ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

АБСТРАКТНАЯ СИНТАКСИЧЕСКАЯ НОТАЦИЯ ВЕРСИИ ОДИН (АСН. 1)

Часть 2

Спецификация информационного объекта

Издание официальное

БЗ 5—2001/107

ГОССТАНДАРТ РОССИИ
Москва

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. № 375-ст

3 Настоящий стандарт содержит полный аутентичный текст международного стандарта ИСО/МЭК 8824-2—98 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН. 1). Часть 2. Спецификация информационного объекта» с Дополнением № 1 (1999 г.)

4 ВВЕДЕН ВПЕРВЫЕ

Содержание

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

АБСТРАКТНАЯ СИНТАКСИЧЕСКАЯ НОТАЦИЯ ВЕРСИИ ОДИН (АСН. 1)

Часть 2

Спецификация информационного объекта

Information technology. Abstract Syntax Notation One (ASN. 1). Information object specification

Дата введения 2002—07—01

1 Область применения

Настоящий стандарт является частью абстрактной синтаксической нотации версии 1 (АСН. 1) и устанавливает нотацию для спецификации классов информационных объектов, информационных объектов и множеств информационных объектов.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие стандарты:

ГОСТ Р ИСО/МЭК 8824-1—2001. Информационная технология. Абстрактная синтаксическая нотация версии один (АСН. 1). Часть 1. Спецификация основной нотации [Рекомендация МККТТ X.680 (1997)]

ИСО/МЭК 8824-3—98 Информационная технология. Абстрактная синтаксическая нотация версии один (АСН. 1). Часть 2. Спецификация ограничения [Рекомендация МККТТ X.682 (1997)]

ИСО/МЭК 8824-4—95 Информационная технология. Абстрактная синтаксическая нотация версии один (АСН. 1). Часть 4. Параметризация спецификации АСН. 1 [Рекомендация МККТТ X.683 (1997)]

3 Определения

3.1 Спецификация базовой нотации

В настоящем стандарте используют термины, определенные в ГОСТ Р ИСО/МЭК 8824-1.

3.2 Спецификация ограничения

В настоящем стандарте используют термин, определенный в ИСО/МЭК 8824-3: **табличное ограничение**

3.3 Параметризация спецификации АСН. 1

В настоящем стандарте используют термины, определенные в ИСО/МЭК 8824-4:

а) **параметризованный тип;**

б) **параметризованное значение.**

3.4 Дополнительные определения

3.4.1 **ассоциированная таблица** (для некоторого информационного объекта или множества информационных объектов): Абстрактная таблица, получаемая из объекта или множества объектов путем выравнивания иерархической структуры, полученной из-за присутствия полей связи (см. 3.4.13).

Примечание — Ассоциированная таблица может быть использована для определения точной природы некоторого ограничения (см. ИСО/МЭК 8824-3), которое было определено с использованием множества объектов.

Издание официальное

3.4.2 **синтаксис по умолчанию:** Нотация, которая должна использоваться для определения информационных объектов тех классов, авторы определения которых не предоставляют определенный синтаксис (см. пример 11.9).

3.4.3 **определяемый синтаксис:** Нотация, предоставленная автором определения класса, которая позволяет определять информационные объекты этого класса дружественным пользователю образом.

Примечание — Например определяемый синтаксис для класса OPERATION может позволить определять экземпляры класса с помощью слова ARGUMENT с последующим &ArgumentType, далее — RESULT с последующим &ResultType и CODE с последующим &operationCode (см. пример 11.10).

3.4.4 **расширяемое множество объектов:** Множество объектов с маркером расширения.

3.4.5 **поле:** Компонент класса информационных объектов. Каждое поле есть поле типа, поле значения фиксированного-типа, поле значения переменного-типа, поле множества значений фиксированного-типа, поле множества значений переменного-типа, поле информационного объекта или поле множества информационных объектов.

3.4.6 **имя поля:** Имя, которое идентифицирует поле некоторого класса; либо класса, который специфицирует поле непосредственно, и в этом случае имя является примитивным именем поля, либо класса, который имеет цепочку полей связи к тому, в котором поле фактически определено (см. 9.13 и 9.14).

3.4.7 **поле идентификатора:** Поле класса значений фиксированного-типа, выбранное для обеспечения однозначной идентификации информационных объектов данного класса. Значения поля идентификатора, если они применяются, должны быть недвусмысленными в пределах любого множества информационных объектов, которое определяется для данного класса. Они могут, но не обязаны служить для недвусмысленной идентификации информационных объектов данного класса в некоторой широкой области.

Примечания

1 Поле идентификатора имеет фиксированный тип АСН. 1, но значения этого типа могут быть введены в протокол для идентификации информационных объектов в классе.

2 Область, в которой идентификатор является недвусмысленным, является множеством информационных объектов. Однако она может быть также недвусмысленной в пределах данного абстрактного синтаксиса или в пределах целого прикладного контекста, или даже может быть глобальной для всех классов и всех прикладных контекстов при использовании типа «идентификатор объекта» для поля идентификатора.

3.4.8 **информационный объект:** Экземпляр некоторого класса информационных объектов, сформированный из множества полей, которые соответствуют спецификациям полей этого класса.

Примечание — Например одним конкретным экземпляром информационного объекта класса OPERATION (упоминаемого в 3.4.9) может быть invertMatrix, который имеет поле &ArgumentType, содержащее тип Matrix, поле &ResultType, содержащее также тип Matrix, и поле &operationCode, содержащее значение 7 (см. пример в 10.13).

3.4.9 **класс информационного объекта (класс):** Множество полей, формирующее шаблон для определения возможно неограниченной совокупности информационных объектов (экземпляров класса).

Примечание — Например класс информационного объекта OPERATION может быть определен так, чтобы соответствовать понятию операции «operation» для удаленных операций. Каждая спецификация различных поименованных полей будет соответствовать некоторому аспекту, который может изменяться от одного экземпляра операции к другому. Таким образом, в ней могут быть поля &ArgumentType, &ResultType и &operationCode, из которых два первых поля определяют тип, а третье — поле значения.

3.4.10 **поле информационного объекта:** Поле, которое содержит информационный объект некоторого заданного класса.

3.4.11 **множество информационных объектов:** Непустое множество информационных объектов, все из которых одного и того же класса информационных объектов.

Примечание — Например одно множество информационных объектов, MatrixOperations, класса OPERATION (использованного в примере 3.4.9) может содержать invertMatrix (упомянутую в 3.4.8) вместе с другими связанными операциями, такими как addMatrices, multiplyMatrices, и т.д. Такое множество объектов может использоваться при определении абстрактного синтаксиса, который обеспечивает вызов и возврат результата всех этих операций (см. пример в 12.7).

3.4.12 **поле множества информационных объектов:** Поле, которое содержит множество информационных объектов некоторого заданного класса.

3.4.13 **тип «экземпляр-из»:** Тип, определяемый указанием класса информационных объектов, который связывает идентификаторы объектов с типами.

3.4.14 **поле связи:** Поле объекта или множества объектов.

3.4.15 **тип поле класса объектов:** Тип, задаваемый путем указания некоторого поля класса информационных объектов. В ИСО/МЭК 8824-3 определена нотация, обеспечивающая возможность ограничения этого типа путем указания множества информационных объектов данного класса.

3.4.16 **примитивное имя поля:** Имя, заданное непосредственно определением класса информационных объектов без использования поля связи.

3.4.16 bis **рекурсивное определение (класса информационных объектов):** Набор определений АСН. 1, который не может быть переупорядочен таким образом, что все классы информационных объектов, использованные в конструкции, определены до определения конструкции.

Примечание — Рекурсивные определения классов информационных объектов допустимы в АСН. 1. Рекурсивные определения информационных объектов и множеств информационных объектов недопустимы.

3.4.17 **поле типа:** Поле, которое содержит произвольный тип.

3.4.18 **поле значения:** Поле, которое содержит значение. Такое поле может быть либо фиксированного-, либо переменного-типа. В первом случае тип значения зафиксирован спецификацией поля. Во втором случае тип значения содержится в некотором (заданном) поле типа того же самого информационного объекта.

3.4.19 **поле множества значений:** Поле, которое содержит непустое множество значений некоторого типа. Такое поле может быть либо фиксированного-, либо переменного-типа. В первом случае тип значений зафиксирован спецификацией поля. Во втором случае тип значений содержится в некотором (заданном) поле типа того же самого информационного объекта.

Примечание — Множество значений в поле множества значений для информационного объекта образует подтип заданного типа.

4 Сокращения

В настоящем стандарте используют следующие сокращения:
АСН. 1 — абстрактная синтаксическая нотация версии 1;
БНФ — Бекуса—Науэра форма.

5 Соглашение

В настоящем стандарте используют соглашения, приведенные в ГОСТ Р ИСО/МЭК 8824-1, раздел 5.

6 Нотация

В настоящем разделе приводят сводку нотаций, определенных в настоящем стандарте.

6.1 Присвоения

В настоящем стандарте определены следующие нотации, которые могут использоваться как альтернативы для «Assignment» (см. ГОСТ Р ИСО/МЭК 8824-1, раздел 12):

- ObjectClassAssignment (см. 9.1);
- ObjectAssignment (см. 11.1);
- ObjectSetAssignment (см. 12.1);

6.2 Типы

6.2.1 В настоящем стандарте определены следующие нотации, которые могут использоваться как альтернативы для встроенного типа «BuiltinType» (см. ГОСТ Р ИСО МЭК 8824-1, 16.2):

- ObjectClassFieldType (см. 14.1);
- InstanceOfType (см. приложение С).

