы абзац, содержащий эти спецификации, вводился конструкцией "<метка-пакета **PRESENT NLY IF**: ".

8.3.3.4 **REGISTERE AS** идентификатор-объекта

Значение идентификатора-объекта обеспечивает глобально однозначный идентификатор определения класса управляемых объектов. Это значение используется с протоколом административного управления для идентификации класса управляемых объектов.

8.4 Шаблон пакета

8.4.1 Обзор

Данный шаблон позволяет определить пакет, состоящий из комбинации определений поведения, атрибутов, атрибутивных групп, операций, сообщений и параметров, для последующей вставки шаблон класса управляемых объектов в конструкциях **CHARACTERIZE BY** или **C N ITI NAL PACKAGES.** Ниже описаны основные элементы определения.

8.4.1.1 Поведение

Определение пакета обеспечивает полную спецификацию поведения, входящего в пакет. Она включает в себя:

- влияние операций на управляемый объект и обстоятельства, при которых создаются сообщения;
- ограничения, которые накладываются на операции для удовлетворения правилам согласованности, в частности, правилам, по которым может осуществляться создание и удаление управляемых объектов и последовательности этих операций;
- спецификацию того, как экземпляры класса управляемых объектов взаимодействуют с другими управляемыми объектами того же или других классов;
- идентификацию любых атрибутов, которые соотносятся с информацией в сообщениях. Это включает в себя идентификацию любых отображений в конкретные поля создаваемых сообщений или самого создание сообщений;
- спецификацию критериев выбора УОНЗ, если они есть;
- полное определение любых других аспектов поведения класса управляемых объектов.

8.4.1.2 Включаемые атрибуты

Должно быть определено множество атрибутов, которые содержат пакет.

8.4.1.3 Операции и сообщения

Определение пакета должно специфицировать, какие могут создаваться экземпляры сообщений класса, использующего этот пакет, какие могут осуществляться экземпляры операций класса и, в случае операций, относящихся к атрибутам, над какими атрибутами могут осуществляться операции. Определение пакета должно также специфицировать любые дополнительные параметры

тех сообщений и операций экземпляров класса управляемых объектов, которые используют данный пакет.

Примечания

1 Операции, идентифицированные в определении класса, относятся к типам операций, определенным в ГОСТ Р ИСО/МЭК 10165-1 (Get attribute value, Replace attribute value, Replace with default value и пр.). В случае операций Actions и Notifications требуются дополнительные определения для характеристики их функций, как описано в 8.10 и 8.11. Операции Create и Delete специфицируются как часть шаблона связывания имен, описанного в 8.6, так как создание и удаление управляемого объекта более тесно связано с соотношением вмещения между старшими и подчиненными объектами, чем со всеми экземплярами класса управляемых объектов.

2 Отложенное связывание, т.е. присвоение дополнительных параметров действиям и сообщениям класса управляемых объектов, может быть осуществлено путем включения в класс управляемых объектов пакета, который содержит (только) соответствующие действия и пакеты и их новые параметры. Правила объединения, приведенные в 8.3.3 для параметров действий и сообщений, означают, что дополнительные параметры будут связаны с сообщением или действием только при реализации пакета.

8.4.2 Структура шаблона

<метка-пакета **PACKAGE**
  [BEHAVIUR <метка-определения-поведения
  [,<метка-определения-поведения]*;
  ]
  [ATTRIBUTES <метка-атрибута список-свойств[<метка-параметра]*
  [,<метка-атрибута список-свойств[<метка-параметра]*]*;
  ]
  [ATTRIBUTEGRUPS <метка-группы[<метка-атрибута]*
  [,<метка-группы[<метка-атрибута]*]*;
  ]
  [ACTINS <метка-действия[<метка-параметра]*
  [,<метка-действия[<метка-параметра]*]*;
  ]
  [NTIFICATINS <метка-сообщения[<метка-параметра]*
  [,<метка-сообщения[<метка-параметра]*]*;
  ]
[REGISTEREAS идентификатор-объекта];
supportingproductions
список-свойств- **[REPLACE-WITH-EFAULT]**
  [EFAULTVALUE спецификатор-значения]
  [INITIALVALUE спецификатор-значения]
  [PERMITTEVALUES указание-типа]
  [REQUIREVALUES указание-типа]
  [получить-заменить]
  [добавить-удалить]
  [SET-BY-CREATE]
  [N-M IFY]
спецификатор-значения-указание-значения|
  ERIVATI N RULE<метка-определения-поведения
получить-заменить - **GET|REPLACE|GET-REPLACE**
добавит-удалить - **A |REM VE|A -REM VE**

8.4.3 Обеспечивающие определения

8.4.3.1 **BEHAVIUR**<метка-определения-поведения
  [,<метка-определения-поведения]*

Конструкция **BEHAVIUR** позволяет полностью описать поведение (семантику), связанное с пакетом. Эта конструкция соотносит внешние аспекты управляемого объекта (его операции и сообщения с его внутренними состояниями. Метка-определения-поведения идентифицирует экземпляр использования шаблона поведения. При некоторых обстоятельствах могут быть определены пакеты, в которых нет требуется спецификация поведения.

8.4.3.2**ATTRIBUTES**<метка-атрибутасписок-свойств
[<метка-параметра]*|,<метка-атрибута
список-свойств[<метка-параметра]*]*

Даннаяконструкцияпозволяетвключатьатрибутывопределениепакета.Список-свойств,который следует за каждой меткой-атрибута, определяет набор операций, которые могут осуществляться над управляемым объектом с указанием этого атрибута, и определяет значения по умолчанию, начальные, допустимые и обязательные значения, связанные с этим атрибутом.

Свойство **REPLACE-WITH-EFAULT** включается в определение, если атрибут имеет значение по умолчанию, которое может быть установлено с помощью операции Replace with default value.

Свойство **EFAULTVALUE** включается в определение, если атрибут имеет значение по умолчанию, которое должно использоваться для обеспечения значения атрибута в операции Replace withdefaultvalueилидолжнозадаватьдляатрибутазначениепоумолчаниюприреализациипакета в соответствии с правилами, определенными в **ГОСТ Р ИСО/МЭК 10165-1.** Если значение по умолчанию не определено, а свойство **REPLACE-WITH-EFAULT** присутствует, то значение по умолчаниюопределяетсяспособами,локальнымидляуправляемойсистемы.Значениеможетбыть задано или с помощью указания-значения, или с помощью конструкции **ERIVATI N RULE,** которая устанавливает, как может быть определено значение по умолчанию.

Свойство **INITIAL VALUE** включается в определение, если атрибут имеет обязательное начальноезначение,котороедолжноиспользоватьсядляатрибутавмоментсоздания.Значениеможет быть задано с помощью или указателя-значения, или конструкции **ERIVATI N RULE,** которая устанавливает, как может быть определено значение по умолчанию.

Еслиприсутствуетсвойство**PERMITTE VALUES,**тоуказание-типаспецифицируетограничения на допустимые значения, которые может принимать атрибут. Указываемая спецификация должна иметь вид подтипа синтаксиса атрибута, определенного с использованием нотации ACH.1 для подтипа.

Примечание 1 — Конструкция **PERMITTE VALUES** требуется только в тех определениях атрибутов, в которых необходимо задать ограничение на множество значений, допустимое синтаксиса атрибута, например при изменении существующей спецификации атрибута. Такое ограничение на множество значений атрибута должно устанавливаться только тогда, когда оно основано на ограничении наследования в семантике атрибута, а не на некоторых произвольных допущениях относительного того, что может образовывать приемлемое множество значений.

Если присутствует свойство **REQUIRE VALUES**, то указание-типа специфицирует значения, которые атрибут должен быть способен принимать. Указываемая спецификация должна иметь вид подтипа синтаксиса атрибута, определенного с использованием нотации ACH.1 для подтипа.

Примечание 2 — Это свойство определяет множество значений, требуемое для соответствия. Например, управляемый объект модема может иметь атрибут скорости передачи данных с допустимыми значениями от 0 до 19,2К; однако соответствие модема стандарту может требовать обеспечения одной конкретной скорости передачи данных из множества допустимых значений. Как и в случае с конструкцией **PERMITTE VALUES,** такое ограничение на множество значений атрибута должно устанавливаться только тогда, когда оно основано на ограничении наследования в семантике атрибута, а не на некоторых произвольных допущениях относительного того, что может образовывать приемлемое множество значений.

Свойство **GET** присутствует, если значение атрибута может быть получено с помощью операции **Getattributevalue.**

Свойство REPLACE присутствует, если атрибут может быть установлен с помощью операций **SetattributevalueиCreate**. Установка с помощью операции Create применяется только тогда, когда эта операция поддерживается связыванием имен экземпляра управляемого объекта.

Свойство **GET-REPLACE** является обозначением того, что присутствуют как свойство **GET**, так и свойство **REPLACE**.

Свойство **A** присутствует, если атрибут может быть установлен с помощью операции **Addmember**.

Свойство **REMVE** присутствует, если атрибут может быть установлен с помощью операции **Removemember**.

Свойство **A -REMVE** является обозначением присутствия как свойства **A** , так и **REMVE**.

Свойство **SET-BY-CREATE** присутствует, если атрибут может быть установлен с помощью операции Create. Это свойство имеет смысл только тогда, когда операция Create поддерживается

связыванием имен экземпляра управляемого объекта. Так как свойство REPLACE присутствует, если атрибут может быть установлен с помощью операций **Set attribute value** и **Create**, то свойство **SET-BY-CREATE** не должно включаться в шаблон, если присутствует свойство **REPLACE**. Аналогично, свойство **SET-BY-CREATE** не должно включаться в шаблон, если присутствует свойство **A , R E M V E** или **A - R E M V E**. Даже когда свойство **SET-BY-CREATE** отсутствует, попытка установить значение с помощью операции Create может оказаться успешной.

Отсутствие свойства **REPLACE** может быть использовано для спецификации того, что атрибут не может быть изменен в экземплярах класса, но его отсутствие не исключает подклассов, в которых добавлено свойство **REPLACE**. Свойство **N - M I F Y** присутствует для того, чтобы явно установить, что атрибут не может быть изменен (доступен только для чтения) в классе, имеющем это свойство, во всех его подклассах и во всех совместимых с ним управляемых объектах (т.е. управляемых объектах, ведущих себя аломорфно этому классу). Это свойство несовместимо, следовательно, не должно присутствовать в определении класса управляемых объектов, которое имеет для того же самого атрибута любое из свойств **REPLACE, GET-REPLACE, A , REM VE** или **A - REM VE.**

Примечания

3 Свойство **N - M I F Y** может быть совместимо со свойством **REPLACE-WITH-EFAULT**, т. к. эта операция часто используется в смысле «установить повторно», что может согласовываться с возможностями управляющего контролировать значения атрибутов.

4 До того как свойство **N - M I F Y** было добавлено в РОУО, было принято специфицировать это свойство в шаблонах **BEHAVI UR** или в документах, на которые эти шаблоны ссылаются.

Если желательно, чтобы часть определения атрибута являлось утверждение о том, что атрибут не может быть изменен ни в каком использующем этот атрибут классе, то это ограничение должно быть установлено в шаблоне BEHAVIOUR, на который ссылается шаблон ATTRIBUTE.

Метки-параметров, если они есть, идентифицируют специфичные для класса управляемых объектов параметры ошибок, связанные с операциями управления над этим атрибутом. Сообщение о них приводятся как ошибки обработки. Синтаксис параметров ошибок определяется в указываемых шаблонах.

8.4.3.3 **ATTRIBUTEGRUPS** <метка-группы [<метка-атрибута]*
[,<метка-группы [<метка-атрибута]*]*

Эта конструкция позволяет идентифицировать атрибутивные группы как часть пакета. В случае расширяемой атрибутивной группы ее исходное определение может быть расширено добавлением меток-атрибутов.

8.4.3.4 **ACTI NS** <метка-действия [<метка-параметра]*
[,<метка-действия [<метка-параметра]*]*

Метки-действий, если они есть, идентифицируют определения действий, которые включаются в пакет. Определения поведения должны специфицировать результаты этих действий на управляемые объекты.

Метки-параметров, если они есть, идентифицируют специфичную для класс управляемых объектов информацию действия или параметры ответа, или специфичные для класса управляемых объектов параметры ошибок, связанные с действием. Синтаксис параметров определяется в указываемых шаблонах.

8.4.3.5 **N TIFICATI NS** <метка-сообщения [<метка-параметра]*
[,<метка-сообщения [<метка-параметра]*]*

Конструкция присутствует, если в пакет включаются сообщения. Метки-сообщений идентифицируют применяемые определения сообщений. Определения поведения должны специфицировать обстоятельства, при которых эти сообщения создаются управляемыми объектами.

Метки-параметров, если они есть, идентифицируют специфичную для класса управляемых объектов информацию сообщения или параметры ответа, или специфичные для класса управляемых объектов параметры ошибок, связанные с сообщением. Здесь могут использоваться дополнительные параметры, например для заполнения полядополнительной информации сообщения, определенного в ИСО/МЭК10164-4. Синтаксис параметров определяется в указываемых шаблонах.

8.4.3.6 **REGISTERE AS** идентификатор-объекта

Значение идентификатора-объекта, если оно есть, обеспечивает глобально однозначный идентификатор определения пакета и регистрацию определенного пакетом группирования поведения,

атрибутов,атрибутивныхгрупп,действийисообщений.Значениеидентификатора-объектаявляется тем значением, которое включается в атрибут Packages всех создаваемых экземпляров класса управляемыхобъектоввсоответствииправилами,определеннымивГОСТРИСО/МЭК10165-1. Этаконструкцияобязательна,когданапакетссылаетсяконструкция**C N ITI N AL PACKAGES** в шаблонеклассауправляемыхобъектов.

8.5Шаблонпараметра

8.5.1 Обзор

Данныйшаблонпозволяетспецифицироватьизарегистрироватьсинтаксиспараметраисоответствующее поведение, которые могут быть связаны с конкретными атрибутами, операциями и сообщениямившаблонахпакета,атрибута,действияисообщения,определенныхв8.4,8.7,8.10и 8.11. Тип, определенный в шаблоне параметра, используется для заполнения конструкции **ANY E F I N E BY**xвПБДадминистративногоуправления,гдех—полеПБД,котороесодержитидентификаторобъекта,присвоенныйпараметру.Этотметодприменим,например,к:

- отказамобработки;
- параметрамзапросовиответовдействий;
- параметрамзапросовиответовсообщений.

Использованиешаблонавкаждомизэтихконтекстовописанов8.5.3.

Основныеэлементыопределенияописанынижe.

8.5.1.1Определениеконтекста

Шаблонспецифицируетконтекст,вкоторомприменяетсяпараметр,аименноонустанавливает,чтопередаетпараметрвконкретномполеПБДадминистративногоуправления.

8.5.1.1.1Информация/ответдействия,информация/ответ события,специфичнаяошибка

Когда контекст недвусмысленно идентифицируется ПБД административного управления, в которомпараметрпередан,этотконтекстможетбытьуказанодним изпятиключевыхслов,определенныхв8.5.3.1.КонтекстнедвусмысленноидентифицируетсяПБДадминистративногоуправлениятольковтомслучае,когдаконструкция**ANY E F I N E BY**встречаетсявэтомПБДровноодин раз.

8.5.1.1.2Ключевоесловоконтекста

Есликонтекстнеидентифицируетсянедвусмысленноп БДадминистративногоуправления,в котором передается параметр, то этот контекст должен быть специфицирован ключевым словом. Ключевоесловоконтекстадолжноидентифицироватьполевп БДадминистративногоуправления, вкоторомможетпередаватьсяпараметр.

8.5.1.1.3Использованиевдругихшаблонах

Втаблице16показано,гдедаются ссылкинашаблонпараметра.

Таблица 16 — Использование шаблона параметра

Использование	Возможные контексты
Конструкция**ATTRIBUTES**вшаблонепакета	**SPECIFIC-ERRR**
Конструкция**ACTI N S**вшаблонепакета	ключевое-слово-контекста, **SPECIFIC-ERRR, ACTI N-INF , ACTI N-REPLY**
Конструкция**N TIFICATI NS**вшаблонепакета	ключевое-слово-контекста, **SPECIFIC-ERRR, EVENT-INF , EVENT-REPLY**
Конструкция**CREATE**вшаблонесвязыванияимен	**SPECIFIC-ERRR**
Конструкция **E L E T E**вшаблонесвязыванияимен	**SPECIFIC-ERRR**
Шаблонатрибута	**SPECIFIC-ERRR**

Продолжение таблицы 16

Использование	Возможные контексты
Шаблон действия	ключевое-слово-контекста, **SPECIFIC-ERRR,** **ACTI N-INF ,** **ACTI N-REPLY**
Шаблон сообщения	ключевое-слово-контекста, **SPECIFIC-ERRR,** **EVENT-INF ,** **EVENT-REPLY**

При использовании в качестве квалификатора в определении пакета, параметр может быть «связан позже» с элементом, который он квалифицирует, например дополнительные параметры могут быть добавлены к ранее определенному сообщению в момент определения пакета, если синтаксис сообщения является расширяемым.

8.5.1.2 Определение синтаксиса

Шаблон позволяет связать с параметром абстрактный синтаксис.

8.5.1.3 Указание атрибута

Вместо явного определения синтаксиса и регистрации в шаблоне параметра шаблон может специфицировать эти два элемента через ссылку на шаблон атрибута. Использование такой конструкции не влияет на смысл существующих зарегистрированных атрибутов.

8.5.1.4 Поведение

Шаблон определяет любое поведение, которое применяется к использованию параметра.

