ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

АБСТРАКТНАЯ СИНТАКСИЧЕСКАЯ НОТАЦИЯ ВЕРСИИ ОДИН (АСН.1)

Часть 4

Параметризация спецификации АСН.1

Издание официальное

БЗ 9—2000/263

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 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Постановлением Госстандарта России от 21 января 2003 г. № 19-ст

3 Настоящий стандарт содержит полный аутентичный текст международного стандарта ИСО/МЭК 8824-4—95 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Параметризация спецификации АСН.1», ИСО/МЭК 8824-4—98/Доп. 1—2000 «АСН.1 — семантическая модель»

4 ВВЕДЕН ВПЕРВЫЕ

© ИПК Издательство стандартов, 2003

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 . . . 1
3 Определения . . . 1
3.1 Спецификация базовой нотации . . . 1
3.2 Спецификация информационного объекта . . . 1
3.3 Спецификация ограничения . . . 1
3.4. Дополнительные определения . . . 1
4 Сокращения . . . 2
5 Соглашение . . . 2
6 Нотация . . . 2
6.1 Присваивания . . . 2
6.2 Параметризованные определения . . . 2
6.3 Символы . . . 3
7 Элементы АСН.1 . . . 3
8 Параметризованные присвоения . . . 3
9 Указания параметризованных определений . . . 6
10 Параметры абстрактного синтаксиса . . . 8
Приложение А Примеры . . . 9
Приложение В Сводка нотаций . . . 13

ГОСУДАРСТВЕННЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

АБСТРАКТНАЯ СИНТАКСИЧЕСКАЯ НОТАЦИЯ ВЕРСИИ ОДИН (АСН.1)

Часть 4

Параметризация спецификации АСН.1

Information technology. Abstract Syntax Notation One (ASN.1).
Part 4. Parameterization of ASN.1 specifications

Дата введения 2004—01—01

1 Область применения

Настоящий стандарт является частью абстрактной синтаксической нотации версии 1 (АСН.1) и определяет нотацию для параметризации спецификации АСН.1.

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 (1994)]

ГОСТ Р ИСО/МЭК 8824-2—2001 Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 2. Спецификация информационного объекта [Рекомендация МККТТ X.681 (1994)]

ГОСТ Р ИСО/МЭК 8824-3—2002. Информационная технология. Абстрактно-синтаксическая нотация версии один (АСН.1). Часть 3. Спецификация ограничения [Рекомендация МККТТ X.682 (1994)]

3 Определения

В настоящем стандарте применяют следующие термины:

3.1 **Спецификация базовой нотации**

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

3.2 **Спецификация информационного объекта**

В настоящем стандарте используют термины, определенные в ГОСТ Р ИСО/МЭК 8824-2.

3.3 **Спецификация ограничения**

В настоящем стандарте используют термины, определенные в ГОСТ Р ИСО/МЭК 8824-3.

3.4 **Дополнительные определения**

3.4.1 **стандартное имя ссылки:** Имя ссылки, определенное без параметров, посредством другого «Assignment» нежели «ParameterizedAssignment». Такое имя указывает полное определение и не обеспечивается фактическими параметрами при использовании.

3.4.2 **параметризованное имя ссылки:** Имя ссылки, определенное с помощью параметризованного присваивания, которое указывает на неполное определение и поэтому должно быть обеспечено фактическими параметрами при использовании.

3.4.3 **параметризованный тип:** Тип, определенный с помощью присваивания параметризованного типа и, таким образом, компоненты которого являются неполными определениями, которые должны быть обеспечены фактическими параметрами при использовании типа.

Издание официальное

3.4.4 **параметризованное значение:** Значение, определенное с помощью присваивания параметризованного значения и, таким образом, не полностью специфицированное, которое должно быть обеспечено фактическими параметрами при использовании.

3.4.5 **параметризованное множество значений:** Множество значений, определенное с помощью присваивания параметризованного множества значений и, таким образом, не полностью специфицированное, которое должно быть обеспечено фактическими параметрами при использовании.

3.4.6 **параметризованный класс объектов:** Класс информационных объектов, определенный с помощью присваивания параметризованного класса объектов, таким образом, спецификации его полей заданы не полностью и должны быть обеспечены фактическими параметрами при использовании.

3.4.7 **параметризованный объект:** Информационный объект, определенный с помощью присваивания параметризованного объекта и, таким образом, его компоненты заданы не полностью и должны быть обеспечены фактическими параметрами при использовании.

3.4.8 **параметризованное множество объектов:** Множество информационных объектов, определенное с помощью присваивания множества параметризованных объектов и, таким образом, его объекты заданы не полностью и должны быть обеспечены фактическими параметрами при использовании.

3.4.9 **переменное ограничение:** Ограничение, применяемое в спецификации параметризованного абстрактного синтаксиса и зависящее от некоторого параметра абстрактного синтаксиса.

4 Сокращения

В настоящем стандарте использовано следующее сокращение:

АСН.1 — абстрактная синтаксическая нотация версии 1.

5 Соглашение

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

6 Нотация

В данном разделе приведена сводка нотации, определенной в настоящем стандарте.

6.1 Присваивания

В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для присваивания «Assignment» (см. ГОСТ Р ИСО/МЭК 8824-1, раздел 12):

- ParametrizedAssignment (см. 8.1).

6.2 Параметризованные определения

6.2.1 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для определяемого типа «DefinedType» (см. ГОСТ Р ИСО/МЭК 8824-1, пункт 13.1):

- ParametrizedType (см. 9.2).

6.2.2 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для определяемого значения «DefinedValue» (см. ГОСТ Р ИСО/МЭК 8824-1, пункт 13.1):

- ParametrizedValue (см. 9.2).