6.2.2 В настоящем стандарте определены следующие нотации, которые могут использоваться как альтернативы для указываемого типа «ReferencedType» (см. ГОСТ Р ИСО/МЭК 8824-1, 16.3):

- TypeFromObject (см. раздел 15);
- ValueSetFromObject (см. раздел 15).

6.3 Значения

6.3.1 В настоящем стандарте определены следующие нотации, которые могут использоваться как альтернативы для встроенного значения «BuiltinValue» (см. ГОСТ Р ИСО/МЭК 8824-1, 16.8):

- ObjectClassFieldValue (см. 14.6);
- InstanceOfValue (см. приложение С).

6.3.2 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для указываемого значения «ReferencedValue» (см. ГОСТ Р ИСО/МЭК 8824-1, 16.9):

- ValueFromObject (см. раздел 15);

6.4 Элементы

В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для элементов «Elements» (см. ГОСТ Р ИСО/МЭК 8824-1, 46.3):

- ObjectSetElements (см. 12.3).

7 Элементы АСН. 1

Дополнительно к элементам АСН. 1, определенным в ГОСТ Р ИСО/МЭК 8824-1, раздел 11, в настоящем стандарте используют элементы АСН. 1, определенные в последующих подразделах. Общие правила, применяемые к этим элементам, те же самые, что и определенные в ГОСТ Р ИСО/МЭК 8824-1, 11.1. Эти новые элементы АСН. 1 используют набор символов АСН. 1, определенный в ГОСТ Р ИСО/МЭК 8824-1, раздел 10, и дополнительно знак амперсанд («&»).

Примечание — Примечание в ГОСТ Р ИСО/МЭК 8824-1, 10.1 также применяется к элементам, специфицированным в 7.1 — 7.9.

7.1 Ссылки на класс информационных объектов

Имя элемента — objectclassreference

Ссылка на класс информационных объектов «objectclassreference» должна состоять из последовательности символов, определенной в ГОСТ Р ИСО/МЭК 8824-1, 11.2 для «typereference», за исключением того, что в нее не должны включаться буквы нижнего регистра.

7.2 Ссылки на информационный объект

Имя элемента — objectclassreference

Ссылка на информационный объект «objectreference» должна состоять из последовательности символов, определенной в ГОСТ Р ИСО/МЭК 8824-1, 11.4 для «valuereference».

7.3 Ссылки на множество информационных объектов

Имя элемента — objectclassreference

Ссылка на множество информационных объектов «objectsetreference» должна состоять из последовательности символов, определенной в ГОСТ Р ИСО/МЭК 8824-1, 11.2 для «typereference».

7.4 Ссылки на поле типа

Имя элемента — typefieldreference

Ссылка на поле типа «typefieldreference» должна состоять из амперсанда («&»), за которым непосредственно следует последовательность символов, определенная в ГОСТ Р ИСО/МЭК 8824-1, 11.2 для «typereference».

7.5 Ссылки на поле значения

Имя элемента — valuefieldreference

Ссылка на поле значения «valuefieldreference» должна состоять из амперсанда («&»), за которым непосредственно следует последовательность символов, определенная в ГОСТ Р ИСО/МЭК 8824-1, 11.4 для «valuereference».

7.6 Ссылки на поле множества значений

Имя элемента — valuesetfieldreference

Ссылка на поле множества значений «valuesetfieldreference» должна состоять из амперсанда («&»), за которым непосредственно следует последовательность символов, определенная в ГОСТ Р ИСО/МЭК 8824-1, 11.2 для «typereference».

7.7 Ссылки на поле объекта

Имя элемента — objectfieldreference

Ссылка на поле объекта «objectfieldreference» должна состоять из амперсанда («&»), непосредственно за которым следует последовательность символов, определенная в 7.2 для «objectreference».

7.8 Ссылки на поле множества объектов

Имя элемента — objectsetfieldreference

Ссылка на поле множества объектов «objectsetfieldreference» должна состоять из амперсанда («&»), непосредственно за которым следует последовательность символов, определенная в 7.3 для «objectsetreference».

7.9 Слово

Имя элемента — word

Слово «word» должно состоять из последовательности символов, определенной в ГОСТ Р ИСО/МЭК 8824-1, 11.2 для «typereference», за исключением того, что в нее не должны включаться буквы или цифры нижнего регистра.

7.10 Дополнительные ключевые слова

Имена CLASS, INSTANCE, SINTAX и UNIQUE приведены в ГОСТ Р ИСО/МЭК 8824-1, 11.18 как зарезервированные слова.

8 Определения ссылок

8.1 Конструкции

DefinedObjectClass : : =
    ExternalObjectClassReference |
    objectclassreference |
    UsefulObjectClassReference
DefinedObject : : =
    ExternalObjectReference |
    objectreference
DefinedObjectSet : : =
    ExternalObjectSetReference |
    objectsetreference

используются для ссылок на определения класса, информационного объекта и множества информационных объектов соответственно.

8.2 За исключением, определенным в ГОСТ Р ИСО/МЭК 8824-1, 12.15, альтернативы «objectclassreference», «objectreference» и «objectsetreference» должны использоваться только в том модуле, в котором класс, информационный объект или множество информационных объектов (см. 9.1, 11.1 и 12.1) присвоены этой ссылке.

Альтернативы «ExternalObjectClassReference», «ExternalObjectReference» и «ExternalObjectSetReference» определяются следующим образом:

ExternalObjectClassReference : : =
    modulereference
    «.»
    objectclassreference
ExternalObjectReference : : =
    modulereference
    «.»
    objectreference
ExternalObjectSetReference : : =
    modulereference
    «.»
    objectsetreference

Эти альтернативы должны использоваться только в том случае, если соответствующим «objectclassreference», «objectreference» или «objectsetreference» были присвоены класс, информационный объект или множество информационных объектов соответственно (см. 9.1, 11.1 и 12.1) в модуле (отличном от ссылающегося модуля), идентифицированном соответствующей ссылкой «modulereference». Это и есть именно те, соответственно, класс, информационный объект или множество информационных объектов, которые указываются.

8.3 Альтернатива «UsefulObjectClassReference» для «DefinedObjectClass» определяется следующим образом:

UsefulObjectClassReference : : = TYPE—IDENTIFIER | ABSTRACT—SYNTAX, где первая альтернатива определена в приложении А, а вторая — в приложении В.

Примечание — Имена TYPE—IDENTIFIER и ABSTRACT—SYNTAX приведены в ГОСТ Р ИСО/МЭК 8824-1, 11.18 как зарезервированные слова.

9 Определение и присвоение класса информационных объектов

9.1 Конструкция «ObjectClassAssignment» используется для присвоения класса информационных объектов имени ссылки («objectclassreference»). Эта конструкция является одной из альтернатив для «Assignment» в ГОСТ Р ИСО/МЭК 8824-1, раздел 12 и определяется следующим образом:

ObjectClassAssignment : : =
objectclassreference
«.»
ObjectClass

9.2 Класс информационного объекта — тот, который определяется конструкцией «ObjectClass».

ObjectClass : : =
DefinedObjectClass |
ObjectClassDefn |
ParametrizedObjectClass

если «ObjectClass» есть:

а) «DefinedObjectClass», то определение класса — то же самое, как определение указываемого класса;

б) «ObjectClasDefn», то класс определяется как описано в 9.3;

в) «ParameterizedObjectClass», то класс определяется как описано в ИСО/МЭК 8824-4, 9.2.

9.3 Каждый класс, в конечном счете, определяется «ObjectClassDefn»:

ObjectClassDefn : : =
CLASS
«{» FieldSpec «,» + «}»
WithSyntaxSpec?
WithSyntaxSpec : : = WITH SYNTAX SyntaxList

Эта нотация позволяет разработчику класса предоставить спецификации поименованных полей, каждая из которых является «FieldSpec», как определено в 9.4. Факультативно автор определения может предоставить синтаксис определения информационных объектов («SyntaxList»), как указано в 10.5. Автор определения класса может также специфицировать семантику, связанную с определением класса.

9.4 Каждая альтернатива «FieldSpec» специфицирует и называет одно из полей, которое будет или может быть связано с экземплярами класса.

FieldSpec : : =
TypeFieldSpec |
FixedTypeValueFieldSpec |
VariableTypeValueFieldSpec |
FixedTypeValueSetFieldSpec |
VariableTypeValueSetFieldSpec |
ObjectFieldSpec |
ObjectSetFieldSpec

Различные альтернативы для «FieldSpec» определяются в последующих пунктах.

9.5 Альтернатива «TypeFildSpec» определяет, что поле является полем типа (см. 3.4.17).

TypeFildSpec : : =
typefieldreference
TypeOptionalitySpec?
TypeOptionalitySpec : : = OPTIONAL | DEFAULT Type

Имя поля есть «typefieldreference». Если продукция «TypeOptionalitySpec» отсутствует, то все определения информационных объектов этого класса обязательно должны включать в себя спецификацию типа для этого поля. Если присутствует «OPTIONAL», то поле может быть оставлено неопределенным. Если присутствует «DEFAULT», то последующий «Type» обеспечивает установку умолчания для поля, если оно опущено в определении.

9.6 Продукция «FixedTypeValueFieldSpec» определяет, что поле является полем значения фиксированного-типа (см. 3.4.18).

FixedTypeValueFieldSpec : : =
valuefieldreference
Type
UNIQUE?

ValueOptionalitySpec?
ValueOptionalitySpec : : = OPTIONAL | DEFAULT Value

Имя поля есть «valuefieldreference». Конструкция «Type» специфицирует тип значения, содержащегося в поле. Продукция «ValueOptionalitySpec», если присутствует, специфицирует, что значение может быть опущено в определении информационного объекта или, в случае «DEFAULT», это опускание порождает последующее значение «Value», которое должно быть того же самого типа. Присутствие ключевого слова «UNIQUE» специфицирует, что это поле является полем идентификатора. Если присутствует ключевое слово, то продукция «ValueOptionalitySpec» не должна быть альтернативой «DEFAULT Value».