8.5.2 Структура шаблона

<метка-параметра **PARAMETER**
CNTEXT тип-контекста;
выбор-синтаксиса-или-атрибута;
[**BEHAVIUR** <метка-определения-поведения
[,<метка-определения-поведения]*;
]
[**REGISTERE AS** идентификатор-объекта];
supportingproductions

тип-контекста	-	ключевое-слово-контекста| **ACTI N-INF** | **ACTIN-REPLY** | **EVENT-INF** | **EVENT-REPLY** | **SPECIFIC-ERRR**
ключевое-слово-контекста	-	указание-типа.<идентификатор
выбор-синтаксиса-или-атрибута	-	**WITHSYNTAX** указание-типа| **ATTRIBUTE**<метка-атрибута

8.5.3 Обеспечивающие определения

8.5.3.1 **CNTEXT** тип-контекста

Эта конструкция определяет контекст, в котором используется параметр, следующим образом:

- ключевое-слово-контекста: — этот выбор является ссылкой на контекст, определенный для шаблона внешним образом. Структура ссылки состоит из указания-типа с последующим идентификатором, который является именем поля в ПБД административного управления, заданного указанием-типа. Следовательно, может быть использована ссылка на контекст, определенный в другом документе. Это можно использовать, например, для указания того, что параметр применяется только для конкретного поля в параметре информации события УОИУ (см. ИСО/МЭК 10164-4) или в параметре ответа действия УОИУ. Если параметр не отображается в конкретное имя поля (например, информация о событии, по определению, должна быть установлена в паре идентификатор

параметра/значение параметра), то может быть задан один из перечисленных ниже более общих контекстов:

- ACTIN-INF: — этот выбор определяет параметр как применяемый для представления параметров, которые могут передавать информацию действия УОИУ;

- ACTIN-REPLY: — этот выбор определяет параметр как применяемый для представления параметров, которые могут передавать ответ действия УОИУ;

- EVENT-INF: — этот выбор определяет параметр как применяемый для представления параметров, которые могут передавать информацию события УОИУ;

- EVENT-REPLY: — этот выбор определяет параметр как применяемый для представления параметров, которые могут передавать ответ события УОИУ;

-SPECIFIC-ERRR:—этот выбор определяет параметр как применяемый для представления или создания сообщений об ошибках обработки УОИУ. Когда этот выбор используется с параметрами, которые применяются для атрибутов, то определение класса управляемых объектов должно специфицировать, должны ли изменяться другие атрибуты, указанные в одном запросе замены значений, если эта ошибка происходит для одного атрибута в операции Replace atribute value или Replace with default value.

8.5.3.2**WITHSYNTAX** указание-типа

Данная конструкция, если она присутствует, идентифицирует тип АСН.1 параметра при передаче в протоколе.

8.5.3.3**ATTRIBUTE** <метка-атрибута

Данная конструкция, если она присутствует, идентифицирует шаблон атрибута, синтаксис и идентификатор объекта которого используется в качестве синтаксиса и идентификатора объекта параметра.

8.5.3.4**BEHAVIUR** <метка-определения-поведения

[,<метка-определения-поведения]*

Данная конструкция, если она присутствует, позволяет специфицировать любое поведение или семантику, связанные с данным параметром. Если используется конструкция ATTRIBUTE, то рассматриваемая конструкция не изменяет поведение атрибута.

8.5.3.5**REGISTERED AS** идентификатор-объекта

Значение идентификатора-объекта, если оно есть, обеспечивает глобально однозначный идентификатор определения параметра. Данное значение используется в протоколе административного управления, когда необходимо идентифицировать параметр. Эта конструкция должна присутствовать только тогда, когда присутствует конструкция **WITHSYNTAX**.

8.6 Шаблон связывания имен

8.6.1 Обзор

Этот шаблон позволяет определить альтернативные структуры наименования для управляемых объектов данного класса с помощью связывания имен. Связывание имен позволяет выбрать атрибут в качестве именующего атрибута, который будет использоваться, когда подчиненному объекту, являющемуся экземпляром заданного класса управляемых объектов, присваивается имя старшим объектом, являющимся экземпляром заданного класса управляемых объектов или класса других объектов, например класса объектов справочника.

Если используется данное связывание имен, то в подчиненном объекте должен присутствовать атрибут, идентифицированный как именующий. Именующий атрибут используется для построения относительных отличающих имен (ООИ) подчиненных объектов этого класса. ООИ строится из идентификатора объекта, присвоенного этому типу атрибута, и значения экземпляра атрибута. Отличающее имя подчиненного объекта получается путем добавления его ООИ к отличающему имени его старшего объекта.

Связывания имен не рассматриваются как часть определения классов, к которым они относятся. Данный подчиненный класс управляемых объектов может иметь несколько относящихся к нему связываний имен. Множество связываний имен определяет множество возможных именующих взаимосвязей со старшими объектами и множество классов управляемых объектов, из которых могут реализовываться подчиненные объекты.

Связывание имен может быть определено так, что будет применяться ко всем подклассам заданного старшего класса объектов, или ко всем подклассам заданного подчиненного класса объектов, или к тем и другим.

Примечание — Связывание имен для класса управляемых объектов может быть установлено после спецификации самого класса.

8.6.2 Структура шаблона

<метка-связывание-имен**NAMEBINING**

SUBRINATE BJECTCLASS <метка-класса [**AN SUBCLASSES**];

NAME BYSUPERIR BJECTCLASS <метка-класса [**AN SUBCLASSES**];

WITHATTRIBUTE <метка-атрибута;

[**BEHAVIUR** <метка-определения-поведения

[,<метка-определения-поведения]*;

]

[**CREATE** [модификатор-создания[, модификатор-создания]]

[<метка-параметра]*;

]

[**ELETE** [модификатор-удаления]

[<метка-параметра]*;

]

REGISTERE ASидентификатор-объекта;

supportingproductions

модификатор-создания - **WITH-REFERENCE-BJECT** |

WITH-AUTMATIC-INSTANCE-NAMING

модификатор-удаления - **NLY-IF-N-CNTAINE-BJECTS** |

ELETES-CNTAINE-BJECTS

8.6.3 Обеспечивающие определения

8.6.3.1**SUBRINATE BJECTCLASS**<метка-класса

[**AN SUBCLASSES**]

Конструкция определяет класс управляемых объектов, экземплярам которого могут присваиваться имена экземплярами класса объектов, определенного конструкцией **NAME BYSUPERIR BJECTCLASS**. Имя экземпляра этого подчиненного класса объектов образуется сцеплением отличающего имени его старшего объекта с относительным отличающим именем подчиненного объекта. Если задано **AN SUBCLASSES**, то это связывание имен применяется и для всех подклассов заданного класса управляемых объектов.

8.6.3.2**NAME BYSUPERIR BJECTCLASS**<метка-класса

[**AN SUBCLASSES**]

Конструкция определяет класс управляемых объектов или класс других объектов, например класс объектов справочника, экземпляры которого могут присваивать имена экземплярам класса управляемых объектов, определенного конструкцией **SUBRINATE BJECTCLASS**. Если задано **AN SUBCLASSES**, то это связывание имен применяется и для всех подклассов заданного класса управляемых объектов.

8.6.3.3**WITHATTRIBUTE**<метка-атрибута

Эта конструкция определяет атрибут, который должен использоваться в контекстерассматриваемого связывания имен для построения относительного отличающего имени экземпляра класса управляемых объектов, определенного конструкцией **SUBRINATE BJECTCLASS**. Значения этого атрибута должны представляться типом данных с единственным значением, удовлетворяющим ограничениям ГОСТРИСО/МЭК10165-1. Если нет подходящего атрибута для использования в качестве именующего, то проектировщикам управляемых объектов рекомендуется обеспечивать управляющий атрибут типа **GraphicString**.

8.6.3.4**BEHAVIUR**<метка-определения-поведения

[,<метка-определения-поведения]*

Если она присутствует, эта конструкция позволяет определить любые конфликты поведения, возникающие вследствие связывания имен. Метка-определения-поведения идентифицирует рассматриваемое определение поведения.

Примечание — Эта конструкция предназначена для использования в качестве средства описания поведения, специфичного для связывания имен. Любое поведение, которое применимо ко всем возможным экземплярам класса управляемых объектов, должно быть определено как часть поведения, указанного в шаблоне пакета, определяющего класс управляемых объектов.

8.6.3.5**CREATE**[модификатор-создания

[, модификатор-создания]][<метка-параметра]*

Конструкцияприсутствует,еслидопускаетсясозданиеновыхэкземпляровклассауправляемых объектов,указанногоконструкцией**SUB RINATE BJECT CLASS,** вконтекстеданногосвязывания именспомощьюоперацииадминистративногоуправлениясистемы.Значениямодификаторов-созданияспецифицируютопции,доступныеприсоздании.Допустимымизначениямиявляются следующие:

- **WITH-REFERENCE-BJECT**:присозданииможетбытьзаданассылканауправляемыйобъект какисточникзначенийпоумолчаниюидляспецификациивыбораусловныхпакетов;

- **WITH-AUTMATIC-INSTANCE-NAMING:**можноопуститьвзапросесозданияспецификациюимениэкземпляранового управляемогообъекта.

Определения поведения должны специфицировать, как должны выбираться действия, когда выбраны связывания имен, которые могут применяться к новому управляемому объекту.

Источники начальных значений атрибутов, используемых в момент создания управляемого объекта, и соответствующие правила предпочтения определены в ГОСТРИСО/МЭК10165-1.

Метки-параметров,еслиониесть,идентифицируютпараметрыспецифичныхдлясвязывания именошибок,связанныхсоперацией Create.Онихсообщаетсякакоботказахвремениобработки. Синтаксиспараметровошибокопределяетсяуказываемымишаблонами.

8.6.3.6 **ELETE**[модификатор-удаления]|<метка-параметра]*

Конструкцияприсутствует,еслидопускаетсяудалениеэкземпляровклассауправляемыхобъектов,указанногоконструкцией**SUB RINATE BJECT CLASS**, вконтекстеданногосвязывания имен.Модификатор-удаления,еслионесть,указываетповедениеприудаленииуправляемогообъекта этогокласса.Допустимымизначениямиявляютсяследующие:

- **NLY-IF-N-CNTAINE-BJECTS:** всевмещаемыеуправляемыеобъектыдолжныбыть явноудалены спомощьюоперацийадминистративногоуправления доудаления вмещающегоуправляемого объекта, т. е. запрос операции elete приведет к ошибке, если имеются вмещаемые управляемыеобъекты;

- **ELETES-CNTAINE-BJECTS:**еслиоперацияeleteприменяетсякуправляемомуобъекту, для которого задан этот модификатор, то запрос операции elete приведет к отказу, если какой-либопрямоиликосвенновмещаемыйуправляемыйобъектимеетмодификатор **NLY-IF-N-CNTAINE-BJECTS** исодержитуправляемыеобъекты;впротивномслучаеуспешный запросоперацииeleteприведеткудалениювсехвмещаемыхуправляемыхобъектов.

Другие правила, описывающие поведение относительно удаления вмещаемых управляемых объектов,могутбытьспецифицированывконструкции**BEHAVIUR.**

Примечание—Учитывая,чтомодификатор**ELETES-CNTAINE-BJECTS**допускаетудаление управляемого объекта независимо от того, содержит ли он другие управляемые объекты, предпочтительнее использовать модификатор **NLY-IF-N-CNTAINE-BJECTS**, если есть какие-либо сомнения относительнотого,какоймодификаторлучшеподходит.

Если существуют ограничения на удаление относительно других взаимосвязей или условий, которыеявляютсяродовымидляклассауправляемыхобъектов,тоонидолжныбытьспецифицированыкачастьповеденияклассауправляемыхобъектов.

Метки-параметров,еслиониесть,идентифицируютпараметрыспецифичныхдлясвязывания именошибок,связанныхсоперацией elete.Онихсообщаетсякакоботказахвремениобработки. Синтаксиспараметровошибокопределяетсяуказываемымишаблонами.

8.6.3.7**REGISTERE AS**идентификатор-объекта;

Значениеидентификатора-объекта,еслионоесть,обеспечиваетглобальнооднозначныйидентификаторопределениясвязыванияимен.Данноезначениеиспользуетсядляидентификациисвязыванияименвцеляхадминистративногоуправления.

8.7Шаблонатрибута

8.7.1 Обзор

Данныйшаблониспользуетсядляопределенияотдельныхтиповатрибутов.Этиопределения могутбытьвдальнейшемобъединенывшаблонеатрибутивнойгруппы.Основныеэлементыопределенияописаныниже.

8.7.1.1Получение

Определениетипаатрибутаможетизменять или ограничивать определение другого типа атрибута.

8.7.1.2Синтаксисатрибута

Определениетипаатрибутадолжновключатьвсебяопределениесинтаксиса,которыйдолжен использоватьсядлявыражениязначенийатрибутавпротоколеадминистративногоуправления.Это достигаетсяспомощьюссылкинаопределениетипаACH.1.Определениесинтаксисаатрибутауказывает,являетсялизначениеатрибутаодно-илимногозначным.Еслибазовымтипомявляется**SET OF**,тотипатрибута—многозначный,впротивномслучае—однозначный.

8.7.1.3Согласованиезначений

Определение типа атрибута может включать в себя допустимые способы, которыми может бытьпроверенозначениеэкземпляратипа,т.е.атрибутможетбытьпроверенаравенство,размеры ит.п.Длянекоторыхтиповатрибутовсогласованиезначенияможетпотребовать,какчастьопределенияповеденияатрибута,спецификациютого,какопределеныиспользуемыеправилаогласования.Отсутствиеправилсогласованиявопределенииатрибутаподразумевает,чтосогласованиезначенийнеопределено.

8.7.1.4Поведение

Определениеатрибутаможетвключатьвсебяопределениеспецифичногодляатрибутаповедения, т. е. поведения, которое применяется для типа атрибута, независимо от того, какой класс управляемыхобъектовсодержитэкземплярытипаатрибута.

8.7.1.5Идентификаторатрибута

Значениеидентификатораобъектадолжнобытьвыделенодлякаждогоатрибута,которыйдолженвключатьсявопределениеклассауправляемыхобъектов.Этозначениеиспользуетсявпротоколеадминистративногоуправлениядляидентификацииатрибута.

8.7.1.6Параметры

Определениеатрибутаможетидентифицироватьпараметрыспецифичныхдляатрибутаошибок,связанныесоперациямиуправлениянадатрибутомэтоготипа.

8.7.2 Структура шаблона

```
<метка-атрибута          ATTRIBUTE
        получен-из-или-синтаксис;
        [MATCHES FOR квалификатор
                [, квалификатор]*;
        ]
        [BEHAVIOUR   <метка-определения-поведения
                [,<метка-определения-поведения]*;
        ]
        [PARAMETERS  <метка-параметра
                [,<метка-параметра]*;
        ]
[REGISTERED AS идентификатор-объекта];
supporting productions
квалификатор             - EQUALITY | ORDERING | SUBSTRINGS |
                           SET-COMPARISON | SET-INTERSECTION
получен-из-или-синтаксис - DERIVED FROM <метка-атрибута |
                           WITH ATTRIBUTE SYNTAX указание-типа
```

8.7.3 Обеспечивающие определения

8.7.3.1 **DERIVED FROM**<метка-атрибута

Еслиэтаконструкцияприсутствует,тоопределениеатрибутавкачествеисходнойточкипринимает все элементы определения, указываемые меткой-атрибута, включая те, которые, в свою очередь,могутбытьполученыиздругихопределенийатрибутов.Вэтомслучаеправилаинтерпретациирезультатаприсутствиялюбогодругогоэлементашаблонаатрибутаследующие:

- **MATCHES FOR**: результирующий набор правил согласования является логическим ИЛИ правилсогласования,задаваемыхэтойконструкцией,совсемизаимствованнымиправиламисогласования;
- **BEHAVIOUR**:—расширяетлюбыезаимствованныеправилаогласования;
- **REGISTERED AS**:—заменяетлюбуюзаимствованнуюрегистрацию.

Этотметодвыводаодногоатрибутаиздругогопозволяет:

-определятьатрибутнаосноведругогосуществующегоопределенияатрибута;

-добавлятьдополнительныеограниченияксуществующемуопределениюатрибута.

8.7.3.2**WITHATTRIBUTESYNTAX**указание-типа

Этаконструкция,присутствующаятольковтомслучае,когдаотсутствуетконструкция **E R I V E F R M**,идентифицируеттипданныхACH.1,которыйописывает,какэкземплярызначенийатрибутапередаютсявпротоколе.

ТипданныхACH.1такжеопределяеттипданныхсамогоатрибута.Еслибазовымтипомсинтаксисаявляется«множество-из»,тоатрибутможетиметьнесколькозначений.Всеостальныетипыданных ACH.1,включаятипы«множество»,«последовательность»и«последовательность-из»,определяюттипыатрибутовсединственнымзначением.

8.7.3.3**MATCHES F R**квалификатор[,квалификатор]*

Этаконструкцияопределяеттипыпроверок,которыемогутприменятьсякзначениюатрибутакакчастьфильтраоперации.Согласованиенаналичиеатрибутанеявноподразумеваетсядлявсех атрибутов.Всесогласованиядругихтипов,еслиэтаконструкцияотсутствует,неопределены,следовательно,недопустимыдляатрибута.Вариантыквалификаторовследующие:

- **EQUALITY**:—значениеатрибута,еслионоесть,можетбытьпроверенонаравенствозаданномузначению;
- **R E R I N G**: — значение атрибута, если оно есть, можно сравнить с заданным значением дляопределениябольшегоизних;
- **SUBSTRINGS**:—значениеатрибута,еслионоесть,можносравнитьсзаданнымзначениемдляопределения,входитоноилинетвзначениеатрибута;
- **SET-C MPARIS N**:—значениеатрибута,еслионоесть,можносравнитьсзаданнымзначениемдляопределениясоотношениясупермножество/подмножествомеждуэтимизначениями;
- **SET-INTERSECTIN**: — значение атрибута, если оно есть, можно сравнить с заданным значениемдлянахождениянепустогопересеченияэтихдвухзначений.

8.7.3.4**BEHAVI UR**<метка-определения-поведения

|,<метка-определения-поведения]*

Любоеповедение,котороеявляетсяродовымдляданноготипаатрибута,можетбытьопределеноспомощьюэтойконструкции.Определениеповедениядолжновключатьвсебялюбыедополнительные спецификации, которые требуются для определения того, как выбранное множество правил согласования применяется к определению атрибута. Поведение, которое является специфичным для класса управляемых объектов, определяется в конструкции BEHAVIOUR шаблона пакета.