9.7 Когда полю идентификатора присваивается значение, то требуется, чтобы это значение было недвусмысленным в определенном множестве информационных объектов.

9.8 Продукция «VariableTypeValueFieldSpec» определяет, что поле является полем значения переменного-типа (см. 3.4.18).

VariableTypeValueFieldSpec : : =
valuefieldreference
FieldName
ValueOptionalitySpec?

Имя поля есть «valuefieldreference». Продукция «FieldName» (см. 9.14), которая относится к определяемому классу, должна быть полем типа; поле типа, которое либо находится в том же информационном объекте как поле значения, либо связывается цепочкой полей объектов, ссылки на которые появляются в «FieldName», должно содержать тип значения. (Все поля связи, ссылки на которые появляются в «FieldName», должны быть полями объектов). Продукция «ValueOptionalitySpec», если присутствует, специфицирует, что значение может быть опущено в определении информационного объекта или, в случае «DEFAULT», это опускание порождает последующее значение «Value». Продукция «ValueOptionalitySpec» должна быть такой, что:

а) если поле типа, обозначенное «FieldName», имеет продукцию «TypeOptionalitySpec» «OPTIONAL», то «ValueOptionalitySpec» должна также быть «OPTIONAL», и

б) если продукция «ValueOptionalitySpec» есть «DEFAULT Value», то поле типа, обозначенное «FieldName», должно иметь продукцию «TypeOptionalitySpec» «DEFAULT Type», и значение «Value» должно быть значением этого типа.

9.9 Продукция «FixedTypeValueSetFieldSpec» определяет, что поле является полем множества значений фиксированного-типа (см. 3.4.19):

FixedTypeValueSetFieldSpec : : =
valuesetfieldreference
Type
ValueSetOptionalitySpec?
ValueSetOptionalitySpec : : = OPTIONAL | DEFAULT ValueSet

Примечание — Продукция «ValueSet» определяется в ГОСТ Р ИСО/МЭК 8824-1, 15.4, 15.5 и позволяет явно перечислять (в фигурных скобках) множество значений или использовать ссылку «typereference» для подтипа «Type».

Имя поля есть «valuesetfieldreference». Конструкция «Type» специфицирует тип значений, содержащихся в поле. Продукция «ValueSetOptionalitySpec», если присутствует, указывает, что поле в определении информационного объекта может быть неспецифицированным или, в случае «DEFAULT», это опускание порождает последующее множество значений «ValueSet», которое должно быть подтипом этого типа.

9.10 Продукция «VariableTypeValueSetFieldSpec» определяет, что поле является полем множества значений переменного-типа (см. 3.4.19).

VariableTypeValueSetFieldSpec : : =
valuesetfieldreference
FieldName
ValueSetOptionalitySpec?

Имя поля есть «valuesetfieldreference». Продукция «FieldName» (см. 9.14), которая относится к определяемому классу, должна быть полем типа; поле типа, которое либо находится в том же информационном объекте как поле множества значений, либо связывается цепочкой полей объектов, ссылки на которые появляются в «FeidlName», должно содержать тип значений. (Все поля связи, ссылки на которые появляются в «FieldName», должны быть полями объектов). Продукция «ValueSetOptionalitySpec», если присутствует, специфицирует, что множество значений может быть опу-

щено в определении информационного объекта или, в случае «DEFAULT», это опускание порождает последующее множество значений «ValueSet». Продукция «ValueSetOptionalitySpec» должна быть такой, что:

а) если поле типа, обозначенное «FieldName», имеет продукцию «TypeOptionalitySpec» «OPTIONAL», то продукция «ValueSetOptionalitySpec» должна также быть «OPTIONAL», и

б) если продукция «ValueSetOptionalitySpec» есть «DEFAULT ValueSet», то поле типа, обозначенное «FieldName», должно иметь продукцию «TypeOptionalitySpec» «DEFAULT Type», а множество значений «ValueSet» должно быть значением этого типа.

9.11 Продукция «ObjectFieldSpec» определяет, что поле является полем информационного объекта (см. 3.4.10).

ObjectFieldSpec : : =
    objectfieldreference
    DefinedObjectClass
    ObjectOptionalitySpec?

ObjectOptionalitySpec : : = OPTIONAL | DEFAULT Object

Имя поля есть «objectfieldreference». Продукция «DefinedObjectClass» указывает класс объекта, содержащегося в поле (который может быть классом «ObjectClass», определяемым в настоящее время). Продукция «ObjectOptionalitySpec», если присутствует, устанавливает, что поле в определении информационного объекта может быть неспецифицированным или, в случае «DEFAULT», это опускание порождает последующий объект «Object» (см. 11.2), который должен быть «DefinedObjectClass».

9.12 Продукция «ObjectSetFieldSpec» определяет, что поле является полем множества информационных объектов (см. 3.4.12).

ObjectSetFieldSpec : : =
    objectsetfieldreference
    DefinedObjectClass
    ObjectSetOptionalitySpec?

ObjectSetOptionalitySpec : : = OPTIONAL | DEFAULT ObjectSet

Имя поля есть «objectsetfieldreference». Продукция «DefinedObjectClass» указывает класс объектов, содержащихся в поле. Продукция «ObjectSetOptionalitySpec», если присутствует, устанавливает, что поле в определении информационного объекта может быть неспецифицированным или, в случае «DEFAULT», это опускание порождает последующее множество объектов «ObjectSet» (см. 12.2), все объекты которого должны быть «DefinedObjectClass».

9.13 Конструкция «PrimitiveFieldName» используется для идентификации поля относительно класса, содержащего его спецификацию:

PrimitiveFieldName : : =
    typefieldreference |
    valuefieldreference |
    valuesetfieldreference |
    objectfieldreference |
    objectsetfieldreference

Имена всех полей, специфицированых в определении класса, должны быть различными.

9.14 Конструкция «FieldName» используется для идентификации поля относительно некоторого класса, который либо непосредственно содержит спецификацию поля, либо имеет цепочку полей связи к содержащему классу. Цепочка указывается списком имен «PrimitiveFieldName», разделенных точками.

FieldName : : = PrimitiveFieldName «.» +

9.15 Если имеется любая цепочка (одна или более) спецификаций полей связи (см. 3.4.14) такая, что:

а) первое поле находится в том классе, который определяется, и

б) каждое последующее является полем класса, использованного при определении предыдущего, и

в) последнее поле определяют, используя класс, который определяется, то, по крайней мере, одна из спецификаций полей должна иметь «ObjectOptionalitySpec» или «ObjectSetOptionalitySpec».

Примечание — Это условие должно предотвратить рекурсивные определения классов информационных объектов без конечного представления для информационных объектов этого рекурсивного класса.

9.16 Примеры

Расширенная версия класса информационных объектов, описанного неформально как пример в 3.4.9, может определяться следующим образом:

```
OPERATION : : = CLASS
{
    &ArgumentType OPTIONAL,
    &ResultType   OPTIONAL,
    &Errors       ERROR OPTIONAL,
    &Linked       OPERATION OPTIONAL,
    &resultReturned BOOLEAN DEFAULT TRUE,
    &code         INTEGER UNIQUE
}
ERROR : : = CLASS
{
    &ParameterType OPTIONAL,
    &code          INTEGER UNIQUE
}
```

Примечания

1 Этот пример основывается на понятиях операции и ошибки из стандартов по удаленным операциям, но упрощен с целью наглядности.

2 Поля, специфицируемые для этого класса, включают два поля типа (&ArgumentType и &ResultType) двух полей множеств объектов (&Errors и &Linked) и двух полей значений (&resultReturned и &code), последнее является полем идентификатора.

3 Множество информационных объектов, образованное классом OPERATION, должно быть таким, чтобы не было двух объектов в множестве, имеющих одно и то же значение для поля &code. (То же самое применяется к множествам объектов ERROR).

4 Класс информационных объектов OPERATION включает цепочку полей связи, описанную в 9.15. Цепочка длиной единица образуется полем &Linked, которое специфицируется (рекурсивно) с помощью OPERATION. Однако это недопустимо, так как поле обозначено как OPTIONAL.

5 Никакой из этих примеров не содержит продукцию «WithSyntaxSpec». Соответствующие примеры приводятся в 10.13.

10 Список синтаксисов

10.1 Часто бывает, что одна спецификация определяет класс информационных объектов, для которого несколько других независимых отдельных спецификаций определяют информационные объекты. Возможно, автору определения класса следует предоставить дружественную пользователю нотацию для определения информационных объектов этого класса.

Примечание — Именно для этого (исторически) использовалась главным образом «макронотация» АСН. 1 до того, как была заменена настоящим стандартом.

10.2 В данном разделе установлена нотация, с помощью которой разработчик класса информационных объектов определяет специфический для класса синтаксис определения информационных объектов этого класса.

10.3 Нотация является синтаксической конструкцией «SyntaxList», которая встречается в синтаксической конструкции «ObjectClassDefn».

10.4 Конструкция «SyntaxList» специфицирует синтаксис для определения единственного информационного объекта определяемого класса. Синтаксис появляется как «DefinedSyntax» в последующих пунктах.

Примечание — Свойством настоящей спецификации является то, что конец любой синтаксической конструкции, определенный «SyntaxList» (экземпляра «DefinedSyntax»), может быть найден путем:

а) игнорирования комментариев АСН. 1;

б) трактовки значений символьных строк как лексических признаков;

в) проставления начальной «{», согласовывая вложенные «{» и «}», и завершения конструкции несогласованной «}».