8.7.3.5**PARAMETERS**<метка-параметра[,<метка-параметра]*

Метки-параметровпозволяютсвязатьпараметрыповедениемтипаатрибутадляопределения отказовобработки.Например,принекоторыхобстоятельствахтипатрибутаможетпродемонстрироватьошибку«нарушениеограничения».Параметр,дающийинформациюотакойошибке,может бытьопределен,используя**CNTEXTSPECIFIC-ERR R**вшаблонепараметра,иуказанизшаблонаатрибута.

8.7.3.6**REGISTERE AS**идентификатор-объекта

Значениеидентификатора-объекта,еслионоесть,обеспечиваетглобальнооднозначныйидентификатор определения атрибута, которое включает в себя все элементы, прямо или косвенно указанные в конструкциях **ERIVE FRM, WITH ATTRIBUTE SYNTAX, MATCHES F R и BEHAVI UR.**Этозначениеиспользуетсявпротоколеадминистративногоуправления,когданеобходимоидентифицироватьтипатрибута.Еслиэтаконструкцияопущена,тонаопределениеатрибутанельзясослатьсявопределенииклассауправляемыхобъектов.Когдаопределениеатрибутавыводитсяизсуществующегоопределенияатрибута,котороесодержитконструкцию**REGISTERE AS**, значениеидентификатора-объекта,присвоенноесуществующемуопределению,неявляетсядопустимымидентификаторомдлявыводимогоопределения.Следовательно,конструкция**REGISTERE AS**должнабытьвключенаввыводимоеопределение,еслинанегонужноссылатьсяизопределения классауправляемыхобъектов.

8.8Шаблонатрибутивнойгруппы

8.8.1 Обзор

Данныйшаблонпозволяетопределятьатрибутивныегруппы;такоегруппированиеприменяетсявситуациях,когдажелательноработатьссовокупностьюатрибутов,которыеявляютсячленами группы.Определенияповедениядляконкретногоклассауправляемыхобъектовустанавливаютсмысл операций получения значения атрибута и замены значением по умолчанию в тех случаях, когда операцииприменяютсякатрибутивнымгруппам.Каждыйчленгруппысамдолженбытьопределен какодно-илимногозначныйтипатрибута.

Шаблон атрибутивной группы определяет минимальный набор атрибутов, которые образуют группу, и значение идентификатора объекта, которое используется для наименования группы. Каждое определение класса управляемых объектов, которое ссылается на атрибутивную группу, может расширить группу, добавив новых членов, если только группа не была определена как фиксированная. Такие расширения применяются только для экземпляров того класса управляемых объектов, в котором расширение определено. Атрибуты, идентифицированные в шаблоне атрибутивной группы, определяют минимальный состав группы во всех определениях классов управляемых объектов, ссылающихся на эту группу.

Если в определении класса управляемых объектов присутствует расширяемая атрибутивная группа, то все атрибуты, включенные в группу либо в тело шаблона атрибутивной группы, либо добавленные во определении класса управляемых объектов, должны присутствовать в пакете, который ссылается на группу, или во одном из обязательных пакетов этого класса.

Если во определении класса управляемых объектов присутствует фиксированная атрибутивная группа, то все атрибуты, включенные в группу, должны присутствовать в пакете, который ссылается на группу.

8.8.2 Структура шаблона

```
<метка-группы                  ATTRIBUTEGRUPS
      [GROUP ELEMENTS  <метка-атрибута
                       [,<метка-атрибута]*;
      ]
      [FIXE;
      ]
      [ESCRIPTI N      выделенная-строка;
      ]
REGISTEREAS идентификатор-объекта;
```

8.8.3 Обеспечивающие определения

8.8.3.1 **GRUPELEMENTS** <метка-атрибута], <метка-атрибута]*

Эта конструкция, если она есть, определяет набор меток-атрибутов, которые идентифицируют отдельные атрибуты, образующие те элементы атрибутивной группы, которые должны присутствовать во всех экземплярах этой группы; каждый из этих элементов должен быть определен с помощью шаблона атрибута. Определения поведения для конкретного класса управляемых объектов устанавливают смысл операций получения значения атрибута и замены значением по умолчанию в тех случаях, когда операции применяются к атрибутивным группам.

Примечание — Это не подразумевает, что существует спецификация поведения самой атрибутивной группы, которая не применяется к отдельным атрибутам.

Все атрибуты в группе должны быть членами определения класса управляемых объектов, ссылающегося на группу, т.е. каждый атрибут, который является членом группы для данного класса управляемых объектов, должен быть указан в конструкции ATTRIBUTES одного или нескольких пакетов, на которые ссылается определение класса управляемых объектов.

8.8.3.2 **FIXE**

Эта конструкция, если она есть, указывает, что атрибутивная группа определяется как фиксированная.

8.8.3.3 **ESCRIPTI N** выделенная-строка

Эта конструкция позволяет описать семантику группы, например: «Группа всех атрибутов состояний управляемого объекта». Не устанавливается никаких ограничений на наборы символов, используемые в данной конструкции, и не определяется какая-либо структура внутри нее.

Данная конструкция не должна использоваться как способ определения поведения группы или ее членов.

8.8.3.4 **REGISTERE AS** идентификатор-объекта

Значение идентификатора-объекта обеспечивает глобально однозначный идентификатор определения атрибутивной группы. Это значение используется в протоколе административного управления, когда необходимо идентифицировать атрибутивную группу. Атрибутивная группа, идентифицированная этим значением идентификатора-объекта в контексте управляемого объекта, включает в себя все атрибуты, определенные в теле шаблона атрибутивной группы, с учетом для рас-

ширяемых групп всех атрибутов, добавленных к группе вследствие определения элементов шаблона класса управляемых объектов, который применяется при реализации управляемого объекта.

Примечание — Шаблон атрибутивной группы определяет набор атрибутов (он может быть пустым), которые всегда являются членами группы. В случае расширяемой атрибутивной группы этот набор может быть расширен конструкцией ATTRIBUTE GROUPS в шаблоне пакета в интересах конкретных определений классов управляемых объектов. Этот метод подходит тогда, когда желательно определить атрибутивную группу, члены которой имеют некоторую общую семантику (например, «атрибуты состояний»), но число атрибутов с этой семантикой, которые могут присутствовать в данном классе управляемых объектов, определяется в момент реализации; или когда в нескольких классах управляемых объектов требуются различные группировки атрибутов со одной и той же семантикой. Вобщем случае можно определить состав расширяемой атрибутивной группы в управляемом объекте только в момент реализации, когда известно, какие пакеты и, следовательно, какие атрибуты должны реализовываться.

8.9 Шаблон поведения

8.9.1 Обзор

Данный шаблон используется для определения поведения классов управляемых объектов, связываний имен, параметров, атрибутов, действий и сообщений. Шаблон поведения предназначен для того, чтобы обеспечивать расширения, но спецификации поведения не должны изменять ранее определенную информацию. Если информация оставлена неопределенной, то определении поведения должно быть явно указано, что именно не определено.

Примечания

1 Шаблоны поведения должны использоваться для выражения семантики, которая не полностью описана в других шаблонах. В частности, авторы определений не должны полагаться на заметки для выражения семантики.

2 Утверждения о поведении должны быть выражены в терминах управляемых объектов того класса, определение которого содержит эти утверждения.

8.9.2 Структура шаблона

<метка-определения-поведения **BEHAVIOUR DEFINED AS** выделенная-строка;

8.9.3 Обеспечивающие определения

8.9.3.1 **DEFINED AS** выделенная-строка

Текст, содержащийся в выделенной-строке, дает определение поведения класса управляемых объектов или соответствующих ему связываний имен, параметров, атрибутов, действий или сообщений. Это определение может быть задокументировано на естественном языке или с использованием формальных методов описания. Текст может быть (текстовой) ссылкой на раздел или подразделы некоторого документа или стандарта. Не устанавливается никаких ограничений на наборы символов, используемые для представления выделенной-строки, и не определяется какая-либо структура в этом тексте.

8.10 Шаблон действия

8.10.1 Обзор

Этот шаблон используется для определения поведения и синтаксисов, связанных с конкретным типом действия. Типы действий, определенные с помощью этого шаблона, могут быть переданы услугой **M-ACTION**, определенной в ГОСТ Р ИСО/МЭК 9595. Ниже описаны основные элементы определения.

8.10.1.1 Поведение

Определение типа действия должно специфицировать функции действия в терминах воздействий, которые оно оказывает на классы управляемых объектов. Когда действие может применяться к нескольким классам управляемых объектов, описание поведения должно ограничиваться теми характеристиками, которые являются общими для управляемых объектов во всех классов; относящееся к этому действию поведение, специфичное для класса управляемых объектов, описывается как часть определения самого класса.

8.10.1.2 Режим работы

Определение типа действия должно указывать, является ли действие всегда подтверждаемым или оно может быть подтверждаемым или не подтверждаемым по усмотрению управляющего.

8.10.1.3 Абстрактный синтаксис

Определение типа действия должно специфицировать все синтаксисы, которые могут использоваться для передачи информации действия и параметров ответа действия услуге

M-ACTIN,определеннойвГОСТРИСО/МЭК9595.Синтаксисыопределяютсяспомощьютипов данныхАСН.1.

Примечания

1Еслитольконетнамеренияспециальнопредотвратитьпоследующиерасширенияаргументовдействий, торекомендуется, чтобы синтаксисы информации и ответа действия определялись расширяемым образом путемвключениявкачествефакультативногополятипаАСН.1**SET FManagementExtension**(определенного вГОСТРИСО/МЭК10165-2).

2 Рекомендуется, чтобы базовым типом данных, выбранным для синтаксисов информации и ответа, был тип **SEQUENCE**.

8.10.1.4Идентификаторыдействий

Значение идентификатора объекта, связанного с определением типа действия, используется для идентификации этого типа в протоколе административного управления.

8.10.1.5Параметры

Определениетипадействияможетидентифицироватьпараметрыинформациидействия,ответадействияилиспецифичныхошибок,связанныхстипомдействия.

8.10.2 Структура шаблона

```
<метка-действия ACTIN
        [ BEHAVIUR    <метка-определения-поведения
                      [,<метка-определения-поведения]*;
        ]
        [ M E C NFIRME ;
        ]
        [ PARAMETERS<метка-параметра
                      [,<метка-параметра]*;
        ]
        [ WITH INFRMATIN SYNTAX указание-типа;
        ]
        [ WITH REPLYSYNTAX указание-типа;
        ]
REGISTERE AS идентификатор-объекта;
```

8.10.3 Обеспечивающие определения

8.10.3.1**BEHAVIUR**<метка-определения-поведения
[,<метка-определения-поведения]*

Эта конструкция, когда присутствует, определяет поведение действия, параметры, которые должны быть определены вместе с действием, результаты, к которым может привести действие, и их смысл. Метки-определений-поведения указывают на описания поведения, определенные с помощью шаблона поведения.

8.10.3.2**M E C NFIRME**

Эта конструкция, если присутствует, указывает, что действие должно осуществляться в подтверждаемом режиме. Если конструкция отсутствует, то действие может осуществляться в подтверждаемом и неподтверждаемом режиме по усмотрению управляющего.

8.10.3.3**PARAMETERS**<метка-параметра[,<метка-параметра]*

Метки-параметров идентифицируют параметры информации или ответа действия, а также отказы обработки, связанные с типом действия. Пример см. в А.7.

8.10.3.4**WITH INFRMATIN SYNTAX**указание-типа

Если эта конструкция присутствует, то указание-типа идентифицирует тип данных АСН.1, описывающий структуру параметра информации действия, которая передается в протоколе административного управления. Если эта конструкция отсутствует, то нет никакой специфичной информации, связанной с вызовом действия.

8.10.3.5**WITH REPLYSYNTAX**указание-типа

Если эта конструкция присутствует, то указание-типа идентифицирует тип данных АСН.1, описывающий структуру параметра ответа действия, которая передается в протоколе административного управления. Если эта конструкция отсутствует, то нет никакой специфичной информации, связанной с ответом действия.

8.10.3.6**REGISTERE AS**идентификатор-объекта

Значение идентификатора-объекта обеспечивает глобально однозначный идентификатор определения типа действия. Это значение используется в протоколе административного управления, когда необходимо идентифицировать тип действия.

8.11 Шаблон сообщения

8.11.1 Обзор

Данный шаблон используется для определения поведения и синтаксисов, связанных с конкретным типом сообщения. Типы сообщений, определенные с помощью этого шаблона, могут передаваться в отчетах о событиях с помощью услуги **M-EVENT-REP RT**, определенной в ГОСТ РИСО/МЭК9595. Ниже описаны основные элементы определения.

8.11.1.1 Поведение

Определение типа сообщения должно специфицировать обстоятельства, при которых создается сообщение данного типа.

8.11.1.2 Абстрактный синтаксис

Определение типа сообщения должно специфицировать все абстрактные синтаксисы, которые могут использоваться для выражения параметров информации и ответа события в услуге **M-EVENT-REP RT**, определенной в ГОСТРИСО/МЭК9595. Шаблон также допускает распределение значений атрибутов по полям синтаксиса.

Примечания

1 Если только нет намерения специально предотвратить последующие расширения аргументов сообщений, то рекомендуется, чтобы синтаксисы информации и ответа сообщения определялись расширяемым образом путем включения в качестве факультативного поля типа ACH.1 **SET F Management Extension** (определенного в ГОСТРИСО/МЭК10165-2).

2 Рекомендуется, чтобы базовым типом данных, выбранным для синтаксисов информации и ответа, был тип **SEQUENCE**.

8.11.1.3 Имя сообщения

Значение идентификатора объекта, связанного с определением сообщения, используется для идентификации типа события в протоколе административного управления.

8.11.1.4 Параметры

Определение типа сообщения может идентифицировать параметры информации события, ответа события или специфичных ошибок, связанных с типом сообщения.

8.11.2 Структура шаблона

```
<метка-сообщения          N TIFICATI N
    [BEHAVI UR<метка-определения-поведения
                    [,<метка-определения-поведения]*;
    ]
    [PARAMETERS     <метка-параметра
                    [,<метка-параметра]*;
    ]
    [WITH INF RMATI N SYNTAXуказание-типа
        [AN ATTRIBUTE I S<имя-поля <метка-атрибута
                            [,<имя-поля <метка-атрибута]*
        ];
    ]
    [WITH REPLY SYNTAXуказание-типа;
    ]
REGISTERE ASидентификатор-объекта;
```

8.11.3 Обеспечивающие определения

8.11.3.1**BEHAVI UR**<метка-определения-поведения

[,<метка-определения-поведения]*

Эта конструкция, когда присутствует, определяет поведение сообщения, данные, которые должны быть определены вместе с сообщением, результаты, к которым может привести сообщение, и их смысл. Метки-определений-поведения указывают на описания поведения, определенные с помощью шаблона поведения.

8.11.3.2**PARAMETERS**<метка-параметра[,<метка-параметра]*

Метки-параметровидентифицируютпараметрыинформацииилиответасобытия,атакжеотказыобработки,связанныестипомсообщения.Примерсм.вА.8.

8.11.3.3 **WITH INFRMATI N SYNTAX**указание-типа

[AN ATTRIBUTE I S<имя-поля<метка-атрибута

[,<имя-поля<метка-атрибута]*]

Если эта конструкция присутствует, то указание-типа идентифицирует тип данных АСН.1, описывающийструктурупараметраинформациисообщения,котораяпередаетсявпротоколеадминистративного управления, и позволяет связать идентификаторы атрибутов с именами полей в абстрактномсинтаксисе.Еслиэтаконструкцияотсутствует,тонетникакойспецифичнойинформации, связанной с вызовом сообщения. Если присутствует конструкция **AN ATTRIBUTE I S**, то имя-поля должно быть меткой, определенной в абстрактном синтаксисе, на который ссылается указание-типа.Типданных,помеченныйименем-поля,используетсядляпередачизначенийатрибута,указанногометкой-атрибута.ТипданныхАСН.1атрибутадолженбытьтемже,чтоиуказанныйименем-поля.

Никакиеметкивнутритипов**S E T F**или**SEQUENCE F**немогутиспользоватьсявкачестве имени-поля, так как метки внутри подобных повторяющихся конструкций не позволяют недвусмысленноссылатьсянаединичныеэкземплярытипаданных.Аналогичноникакиеметкикомпонентов типов**C H I CE**, **SET** и л и **SEQUENCE** не могут использоваться в качестве имени-поля, если помеченныекомпонентынеоднократновстречаютсявопределениитипа.

8.11.3.4**WITHREPLYSYNTAX**указание-типа

Если эта конструкция присутствует, то указание-типа идентифицирует тип данных АСН.1, которыйописываетструктурупараметраответасообщения,котораяпередаетсявпротоколеадминистративногоуправления.Еслиэтаконструкцияотсутствует,тонетникакойспецифичнойинформации,связанной с ответом сообщения.

Синтаксисответаиспользуется,когдасообщениеотправляетсявподтверждаемомрежимеуслуги**У О И У М-EVENT-REPRT**.Подтверждениесобытияневозвращаетсяуправляемомуобъекту. Решениеоботправкесообщениявподтверждаемомилиподтверждаемомрежимеявляетсявопросом соответствующего агента, который принимает решение на основе политики, связанной с управляющим. Когда конструкция **WITH REPLY SYNTAX** опущена в определении сообщения, но сообщениеотправленовподтверждаемомрежиме,топодтверждениенебудетсодержатьинформациииответа.

8.11.3.5**REGISTERE AS**идентификатор-объекта

Значениеидентификатора-объектаобеспечиваетглобальноооднозначныйидентификаторопределениятипасообщения.Этозначениеиспользуетсявпротоколеадминистративногоуправления, когданеобходимоидентифицироватьтипсообщения.

9Руководствопоразработкеэквивалентныхмодулей АСН.1:1994иАСН.1:1990

Возможнаразработкастандартовнаоснове АСН.1:1994(ИСО/МЭК8824-1).Длятогочтобы можнобылоиспользоватьАСН.1:1994,рекомендуетсяразрабатыватьэквивалентныйнормативный модульАСН.1:1990(ГОСТРИСО/МЭК8824)сследующимисвойствами:

1)ондолжениметьтотжесамыйидентификаторобъекта,чтоимодульАСН.1:1994;

2) он является нормативным, но стандарт устанавливает, что в случае несогласованности междумодулямиАСН.1:1990иАСН.1:1994,предпочтениеимеетпоследнийизних;

3) стандарт устанавливает, что использование АСН.1:1990 сохраняется так долго, как это будетнеобходимо.

Примечание1—ПравиламиИСО/МЭКСТК1/ПК21установленпериодический(одинразвгод) обзорсцельюобновлениямеждународныхстандартовАСН.1:1990*.Национальныморганампостандартизации рекомендовано учитывать это при пересмотре стандартов АСН.1:1990. Тем самым обеспечивается, что стандартыАСН.1:1990будутсохраняться*так долго, как этонеобходимо.

*ИСО/МЭКСТК1ПК21подтвердилпродолжениедействиястандартовАСН.1:1990посоображениям соответствияипереносимости.ПК21потребовалотсвоихрабочихгрупппродолжатьподдерживатьэтистандарты.Соответствующаярезолюция ПК21будетприниматьсянакаждомзаседанииПК21(внастоящеевремя —ежегодно).

Для уменьшения количества ошибок рекомендуется, чтобы модуль АСН.1:1990 генерировался в результате машинного преобразования АСН.1:1994, так как это преобразование легко осуществить автоматически.

Примечание 2 — Если для преобразования АСН.1:1994 в АСН.1:1990 желательно использовать коммерческое средство (например средство АСН.1 поставщика XXX), то рекомендуется вначале сгенерированного кода добавить комментарий, который гласит приблизительно следующее:

--Использовано средство XXX АСН.1--

--для преобразования АСН.1:1994 в АСН.1:1990--

и примечание:

«Примечание — Хотя ИСО не отдает предпочтение ни одному программному средству, XXX АСН.1 позволяет преобразовать АСН.1:1994 в АСН.1:1990.»

Следует учитывать, что проблем можно избежать только в том случае, когда используется общее подмножество АСН.1:1990 и АСН.1:1994. В таком случае в стандарт следует включать только модуль АСН.1:1994.

9.1 Руководство

Рекомендуется придерживаться следующих правил.

1) Модули АСН.1:1990 и АСН.1:1994 должны ссылаться на одни и те же документы административного управления системы. Требуется, чтобы данный модуль полностью соответствовал либо АСН.1:1990, либо АСН.1:1994, а директивы, определенные в разделе 10, используются для идентификации того, какая версия нотации используется в конкретном модуле.

2) Указания типов и значений могут быть импортированы в модуль АСН.1:1994 из модуля АСН.1:1990 за следующими исключениями:

а) АСН.1:1990 MACRO не может быть импортирована в модуль АСН.1:1994; следовательно, невозможно создать экземпляр **MACR** в модуле АСН.1:1994.

б) Идентификаторы значений **SET, SEQUENCE** и **CHICE**.

3) Указания типов и значений могут быть импортированы в модуль АСН.1:1990 из модуля АСН.1:1994 за следующим исключением:

типы АСН.1:1994 **CHARACTERSTRING, BMPString, UniversalString, EMBEEPV** не могут быть импортированы. Так как в АСН.1:1990 нет эквивалентов для этих типов АСН.1:1994, то их использование не рекомендуется в тех модулях АСН.1:1994, для которых требуются эквивалентные модули АСН.1:1990. По тем же причинам запрещается использование типа АСН.1:1994 Tuple в тех модулях АСН.1:1994, для которых требуются эквивалентные модули АСН.1:1990*.

Примечание 1 — Если следовать предлагаемому руководству, то противоречия при импорте указаний типов и значений из одной версии АСН.1 в другую не возникает, т. к. эквивалентные конструкции существуют в любом случае.

4) Для определений АСН.1 классов информационных объектов, которые импортируются в модули АСН.1:1994, следует использовать следующий модуль АСН.1:1994:

```
--<МодульАСН.1версии1994годаSMModule
--<{joint-iso-itu-tms(9)smi(1)part4(4)asn1Module(2)2}--
SMModule{joint-iso-itu-tms(9)smi(1)part4(4)asn1Module(2)2}
    EFINITINS::=BEGIN
    REGISTERE-AS::=TYPE-IENTIFIER
--TYPE-IENTIFIERопределенвГОСТРИСО/МЭК8824—1,доступенвлюбоммодулебез
--егоимпортаиопределенкак:
--TYPE-IENTIFIER::=CLASS
--{
--&idBJECTIENTIFIERUNIQUE,
--&Type
--}
```

*Tuple является именем для продукции АСН.1:1994, которая позволяет вставлять управляющие символы в нотацию значения IA5String, чего нельзя сделать в АСН.1:1990. Например, ... **greetingsIA5String**::= {**«hello»**, cr, **«there»**} вставляет возврат каретки между **«hello»** и **«there»** (cr импортировано из модуля, определенного в ИСО/МЭК 8824-1, и эквивалентно литералу «возврат каретки»).

```
- - WITH SYNTAX {&Type IENTIFIE BY &id}
      INF-REPLY-IENTIFIER : : = CLASS
            {
               &Info PTIONAL,
               &Reply PTIONAL,
               &registeredAs BJECTIENTIFIERUNIQUE
            }
      WITH SYNTAX {INF &InfoREPLY &Reply IENTIFIE BY &registeredAs}
      RegisteredAsTableREGISTRERE-AS: : = {...}
      InfoReplyTableINF-REPLY-IENTIFIER: : = {...}
- - RegisteredAsTableдолжнабытьзаполненашаблонамиРОУО
- - ATTRIBUTEsPARAMETER.
--InfoReplyTable должнабытьзаполненашаблонамиРОУО
--ACTIONиNOTIFICATION.
      E N
      5) МодулиACH.1:1994определяютсякакв следующемпримере:
--<МодульACH.1версии1994годаExampleModule - -
      ExampleModule{--здесьдолженбытьдопустимыйидентификаторобъекта--
      EFINITI NS: : = BEGIN
      IMP RTS
      REGISTERE-AS,
      INF-REPLY-IENTIFIER,
      RegisteredAsTable,
      InfoReplyTable
      FRMSMModule{joint-iso-itu-tms(9)smi(1)part4(4)asn1Module(2)2};
      Foo : : =          SEQUENCE{
                         id1REGISTERE-AS.&id({RegisteredAsTable}),
                         syntax1REGISTERE-AS.&Type({RegisteredAsTable}{@.id1})}
      Bar: : =           SEQUENCE{
                         id2REGISTERE-AS.&id({RegisteredAsTable}),
                         syntax2SEQUENCEFREGISTERE-AS.&Type({RegisteredAsTable}{@id2})}
      firstExtensionId BJECTIENTIFIER: : = {1 3 1 7 1 0 3 1 0 1}
      firstExtensionInfo: : = PrintableString
--Иллюстрируетиспользованиесодержащегосяподтипа,ограничивающегооткрытыйтип.--
      FooBar: : =        Foo(WITH CMP NENTS{
                         id1(firstExtensionId),
                         syntax1(FirstExtensionInfo)})
      E N
```

ТаккакРОУОиспользуетсявместесклассоминформационныхобъектовACH.1**REGISTERE-AS**, то FooBar является дублированием информации. А именно, спецификация РОУО плюс **REGISTERE-AS**ACH.1эквивалентныограничениювнутреннеготипав**Foo**.FooBarпростоиллюстрирует,чтооткрытый тип может быть ограничен до любого типа, а**ANY/ANY EFINE BY** не можетбытьограниченвACH.1:1990дотипа,отличногоот**ANY/ANY EFINE BY**.Этаконструкция показывает, как отображать ограниченный таким образом тип ACH.1:1994 в комментарии в ACH.1:1990.

6) МодулиACH.1:1994преобразуютсявмодулиACH.1:1990последующиминструкциям.

а) Удалить часть утверждения**IMP RTS**, ссылающуюся на модуль SMModule. Это позволит избавиться от импортированных определений классов информационных объектов **REGISTERE-AS** и **INF-REPLY-IEN-TIFIER**.

б) Преобразовать все ссылки на открытые типы ктипам **ANY** или **ANY EFINE BY**. Для этого преобразовать весь синтаксис ACH.1 следующимобразом:

во-первых, преобразовать

из	в
REGISTERE-AS.&id	**BJECTIENTIFIER**

Если «**REGISTERE-AS.&Type**» является компонентом **SET** или **SEQUENCE** и определен в той же самой конструкции **SET** или **SEQUENCE** как к «**id**», то преобразовать

из	в
REGISTERE-AS.&Type({RegisteredAsTable}{@.id1}) **REGISTERE-AS.&Type({RegisteredAsTable}{@.id1})**	**ANY EFINE BY id** **ANY EFINE BY id**

в противном случае преобразовать

из	в
REGISTERE-AS.&Type({RegisteredAsTable}{@.id1}) **REGISTERE-AS.&Type({RegisteredAsTable}{@.id1})**	**ANY** **ANY**

в) Если для модуля ACH.1 действует **AUTMATIC TAGS**, то применить ИСО/МЭК 8824-1, пп. 22.5—22.7, где описано, как действует автоматическое тегирование на компоненты типов **SET**, **SEQUENCE** и **CHICE**, и удалить конструкцию «**AUTMATICTAGS**» из предложения определения модуля.

г) Если открытый тип ограничен с использованием нотации подтипа TypeConstraint, то удалить ограничение, так как **ANY** и **ANY EFINE BY** не могут быть ограничены в ACH.1:1990 до типа, отличного от **ANY** или **ANY EFINE BY**.

д) Если определение типа **ENUMERATE** использует синтаксис «**identifier**» для EnumerationItem, то его следует изменить на «**identifier(number)**». Например, изменить

```
ENUMERATE {a, b, c, d}
```

на

```
ENUMERATE {a(0), b(1), c(2), d(3)}
```

е) Удалить все маркеры расширения (т.е. «...»). Например, изменить

```
SEQUENCE{
        i       IA5String,
        b       B LEAN,
        ...
        }
```

на

```
SEQUENCE{
        i       IA5String,
        b       B LEAN
        }
```

ж) Удалить из предложения определения модуля все конструкции **EXYENSIBILITYIMPLIE.**

и) Удалить все символы пробелов и новой строки из hstring и bstring; удалить все символы новой строки из cstring. Например, изменить

```
bBITSTRING : : = '0001110011010110111001111101010 1'B
o CTET STRING : : = '8F3CE4830192B3459325EF28AA3E700'H
pPrintableString : : = «Hello,
                        world|»
```

на

```
bBITSTRING : : = '000111001101011011100111110101 01'B
o CTET STRING : : = '8F3CE4830192B3459325EF28AA3E700'H
pPrintableString : : = «Hello, world|»
```

к) Преобразовать все нотации CharacterStringList в эквивалентные имcstring. Например, изменить

```
namePrintableString : : = {«Thisisalongstring,thatis
                        spread across two lines»}
```

на

```
namePrintableString : : =
«Thisisalongstring,thatisspreadacrosstwolines»
```

л) Преобразовать все ссылки на множества значений все ссылки на ограниченные типы. Например, изменить

```
AgesINTEGER: : = {1 | 4 | 7 . . 20}
```

на

```
AgesINTEGER: : = {1 | 4 | 7 . . 20}
```

Примечание 2 — Предпочтительнее использовать ограниченные типы вместо множеств значений, если только не используются интенсивно параметризация и классы информационных объектов, для которых полезно использование множеств значений.

м) Преобразовать все появления типа **INSTANCE F** в эквивалентный тип **SEQUENCE**. Например, изменить

```
A: : = INSTANCE F REGISTERE-AS
```

на

```
A: : = SEQUENCE{
        type-id                 BJECT I ENTIFIER,
        value                   [0] ANY EFINE BY type-id
}
```

н) Удалить все идентификаторы «**mantissa**», «**base**» и «**exponent**» из всех значений типа **REAL** и заменить все внутренние ограничения типа **REAL** на комментарий. Например, изменить

```
ten REAL: : = {mantissa 1, base 10, exponent 1}
ecimalReal: : = REAL (WITH C MP NENTS { . . . , base 10})
```

на

```
ten REAL: : = {1, 10, 1}
ecimalReal: : = REAL - - должно кодироваться по основанию 10
```

о) Заменить все случаи нотации значения **EXTERNAL** на эквивалентные ей в АСН.1:1990. Например, изменить

```
extern1990EXTERNAL: : = {
    direct-reference        {1 2 3 4 5 6},
    indirect-reference      3,
    encoding                single-ASN1-type:IA5String:«hello»
}
```

на

```
extern1994EXTERNAL: : = {
  identificationcontext-negotiation: {
    presentation-context-id       3,
    transfer-syntax               {1 2 3 4 5 6}
  },
  data-value      notation:IA5String:«hello»
}
```

п) Заменить все допустимые алфавиты АСН.1:1994 на эквивалентные им в АСН.1:1990. Например, изменить

```
UpperCaseAndSpacenly: : = PrintableString (FRM("A" . . "Z"|""))
```

и

```
UpperCaseAndSpacenly: : = PrintableString (FRM("A"|"B"|"C"|""|
                "E"|"F"|"G"|"H"|"I"|"J"|"K"|"L"|"M"|"N"|""|
                "P"|"Q"|"R"|"S"|"T"|"U"|"V"|"W"|"X"|"Y"|"Z"))
```

р) Изменить все выражения множеств АСН.1:1994, используемые в нотации подтипа, на эквивалентные им в АСН.1:1990. Например, изменить

```
PartNumber: : = NumericString(SIZE(8)^FRM("0" . . "9"))
```

на

```
PartNumber: : = NumericString(SIZE(8)) (FRM("0"|"1"|"2"|"3"|"4"|
                "5"|"6"|"7"|"8"|"9"))
```

с) Когда требование р) не может быть выполнено, так как результатом будет бесконечное множество, часть нотации подтипа АСН.1:1994, которая приводит к бесконечному множеству, следует заменить комментарием. Например, изменить

AllButZeroToTen : : = **INTEGER(ALLEXCEPT(0..10))**

на

AllButZeroToTen : : = **INTEGER** - - все целые значения, кроме 0 — 10

Применение приведенных выше инструкций к модулю АСН.1:1994 **ExampleModule** из перечисления 5) дает:

- - < Модуль АСН.1 версии 1990 года **ExampleModule** - -

ExampleModule{- - здесь должен быть допустимый идентификатор объекта-

EFINITI NS : : = **BEGIN**

Foo : : = **SEQUENCE{**

id1 BJECTI ENTIFIER,

syntax1 ANY EFINE BY id1}

Bar : : = **SEQUENCE{**

id2 BJECTI ENTIFIER,

syntax2ANY}

- - В модуле АСН.1:1994 **ExampleModule** синтаксис **syntax2** в **Bar** был определен как

- - **SEQUENCE F,** а не как открытый тип, и, таким образом, не мог быть преобразован в

- - **ANY EFINE BY.**

firstExtensionId BJECTI ENTIFIER : : = **{1 3 1 7 1 0 3 1 0 1}**

FirstExtensionInfo : : = **PrintableString**

- - **firstExtensionId** и **FirstExtensionInfo** должны использоваться в типе **Foo**, где **firstExtensionId**

- - является значением **id1 (BJECTI ENTIFIER),** которое указывает, что **syntax1** имеет тип

- - синтаксиса **FirstExtensionInfo(PrintableString).**

E N

10 Соглашения для АСН.1 и директив РОУО

В настоящем разделе вводятся соглашения для ясной идентификации спецификации и других пользовательских возможностей, связанных с шаблонами РОУО, и относящихся к ним модулей АСН.1. Это осуществляется с помощью директив в потоке. Настоящие соглашения могут быть полезны в качестве директив для компиляторов как АСН.1, так и РОУО. Если используются настоящие соглашения, то не требуется, чтобы автор изменял спецификации АСН.1 и РОУО для использования этих директив; а именно, директивы могут находиться в том же тексте, что и модули АСН.1 или шаблоны РОУО, но могут находиться и в других спецификациях. Если используются настоящие соглашения, то они не изменяют спецификаций АСН.1 или РОУО; а именно, эти директивы не изменяют синтаксиса спецификаций АСН.1 или РОУО. Настоящие соглашения являются рекомендуемыми, но не обязательными.

Предлагаемые директивы построены так, чтобы удовлетворить следующим требованиям:

- входные файлы с директивами (т.е. модулями АСН.1, библиотеками РОУО и прочими директивами) должны быть приняты компиляторами АСН.1 и РОУО, которые не распознают директив;

- входные файлы без директив должны быть приняты компиляторами АСН.1 и РОУО.

Примечание — Эти соглашения допускают расширения путем использования специфичных для реализации директив.

Каждая спецификация директивы является структурированным комментарием, содержащим единственную директиву, которая состоит из ключевого слова, квалифицированного областью действия, с последующими нулем или несколькими операндами. Регистр принимается во внимание. Следовательно, директивы рассматриваются как комментарии либыми компиляторами, которые этих директив не поддерживают.

Для описания директив использованы следующие соглашения:

- текст, набранный жирным шрифтом (например, **--<** и **ASN1**), должен вводиться так, как он приведен, без добавления пробелов или изменения регистра;

- текст, набранный курсивом (например, *ключевое_слово*), должен быть заменен подходящим текстом;

- факультативные элементы директив заключены в квадратные скобки (например, [операнды]);

- за элементом директивы, который может повторяться (произвольное число раз), стоят три точки (например, [операнды]...).

Общий формат директивы:

--<директива--

где директива есть

область_действия.ключевое_слово[операнд][,операнд]...

Таким образом, директива начинается с двумя последовательными дефисами и знаком «меньше, чем» (--<) и завершается знаком «больше, чем» и двумя последовательными дефисами (--). Между дефисами и знаками «меньше, чем» и «больше, чем» не должно быть пробелов. В результате компиляторами, которые не поддерживают эти директивы, они трактуются как комментарии ACH.1 или РОУО. Директива не может содержать комментарии в ACH.1.