10.5 Конструкция «SyntaxList» специфицирует последовательность «DefinedSyntaxToken», которая должна появляться в «DefinedSyntax» (см. 11.5).

```
SyntaxList : : = «{» TokenOrGroupSpec empty + «}»
TokenOrGroupSpec : : = RequiredToken | OptionalGroup
```

OptionalGroup : : = «[» TokenOrGroupSpec empty + «|»
RequiredToken : : =
Literal |
PrimitiveFieldName

Примечание — Разработчику конструкции «SyntaxList» не предоставляются полные возможности БНФ. Грубо говоря, мощность нотации эквивалента той, которая обычно используется при спецификации синтаксисов командной строки для интерпретаторов команд. Список возможных признаков «RequiredToken» дается в том порядке, в каком они допустимы; один или несколько последовательных признаков могут быть сделаны факультативными путем заключения их в квадратные скобки.

10.6 Признак «word», используемый как литерал «Literal», не должен быть одним из следующих:
BIT
BOOLEAN
CHARACTER
CHOICE
EMBEDDED
END
ENUMERATED
EXTERNAL
FALSE
INSTANCE
INTEGER
INTERSECTION
MINUS—INFINITY
NULL
OBJECT
OCTET
PLUS—INFINITY
REAL
RELATIVE—OID
SEQUENCE
SET
TRUE
UNION

Примечание — Этот список включает только те (и все те) зарезервированные слова АСН. 1, которые могут появляться в первом элементе продукций «Type», «Value», «ValueSet», «Object» или «ObjectSet», а также зарезервированное слово «END». Использование других зарезервированных слов АСН. 1 не вызывает двусмысленности и разрешается. Когда определяемый синтаксис используется в окружении, в котором «word» является также «typereference» или «objectsetreference», то их использование в качестве «word» имеет предпочтение.

10.7 Продукция «Literal» специфицирует фактическое включение того литерала «Literal», который является либо «word», либо запятой («,»), в данной позиции в определяемом синтаксисе.
Literal : : =
word |
«,»

10.8 Каждая альтернатива «PrimitiveFieldName» специфицирует включение (в данной позиции в новом синтаксисе) продукции «Setting» (см. 11.6) для соответствующего поля.

10.9 Каждая альтернатива «PrimitiveFieldName» класса информационных объектов должна появляться ровно один раз.

10.10 Когда в процессе синтаксического разбора встречается альтернатива «OptionalGroup», а следующий элемент АСН. 1 является синтаксически приемлемым в качестве первого элемента АСН. 1 в факультативной группе, то принимается, что эта группа присутствует. Если следующий элемент не является синтаксически приемлемым в качестве первого элемента АСН. 1 в факультативной группе, то принимается, что эта группа отсутствует.

Примечание — Для того чтобы предотвратить неожиданные эффекты, разработчики обычно делают первым элементом АСН. 1 в факультативной группе литерал «Literal».

10.11 Экземпляр использования «DefinedSyntax» является недопустимым, если он не специфицирует все обязательные поля для класса информационных объектов.

10.12 Для того чтобы обеспечить простой синтаксический разбор нового синтаксиса и предотвратить правильное употребление, следующие дополнительные ограничения накладываются на автора определения нового синтаксиса:

а) требуется, чтобы каждая продукция «OptionalGroup» содержала в себе по крайней мере одну продукцию «PrimitiveFieldName» или «OptionalGroup»;

Примечание 1 — Это помогает предотвратить видимое скопление информации, которое не отражается ни в каком поле информационного объекта;

б) использование продукции «OptionalGroup» должно быть таким, чтобы в процессе синтаксического разбора никогда не могла появиться продукция «Setting», которая потенциально может быть установкой для более чем одного «FieldName»;

в) если продукция «OptionalGroup» начинается с альтернативы «Literal», то первый признак, следующий за «OptionalGroup», должен также быть «Literal» и отличаться от первого литерала «Literal» во всех непосредственно следующих конструкциях «OptionalGroup».

Следующие ограничения накладываются на пользователя продукции «DefinedSyntax»:

г) всякий раз, когда альтернатива «Literal» присутствует в «DefinedSyntax», которая встречается в «OptionalGroup», альтернатива «Setting» для «PrimitiveFieldName» в этой «OptionalGroup» также должна присутствовать.

Примечания

2 Это ограничение помогает предотвратить видимое скопление информации, которая не отражается ни в каком поле информационного объекта.

3 Следующий пример является допустимым синтаксисом, но ограничение г) не позволяет пользователю писать «LITERAL» без следующих за ним одной или обеих факультативных групп.

[LITERAL [A &field] [B &field2]]

10.13 Примеры

Примеры определений классов из 9.16 могут быть дополнены определяемым синтаксисом для обеспечения дружественного пользователям способа определения экземпляров классов. (Этот определяемый синтаксис используется в примере в 11.10).

```
OPERATION : : = CLASS
{
    &ArgumentType    OPTIONAL,
    &ResultType      OPTIONAL,
    &Errors          ERROR OPTIONAL,
    &Linked          OPERATION OPTIONAL,
    &resultReturned  BOOLEAN DEFAULT TRUE,
    &operationCode   INTEGER UNIQUE
}
WITH SYNTAX
}
    [ARGUMENT    &ArgumentType]
    [RESULT      &ResultType]
    [RETURN RESULT &resultReturned]
    [ERRORS      &Errors]
    [LINKED      &Linked]
    CODE         &operationCode
}
ERROR : : = CLASS
{
    &ParameterType   OPTIONAL
    &errorCode       INTEGER UNIQUE
}
WITH SYNTAX
}
    [PARAMETER   &ParameterType]
    CODE         &errorCode
}
```

11 Определение и присвоение информационного объекта

11.1 Синтаксическая конструкция «ObjectAssignment» используется для присвоения информационному объекту заданного класса ссылочного имени («objectreference»). Эта конструкция является одной из альтернатив для «Assignment» в ГОСТ Р ИСО/МЭК 8824-1, раздел 12 и определяется следующим образом:

ObjectAssignment : : =
objectreference
DefinedObjectClass
«: : =»
Object

11.2 Информационный объект, который должен быть класса, указанного «DefinedObjectClass», является объектом, определяемым конструкцией «Object».

Object : : =
DefinedObject |
ObjectDefn |
ObjectFromObject |
ParameterizedObject

Если «Object» является:

а) «DefinedObject», то объект — тот, который указан;

б) «ObjectDefn», то объект определяется, как специфицировано в 11.3;

в) «ObjectFromObject», то объект определяется, как специфицировано в разделе 15;

г) «ParameterizedObject», то объект определяется, как специфицировано в ИСО/МЭК 8824-4, 9.2.

11.3 Каждый информационный объект, в конечном счете, определяется «ObjectDefn».:

ObjectDefn : : =
DefaultSyntax |
DefinedSyntax

Продукция «ObjectDefn» должна быть альтернативой «DefaultSyntax» (см. 11.4), если определение класса не содержит «WithSyntaxSpec», и «DefinedSyntax» (см. 11.5), если содержит.

11.4 Синтаксис по умолчанию «DefaultSyntax» определяется следующим образом:

DefaultSyntax : : = «{» FieldSetting», «,» * «}»

FieldSetting : : = PrimitfveFieldNameSetting

Здесь должна быть ровно одна альтернатива «FieldSetting» для каждой конструкции «FieldSpec» в определении класса, для которой не указано «OPTIONAL» или «DEFAULT», и должно быть не более одной альтернативы «FieldSetting» для каждой конструкции «FieldSpec». Альтернативы «FieldSetting» могут появляться в произвольном порядке. Конструкция «PrimitiveFieldName» в каждой конструкции «FieldSetting» должна быть именем соответствующей продукции «FieldSpec». Конструкция «Setting» сопределяется в 11.6.

11.5 Определяемый синтаксис «DefinedSyntax» задается следующим образом:

DefinedSyntax : : = «{» DefinedSyntaxToken emply * «}»

DefinedSyntaxToken : : =
Literal |
Setting

Конструкция «SintaxList» в конструкции «WithSyntaxSpec» (см. раздел 10) определяет последовательность конструкций «DefinedSyntaxToken», которая должна появляться в конструкции «DefinedSyntax». Конструкция «Setting» определяется в 11.6; каждое ее появление специфицирует установку для некоторого поля информационного объекта. Конструкция «Literal» определена в 10.7; конструкции «Literal» присутствуют для удобочитаемости.

11.6 Конструкция «Setting» специфицирует установку некоторого поля в определяемом информационном объекте.

Setting : : =
Type |
Value |
ValueSet |
Object |
ObjectSet

Если поле является:

a) полем типа, то выбирается альтернатива «Type»;

b) полем значения, то выбирается альтернатива «Value»;

c) полем множества значений, то выбирается альтернатива «ValueSet»;

d) полем информационных объектов, то выбирается альтернатива «Object»;

e) полем множества информационных объектов, то выбирается альтернатива «ObjectSet».

Примечание — Установка ограничивается, как описано в 9.5 — 9.12 и 11.7 — 11.8.

11.7 Установка поля значений переменного-типа должна быть значением типа, заданного соответствующим полем типа того же самого или связанного объекта (то есть нотация значения для открытого типа не допускается).

11.8 Установка поля множества значений переменного-типа должна быть множеством значений типов, заданных соответствующим полем типа того же самого или связанного объекта (то есть нотация значения для открытого типа не допускается).

11.9 Примеры (синтаксис по умолчанию)

Для определений классов информационных объектов из 9.16 (которые не содержат «WithSyntaxSpec») экземпляры классов определяются с использованием «DefaultSyntax». Например (расширенная версия примера, данного в 3.4.7):

```
invertMatrix OPERATION : : =
{
    &ArgumentType     Matrix
    &ResultType       Matrix
    &Errors           {determinant IsZero}
    &operationCode    7
}
determinantIsZero ERROR : : =
{
    &errorCode 1
}
```

11.10 Примеры (определяемый синтаксис)

В 10.13 примеры классов содержат «WithSyntaxSpec» и, таким образом, элементы классов определяются с использованием «DefinedSyntax». Примеры 11.9 могли бы быть написаны следующим образом:

```
invertMatrix OPERATION : : =
{
    ARGUMENT Matrix
    RESULT Matrix
    ERRORS {determinantIsZero}
    CODE 7
}
determinantIsZero ERROR : : =
{
    CODE 1
}
```

12 Определение и присвоение множества информационных объектов

12.1 Синтаксическая конструкция «ObjectSetAssignment» используется для присвоения множеству информационных объектов заданного класса, ссылочного имени («objectsetreference»). Эта конструкция является одной из альтернатив для «Assignment» в ГОСТ Р ИСО/МЭК 8824-1, раздел 12 и определяется следующим образом:

```
ObjectSetAssignment : : =
    objectsetreference
    DefinedObjectClass
    «: : = »
    ObjectSet
```

12.2 Множество информационных объектов, которые должны быть класса, указанного «DefinedObjectClass», является множеством, определяемым конструкцией «ObjectSet»

ObjectSet : : = «{» ObjectSetSpec «}»

ObjectSetSpec : : =

RootElementSetSpec |

RootElementSetSpec «, » «. . .» |

«. . .» |

«. . .» «, » AddidionalElementSetSpec |

RootElementSetSpec «,» «. . .» «,» AdditionalElementSetSpec

Конструкции «RootElementSetSpec» и «AdditionalElementSetSpec» определены в ГОСТ Р ИСО/МЭК 8824-1 и позволяют специфицировать множество информационных объектов в терминах информационных объектов или их множеств из управляющего класса. В множестве должен быть хотя бы один информационных объект, если в «ObjectSetSpec» не задается третья альтернатива («. . . »). В последнем случае многоточие указывает, что множество объектов первоначально пусто, но объекты будут динамически добавляться к нему прикладной программой.

Примечания

1 Элементы, на которые указывает «ObjectSetSpec», являются объединением элементов, указываемых «RootElementSetSpec» и «AdditionalElementSetSpec».

2 В отличие от расширяемых типов, таких как множество или последовательность, или ограничений расширяющих подтипов, которые являются статическими относительно множества «понятных» значений, устанавливаемых для каждой версии спецификации АСН. 1, расширяемое множество объектов может динамически расти и сжиматься в пределах данной версии. Действительно, оно может расширяться и сжиматься с данным экземпляром использования прикладной программы по мере того, как программа динамически определяет и уничтожает объекты.

12.2.1 Результат арифметической установки, применяемой к множествам расширяемых объектов, определен в ГОСТ Р ИСО/МЭК 8824-1, раздел 46.

12.3 Если расширяемое множество объектов А указывается в определении другого множества объектов В, то его маркер расширения наследуется В.

12.4 Если конструкция «ValueSetFromObjects» (см. раздел 15) определена с использованием расширяемого множества объектов, то результирующее множество значений не наследует маркер расширения от множества объектов.

12.5 Если тип ограничен табличным ограничением (см. 10.3 ИСО/МЭК 8824-3) и множество объектов, указанных в табличном ограничении, является расширяемым, то тип не наследует маркер расширения от множества объектов. Если тип предназначается для того, чтобы быть расширяемым, то маркер расширения должен быть явно добавлен к его «ElementSetSpecs».

12.6 Нотация для «ObjectSetElements» следующая:

ObjectSetElements : : =

Object |

DefinedObjectSet |

ObjectSetFromObjects |

ParameterizedObjectSet

Элементы, специфицированные этой нотацией, определяются использованной альтернативой следующим образом:

а) если используется альтернатива «Object», то определяется только объект, который также обозначен. Этот объект должен быть объектом управляющего класса;

б) если используется любая из оставшихся альтернатив, то определяются все объекты множества, обозначенные так же. Объекты должны быть объектами управляющего класса. Если используется альтернатива «DefinedObjectSet», то множество объектов есть то, которое указано. Если используется альтернатива «ObjectSetFromObjects», то множество объектов такое, как определено в разделе 15. Если используется альтернатива «ParameterizedObjectSet», то множество объектов такое, как определено в ИСО/МЭК 8824-4, 9.2.

12.7 Пример

Множество информационных объектов, неформально описанное в примечании к 3.4.11, может быть специфицировано следующим образом:

```
MatrixOperations OPERATION : : =
{
    invertMatrix |
    addMatrices |
    subtractMatrices |
    multiplyMatrices
}
```

13 Ассоциированные таблицы

13.1 Каждый информационный объект или множество информационных объектов могут быть представлены как таблица — его ассоциированная таблица. Каждая ячейка ассоциированной таблицы соответствует установке некоторого поля информационного объекта или является пустой. Совокупность столбцов ассоциированной таблицы определяется классом, к которому объект или объекты относятся; совокупность строк определяется объектом или объектами, которые вызваны.

13.2 Для заданного определения класса совокупность столбцов определяется следующим образом:

а) имеется по одному столбцу для каждой спецификации поля в определении класса. Каждый такой столбец называется соответствующим именем «PrimitiveFieldName»;

б) имеется дополнительная совокупность столбцов, соответствующих каждой спецификации поля связи. Эта совокупность определяется применением настоящих правил к управляющему классу поля связи, за исключением того, что их имена дополняются префиксом «PrimitiveFieldName» поля связи и точкой («.»).

Примечание — Эти правила являются рекурсивными, и таким образом, если класс, прямо или косвенно, является самоссылающимся, то совокупность столбцов не ограничена. Это не запрещается.

13.3 Для данного информационного объекта некоторого класса ассоциированной таблицей является та, которая появляется в результате применения 13.4 к множеству объектов, содержащему только этот объект.

13.4 Для данного множества информационных объектов некоторого класса совокупность строк в ассоциированной таблице является той, которая получается при осуществлении следующей рекурсивной процедуры:

а) сначала имеется одна строка для каждого объекта в множестве объектов. В каждой такой строке ячейки в столбцах, названных именами «PrimitiveFieldName», будут соответствовать установке соответствующего поля в объекте, а все другие ячейки будут пустыми;

б) для каждого поля связи, появляющегося в некоторой строке в совокупности:

1) создается (подчиненная) ассоциированная таблица содержимого поля связи;

2) далее строка, в которой появилось поле связи, заменяется совокупностью строк по одной для каждой строки подчиненной ассоциированной таблицы. Каждая строка в этой совокупности является такой же, как и замененная, за исключением того, что ячейки из выбранной строки подчиненной ассоциированной таблицы используются для наполнения соответствующих ячеек (прежде пустых), имена «FieldName» которых имеют в качестве префикса «PrimitiveFieldName» полей связи.

Примечание — Эти правила рекурсивны и такие, что, если информационный объект, прямо или косвенно, является самоссылающимся, то процедура не будет завершаться. Это не запрещается. На практике необходимо только знать содержимое ячеек с именами конечной длины, а для них может быть создана конечная процедура.

13.5 Примеры допустимых конструкций «FieldName»

Следующие конструкции «FieldName» относятся к допустимым для ассоциированной таблицы для объектов или множеств объектов класса OPERATION (определенного в 10.13):

```
&ArgumentType
&Errors.&Parameter
&EiTors.&errorCode
&Linked.&ArgumentType
&Linked.&Linked. × operationCode
&Linked.&Linked.&Linked.&Linked.&Linked.&Errors.&ErrorCode
```

Так как класс OPERATION является самоссылающимся (из-за поля &Linked), то количество столбцов не ограничено.

14 Нотация для типа «поле класса объектов»

Тип, который указывается этой нотацией, зависит от категории имени поля. Для различных категорий имен полей в 14.2 — 14.5 специфицируется тип, который они указывают.

14.1 Нотацией для типа «поле класса объектов» (см. 3.4.15) должна быть «ObjectClassFieldType»;

ObjectClassFieldType : : =
DefinedObjectClass
«.»
FieldName

где «FieldName», определяется как в 9.14 относительно класса, идентифицированного «DefinedObjectClass».

14.2 Для поля типа нотация определяет открытый тип, то есть тот, множество значений которого является полным множеством всех возможных значений, которые можно специфицировать, используя АСН. 1. Спецификация ограничений, использующая соответствующее множество информационных объектов (см. ИСО/МЭК 8824-3), может ограничивать этот тип конкретным типом. Следующие ограничения на использование данной нотации применяются, когда «FieldName» указывает поле типа.

а) Эта нотация не должна, прямо или косвенно, использоваться в определении типа поля значения или множества значений класса информационных объектов.

б) Эта нотация имеет неопределенный тег и, таким образом, не может быть использована там, где требуется, чтобы тег отличался от некоторых других типов.

Примечания

1 Это ограничение обычно можно обойти (явным) тегированием типа.

2 Несмотря на утверждение в ГОСТ Р ИСО/МЭК 8824-1, 47.7.3, что концептуально добавляемый элемент для маркера расширения имеет тег, отличный от тегов всех известных типов АСН. 1, открытый тип не должен использоваться, когда требуется, чтобы он имел тег, отличный от тега концептуально добавляемого элемента.

в) Эта нотация не должна быть неявно тегированной.

Примечание 3 — Причиной этого ограничения является то, что, когда данный открытый тип ограничивается до конкретного типа, последний может оказаться выборочным типом.

г) Требуется, чтобы правила кодирования для значения, присвоенного определенному таким образом компоненту, были такими, чтобы получатель мог успешно определить абстрактные значения, соответствующие всем другим частям конструкции, в которую компонент вставлен, без какого-любого знания о фактическом типе этого компонента.