Каждая конструкция --<-- является структурированным комментарием, содержащим единственную директиву, которая состоит из ключевого слова, квалифицированного областью действия, с последующими нулем или несколькими операндами. Регистр учитывается.

Директива может продолжаться на следующих строках, первыми и отличными от пробела символами которых должны быть два дефиса (--).

Операнды разделяются одним или несколькими последовательными пробелами или символами табуляции — для не схожих элементов, или запятой — для списка схожих элементов (таких как имена рабочих наборов). Пробелы или символы табуляции могут находиться до и после других элементов директивы. В данном контексте символы возврата каретки, новой строки и вертикальной табуляции не рассматриваются как пробелы и являются недопустимыми.

10.1 Соглашения для директив ACH.1

Для директив ACH.1 приняты следующие соглашения.

Директива может находиться в том же файле, что и элемент ACH.1. В таком случае директива может располагаться вне области действия модуля ACH.1 (до его начала или после его конца). Директива может находиться в теле модуля, в любом месте, где допустим пробел. В этом случае каждая директива должна помещаться сразу до того элемента ACH.1, к которому она применяется.

Для директивы ACH.1 символом область-действия является ASN1. В дополнение к нему могут быть определены (например, реализацией) другие символы область-действия.

Ключевым-словом может быть следующее:

-Version.

Операндом, в общем случае, могут быть:

- текстовая строка (без пробелов, запятых, последовательной пары дефисов и знака «больше, чем»);
- числовая строка (без пробелов, запятых, последовательной пары дефисов и знака «больше, чем»);
- cstring («xxxxx»), занимающая одну строку;
- {идентификатор-объекта};
- другие конструкции ACH.1, например bstring или hstring.
- текстовая строка (без пробелов, запятых, последовательной пары дефисов и знака «больше, чем»);
- числовая строка (без пробелов, запятых, последовательной пары дефисов и знака «больше, чем»);
- cstring («xxxxx»), занимающая одну строку;
- {идентификатор-объекта};
- другие конструкции ACH.1, например bstring или hstring.

10.1.1 Директива версии

Директива **Version** используется для указания того, по какой версии написан модуль ACH.1: ACH.1:1990 или ACH.1:1994.

Директива имеет следующий формат:

--<**ASN1.Version** версия имя_модуля [фио] --

Элементы, выделенные жирным шрифтом (например, **ASN1.Version**) пишутся так, как показано, а элементы, набранные курсивом, заменяются следующим образом:

версия — либо **1990**, либо **1994**, либо **1990, 1994**;

имя_модуля — имя модуля ACH.1;

фио — факультативное значение идентификатора объекта АСН.1, используемое для недвусмысленной идентификации модуля АСН.1.

Директива **Version** является единственной директивой для модуля АСН.1. Эта директива со значением операнда 1990, 1994 означает, что модуль АСН.1 соответствует как версии АСН.1:1990, так и версии АСН.1:1994. Компилятор может использовать свои сведения о любой из этих версий или об их общем подмножестве. Если директива версии для модуля АСН.1 не предоставлена, то реализация может использовать какой-либо иной способ для идентификации версии модуля, например, опции командной строки компилятора или распознавание синтаксиса, специфичного для конкретной версии (как макров АСН.1:1990 или информационные объекты в АСН.1:1994).

Примеры:

--<**ASN1.Version1990Attribute-ASN1Module**

--{**joint-iso-itu-tms(9)smi(3)part2(2)asn1Module(2)1**} --

--<**ASN1.Version1994SMModule**

--{**joint-iso-itu-tms(9)smi(3)part4(4)asn1Module(2)2**} --

10.2 Соглашения для директив РОУО

Для директив РОУО существуют следующие дополнительные соглашения.

Директива не может содержать комментарии в АСН.1.

Директива может находиться как в файле, отличном от содержащего шаблон РОУО, к которому она относится, так и в самом файле. Когда директива находится в том же файле, она может быть вне области действия документа РОУО (до начала или после его). Директива может находиться в теле документа, в любом месте, где допустим пробел.

Когда директива находится внутри документа, она должна располагаться сядом шаблона РОУО, к которому она относится. Для конкретного шаблона РОУО может существовать не более одной директивы конкретного вида, но могут существовать разные директивы для одного и того же шаблона. Например, на один и тот же шаблон могут ссылаться директивы Nickname и WorkingSet, но только по одной каждого вида. Для других элементов АСН.1 могут существовать другие директивы Nickname.

Символом область_действия является GMO. Дополнительно могут быть определены (например, реализацией) другие символы область_действия.

Ключевым_словом может быть одно из следующих:

- **Alias;**
- **ocument;**
- **Endocument;**
- **Version.**

Операндом, в общем случае, могут быть:

- текстовая строка (без пробелов, запятых, последовательной пары дефисов и знака «больше, чем»);
- числовая строка (без пробелов, запятых, последовательной пары дефисов и знака «больше, чем»);
- cstring («xxxxx»), занимающая одну строку;
- {идентификатор-объекта};
- другие конструкции АСН.1, например bstring или hstring.

10.2.1 Директива альтернатив

Директива **Alias** используется для предоставления альтернативных или алиасных идентификаторов документа РОУО. Эти алиасы используются для согласования ссылок между документами РОУО, когда используются короткие или противоречивые идентификаторы.

Формат директивы:

--<**GM.Alias** идентификатор_документа алиас_документа

--[,алиас_документа]... --

Элементы, выделенные жирным шрифтом (например **GM.Alias**) пишутся так, как показано, а элементы, набранные курсивом (например *идентификатор_документа*), заменяются так, как описано ниже.

Может присутствовать несколько разделенных запятыми элементов алиас_документа.

- идентификатор_документа — идентификатор документа РОУО в виде либо строки, взятой в кавычки ("), либо идентификатора объекта в фигурных скобках ({});
- алиас_документа — альтернативный идентификатор документа РОУО в виде либо строки, взятой в кавычки ("), либо идентификатора объекта в фигурных скобках ({}).

Примеры:

--<GM.Alias"CCITTRec.X.721(1992)|IS/IEC10165-2:1992"
--"Rec.X.721|IS/IEC10165-2:1992",
--"CCITTRec.X.721|IS/IEC10165-2",
--"Rec.X.721|IS/IEC10165-2",
--"CCITTRec.X.721(1992)",
--"CCITTRec.X.721(1992)|IS/IEC10165-2:1992",
--"M1" --
--<GM.Alias"RecomendationM.3100:1992"
--"Rec.M.3100:1992",
--"M.3100:1992",
--"M.3100" --

10.2.2 Директива документа

Директива ocumentобеспечиваетидентификатордокументаРОУО,являющийсялибосимвольной строкой, либо идентификатором объекта, либо тем и другим. Формат директивы может иметьоднуизследующихтрехформ:

--<GM.ocument строка_документа --
--<GM.ocument ИО-документа --
--<GM.ocument строка_документа ИО-документа --

Элементы, выделенные жирным шрифтом (например **GM.ocument**) пишутся так, как показано, а элементы, набранные курсивом (например *строка_документа*), заменяются следующим образом:

строка_документа—символьнаястрока,идентифицирующаядокументРОУО,вкавычках(");

ИО-документа—идентификаторобъектаАСН.1,присвоенныйдокументу,вфигурныхскобках({}).

ДирективаocumentдолжнанаходитьсядопервогошаблонаРОУОилимодуляАСН.1,образующегодокумент.Такимобразом,документРОУОрассматриваетсякаксостоящийизвсехшаблоновРОУОимодулейАСН.1отдирективыocumentдосоответствующейдирективыEndocument, илидоследующейдирективыocument,илидоконцафайла,содержащегоэтоттекстРОУО.

Длясохранениявфайлахтекстов РОУО,имеющихсоответствующиедирективы,существуют следующиеправила:

-КаждыйдокументРОУОдолжениметьдирективуocumentдляобеспечения«правильного» именидокумента.Еслиэтойдирективынет,тоимядокументанеопределеноилиобеспечивается каким-либодругимметодом.

-Директива ocumentдолжнанаходитьсявтомжесамомфайле,чтоитекстРОУО,ктоторомуонаприменяется.

Одноитожеимядокумента РОУОможетпоявитьсявнесколькихдирективахocumentприменительнокнесколькимблокамтекста РОУО,такимкаксразличныефайлы.Вэтомслучаевесь текстРОУОвразныхфайлахрассматриваетсякакчастьодногоитогожедокумента РОУО.Это аналогичнопространствуименвС++,котороепозволяетвнесколькихразныхфайлахзаголовков содержатьматериал,определенныйводномитомжепространствеимен.

Вобщемслучаедолженбытьединственныйдокумент РОУОнаодинвходнойфайл.Файл можетсодержатьнесколькодокументовРОУОтольковтомслучае,когдаонсодержитнесколько согласованныхдирективocumentиEndocument.

Пропуски(одинилинесколькопоследовательныхпробеловидисимволовтабуляции)неучитываютсявидентификаторе_документаивалиасе_документа.

Примеры:

--<GM.ocument"CCITTRec.X.721(1992)|IS/IEC10165-2:1992" --
--<GM.ocument"RecomendationM.3100:1992" --
--<GM.ocument"P1LibraryVol.4" --
--<GM.ocument{iso(1)21243605011513 1} --
--<GM.ocument"IIMCM1BTranslation"{IS (1)21243605011513 1} --

10.2.3 Директива конца документа

ДирективаEndocumentотмечаетконец"документа"РОУО.Форматдирективыможетиметь одноизследующихчетырехформ:

--<**G M . End ocument** - -

--<**G M . End ocument** строка_документа --

--<**G M . End ocument** ИО-документа --

--<**G M . End ocument** строка_документа ИО-документа --

Элементы, выделенные жирным шрифтом (например **G M . End ocument**), пишутся так, как показано, а элементы, набранные курсивом (например *строка_документа*), заменяются следующим образом:

строка_документа — символьная строка, идентифицирующая документ РОУО, в кавычках (");

ИО-документа — идентификатор объекта АСН.1, присвоенный документу, в фигурных скобках ({}).

Документ РОУО рассматривается как состоящий из всех шаблонов РОУО и модулей АСН.1 от директивы ocument до соответствующей директивы Endocument, или до следующей директивы ocument, или до конца файла, содержащего этот текст РОУО. Таким образом, использование директивы Endocument не обязательно. Если же директива Endocument используется, то она должна следовать за последним шаблоном РОУО или модулем АСН.1, образующем документ. Строка_документа и (или) ИО-документа в директиве **End ocument** должна(ы) соответствовать предшествующей директиве **ocument.**

Примеры

--<**G M . End ocument "** - -

--<**G M . End ocument "CCITT Rec. X. 721 (1992) \| IS / IES 10165-2:1992"** - -

--<**G M . End ocument "Recomendation M. 3100:1992"** - -

--<**G M . End ocument " P1 Library Vol. 4"** - -

--<**G M . End ocument {iso (1) 2 12436050115131}** - -

--<**G M . End ocument "IIMC MIB Translation" {iso (1) 2 12436050115131}** - -

10.2.4 Директива версии

Директива **Version** используется для указания того, какая версия РОУО использована в документе. Формат директивы может иметь одну из следующих двух форм:

--<**G M . Version** версия - -

--<**G M . Version** версия идентификатор_документа --

Элементы, выделенные жирным шрифтом (например **G M . Version**), пишутся так, как показано, а элементы, набранные курсивом (например *версия*), заменяются следующим образом:

- версия — номер, указывающий версию РОУО, в соответствии со следующей таблицей:

Индекс	Определение версии
1	РОУО, 1992
1.1	Дополнение 1: конструкция **SET-BY-CREATE**
1.2	Дополнение 2: конструкции **N - M 1F1, ASN. 1: 1994** и директивы
1.3	Дополнение 3: использование Z в поведении класса управляемых объектов

Новые версии определяются при каждом обновлении РОУО (т. е., пересмотре, поправках, дополнениях). Обеспечение данной версии является кумулятивным. Это значит, что любой документ РОУО, действительный для версии n, должен быть допустимым и для предшествующих версий. Например **G M . Version 1. 2** указывает, документ РОУО поддерживает версию РОУО 1992 (версия 1), дополнение конструкции SET-BY-CREATE (версия 1.1) и дополнение конструкций NO-MO1F1, ASN.1:1994 и директив (версия 1.2).

- идентификатор_документа — идентификатор документа РОУО в виде символьной строки в кавычках (") или идентификатора объекта АСН.1 в фигурных скобках ({}).

Первая форма директивы (без идентификатора_документа) может находиться в документе РОУО, но до любого шаблона РОУО.

Примеры

--<**G M . Version 1 "CCITT Rec. X. 721 (1992) \| IS / IEC 10165-2:1992"** - -

--<**G M . Version 1. 1 "Recomendation M. 3100"** - -

--<**G M . Version 1. 2** - -

ПРИЛОЖЕНИЕА
(справочное)

Примеры

Примеры,приведенныевнастоящемприложении,иллюстрируютиспользованиешаблонов,анесодержатполезныедляреализацииопределения.Вчастности,определенияповедениядовольноискусственные.Примеры,имеющиепрактическоезначениедляразработкиопределенийклассовуправляемыхобъектов,приведенывГОСТРИСО/МЭК10165-2.

А.1Определениеклассауправляемыхобъектов

```
exampleObjectClassMANAGED OBJECTCLASS
   DERIVED FROM "CCITT Rec.X.721(1992) | IS/IEC 10165-2:1992":top
   CHARACTERIZED BY examplePackage2;
   CONDITIONAL PACKAGES
      examplePackage1   PACKAGE
         ACTIONS        qSResetAction,
                        activate;
         NOTIFICATIONS  communicationError;
      REGISTERED AS{joint-iso-ccitt ms(9) smi(3) part4(4) package(4) examplepack1(0)};
      PRESENT IF|применяетсякласссоответствия2нижележащегоресурса,
            описанныйвИСО/МЭКXXXX|;
REGISTERED AS{joint-iso-ccitt ms(9) smi(3) part4(4)
            managedObjectClass(3) exampleclass(0)};
```

Примечание—Этотшаблонисполь зуетвозможностьвстроенногодокументированияусловного пакета.

А.2Определениесвязыванияимен

```
exampleNameBindingNAMEBINDING
         SUBORDINATE OBJECTCLASS exampleObjectClass;
         NAMED BY
         SUPERIOR OBJECTCLASS "CCITT Rec.X.721(1992) | IS/IEC 10165-2:1992":system;
         WITH ATTRIBUTE objectName;
         BEHAVIOUR
            containmentBehaviourBEHAVIOUR
               DEFINED AS |Влюбомэкземпляре"CCITT Rec.X.721(1992)
                          | IS/IEC 10165-2:1992":systemможетсодержаться
                          неболеетрехэкземпляров exampleObjectClass|
            ;
         ;
         CREATE WITH-AUTMATIC-INSTANCE-NAMING createErrorParameter;
         DELETE DELETES-CONTAINED-OBJECTS;
REGISTERED AS{joint-iso-ccitt ms(9) smi(3) part4(4) nameBinding(6) examplenb(0)};
```

Примечание—Этотшаблонисполь зуетвозможностьвстроенногодокументированияповедения.

А.3Определениепараметров

```
pUHeaderPARAMETER
         CONTEXT            EVENT-INFO;
         WITH SYNTAX        ParameterModule.PUString;
         BEHAVIOUR
            pUHeaderBehaviour BEHAVIOUR
               DEFINED AS  |ЗаголовокПБД.
                           ПередаетсявполеПОИУeventInfo.|
            ;
         ;
REGISTERED AS{joint-iso-ccitt ms(9) smi(3) part4(4) parameter(5) pduheaderparam(0)};
createErrorParameterPARAMETER
   CONTEXT          SPECIFIC-ERROR;
   WITH SYNTAX      ParameterModule.ErrorInfo1;
   BEHAVIOUR
```

```
        createErrorBehaviour BEHAVIUR
          EFINE AS|Еслиовмещающемуправляемомобъектесуществуетмаксимальновозможноечисло
                  экземпляровexamplebjectClass,топопыткасозданиядополнительныхэкземпляров
                  приведеткквозвратусообщенияобошибкеПОИУ«Отказобработки»,вкоторомполе
                  SpecificErrorInfoимеетвид
                  SpecificErrorInfo : : = SEQUENCE{
                  errorid BJECTIENTIFIER,
                  errorinfoANY EFINE BYerrorid}
                   BJECTIENTIFIER, передаваемый вerrorid,долженбытьзначениемпараметра,
                  подкоторымонозарегистрировано.Тип,передаваемыйвerrorinfo,долженбытьиден-
                  тифицированнымконструкциейWITHSYNTAXэтогоопределенияпараметра.Значе-
                  ние,передаваемоетипом,указываетчислоэкземпляровэтогоклассауправляемых
                  объектов,которыевданныймоментсуществуютвовмещающемуправляемом
                  объекте.|
        ;
    ;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)parameter(5)createrror(1)};
serviceProviderErrorResponseReason      PARAMETER
    CNTEXT          ACTIN-REPLY;
    WITHSYNTAX      ParameterModule.ServiceProviderErrorResponseReason;
    BEHAVIUR
        serviceProviderErrorResponseReasonBehaviour BEHAVIUR
          EFINE AS|ВозвращаетсявполеresponsePARAMETERSПОИУactionReplyInfo,еслиresponceCode
                  имееттекущеезначениеserviceProviderErrorResponse.|
        ;
    ;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)parameter(5)sperrorrsp(2)};
```

Примечание—Этотшаблониспользуетвозможностьвстроенногодокументированияповедения.