Примечание 4 — Такая конструкция «Type» обычно ограничивается использованием множества информационных объектов и «AtNotation», как определено в ИСО/МЭК 8824-3, раздел 10. Однако пользователи АСН. 1 должны учитывать, что применение этой нотации без ограничения может привести к двусмысленности в требованиях к реализации, и обычно ее следует избегать.

14.3 Для поля значения фиксированного-типа или множества значений фиксированного-типа нотация обозначает тип «Type», который появляется в спецификации этого поля в определении класса информационных объектов.

14.4 Для поля значения переменного-типа или множества значений переменного-типа нотация определяет открытый тип. Она используется при тех же ограничениях, оговоренных в 14.2.

14.5 Эта нотация недопустима, если поле является полем объекта или полем множества объектов.

14.6 Нотацией для определения значения этого типа должно быть «ObjectClassFieldValue»:

ObjectClassFieldValue : : =
OpenTypeFieldVal |
FixedTypeFieldVal

OpenTypeFieldVal : : = Type «:» Value

FixedTypeFieldVal : : = BuiltinValue | ReferencedValue

14.7 Для поля значения или множества значений фиксированного-типа, используемого в «ObjectClassFieldType», должна использоваться альтернатива «FixedTypeFieldVal» и должно быть значение «Type», указанное в определении класса информационных объектов.

14.8 Для поля типа, поля значения или множества значений переменного-типа, использованного в «ObjectClassFieldType», должна использоваться альтернатива «OpenTypeFieldVal». Тип «Type»

в «OpenTypeFieldVal» должен быть любым типом АСН. 1, а значение «Value» — любым значением того типа.

14.9 Пример использования конструкции «ObjectClassFieldType»

Каждый из следующих примеров основывается на примере в 10.13 и показывает: а) возможный «ObjectClassFieldType», б) тип, которому тип примера а) эквивалентен (когда используется неограниченно), и в) нотацию для примера значения того типа.

1(a) OPERATION.&operationCode
(b) INTEGER
(c) 7
2(a) OPERATION.&ArgumentType
(b) открытый тип
(c) Матрица:
{{1, 0, 0, 0},
{0, 1, 0, 0},
{0, 0, 1, 0},
{0, 0, 0, 1}}
3 (a) OPERATION.&Linked.&Linked.&Errors.&errorCode
(b) INTEGER
(c) 1
4 (a) OPERATION.&Linked.&ArgumentType
(b) открытый тип
(c) UniversalString: {planckConstant, «and», hamiltonOperator}

15 Информация из объектов

15.1 Информация из столбцов ассоциированной таблицы для объекта или множества объектов может быть указана различными способами в нотации «InformationFromObjects».

```
InformationFromObjects : : =
    ValueFromObject |
    ValueSetFromObjects |
    TypeFromObject |
    ObjectFromObject |
    ObjectSetFromObjects
ValueFromObject : : =
    ReferencedObjects
    «.»
    FieldName
ValueSetFromObjects : : =
    ReferencedObjects
    «.»
    FieldName
TypeFromObject : : =
    ReferencedObjects
    «.»
    FieldName
ObjectFromObject : : =
    ReferencedObjects
    «.»
    FieldName
ObjectSetFromObjects : : =
    ReferencedObjects
    «.»
    FieldName
ReferencedObjects : : =
    DefinedObject | ParameterizedObject |
    DefinedObjectSet | ParameterizedObjectSet
```

15.2 Эта нотация ссылается на все содержимое указываемого столбца ассоциированной таблицы для объектов «ReferencedObjects».

15.3 В зависимости от формы «ReferencedObjects» и «FieldName» данная нотация может обозначать значение, множество значений, тип, объект или множество объектов. Эти пять случаев обозначаются конструкциями «ValueFromObject», «ValueSetFromObjects», «TypeFromObject», «ObjectFromObject» и «ObjectSetFromObjects» соответственно. Каждая такая конструкция является частным случаем нотации «InformationFromObjects».

15.4 Продукция «InformationFromObjects» может быть разделена на две части. Первая часть образуется удалением последнего (или единственного) «PrimitiveFieldName» и предшествующей точки. Если первая часть обозначает объект или множество объектов, то применяются положения 15.5 — 15.9. В противном случае нотация недопустима. Вторая часть является последним (или единственным) «PrimitiveFieldName».

Примечание — Применимо в качестве примера следующее определение:
obj.&a.&b.&c.&d
Первая часть в определении есть obj.&a.&b.&c, а вторая часть есть &d.

15.5 В первом столбце таблицы 1 показана первая часть, во втором столбце — вторая часть, определенные в 15.4. В третьем столбце указано, какой из пяти случаев «InformationFromObjects» (из перечисленных в 15.3) применяется.

Таблица 1 — Допустимые варианты «InformationFromObjects»

Первая часть InformationFromObjects	Вторая часть InformationFromObjects	Конструкция
Объект	Поле значения фиксированного-типа	«ValueFromObject»
	Поле значения переменного типа	«ValueFromObject»
	Поле множества значений фиксированного-типа	«ValueSetFromObjects»
	Поле множества значений переменного-типа	«ValueSetFromObjects»
	Поле типа	«TypeFromObject»
	Поле объекта	«ObjectFromObject»
	Поле множества объектов	«ObjectSetFromObject»
Множество объектов	Поле значения фиксированного-типа	«ValueSetFromObjects»
	Поле значения переменного типа	Не допускается
	Поле множества значений фиксированного-типа	«ValueSetFromObjects»
	Поле множества значений переменного-типа	Не допускается
	Поле типа	Не допускается
	Поле объекта	«ObjectSetFromObjects»
	Поле множества объектов	«ObjectSetFromObjects»

15.6 Если имеются множества объектов, и последнее имя «PrimitiveFieldName» идентифицирует поле множества значений фиксированного-типа, то «ValueSetFromObjects» является объединением выбранных множеств значений.

Примечание — ГОСТ Р ИСО/МЭК 8824-1, 44.6 запрещает определение множества значений, не содержащего значений.

15.7 Если имеются множества объектов, и последнее имя «PrimitiveFieldName» идентифицирует поле множества объектов, то «ObjectSetFromObjects» является объединением выбранных множеств объектов.

15.8 Как показано в таблице 1, нотация не допускается, если включается множество объектов, а последнее имя «PrimitiveFieldName» идентифицирует поле значения множества значений переменного-типа или поле типа.

15.9 Использование этой нотации не допускается, если все ячейки в столбце, который указывается, пусты, за исключением случаев, когда это используется для непосредственного опре-

деления поля информационного объекта, для которого указано «OPTIONAL» (или «DEFAULT»), что приводит к полю, являющемуся пустым (или принимаемым по умолчанию).

15.10 Пример информации из объектов

Для определений в примерах 11.9, 11.10 и 12.7 следующие конструкции (в левых столбцах) являются допустимыми и могут использоваться в качестве эквивалентов выражений в правых столбцах.

«ValueFromObject»

invertMatrix.&operationCode	7
determinantIsZero.&errorCode	1

«TypeFromObject»

invertMatrix.&ArgumentType	Matrix

«ValueSetFromObject»

invertMatrix.&Errors.&errorCode	{1}
MatrixOperations.&operationCode	{7 | и другие}

«ObjectSetFromObjects»

invertMatrix.&Errors	{determinantIsZero}
MatrixOperations.&Errors	{determinantIsZero | и другие}

ПРИЛОЖЕНИЕ А
(обязательное)

Класс информационных объектов TYPE-IDENTIFIER

А.1 В данном приложении определяется полезный класс информационных объектов со ссылкой на класс TYPE-IDENTIFIER.

Примечание — Этот класс информационных объектов является простейшим классом, имеющим ровно два поля — поле идентификатора типа OBJECT IDENTIFIER и поле типа, который определяет тип АСН. 1 для передачи всей информации, касающейся любого конкретного объекта этого класса. Он определяется в настоящем стандарте из-за широко распространенного использования информационных объектов этой формы.

А.2 Класс информационных объектов TYPE-IDENTIFIER определяется как:

```
TYPE-IDENTIFIER : : = CLASS
{
        &id OBJECT IDENTIFIER UNIQUE,
        &Type
}
WITH SYNTAX {&Type IDENTIFIED BY &id}
```

А.3 Этот класс определяется как «полезный» класс информационных объектов и доступен в любом модуле без необходимости его импорта.

А.4 Пример

Тело связи MHS может быть определено как:

```
MHS-BODY-CLASS : : = TYPE-IDENTIFIER

g4FaxBody MHS-BODY-CLASS : : =
    {BIT STRING IDENTIFIED BY {mhsbody 3}}
```

Разработчик протокола мог бы определять компонент для передачи MHS-BODY-CLASS, специфицируя тип «INSTANCE OF MHS-BODY-CLASS», определенный в С.9.

ПРИЛОЖЕНИЕ В
(обязательное)

Определения абстрактных синтаксисов

В.1 В данном приложении определяется полезный класс информационного объекта, ABSTRACT-SYNTAX, для определения абстрактных синтаксисов.

Примечание — Рекомендуется, чтобы экземпляр этого класса информационного объекта определялся каждый раз, когда абстрактный синтаксис определяется как значения единственного типа ACH. 1.

В.2 Класс информационных объектов ABSTRACT-SYNTAX определяется как:

```
ABSTRACT-SYNTAX : : = CLASS {
    &id                 OBJECT IDENTIFIER,
    &Type,
    &property BIT STRING {handles-invalid-encodings (0)} DEFAULT {}
}WITH SYNTAX {
    &Type IDENTIFIER BY &id [HAS PROPERTY &property]
}
```

Поле &id каждого объекта ABSTRACT-SYNTAX является именем абстрактного синтаксиса, а поле &Type содержит единственный тип ACH. 1, значения которого образуют абстрактный синтаксис. Свойство «handles-invalid-encodings» указывает, что недопустимое кодирование не должно рассматриваться как ошибка во время процесса декодирования, а решение о том, как рассматривать такое недопустимое кодирование, остается за приложением.

В.3 Этот класс информационных объектов определен как «полезный» из-за его общей употребимости, и он доступен в любом модуле без необходимости его импорта.

В.4 Пример

Если определен тип ACH. 1, названный XXX-PDU, то может быть определен абстрактный синтаксис, который содержит все значения XXX-PDU, с помощью нотации:

```
xxx-Abstract-Syntax ABSTRACT-SYNTAX : : =
    {XXX-PDU IDENTIFIED BY {xxx 5}}
```

Подробные примеры использования класса информационных объектов ABSTRACT-SYNTAX см. в ГОСТ Р ИСО/МЭК 8824-1, С. 3.

В.5 Часто бывает, что абстрактный синтаксис определяется в терминах параметризованного типа (как определено в ИСО/МЭК 8824-4), например с параметрами, представляющими границы некоторых компонентов протокола. Такие параметры, ограниченные, как определено в ИСО/МЭК 8824-4, раздел 10, могут быть разрешены при определении абстрактного синтаксиса или переданы дальше как параметры абстрактного синтаксиса.

ПРИЛОЖЕНИЕ С
(обязательное)

Тип «экземпляр-из»

С.1 В настоящем приложении специфицирована нотация типа и значение для типов «экземпляр-из» (см. 3.4.13). Такие типы могут передавать любые значения из любого информационного объекта в классе информационных объектов, определенного как класс TYPE-IDENTIFIER (см. приложение А), используя присвоение класса информационных объектов (ссылка на класс информационных объектов определяется как часть этой нотации).

С.2 Нотация «InstanceOfType», указанная в ГОСТ Р ИСО/МЭК 8824-1, 16.2 как одна из нотаций, образующих «Type», определяется следующим образом:

InstanceOfType : : = INSTANCE OF DefinedObjectClass

Примечание — В ИСО/МЭК 8824-3, раздел 10 установлен способ, в котором этот тип может быть ограничен с использованием «табличного ограничения». При этом значения типа ограничиваются теми, которые представляют некоторое конкретное множество информационных объектов этого класса.

С.3 Данная нотация специфицирует тип, который переносит поле &id (OBJECT IDENTIFIER) и значение поля &Type из любого экземпляра класса «DefinedObjectClass».

Примечание — Обычно эта конструкция будет ограничиваться множеством объектов, которое будет (но не обязательно) пустым именем ссылки, как определено в ИСО/МЭК 8824-4, 8.3 — 8.11, с фактическим множеством объектов, определенным в другом месте.

С.4 Все типы «экземпляр-из» имеют тег универсального класса 8.

Примечание — Это тот же самый универсальный тег, что и для внешнего типа, и использование типа «экземпляр-из» может быть бит-совместимым с внешним типом, когда используются базовые правила кодирования АСН. 1.

С.5 Тип «экземпляр-из» имеет ассоциированный тип «последовательность», который используется для определения значений и подтипов типа «экземпляр-из».

Примечание — Когда этот тип ограничивается нотацией ограничения ИСО/МЭК 8824-3, то также ограничивается ассоциированный тип «последовательность». Ограничения на ассоциированный тип «последовательность», получающиеся из ограничения на тип «экземпляр-из», определены в ИСО/МЭК 8824-3, приложение А.

С.6 Принимается, что ассоциированный тип «последовательность» должен определяться в окружении, в котором установлено тегирование «EXPLICIT TAGS».

С.7 Ассоциированный тип «последовательность» должен быть:

```
SEQUENCE
{
    type-id     <DefinedObjectClass>.&id,
    value       [0] <DefinedObjectClass>.&Type
}
```

где «<DefinedObjectClass>» замещается конкретным классом «DefinedObjectClass», использованным в нотации «InstanceOfType».

С.8 Нотацией значения «InstanceOfValue» для нотации «InstanceOfType» должна быть нотация значения для ассоциированного типа «последовательность».

InstanceOfValue : : = Value

С.9 Пример

На примере, данном в А.4, можно построить следующий пример.

Тип

INSTANCE OF MHS-BODY-CLASS

имеет ассоциированный тип «последовательность»

```
SEQUENCE
{
        type-id MHS-BODY-CLASS.&id,
        value   [0] MHS-BODY-CLASS.&Type
}
```

Пример применения табличного ограничения к этому типу приведен в ИСО/МЭК 8824-3, приложение А.

ПРИЛОЖЕНИЕ D
(справочное)

Примеры

D.1 Пример использования упрощенного класса OPERATION

Применим следующее упрощенное определение классов информационных объектов OPERATION и ERROR:

```
OPERATION : : = CLASS
{
&ArgumentType OPTIONAL,
&ResultType OPTIONAL,
&Errors ERROR OPTIONAL,
&Linked OPERATION OPTIONAL,
&resultReturned BOOLEAN DEFAULT TRUE,
&operationCode INTEGER UNIQUE
}
WITH SYNTAX
{
[ARGUMENT    &ArgumentType]
[RESULT    &ResultType]
[RETURN RESULT &resultReturned]
[ERRORS    &Errors]
[LINKED    &Linked]
CODE    &operationCode
}
ERROR : : = CLASS
{
&ParameterType OPTIONAL
&errorCode INTEGER UNIQUE
}
WITH SYNTAX
}
[PARAMETER &ParameterType]
CODE &errorCode
}
```

Можно определить следующее множество объектов, которое содержит два объекта OPERATION:

```
My-Operations OPERATION : : = {operationA | operationB}
operationA OPERATION : : = {
    ARGUMENT INTEGER
    ERRORS {{PARAMETER INTEGER CODE 1000} | {CODE 1001}}
    CODE 1
}
operationB OPERATION : : = {
    ARGUMETN IA5String
RESULT          BOOLEAN
ERRORS          {{CODE 1002} | {PARAMETER IA5String CODE 1003}}
CODE            2
}
```

Извлечение множества объектов ERROR из приведенного выше множества объектов проводят следующим образом:

```
My-OperationErrors ERROR : : = {My-Operations.&Errors}
```

Результирующее множество объектов есть:

```
My-OperationErrors ERROR : : = {
        { PARAMETER INTEGER CODE 1000} |
        { CODE 1001} |
        { CODE 1002} |
        { PARAMETER IA5String CODE 1003}
```

Извлечение множества кодов ошибок операций проводят следующим образом:

```
My-OperationErrorCodes INTEGER : : = {My-Operations.&Errors.&errorCode}
```

Результирующее множество значений есть:

```
My-OperationErrorCodes INTEGER : : = {1000 | 1001 | 1002 | 1003}
```

D.2 Пример использования «ObjectClassFieldType»

Продукция «ObjectClassFieldType» может использоваться в спецификации типов, например:

```
- - Типы «ObjectClassFieldType» извлекаются из этого класса.
- - Только первые пять полей могут быть использованы при извлечении.
EXAMPLE-CLASS : : = CLASS {
&TypeField                                  OPTIONAL,
&fixedTypeValueField                        INTEGER OPTIONAL,
&variableTypeValueField     &TypeField      OPTIONAL,
&FixedTypeValueSetField          INTEGER OPTIONAL,
&VariableTypeValueSetField  &TypeField      OPTIONAL,
&objectField                SIMPLE-CLASS    OPTIONAL,
&ObjectSetField             SIMPLE-CLASS    OPTIONAL
}
WITH SYNTAX {
[TYPE-FIELD                             &TypeField]
[FIXED-TYPE-VALUE-FIELD                 &fixedTypeValueField]
[FIXED-TYPE-VALUE-SET-FIELD             &variableTypeValueField]
[FIXED-TYPE-VALUE-SET-FIELD             &FixedTypeValueSetField]
[VARIABLE-TYPE-VALUE-SET-FIELD          &VariableTypeValueSetField]
[OBJECT-FIELD                           &objectField]
[OBJECT-SET-FIELD                       &ObjectSetField]
}
SIMPLE-CLASS : : = CLASS {
&value INTEGER
}
WITH SYNTAX {
&value
}
- - Этот тип содержит компоненты, которые специфицированы с
- - использованием нотации «ObjectClassFieldType».
- - В случае полей типа, полей значений и множеств значений
- - переменного-типа результирующий тип компонента является
- - открытым типом. В случае полей значения и множества значений
- - фиксированного-типа результирующим типом компонента является INTEGER.
- - Примечание:
- - Ограничения опущены из всех последующих
- - использований «ObjectClassFieldType»; обычно можно использовать
- - ограничения при ссылке на «ObjectClassFieldType».

ExampleType : : = SEQUENCE {
openTypeComponent1      EXAMPLE-CLASS.&TypeField,
integerComponent1       EXAMPLE-CLASS.&fixedTypeValueField,
openTypeComponent2      EXAMPLE-CLASS.&variableTypeValueField,
integerComponent2       EXAMPLE-CLASS.&FixedTypeValueSetField,
openTypeComponent3      EXAMPLE-CLASS.&VariableTypeValueSetField
}
exampleValue ExampleType : : = {
openTypeComponent1      BOOLEAN : TRUE,
integerComponent1       123,
openTypeComponent2      IA5String: «abcdef»,
integerComponent2       456,
openTypeComponent3      BIT STRING : '0101010101' B
```