А.4Определениепакета

```
examplePackage2 PACKAGE
      BEHAVIUR          exampleClassBehaviour;
      ATTRIBUTES        objectName
                            GET,
                        qS-Error-Cause
                            GET,
                        qS-Error-Counter
                          PERMITTERVALUESAttributeModule.QSCounterRange
                          REQUIREVALUESAttributeModule.QSCounterRange
                            GET;
      ATTRIBUTEGRUPSqS-Group;
      NTIFICATINS protocolError;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)package(4)examplepack2(1)};
```

Примечание—Таккакэтотшаблоннеиспользуетсявкачествеусловногопакета,токонструкция REGISTEREASнеявляетсястрогообязательной,нопрощевключитьрегистрациювовремяспецификации, чемдобавлятьеепозже,еслионастанетнеобходимойдляиспользованияэтогопакетавкачествеусловного.

А.5Определенияатрибутов

```
objectName ATTRIBUTE
      WITHATTRIBUTESYNTAXAttributeModule.bjectName;
      MATCHESFR             EQUALITY;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)attribute(7)objectname(0)};
qS-Error-CauseATTRIBUTE
      WITHATTRIBUTESYNTAXAttributeModule.QSErrorCause;
      MATCHESFR             EQUALITY;
      BEHAVIUR              qSErrorBehaviour;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)attribute(7)qoscause(1)};
qS-Error-CounterATTRIBUTE
      WITHATTRIBUTESYNTAXAttributeModule.QSErrorCounter;
      MATCHESFR             EQUALITY, RERING;
      BEHAVIUR              qSCounterBehaviour;
      REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)attribute(7)qoscount(2)};
```

А.6Определениеатрибутивнойгруппы

```
qS-GroupATTRIBUTE GRUP
    GRUPELEMENTSqS-Error-Cause,qS-Error-Counter;
    ESCRIPTIN    |Атрибутивнаягруппа,котораявключаетвсебявсеатрибуты,относящиесяккаче-
                 ствууслугивклассеуправляемыхобъектов.|;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)attributeGroup(8)qosgroup(0)};
```

А.7Определениядействий

```
qSResetActionACTIN
      BEHAVIUR
       resetBEHAVIUR
         EFINE AS   |<Определениеповеденияresetиеговлияниянаработууправляемогообъекта
                    и т. п. |
        ;
      ;
M E C NFIRME ;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)action(9)reset(0)};
```

Примечание—Этоопределениедействияиспользуетвозможностьвстроенногодокументирования поведения.Абстрактныесинтаксисыдлявывозаиответанеопределены.

```
activateACTIN
    BEHAVIUR
        activateBehaviourBEHAVIUR
          EFINE AS  | Делаетуправляемыйобъектдоступнымдляработы.Еслийдействиеуспешное,
                    товозвращаетсязначениеsuccessResponseвпараметреresponseCode ПОИУ
                    actionReplyInfo.Еслидействиенеудачноеиз-запроблемепоставщикомнижесле-
                    дующихуслуг,тогresponseCodeустанавливаетравнымserviceProviderErrorResponse
                    ивозвращаетсяпараметрserviceProviderErrorResponseReasonдляуказанияпричи-
                    ныпроблемы.|
        ;
    ;
    M E C NFIRME ;
    PARAMETERS          serviceProviderErrorResponseReason;
    WITHREPLYSYNTAX  ActionModule.ActivateReply;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)action(9)activate(1)};
```

А.8Определениясообщений

```
communicationErrorNTIFICATIN
      BEHAVIUR          communicationErrorBehaviour;
      WITHINFRMATINSYNTAX   NotificationModule.ErrorInfo;
      WITHREPLYSYNTAX       NotificationModule.ErrorResult;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)notification(10)commerror(0)};
protocolErrorNTIFICATIN
      BEHAVIUR
       protocolErrorBehaviour        BEHAVIUR
          EFINE AS  |Создается,когдапротокольныйобъектполучаетПБД,которыйявляетсянедо-
                    пустимымилисодержитошибкупротокола.Сообщениевключаетвсебязаго-
                    ловокполученногоПБД.|
          ;
      ;
      PARAMETERS      pUHeader;
      WITHINFRMATINSYNTAXNotificationModule.ProtocolError;
REGISTEREAS{joint-iso-ccittms(9)smi(3)part4(4)notification(10)protoerror(1)};
```

Примечание—Этотшаблонисполъзуетвозможностьвстроенногодокументированияповедения.

А.9Определенияповедения

```
qSCounterBehaviour BEHAVIUR
    EFINE AS  | Атрибутqserrorcounterявляетсясчетчиком,которыйувеличиваетсянаединицупри
              каждомпоявлениишибкиКУ.Егозначениеявляетсяположительнымцелым,диапазон
              которогоспецифицируетсявпакетах,ссылающихсянаэтоопределение.Когдасчетчик
              достигаетмаксимальногозначения,следующееувеличениевызываетвозвратзначения к
              нулю.|;
```

qSErrorBehaviour BEHAVIUR
EFINE AS | Атрибут **qSErrorCause** указывает причину отказа КУ, связанной с управляемым объектом.

Примечание — Взаимоотношения между допустимыми значениями атрибута и работой самого управляемого объекта определяются определениями поведения, связанного с определением класса управляемых объектов.|;

communicationErrorBehaviour BEHAVIUR
EFINE AS | Сообщение CommunicationError создается классом управляемых объектов, когда управляемый объект обнаруживает ошибку коммуникации. Сообщение может содержать любую комбинацию параметров **ProbableCause, Severity, TrendIndication, BackedUpStatus, iagnosticInfo, ProposedRepairAction, TresholdInfo, StateChange u therInfo.**

Примечание — Точное определение того, что представляет собой ошибка коммуникации и используемых значений параметров, специфично для классов управляемых объектов. На практике это определение поведения может ссылаться на базовые стандарты.|;

exampleClassBehaviour BEHAVIUR
EFINE AS | <...Описание поведения класса управляемых объектов, включая описания:
- как атрибуты получают конкретные значения и какой они имеют смысл;
- какие обстоятельства вызывают создание сообщений;
- и т. п.. ;

А.10 Модули АСН.1

AttributeModule {joint-iso-ccittms(9) smi(3) part4(4) asn1Module(2) attributes(0)};
EFINITI NS: : = BEGIN
bjectName: : = GraphicString
QSErrorCause: : = INTEGER {
responceTimeExcessive (0),
queueSizeExceeded (1),
bandwidthReduced (2),
retransmissionRateExcessive (3) }
QSErrorCounter: : = INTEGER
QSCounterRange: : = QSErrorCounter {0..4294967296} — Диапазон 32 бита.
E N
NotificationModule {joint-iso-ccittms(9) smi(3) part4(4) asn1Module(2) notifications(1)};
EFINITI NS: : = BEGIN
IMP RTS
ProbableCause, PerceivedSeverity, TrendIndication, BackedUpStatus, ProposedRepairActions, ThresholdInfo, ManagementExtension
FRMAttribute.ASN1Module {joint-iso-ccittms(9) smi(3) part2(2) asn1Module(2) 1};
ErrorInfo: : = SET {
[0] ProbableCause PTI NAL,
[1] PerceivedSeverity PTI NAL,
[2] TrendIndication PTI NAL,
[3] BackedUpStatus PTI NAL,
[4] ProposedRepairActions PTI NAL,
[5] ThresholdInfo PTI NAL,
[6] therInfo PTI NAL }
ErrorResult; : = NULL
therInfo : : = SET F ManagementExtension
ProtocolError : : = SET F ManagementExtension
E N
ActionModule {joint-iso-ccittms(9) smi(3) part4(4) asn1Module(2) actions(2)}
EFINITI NS: : = BEGIN
IMP RTS
perationalState, ManagementExtension
FRMAttribute.ASN1Module {joint-iso-ccittms(9) smi(3) part2(2) asn1Module(2) 1};
ActivateReply: : = SEQUENCE {
operationalStatus [0] perationalState,
responseCode [1] INTEGER {successResponse (0),
serviceProviderErrorResponse (1)},
responseParams [2] SET F ManagementExtension PTI NAL}
E N

```
ParameterModule{joint-iso-ccittms(9)smi(3)part4(4)asn1Module(2)parameters(3)}
EFINITI NS: : = BEGIN
ErrorInfo1: : = INTEGER
ServiceProviderErrorResponseReason: : = ENUMERATE {
                    insufficientResources          (0),
                    provideroesNotExist            (1),
                    providerNotAvailable           (2),
                    requiredServiceNotAvailable    (3) }
P UString: : = CTETSTRING
E N
```

ПРИЛОЖЕНИЕВ
(справочное)

Руководствопоприменению Zприформализацииповеденияуправляемыхобъектов

В.1 Введение

Настоящее приложение содержит техническое руководство по применению языка Z для определения поведенияуправляемыхобъектов,которыеподдерживаютадминистративноеуправлениевзаимодействиемВОС. Оноявляетсясправочным,а не нормативным. Оно не требует, чтобы для спецификации поведения УО использовалисьметодыформальногоопределения(МФО).Еслитребуется,чтобыиспользовались МФО,тотем самым нетребуется,чтобыиспользовалсяZ;могутиспользоватьсядругиеязыки,такиекакSL.Даженогда должен использоватьсяZ,возможныидругиеспособыспецификациионведенияУО.

Формальныеспецификацииповедения УОмогутбытьнепосредственнополезными,таккакониясныи недвусмысленны.Процесссозданияформальнойспецификациизаставляетпроводитьподробныйанализповедения.Следовательно,формальнаяспецификациюжетиспользоватьсякакинструментдляидентификации и исправления двусмысленностей, которые могли остаться невыявленными в спецификации, полностьюоснованной на естественном языке. По этим причинам формальная спецификация может быть полезной для совершенствованияспецификацииповедения.

Настоящееприложениесодержитиллюстрированный пример,который показываетсовременнуюпрактику использования Z. Его целью является установление тех общих основ и понимания этого конкретного формального подхода, которые помогут достичь согласованности в сходных разработках. Он предоставляет полезную начальную точку для пользователей РОУО, которые хотят использовать Z для улучшения своих спецификаций.

Приложение предназначенодля пользователей, знакомых с основными понятиями спецификаций управляемыхобъектов, использующихшаблоны РОУО, ис языкомZ.

Внастоящемприложениитермин«управляемыйобъект»(или«УО»)используетсядляопределенияклассауправляемыхобъектов,данныхсиспользованиемшаблоновРОУО.

В.2 Вопросыязыка

НотацияZявляетсяформальноопределеннойнотацией,основанойнатеориимножествиисчислении предикатов.Онаимеетдостаточномощныесредствадляописанияотдельныхклассовуправляемыхобъектов.

ОднаковZнесуществуетпонятиеинкапсуляции.СпецификацияZобычносостоитизмоделинекоторогосостоянияисовокупностиоперацийдляизмененияэтогосостояния.ВZнетметодовдляделениясостояния иегоопераций наотдельные модули и их повторного использования в других спецификациях. Очевидным следствием этого является необходимость описания управляемых объектов, которые наследуют переменные и поведение из других определений классов управляемых объектов.

Эффектнаследованияможетбытьполученметодомвключенияхемыценойнекоторойпотериясности. ВовсехостальныхотношенияZподходитдлявыраженияотдельныхклассовуправляемыхобъектов.

В.3 Элементы, подлежащиепреобразованию

Требуетсяпреобразоватьопределениеповедения(илиегочасть)изнеформальногоописаниявописание Z. Степень, в которой должны быть формализованы оставшиеся части шаблонов РОУО, зависит, главным образом, от потребностей разрабочика спецификации.

Шаблоны РОУО уже включают в себя полуформальные определения типов данных в АСН.1. Можно написатьспецификациюZ,используяэтиопределенияАСН.1вкачествеосновыдлятипов,используемыхв спецификации Z, чтоэкономитсущественнуючастьработы.

Однакоесли спецификация написанатаким образом, тобольшой проблемой для разработчика становитсягарантияеесинтаксическойкорректности.ВспецификацияхZбезопределенийАСН.1можноисполь-

зовать существующие средства Z, которые обеспечивают поддержку проверки синтаксиса и статической семантики спецификации Z.

Таким образом, можно улучшить определения поведения, используя Z без переписывания типов данных АСН.1, но существенные преимущества могут быть получены при полном преобразовании типов данных АСН.1 в Z. Примеры того, как преобразовываются основные типы АСН.1 в Z, приведены в В.7.1.

В.3.1 Преобразование из шаблонов РОУО в Z

Ниже дано общее руководство по преобразованию управляемых объектов из их неформального описания настоящего стандарта в Z. Такое преобразование может быть осуществлено только неформально, т.к. формальное преобразование требует, как минимум, чтобы исходный и целевой языки были формализованы. Более того, как и любое отображение двух различных языков, оно ограничено некоторым несоответствием между конструкциями этих языков. Эта проблема усиливается, когда один из языков оказывается неформальным или включает в себя неформальные компоненты.

Ниже перечислены некоторые основные характеристики шаблонов, определенных в настоящем стандарте, с указанием их отличий от соответствующих конструкций Z. Попутно предлагаются общие методы разрешения несогласованности или рекомендации по их разрешению в конкретных случаях.

В настоящем приложении основное внимание уделяется тому, что необходимо описывать в поведении управляемого объекта. Дополнительная информация по преобразованию типов АСН.1 приведена в В.6.

В.3.2 Типы данных

Первым шагом является переписывание типов данных настоящего стандарта в виде типов Z. АСН.1 предоставляет полезные возможности по созданию типов данных, но их построение ориентировано на описание потоков данных, передаваемых между системами.

В АСН.1 построение типа определяется в виде списка. В Z типы являются множествами. Хотя можно моделировать построение типа АСН.1 в виде последовательности в Z, иногда бывает более естественным рассмотреть операции, доступные над типами АСН.1, и отобразить их в типы Z, чтобы более ясно описывать их структуру. Типы АСН.1 «последовательность» и «множество» могут быть отображены в тип Z «кортеж». Тип АСН.1 «последовательность-из» может быть отображен в тип Z «последовательность». Тип АСН.1 «множество-из» может быть отображен в тип Z «множество».

АСН.1 включает в себя специальное обеспечение кодирования, такое как метки и значения по умолчанию. Они не обязательно должны быть представлены в Z, т.к. не влияют на определение поведения.

В пункте В.6.2 приведена дополнительная информация о преобразовании типов АСН.1.

В.3.3 Атрибуты УО

Управляемые объекты определены таким образом, что имеют некоторые атрибуты административного управления. Эти атрибуты имеют типы данных, определенные в АСН.1. Им присвоены идентификаторы объектов. Они могут иметь свойства согласования значений. Предлагаются два способа моделирования таких атрибутов:

- простые типы атрибутов и
- типы атрибутов как схемы.

Простейшим подходом является представление атрибута УО как переменной Z с соответствующим типом данных. Тогда отдельно необходимо определение постоянной, представляющей идентификатор объекта для этого атрибута. Эта постоянная будет связана с атрибутом только по соглашению. Когда для атрибута определена операция согласования, можно использовать свойство фактического фиксированного согласования. Пример приведен в В.6.3.

Можно включить все эти свойства атрибута в единственном типе схемы, который будет типом переменной Z, моделирующей атрибут УО. Схема будет включать в себя значение атрибута, идентификатор объекта и свойство согласования. Пример приведен в В.6.4. Когда требуются правила согласования, отличные от равенства, можно определить параметр согласования как отношение Z над типом значения атрибута. Этот подход допускает формальное представление произвольных конкретных правил согласования, что может быть важным для области действия, фильтрации и выбора объекта.

Трудно моделировать тип ANY АСН.1 в Z. В некоторых случаях можно дать список значений атрибута. Таким образом полная формальная модель, вероятно, потребует комбинации свободного типа Z с уже определенными типами атрибутов. Примеры приведены в В.6.1 и В.6.5.

Идентификаторы объектов формально моделируются множеством:

[OBJECTI]

В.3.4 Другие идентификаторы объектов

Многие элементы, кроме атрибутов, имеют идентификаторы объектов. Удобно ввести их все как постоянные в аксиоматические определения. Можно использовать соглашение о дополнении их суффиксом «Oid». Такие постоянные будут необходимы для классов, пакетов и сообщений.

Пример

packagesPackageOid : *OBJECTI*
allomorphsPackageOid : *OBJECTI*
topClassOid : *OBJECTI*

В.3.5 Наследование и совместимость

ZможетбытьиспользовандляпостроенияиерархийнаследованияУОпутемвключенияхемывмодель наследованияиспециализации.ТакимобразомкорректномоделируетсяповедениеклассовиподтиповУО,но это не позволяет явно выразить взаимоотношение строго подтипа, которое реально существует. Необходим язык, который явно моделирует наследование.

Таким образом Z может использоваться для удовлетворительного определения отдельных УО, но для того, чтобы можно было говорить о наследовании и совместимости, необходимы дополнительные возможности языка.

Наследованиенеподдерживаетсявz.Ономожетбытьсмоделировановключениемсхемвсхемысостояния.

Определение наследования УО требует, чтобы подклассы были совместимы. Но не требует, чтобы подклассы были подтипами в Z. Обычно УО может сообщить свой фактический класс. Так как атрибут «фактический класс» всегда содержит фактический класс объекта, то подкласс не может сообщить класс своего суперкласса. Следовательно, подкласс не может демонстрировать при возврате значения атрибута «фактический класс» то же самое поведение, что и его суперкласс (т.е. они не взаимозаменяемы), даже если их поведение алломорфно. Следовательно, подклассы управляемых объектов не эквиваленты подтипам в Z, где подтип будет демонстрировать то же самое поведение, что и супертип. Однако подкласс демонстрирует очень мало «неподходящее» поведение.

Можно показать, что определенное в настоящем стандарте наследование УО позволяет специфицировать поведение родителя, которое будет несовместимо с поведением потомков. Так как такое неподходящее поведение очень ограничено, то класс УО может быть представлен двумя спецификациями классов. Одна из этих спецификаций охватывает поведение, которое может демонстрировать любой экземпляр или любое расширение УО, другая — поведение, демонстрируемое только экземплярами совместимого класса, но не расширениями. Эта последняя спецификация является как раз той, которая реализуется для получения полного поведения фактического экземпляра УО.

В.3.6 Пакеты

Многие части функциональных возможностей класса могут присутствовать в одних конкретных УО и отсутствовать в других. В настоящем стандарте это описывается с помощью группирования функциональных возможностей в условные пакеты. Тогда каждый УО реализует подходящие пакеты. В Z функциональные возможности не могут быть предоставлены таким способом, но можно сделать поведение УО зависящим от того, какие пакеты реализованы. Это можно сделать непосредственным образом потому, что УО должен содержать атрибут административного управления, называемый «пакеты», в котором перечислены идентификаторы объектов фактически реализованных пакетов. Таким образом для моделирования поведения условного пакета само поведение становится зависящим от присутствия идентификатора пакета в атрибуте «пакеты».

В.3.7 Класс

Для определения класса УО необходимо представить его атрибуты и операции. Атрибуты становятся частью схемы состояния Z, а операции становятся схемами операций Z.

В.3.7.1 Атрибуты

Атрибуты управляемого объекта объявляются в схеме состояния. Каждый атрибут имеет заданный тип, который может быть типом, объявленным в части АСН.1 шаблона РОУО или в полной формальной модели Z.

В.3.7.2 Операция Get

Управляющий может запросить выполнение операции Get над УО. В определении УО ИУМ-Get имеет много параметров, но большинство из них относится к управлению доступом, выбору объектаит.п. В конкретном экземпляре Get может быть смоделирована на границе управляемого объекта, игнорируя указанные вопросы и заменяя единственную операцию Get несколькими операциями Get<имя>, где <имя> — единственный атрибут.

В.3.7.3 Операция GetAll

Операция GetAll, которая не имеет входных параметров, может быть смоделирована таким же образом. Она возвращает непустой набор значений атрибутов.

В.3.7.4 Операция Replace

Установки в конкретном УО запрашиваются операцией УО ИУМ-Set. В настоящей спецификации операции Replace моделируются как видимые на границе УО. Эти операции относятся к установкам атрибутов, установкам значений по умолчанию, добавлению и удалению значений. Следовательно, для представления каждого из этих изменений должна быть специфицирована схема Z.

В.3.7.5 Сообщения

Являются незапрошенными посланиями УО для отчетов о событиях в УО. Они моделируются не как операции, а как результаты операций, происходящих над УО. Таким образом, любая операция (вызванная управляющим или внутренне, ресурсом) может создать выходной результат и, если операция приводит к сообщению, то оно будет частью выходного результата операции. Это означает, что выходным результатом схемы операции Z, которая может вызвать сообщения, должно быть множество сообщений. Случай, когда сообщение не создается, может быть представлен как пустое множество в качестве выходного результата.

Данные в сообщении состоят из типа события EventType, который является идентификатором объекта его стандартного определения. Идентификатор объекта может быть определен как постоянная, конкретные данные — как тип схемы. Поведение сообщения включается в каждый объект, который может это сообщение создавать.

В.3.7.6 Действия

Являются операциями, осуществляемыми управляющим над УО. Они естественным образом представляются операциями Z.

В.3.8 Спецификация системы объектов

Во вставшейся части настоящего приложения описано представление поведения единичного объекта. При рассмотрении создания и удаления объектов, связывания имен, вмещения и наименования необходимо описать состояние системы, в которой находятся объекты. Создание и удаление объекта могут быть представлены как изменения состояния этой системы. Связывание имени вмещением могут быть представлены отношением на множестве объектов. Наименование может быть определено в терминах этого отношения.

В.4 Пример

В настоящем подразделе приведен пример определений высшего класса УО и атрибутов административного управления. Так как в данном руководстве основное внимание уделяется моделированию поведения, то в этом подразделе не показано создание типов Z и типов АСН.1. Полное формальное определение дано в В.7.

В.4.1 Класс top

Первым классом, который должен быть определен, является *top*, представляющий собой основного родителя (в иерархии наследования) для всех УО.

Класс *top* имеет четыре атрибута административного управления: *objectClass, packages, allomorphs* и *nameBinding*. В атрибуте *objectClass* хранится идентификатор объекта класса, в *packages* — идентификаторы объектов реализованных пакетов, в *nameBinding* — идентификатор объекта связывания имен, использованного при реализации объекта, а в *allomorphs* — идентификаторы объектов классов, с которыми объект может быть алломорфен. Так как атрибуты административного управления могут находиться в пакетах, атрибуты, присутствующие в УО данного класса, могут изменяться. Это моделируется включением дополнительного атрибута *attributes*, в котором хранятся идентификаторы объектов атрибутов, фактически реализованных в данном УО. Все атрибуты, присутствующие в классе *top*, фиксированы на протяжении жизни конкретного УО.

Интерфейсы в Z не моделируются явно и, таким образом, не возможно формально определить, какие операции вызываются внутренне управляющим, какие — внешне:

TopState

$$
\begin{array}{|l}
\hline
allomorphs : \mathbb{F}\, OBJECTI \\
objectClass : OBJECTI \\
nameBinding : OBJECTI \\
packages : \mathbb{F}\, OBJECTI \\
attributes : \mathbb{F}\, OBJECTI \\
\hline
\{objectClassOid, nameBindingOid\} \subset attributes \\
allomorphsPackageOid \in packages \Rightarrow allomorphsOid \in attributes \\
packagesPackageOid \notin packages \\
packages \neq \emptyset \Rightarrow packagesOid \in attributes \\
\hline
\end{array}
$$

attributes является не атрибутом УО, а новым компонентом состояния, определенным для удобства. В нем перечислены атрибуты, которые содержатся в УО. Следовательно, инвариант требует, чтобы он содержал идентификаторы объектов подходящих атрибутов, как описано в В.3.3 (и определено в В.7.4). Атрибуты *objectClass* и *nameBinding* являются обязательными. Атрибут *packages* присутствует только в том случае, когда реализован любой зарегистрированный пакет, отличный от *packagesPackage*. В последнем случае этот атрибут обозначает *allomorphsPackage*.

Операция *TopGetNameBinding* опрашивает УО и возвращает значение атрибута *nameBinding* без изменения *TopState*. Операция *TopGetNameBinding* вызывается управляющим:

TopGetNameBinding

$$
\begin{array}{|l}
\hline
\Xi TopState \\
result! : OBJECTI \\
\hline
result! = nameBinding \\
\hline
\end{array}
$$

Операция *TopGetAllomorphs*, *TopGetObjectClass* и *TopGetPackages* здесь не определяются. Нет операции для получения *attributes*, так как *attributes* не является атрибутом УО, определенным в шаблоне РОУО.

Операция *TopGetAll* получает значения атрибутов объекта. Она всегда возвращает значения objectClass и nameBinding. Если присутствуют условные пакеты или алломорфизм, то возвращаются и они. Операция *TopGetAll* вызывается управляющим:

TopGetAll

Ξ *TopState*
result| : *PAttributeValues*

 # attributes = *# result*|
ObjectClassValue objectClass ∈ *result* |
NameBindingValue nameBinding ∈ *result* |
PackagesOid ∈ *attributes* ⇒ *packagesValue packages* ∈ *result* |
AllomorphsOid ∈ *attributes* ⇒ *allomorphsValue allomorphs* ∈ *result* |

Конструкция *TopEventReport* является способом моделирования сообщений. *TopEventReport* возникает спонтанно и представляет способ отчетов о событиях, которые не контролируются управляющим:

TopEventReport

Ξ *TopState*
notification| : *EventInfo*

B.4.2 Класс StateManagement

Не отражает никакой конкретный класс УО. Он отражает поведение любого объекта, содержащего любой из стандартных атрибутов *administrativeState*, *operationalState* и *usageState*. В рамках данного подхода он удобен для понимания включения этих атрибутов как наследования и не является примером для использования.

Схема состояния включает в себя определения *TorState* и предикаты и специфицирует некоторые дополнительные переменные и соединяющие предикаты:

StateManagementState

TopState
administrativeState:
AdministrativeState
operationalState : *OperationalState*
usageState : *UsageState*

operationalState = *disabled* ⇒ *usageState* = *idle*
administrativeState = *locked* ⇒ *usageState* = *idle*
usageState = *idle* ⇒ *administrativeState* ≠ *shuttingown*

Класс StateManagement наследует операции от Top. Хотя нет способа встроить в Z наследование операций, непосредственным методом моделирования является повторное определение операций в терминах нового состояния. Предикаты состояния *StateManagement* следуют из определения функции StateManagement в ГОСТРИСО/МЭК 10165-2 и ГОСТРИСО/МЭК 10164-2.

Может быть легко определена операция *SMGetNameBinding*, так как она не оказывает действия на новые переменные состояния, объявленные в *StateManagementState*. Определение *TopGetNameBinding* может быть использовано повторно:

SMGetNameBinding

TopGetNameBinding
Ξ *StateManagementState*

Операции для получения других атрибутов *StateManagementState* также опущены в этом примере. Для операций *SMGetAllomorphs*, *SMGetObjectClass* и *SMGetPackages* могут быть повторно использованы определения из Top, как для *SMGetNameBinding*. Необходимо определить новые операции *GetSMAdministrativeState*, *GetSMOperationalState* и *SMGetUsageState*. Можно повторно использовать *SMEventReport*.

Схема *SMGetAll* также использует операции, определенные в TopState. Она включает в себя определение *TopGetAll* и усиленные постусловия:

SMGetAll

Ξ *StateManagementState*
TopGetAll

administrativeStateOid ∈ *attributes*
⇒ *administrativeStateValue administrativeState* ∈ *result* |
OperationalStateOid ∈ *attributes*
⇒ *operationalStateValue operationalState* ∈ *result* |
UsageStateOid ∈ *attributes*
⇒ *usageStateValue usageState* ∈ *result* |

Операция *SMReplaceAdministrativeState* описывает поведение, специфичное для класса *StateManagement* путем замены административного состояния другим значением, предоставленным в качестве входного. При

осуществлениииоперациивзависимостиотсостоянияобъектаможетизменитьсяистатусиспользования.Операционныйстатусэтойоперациинеизменяется:

SMReplaceAdministrativeState

$\Delta StateManagementState$
$\Xi TopState$
$input?: AdministrativeState$

$administrativeState' \in$
IF $usageState \neq idle$
THEN $\{ unlocked \rightarrow unlocked, locked \rightarrow locked,$
$shuttingown \rightarrow locked, locked \rightarrow shuttingown,$
$shuttingown \rightarrow shuttingown\} (\{input?\})$
ELSE $\{ unlocked \rightarrow unlocked, locked \rightarrow locked,$
$shuttingown \rightarrow locked\} (\{input?\})$
$administrativeState' \in locked \rightarrow usageState' = idle$
$administrativeState' \neq locked \rightarrow usageState' = usageState$
$operationalState' = operationalState$

Поведение,специфицированноевпредикатахсхемы,являетсяформализациейнеформальногоописаниявГОСТРИСО/МЭК10164-2.Дляполнотыдолжнысбытьопределеныоперациизаменыоперационногостатусаистатусаиспользования.

Существуетряддругихопераций,которыеописываютповедение,специфичноедляклассаStateManagement.Этиоперациинеперечисленыздесь,ноприведеныв*B.7*.Вихчисловходят*SMCapacityecrease, SMCapacityIncrease, SMisable, SMEnable, SMNewUseruSMUserQuit.*

B.4.3 Реализуемыеклассы

Ниодинизописанныхвышеклассовнеможетбытьреализован.Использованнаяпроцедурапостроенияклассовможетбытьпродолжена.Класс*StateManagement*можноповторноиспользоватьдляопределениякласса,названного*CIRCUIT*,который,всвоюочередь,можетбытьиспользовандляопределениякласса*ECIRCUIT* и,следовательно,реализуемогокласса*ActualECircuit*.Этачастьработынеприведенавруководстве,таккакпроцедураточнотакаяже,чтоописаннаявыше,иповторениенедобавитничегонового.

B.5Нерассмотренныевопросы

Вданномподразделеперечисленыосновныевопросы,встречающиесяприпереводеспецификацийуправляемыхобъектовРОУОвZ.ТакжеприведенапредлагаемаянеформальнаятрактовкатехотносящихсякZ случаев,когданетсоответствующихконструкцийдляконкретныххарактеристикспецификацииуправляемогообъекта.

B.5.1 Определениеповедениявуправляемыхобъектах

Вшаблонахтермин«определениеповедения»используетсяпочтидлявсехкатегорий,являющихсяданнымиилиобработкой.Впоследнемслучаэтоопределениеможетвключатьвсебяинформациюофактическомповедении(вточномсмыслеэтогослова)илистатическуюинформациюокатегории,такуюкакееназначение,илитоидругое.Припереводенеобходимопроанализироватьтекст,приведенныйподуказаннымзаголовком,иизвлечьизнегоинформацию,относящуюсякповедениюрассматриваемойкатегории.Этаинформациябудетиспользоватьсяприформальномпереводе,аимеющийсятекстможетбытьвключенвспецификациюZ вкачествекомментария.

B.5.2 Внутренниеоперациивz

Внутренняяоперацияуправляемогообъектапредставляетслучай,когдасообщениесоздаетсяспонтанно(безучастиявызоваадминистративногоуправления).Внутренниеоперацииявляютсяжелательнойхарактеристикоймногихсистем.Внастоящеевремявzэтахарактеристикапредставляетсянеформальноспомощьюкомментариянаестественномязыке.

B.5.3 Абстрактныеклассывz

Иногдаполезноидентифицироватьабстрактныеклассы,т.е.классы,которыенеимеютсвоейсобственнойреализации.НекоторыеклассыУО(например,top)немогутбытьреализованы.Былобыполезнопоказать,какиечастиспецификацииZпредставляютклассы,которыенемогутбытьреализованы.Внастоящеевремяэтоделаетсяспомощьюнеформальнойаннотации.

B.5.4 Семантикаконструкции PARAMETERS

Включениесемантикиконструкции PARAMETERS вобъектынерассматриваетсявнастоящем стандарте.

B.6ПреобразованиетиповданныхACH.1вZ

НижерассмотренывопросыпереводаконструкцийACH.1.

B.6.1 Простыетипы

ACH.1включаетвсебянекоторыепростыетипы,которыеявляютсявстроенными.Ониимеютстандартнуюструктуру,нообычноэтонепредставляетинтересавконтекстенастоящегостандартаиихследуетпредставлятькакзаданныемножества.Имеетсябольшойнабортиповсимвольныхстрок:

[*NUMERICSTRING, PRINTABLESTRING, TELETEXSTRING, VIEOTEXSTRING, VISIBLESTRING, IA5STRING, GRAPHICSTRING, GENERALSTRING*]

Два из них имеют синонимы:

T61STRING = = *TELETEXSTRING*

ISO64STRING = = *VISIBLESTRING*

Из других простых типов целый может быть представлен как Z, а булевский и вырожденный — как свободные типы:

Boolean : : = *btrue* | *bfalse*

Null : : = *null*

Эти три типа определяют и нотацию их значений.

Вещественный тип, строки битов и октетов могут быть представлены как заданные множества (хотя иногда для строк битов и октетов может потребоваться структура):

[*REAL, BITSTRING, OCTETSTRING*]

В настоящем стандарте описано еще один специальный тип, который может быть представлен как заданное множество:

[*OBJECTI*]

Он представляет идентификатор объекта АСН.1.

Идентификаторы объектов фактически являются непустыми последовательностями из N и, вместо заданных множеств, могут быть смоделированы именно так. В некоторых случаях соответствующей нотации значения должен быть приданнекоторый смысл.

ВАСН.1 определено еще несколько «полезных» типов. Хотя они могут быть определены в терминах других конструкций АСН.1, удобно представлять их в виде заданных множеств:

[GENERALIZETIME, UTCTIME, OBJECTESCRIPTOR, EXTERNAL]

ANY

ВАСН.1 имеется специальный тип ANY, который может содержать АСН.1 любого другого типа. Такой тип недопустим в Z и было бы трудно расширить этот язык для включения подобного типа. Однако, задавая некоторое известное множество типов, можно определить свободный тип Z, который может включать в себя любой из этих типов. Альтернативный подход состоит в том, чтобы определить ANY как заданное множество для целей проверки типа. Это приемлемо до тех пор, пока нам не нужно делать ничего другого. Тип AttributeValues обычно замещает ANY, что описано ниже.

В.6.2 Структурированные типы

Другие типы АСН.1 строятся на основе конструкций.

Множество

Множества АСН.1 могут быть представлены в Z либо как кортежи, либо как схемы. Кортежи Z не позволяют именовать компоненты, и в этом отношении больше подходят схемы. Однако нотация значений Z для схем менее удобна. Тегирование в Z невозможно и не нужно, так как компоненты множества всегда могут быть опознаны либо по их положению в кортеже, либо по имени компонента в схеме.

Компоненты этого и других структурированных типов могут быть факультативными (*OPTIONAL*). Это может быть представлено в Z дополнением типа факультативного компонента специальным значением «отсутствует». Значения по умолчанию *EFAULT* не могут быть представлены удобным образом как свойства типа данных. Можно представить поведение, подразумеваемое умолчанием, во всех операциях над этим типом данных.

Последовательность

Последовательности АСН.1 могут моделироваться точно таким же образом, каким множества АСН.1, так как единственным различием является явно установленный порядок. Поэтому более подходящими являются кортежи, но можно использовать и схемы.

Множество-из

Множества-из АСН.1 могут моделироваться как последовательности Z.

Выбор

Выборочные типы АСН.1 непосредственно являются перечислениями и могут моделироваться свободными типами Z.

С этим типом связана серьезная проблема область действия. В АСН.1 конструкции в пределах выборочного типа являются локальными и для этого типа. Таким образом одно и то же имя конструкции может использоваться в нескольких перечислениях. В Z имена являются глобальными и могут быть повторно использованы. Эта проблема обычно может быть разрешена изменением имен конструкций, как правило, путем добавления к ним в качестве префикса имени их типа.

Аналогичная проблема возникает, когда создаются типы АСН.1, являющиеся синонимами, например целого, но с поименованными значениями. Эти поименованные значения являются локальными для типа-синонима в АСН.1, но глобальными и синонимами для целых в Z. Для разрешения этой проблемы также можно изменить имена конструкций.