D.3 Пример использования объектов и множества объектов

Пример использования класса объектов, определенного в D.2:

```
objectA EXAMPLE-CLASS : : = {
FIXED-TYPE-VALUE-FIELD              123
FIXED-TYPE-VALUE-SET-FIELD          {1|2|3}
OBJECT-FIELD                        {1}
OBJECT-SET-FIELD                    {{2} | {3}}
}
```

```
objectB EXAMPLE-CLASS : : = {
TYPE- FIELD                              IA5String
FIXED-TYPE-VALUE-FIELD                   456
VARIABLE-TYPE-VALUE-FIELD                «abc»
VARIABLE-TYPE-VALUE-SET-FIELD            {«d» | «e» | «f»}
}
- - Следующее множество объектов содержит два определяемых
- - объекта и один встроенный.
ObjectSet EXAMPLE-CLASS : : = {
    objectA |
    objectB |
    {
    TYPE-FIELD                           INTEGER
    FIXED-TYPE-VALUE-FIELD               789
    VARIABLE-TYPE-VALUE-SET-FIELD        {4 | 5 | 6}
    }
}
- - Следующие определения извлекают информацию из объектов и
- - множества объектов.
integerValue INTEGER : : = objectA.&fixedTypeValueField
stringValue IA5String : : = objectB.&variableTypeValueField
IntegerValueSetFromObjectA INTEGER : : = {objectA.&FixedTypeValueSetField}
StringValueSet IA5String : : = {objectB.&VariableTypeValueSetField}
StringType : : = objectB.&TypeField
objectFromObjectA SIMPLE-CLASS : : = objectA.&objectField
ObjectSetFromObjectA SIMPLE-CLASS : : = {objectA.&ObjectSetField}
SetOfValuesInObjectSet INTEGER : : = {ObjectSet.&fixedTypeValueField}
SetOfValueSetsInObjectSet INTEGER : : = {ObjectSet.&FixedTypeValueSetField}
SetOfObjectsInObjectSet SIMPLE-CLASS : : = {ObjectSet.&objectField}
SetOfObjectSetsInObjectSet SIMPLE-CLASS : : = {ObjectSeL&ObjectSetField}
```

ПРИЛОЖЕНИЕ Е
(справочное)

Руководство по модели АСН. 1 расширения множества объектов

Спецификация АСН. 1 может определять множества информационных объектов, и эти множества объектов с помощью маркера расширения могут быть помечены как расширяемые. Использование маркера расширения с множествами объектов отличается от использования с типами тем, что часто этот маркер указывает на требование к приложению динамически добавлять объекты в множество или удалять из него. Табличные ограничения и ограничения связи компонентов, которые при этом не удовлетворяются, сами по себе не рассматриваются как ошибки, если множество объектов является расширяемым. В таких случаях не является ошибкой, если значение указанного типа не найдено в множестве объектов, но если оно найдено, то должно удовлетворяться ограничение для ссылающегося типа.

ПРИЛОЖЕНИЕ F
(справочное)

Сводка нотации

Следующие элементы определены в разделе 7:
objectclassreference
objectreference
objectsetreference
typefieldreference
valuefieldreference
valuesetfieldreference
objectsetfieldreference
word
CLASS
INSTANCE
SYNTAX
UNIQUE
Следующие элементы определены в ГОСТ Р ИСО/МЭК 8824-1 и используются в настоящем стандарте:
empty
modulereference
«: : =»
«{»
«}»
«,»
«.»
«|»
«|»
«:»
DEFAULT
OF
OPTIONAL
WITH
Следующие продукции определены в ГОСТ Р ИСО/МЭК 8824-1 и используются в настоящем стандарте:
ElementSetSpec
Type
Value
ValueSet
Следующие продукции определены в ИСО/МЭК 8824-4 и используются в настоящем стандарте:
ParameterizedObjectClass
ParameterizedObjectSet
ParameterizedObject
Следующие продукции определены в настоящем стандарте:

```
DefinedObjectClass : : =
        ExternalObjectClassReference | objectclassreference |
        UsefulObjectClassReference
ExternalObjectClassReference : : = modulereference «.»
    objectclassreference
UsefulObjectClassReference : : =
    TYPE-IDENTIFIER |
    ABSTRACT-SYNTAX
ObjectClassAssignment : : = objectclassreference «: : = » ObjectClass
ObjectClass : : = DefinedObjectClass | ObjectClassDefn |
    ParameterizedObjectClass
ObjectClassDefn : : = CLASS «{» FieldSpec «,» + «}» WithSyntaxSpec?
FieldSpec : : =
        TypeFieldSpec |
        FixedTypeValueFieldSpec |
        VariableTypeValueFieldSpec |
        FixedTypeValueSetFieldSpec |
```

```
            VariableTypeValueSetFieldSpec |
            ObjectFieldSpec |
            ObjectSetFieldSpec
    PrimiitiveFieldName : : =
            typefieldreference |
            valuefieldreference |
            valuesetfieldreference |
            objectfieldreference |
            objectsetfieldreference
    FieldName : : = PrimitiveFieldName «.» +
    TypeFieldSpec : : = typefieldreference TypeOptionalitySpec?
    TypeOptionalitySpec : : = OPTIONAL | DEFAULT Type
    FixedTypeValueFieldSpec : : = valuefieldreference UNIQUE? ValueOptionalitySpec?
    ValueOptionalitySpec : : = OPTIONAL | DEFAULT Value
    VariableTypeValueFieldSpec : : = valuefieldreference FieldNameValueOptionalitySpec?
    FixedTypeValueSetFieldSpec : : = valuesetfieldreference TypeValueSetOptionalitySpec?
    ValueSetOptionalitySpec : : = OPTIONAL | DEFAULT ValueSet
    VariableTypeValueSetFieldSpec : : = valuesetfieldreference FieldNameValueSetOptionalitySpec?
    ObjectFieldSpec : : = objectfieldreference DefinedObjectClassObjectOptionalitySpec?
    ObjectOptionalitySpec : : = OPTIONAL | DEFAULT Object
    ObjectSetFieldSpec : : = objectsetfieldreference DefinedObjectClassObjectSetOptionalitySpec?
    ObjectSetOptionalitySpec : : = OPTIONAL | DEFAULT ObjectSet
    WithSyntaxSpec : : = WITH SYNTAX SyntaxList
    SyntaxList : : = «{» TokenOrGroupSpec empty * «}»
    TokenOrGroupSpec : : = RequiredToken | OptionalGroup
    OptionalGroup : : = «[» TokenOrGroupSpec empty + «]»
    RequiredToken : : = Literal | PrimitiveFieldName
    Literal : : = word | «,»
    DefinedObject : : = ExternalObjectReference | objectreference
    ExternalObjectReference : : = modulereference «.» objectreference
    ObjectAssignment : : = objectreference DefinedObjectClass «: : = » Object
    Object : : = DefinedObject | ObjectDefn | ObjectFromObject | ParameterizedObject
    ObjectDefn : : = DefaultSyntax | DefinedSyntax
    DefaultSyntax : : = «{» FieldSetting «,» * «}»
    FieldSetting : : = PrimitiveFieldName Setting
    DefinedSyntax : : = «{» DefinedSyntaxToken empty * «}»
    DefinedSyntaxToken : : = Literal | Setting
    Setting : : = Type | Value | ValueSet | Object | ObjectSet
    DefinedObjectSet : : = ExternalObjectSetReference | objectsetreference
    ExternalObjectSetReference : : = modulereference «.» objectsetreference
    ObjectSetAssignment : : = objectsetreference DefinedObjectClass «: : =» ObjectSet
    ObjectSet : : = «{» ObjectSetSpec «}»
    ObjectSetSpec : : =
        RootElementSetSpec |
        RootElementSetSpec «,» «. . .» |
        «. . .» |
        «. . .» «,» AdditionalElementSetSpec |
        RootElementSetSpec «,» «. . .» «,» AdditionalElementSetSpec
    ObjectSetElements : : =
        Object | DefinedObjectSet | ObjectSetFromObjects |
        ParameterizedObjectSet
    ObjectClassFieldType : : = DefinedObjectClass «.» FieldName
    ObjectClassFieldValue : : = OpenTypeFieldVal | FixedTypeFieldVal
    OpenTypeFieldVal : : = Type «:» Value
    FixedTypeFieldVal : : = Value
    InformationFromObjects : : = ValueFromObject | ValueSetFromObjects | TypeFromObject | ObjectFromObject |
        ObjectSetFromObjects
    ReferencedObjects : : =
        DefinedObject | ParameterizedObject |
        DefinedObjectSet | ParameterizedObjectSet
```

ValueFromObject : : = ReferencedObjects «.» FieldName
ValueSetFromObjects : : = ReferencedObjects «.» FieldName
TypeFromObject : : = ReferencedObjects «.» FieldName
ObjectFromObject : : = ReferencedObjects «.» FieldName
ObjectSetFromObjects : : = ReferencedObjects «.» FieldName
InstanceOfType : : = INSTANCE OF DefinedObjectClass
InstanceOfValue : : = Value

УДК 681.324:006.354 ОКС 35.100.60 П85 ОКСТУ 4002

Ключевые слова: информационная технология, обработка данных, информационный обмен, взаимосвязь открытых систем, уровень представления, спецификации, абстрактная синтаксическая нотация, информационный объект

Редактор *В.П. Огурцов*
Технический редактор *Н.С. Гришанова*
Корректор *Р.А. Ментова*
Компьютерная верстка *Л.А. Круговой*

Изд. лиц. № 02354 от 14.07.2000. Сдано в набор 17.09.2001. Подписано в печать 31.10.2001. Усл. печ. л. 3,72.
Уч.-изд. л. 3,40. Тираж 400 экз. С 2442. Зак. 1032.

ИПК Издательство стандартов, 107076, Москва, Колодезный пер., 14.
http://www.standards.ru e-mail: info@standards.ru
Набрано в Издательстве на ПЭВМ
Филиал ИПК Издательство стандартов — тип. "Московский печатник", 103062, Москва, Лялин пер., 6.
Плр № 080102