В.6.3 Простые типы атрибутов

Прощевсегопредставитьатрибутв УОкакпеременнуюZ ссоответствующимтипомданных. Потребуется такжеопределениепостоянной, представляющей *OBJECTI* этогоатрибута. Когдадляатрибутаопределеныоперациисогласования, можетбытьиспользованофактическоестандартноезначениепараметрадлясогласования.

Рассмотриматрибут УО administrativeState. Имеетсяопределениетипа:

AdministrativeState : : = *unlocked* | *locked* | *shuttingown*

Можетбытьопределенапостояннаядляпредставленияидентификатораобъектаатрибута:

administrativeStateOid : *OBJECTI*

Фактическоезначениеидентификатораможетбытьпредставленокакограничениенаэтоаксиоматическоеопределение.

Можетбытьопределенапеременнаясостояниявуо:

MOState

administrativeState: *AdministrativeState*

Такоерешениеявляетсяпрямымиудобным, нотребуетсписказаксиоматическихопределений *OBJECTI*. Онотакжеделаетсвязьмеждуименематрибутаиего *OBJECTI* чистосинтаксической. Втомрешениибыло принятосоглашениеобиспользованиисуффикса *Oid*.

В.6.4 Типы атрибутов как схемы

Можновключитьвсеэтихарактеристикиатрибутаводинтипсхемы, которыйбудеттипомпеременной Z, моделирующейатрибут УО:

AdministrativeStateType

value: *AdministrativeState* *Oid*: *OBJECTI*
Oid= <4,3,19,27,1,3

Важно подставить именно здесь значение *OBJECTI*, так как необходимо гарантировать, что оно не можетбытьизменено.

Структура *OBJECTI* пока не определена, но запись

OBJECTI = = ***seqIN***

является одной из возможностей придать смысл предыдущей схеме. Этажесхемаможетхранитьпараметрдля согласования, если окажется важным представить его в спецификации.

Теперь УО может содержать атрибут этого типа:

MOState

administrativeState: *AdministrativeStateType*

Ссылка на его значение (или на его Oid) может быть сделана через выбор компонента, например *administrativeState.value*.

Этотметодудобнопередаетсемантикудлясвязимеждуатрибутомиего *OBJECTI*. Однакодлячитателей спецификации может показаться странным, что Oid присутствует в состоянии УО, хотя он и не может изменяться (ифактическиявляетсяглобальнойпостоянной, известнойнамоментспецификации).

В.6.5 Тип AttributeValues

Какужеотмечалось, вZтрудномоделироватьтипACH.IANY. Общимподходомявляетсязаданиесписковзначенийатрибута. Требуетсяопределениесвободноготипа Z вкомбинациисужеопределеннымитипами. Этотподходработаетдотехпор, покамножествоиспользуемыхатрибутовфиксированонамоментспецификации. Тогдарешениебудетвыглядетьприблизительнотак:

AttributeValues : : = *administrativeStateValue* <<*AdministrativeState* |
objectClassValue <<*OBJECTI* |
nameBindingValue <<*OBJECTI* |
packagesValue <<***P OBJECTI*** |
allomorphsValue <<***P OBJECTI*** |
operationalStateValue <<*OperationalState* |
usageStateValue <<*UsageState*

В.7 Полный пример

В настоящем подразделе представлена полная формальная модель, на которой основан пример в В.4. Модель представлена в традиционном для Z стиле объявления до использования; а именно, определения типов, преобразованные из ACH.1, появляются в начале, а определения поведения — в конце.

Следует прокомментировать одно место в стиле спецификации. Определения *AttributeValues* и *OBJECTINSTANCE* являются взаимнорекурсивными. В Z этотехническинедопустимо, идлятого, чтобыдать определения, было сделано следующее. Тип *AttributeValues* был введен как заданное множество. Затем

OBJECTINSTANCE было определен с использованием заданного множества *AttributeValues*. Определение *OBJECTINSTANCE* было использовано для введения ограничений того, что может принимать *AttributeValues*. Была выполнена проверка ограничения для того, чтобы показать, что такие множества фактически существуют, но в примере это не показано.

B.7.1 Основные типы АСН.1

```
[NUMERICSTRING,PRINTABLESTRING,TELETEXSTRING,VIEOTEXSTRING]
[VISIBLESTRING,IA5STRING,GRAPHICSTRING,GENERALSTRING]
T61STRING = = TELETEXSTRING
ISO64STRING = = VISIBLESTRING
Boolean : : = btrue | bfalse
Null : : = null
[REAL, BITSTRING, OCTETSTRING]
[OBJECTI ]
[ANY]
[GENERALIZETIME,UTCTIME,OBJECTESCRIPTOR,EXTERNAL]
```

B.7.2 Атрибуты УО

Следующее заданное множество резервирует место для более сложного и полного определения свободного типа, которое будет расширяться по мере определения новых классов.

```
[AttributeValues]
AttributeValuesOptional : : = present<<AttributeValues | absent
RelativeistinguishedName = = AttributeValues
RNSequence = = seqRelativeistinguishedName
istinguishedName = = RNSequence
OBJECTINSTANCE : : = distinguishedName<<istinguishedName | nonSpecificForm<<N
                   | localistinguishedName<<RNSequence
```

B.7.3 Сообщения

```
ProbableCause = = OBJECTI
SpecificIdentifier : : = globalvalue<<OBJECTI | localValue<<N
SpecificProblems = = PSpecificIdentifier
SpecificProblemsOptional : : = sPPresent<<SpecificProblems | sPAbsent
PerceivedSeverity : : = indeterminate|critical| major|minor |warning|cleared
BackedUpStatus = = Boolean
BackedUpStatusOptional : : = bUSPresent<<BackedUpStatus | bUSAbsent
ObjectInstanceOptional : : = oISPresent<<OBJECTINSTANCE | oIAbsent
TrendIndication : : = lessSevere | noChange | moreSevere
TrendIndicationOptional : : = tIPresent<<TrendIndication | tIAbsent
ObservedValue : : = int<<N | real<<REAL
ObservedValueOptional : : =oVPresent<<ObservedValue | oVAbsent
ThresholdLevelInd : : =
    up<<ObservedValue- ObservedValueOptional
    | down <<ObservedValue- ObservedValueOptional
ThresholdLevelIndOptional : : = tLIPresent<<ThresholdLevelInd |tLIAbsent
ArmTimeOptional : : = aTPresent<<GENERALIZETIME |aTAbsent
ThresholdInfo = =
    OBJECTI - ObservedValue- ThresholdLevelIndOptional - ArmTimeOptional
ThresholdInfoOptional : : = thIPresent<<ThresholdInfo |thIAbset
NotificationIdentifier==N
NotificationIdentifierOptional : : = nIPresent<<NotificationIdentifier |nIAbsent
CorrelatedNotifications = = P((P NotificationIdentifier) - ObjectInstanceOptional)
CorrelatedNotificationsOptional : : = cNPresent<<CorrelatedNotifications |cNAbset
AttributeValueChangeefinition==P(OBJECTI - AttributeValuesOptional - AttributeValues)
AttributeValueChangeefinitionOptional : :
  aVCPresent<<AttributeValueChangeefinition |aVCAbsent
MonitoredAttributes==POBJECTI
MonitoredAttributesOptional : : = mAPresent<<MonitoredAttributes |mAAbset
ProposedRepairActions==PSpecificIdentifier
ProposedRepairActionsOptional : : = pRAPresent<<ProposedRepairActions |pRAAbsent
AdditionalTextOptional : : = adTPresent<<GRAPHICSTRING |adTAbsent
ManagementExtension = = OBJECTI - Boolean - ANY
AdditionalInformation==PManagementExtension
AdditionalInformationOptional : : = aIPresent<<AdditionalInformation |aIAbsent
SourceIndicator : : = resourceOperation |managementOperation|sIUnknown
```

SourceIndicatorOptional: : = *sIPresent* << *SourceIndicator* | *sIAbsent*
AttributeIdentifierList = = *P OBJECTI*
AttributeIdentifierListOptional: : = *atIPresent* << *AttributeIdentifierList* | *atIAbsent*
Attribute = = *OBJECTI* · *AttributeValues*
AttributeList = = *PAttribute*
AttributeListOptional: : = *aLPresent* << *AttributeList* | *aLAbsent*

AlarmInfo

probableCause:ProbableCause
specificProblems:SpecificProblemsOptional
perceivedSeverity:PerceivedSeverity
backedUpStatus:BackedUpStatusOptional
backUpObject:ObjectInstanceOptional
trendIndication:TrendIndicationOptional
thresholdInfo:ThresholdInfoOptional
notificationIdentifier:NotificationIdentifierOptional
correlatedNotifications:CorrelatedNotificationsOptional
stateChangeefinition:AttributeValueChangeefinitionOptional
monitoredAttributes:MonitoredAttributesOptional
proposedRepairActions:ProposedRepairActionsOptional
additionalText:AdditionalTextOptional
additionalInformation:AdditionalInformationOptional

AttributeValueChangeInfo

sourceIndicator:SourceIndicatorOptional
attributeIdentifierList:AttributeIdentifierListOptional
attributeValueChangeefinition:AttributeValueChangeefinitionOptional
notificationIdentifier:NotificationIdentifierOptional
correlatedNotifications:CorrelatedNotificationsOptional
additionalText:AdditionalTextOptional
additionalInformation:AdditionalInformationOptional

ObjectInfo

sourceIndicator:SourceIndicatorOptional
attributeList:AttributeListOptional
notificationIdentifier:NotificationIdentifierOptional
correlatedNotifications:CorrelatedNotificationsOptional
additionalText:AdditionalTextOptional
additionalInformation:AdditionalInformationOptional

RelationshipChangeInfo

sourceIndicator:SourceIndicatorOptional
attributeIdentifierList:AttributeIdentifierListOptional
relationshipChangeefinition:AttributeValueChangeefinitionOptional
notificationIdentifier:NotificationIdentifierOptional
correlatedNotifications:CorrelatedNotificationsOptional
additionalText:AdditionalTextOptional
additionalInformation:AdditionalInformationOptional

SecurityAlarmInfo

notificationIdentifier:NotificationIdentifierOptional
correlatedNotifications:CorrelatedNotificationsOptional
additionalText:AdditionalTextOptional
additionalInformation:AdditionalInformationOptional

StateChangeInfo

sourceIndicator:SourceIndicatorOptional
attributeIdentifierList:AttributeIdentifierListOptional
stateChangeefinition:AttributeValueChangeefinitionOptional
notificationIdentifier:NotificationIdentifierOptional
correlatedNotifications:CorrelatedNotificationsOptional
additionalText:AdditionalTextOptional
additionalInformation:AdditionalInformationOptional

EventInfo : : = *attributeValueChange*<<*AttributeValueChangeInfo*
| *communicationsAlarm*<<*AlarmInfo*
| *environmentalAlarm*<<*AlarmInfo*
| *equipmentAlarm*<<*AlarmInfo*
| *integrityViolation*<<*SecurityAlarmInfo*
| *objectCreation*<<*ObjectInfo*
| *objecteletion*<<*ObjectInfo*
| *operationalViolation*<<*SecurityAlarmInfo*
| *physicalViolation*<<*SecurityAlarmInfo*
| *processingError*<<*AlarmInfo*
| *qualityOfServiceAlarm*<<*AlarmInfo*
| *relationshipChange*<<*RelationshipChangeInfo*
| *securityServiceOrMechanismViolation*<<*SecurityAlarmInfo*
| *stateChange*<<*StateChangeInfo*
| *timeomainViolation*<<*SecurityAlarmInfo*

B.7.4 «CCITT Rec. X.721 (1992)» | ISO/IEC 10165-2:1992:Top

allomorphsOid : *OBJECTI*
nameBindingOid : *OBJECTI*
objectClassOid : *OBJECTI*
packagesOid : *OBJECTI*

---

allomorphsValue : (*FOBJECTI*) → *AttributeValues*
nameBindingValue : *OBJECTI* → *AttributeValues*
objectClassValue : *OBJECTI* → *AttributeValues*
packagesValue : (*FOBJECTI*) → *AttributeValues*

---

disjoint<**ran***allomorphsValue*, **ran***nameBindingValue*,
ran*objectClassValue*, **ran***packagesValue*

packagesPackageOid: *OBJECTI*
allomorphsPackageOid : *OBJECTI*

TorState

allomorphs : *FOBJECTI*
objectClass : *OBJECTI*
nameBinding : *OBJECTI*
packages : *FOBJECTI*
attributes : *FOBJECTI*

{*objectClassOid*, *nameBindingOid*} ⊂ *attributes*
allomorphsPackageOid ∈ *packages* ⇒ *allomorphsOid* ∈ *attributes*
packagesPackageOid ∉ *packages*
packages ≠ ∅ ⇒ *packagesOid* ∈ *attributes*

TopGetNameBinding

Ξ *TopState*
result′ : *OBJECTI*

---

result′ = *nameBinding*

TopGetAll

Ξ *TopState*
result′ : *PAttributeValues*

---

attributes = # *result*′
ObjectClassValue~*objectClass* ∈ *result*′
NameBindingValue~*nameBinding* ∈ *result*′
PackagesOid ∈ *attributes* ⇒ *packagesValue*~*packages* ∈ *result*′
AllomorphsOid ∈ *attributes* ⇒ *allomorphsValue*~*allomorphs* ∈ *result*′

TopEventReport

Ξ TopState
notification| : EventInfo

B.7.5 Класс StateManagement

administrativeStateOid: OBJECTI
operationalStateOid: OBJECTI
usageStateOid: OBJECTI

AdministrativeState: : = unlocked|locked . shuttingown
OperationalState : : = enabled | disabled
UsageState: : = idle|active|busy

administrativeStateValue: AdministrativeState→AttributeValues
operationalStateValue: OperationalState→AttributeValues
usageStateValue: UsageState→AttributeValues

disjoint *<* ***ran****allomorphsValue,* ***ran****nameBindingValue,*
ran*objectClassValue,* ***ran****packagesValue,*
ran*administrativeStateValue,*
ran*operationalStateValue,* ***ran****usageStateValue*

StateManagementState

TopState
administrativeState:
AdministrativeState
operationalState: OperationalState
usageState: UsageState

operationalState= disabled ⇒ usageState=idle
administrativeState= locked ⇒ usageState=idle
usageState=idle⇒ administrativeState≠ shuttingown

SMGetNameBinding

TopGetNameBinding
Ξ StateManagementState

SMEventReport

TopEventReport
Ξ StateManagementState

SMGetAll

Ξ StateManagementState
TopGetAll

administrativeStateOid ∈ attributes
⇒ administrativeStateValueadministrativeState ∈ result |
OperationalStateOid ∈ attributes
⇒ operationalStateValueoperationalState ∈ result |
UsageStateOid ∈ attributes
⇒ usageStateValueusageState ∈ result |

SMCapacityecrease

Δ StateManagementState
Ξ TopState

usageState=active⇒ usageState' ∈ {active,busy}
usageState≠active⇒ usageState' = usageState
administrativeState' = administrativeState
operationalState' = operationalState

SMCapacityIncrease

$\Delta StateManagementState$
$\Xi TopState$

$usageState = busy \Rightarrow usageState' = active$
$usageState \neq busy \Rightarrow usageState' = usageState$
$administrativeState' = administrativeState$
$operationalState' = operationalState$

SMisable

$\Delta StateManagementState$
$\Xi TopState$

$administrativeState' =$
IF $administrativeState = shuttingown$
THEN $locked$
ELSE $administrativeState$
$operationalState' = disabled$
$usageState' = idle$

SMEnable

$\Delta StateManagementState$
$\Xi TopState$

$operationalState = disabled$
$operationalState' = enabled$
$administrativeState' = administrativeState$
$usageState' = usageState$

SMNewUser

$\Delta StateManagementState$
$\Xi TopState$

$operationalState = enabled$
$administrativeState = unlocked$
$usageState \in \{idle, active\}$
$usageState' \in \{active, busy\}$
$administrativeState' = administrativeState$
$operationalState' = operationalState$

SMUserQuit

$\Delta StateManagementState$
$\Xi TopState$

$administrativeState = shuttingown\ usageState' = idle$
$\Rightarrow administrativeState' = locked$
$administrativeState \neq shuttingown] usageState' \neq idle$
$\Rightarrow administrativeState' = administrativeState$
$usageState \in \{active, busy\}$
$usageState' \in \{idle, active\}$
$operationalState' = operationalState$

SMReplaceAdministrativeState

$\Delta StateManagementState$
$\Xi TopState$

$administrativeState' \in$
IF $usageState \neq idle$
THEN $\{ unlocked \rightarrow unlocked, locked \rightarrow locked,$
$shuttingown \rightarrow locked, locked \rightarrow shuttingown,$
$shuttingown \rightarrow shuttingown\}(\{input?\})$
ELSE $\{ unlocked \rightarrow unlocked, locked \rightarrow locked,$
$shuttingown \rightarrow locked\}(\{input?\})$
$administrativeState' = locked \rightarrow usageState' = idle$
$administrativeState' \neq locked \rightarrow usageState' = usageState$
$operationalState' = operationalState$

УДК 681.324:006.354 ОКС 35.100.70 П85 ОКСТУ 4002

Ключевые слова: информационная технология, взаимосвязь открытых систем, обработка данных, информационный обмен, сетевое взаимодействие, административное управление, информация

Редактор *В. П. Огурцов*
Технический редактор *Н. С. Гришанова*
Корректор *Н. И. Гаврищук*
Компьютерная верстка *Т. Ф. Кузнецовой*

Изд. лиц. № 02354 от 14.07.2000. Сдано в набор 25.09.2001. Подписано в печать 16.11.2001. Усл. печ. л. 7,90. Уч.-изд. л. 8,00.
Тираж 429 экз. С 2811. Зак. 2300

ИПК Издательство стандартов, 107076, Москва, Колодезный пер., 14.
http://www.standards.ru e-mail:info@standards.ru
Набран в Калужской типографии стандартов на ПЭВМ.
Калужская типография стандартов, 248021, Калуга, ул. Московская, 256.
ПЛР № 040138

gost.e-ecolog.ru