6.2.3 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для определяемого типа «DefinedType» (см. ГОСТ Р ИСО/МЭК 8824-1, пункт 13.1):

- ParametrizedValueSetType (см. 9.2).

6.2.4 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для класса объектов «ObjectClass» (см. ГОСТ Р ИСО/МЭК 8824-2, пункт 9.2):

- ParametrizedObjectClass (см. 9.2).

6.2.5 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для объекта «Object» (см. ГОСТ Р ИСО/МЭК 8824-2, пункт 11.2):

- ParametrizedObject (см. 9.2).

6.2.6 В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для множества объектов «ObjectSet» (см. ГОСТ Р ИСО/МЭК 8824-2, пункт 12.2):

- ParametrizedObjectSet (см. 9.2).

6.3 Символы

В настоящем стандарте определена следующая нотация, которая может использоваться как альтернатива для символа «Symbol» (см. ГОСТ Р ИСО/МЭК 8824-1, пункт 12.1):

- ParametrizedReference (см. 9.1).

7 Элементы АСН.1

В настоящем стандарте используют элементы АСН.1, определенные в ИСО/МЭК 8824-1, раздел 11.

8 Параметризованные присвоения

8.1 Существуют операторы параметризованного присвоения, соответствующие каждому из операторов присвоения, определенному в ГОСТ Р ИСО/МЭК 8824-1 и ГОСТ Р ИСО/МЭК 8824-2. Конструкция «ParameterizedAssignment» есть:

```
ParameterizedAssignment : : =
   ParameterizedTypeAssignment          |
   ParameterizedValueAssignment         |
   ParameterizedValueSetTypeAssignment  |
   ParameterizedObjectClassAssignment   |
   ParameterizedObjectAssignment        |
   ParameterizedObjectSetAssignment     |
```

8.2 Каждая конструкция «Parameterized<X>Assignment» имеет тот же самый синтаксис, что и «<X>Assignment», за исключением того, что имеется начальный элемент «ParameterList». Таким образом, начальный элемент становится параметризованным именем ссылки (см. 3.4.2).

Примечание — ГОСТ Р ИСО/МЭК 8824-1 налагает требование, заключающееся в том, что все имена ссылки, назначенные в пределах модуля, параметризованные или нет, должны различаться.

```
ParameterizedTypeAssignment : : =
        typereference
        ParameterList
        ": : = "
        Type

ParameterizedValueAssignment : : =
        valuereference
        ParameterList
        Type
         ": : = "
        Value

ParameterizedValueSetTypeAssignment : : =
        typereference
        ParameterList
        Type
        ": : ="
        ValueSet

ParameterizedObjectClassAssignment : : =
        objectclassreference
        ParameterList
        ": : ="
        ObjectClass
```

ParameterizedObjectAssignment : : =
objectreference
ParameterList
DefinedObjectClass
": : ="
ObjectClass

ParameterizedObjectSetAssignment : : =
objectsetreference
ParameterList
DefinedObjectClass
": : ="
ObjectSet

8.3 Конструкция "ParameterList" есть список параметров "Parameter", заключенных в фигурные скобки.

ParameterList : : = "{" Parameter "," + "}"

Каждый параметр "Parameter" состоит из пустой ссылки "DummyReference" и, возможно, из параметра управляющего слова "ParamGovernor".

Parameter : : = ParamGovernor ":" DummyReference | DummyReference
ParamGovernor : : = Governor | DummyGovernor
Governor : : = Type | DefinedObjectClass
DummyGovernor : : = DummyReference
DummyReference : : = Reference

"DummyReference" в "Parameter" может замещаться:

а) "Type" или "DefinedObjectClass", в том случае, когда не должно быть "ParamGovernor";

б) "Value" или "ValueSet"; в этом случае должен присутствовать "ParamGovernor"; когда "ParamGovernor" есть "Governor", он должен быть "Type", когда "ParamGovernor" есть "DummyGovernor", фактическим параметром для "ParamGovernor" должен быть "Type";

в) "Object" или "ObjectSet"; в этом случае должен присутствовать "ParamGovernor"; если "ParamGovernor" есть "Governor", это должен быть "DefinedObjectClass", если "ParamGovernor" есть "DummyGovernor", то фактическим параметром для "ParamGovernor" должен быть "DefinedObjectClass";

"DummyGovernor" должен быть "DummyReference", который не имеет "Governor".

8.4 Областью действия "DummyReference", появляющейся в конструкции "ParameterList", является сам "ParameterList" вместе с той частью "ParameterizedAssignment", которая следует за ": : =". "DummyReference" скрывает любую другую ссылку "Reference" с таким же именем в этой области действия.

8.5 Использование пустой ссылки "DummyReference" в ее области действия должно быть согласовано с ее синтаксической формой, и (там, где применимо) с управляющим параметром, а все использования той же самой "DummyReference" должны быть согласованы друг с другом.

Примечание — Когда синтаксическая форма имени пустой ссылки двусмысленна (например, не ясно, используется "objectclassreference" или "typereference"), неоднозначность может быть разрешена при первом использовании имени пустой ссылки справа от оператора присваивания. После этого характер имени пустой ссылки становится известным. Однако характер пустой ссылки не определяется полностью по правой стороне оператора присваивания, когда он, в свою очередь, используется только как фактический параметр в параметризованной ссылке; в этом случае характер пустой ссылки должен быть определен при рассмотрении определения этой параметризованной ссылки. Пользователи нотации должны учитывать, что такая практика может сделать спецификации ASN.1 менее понятными, поэтому рекомендуется предусматривать соответствующие комментарии для пояснений.

Пример

Рассмотрим следующее присваивание параметризованного класса объектов:

```
PARAMETERIZED-OBJECT-CLASS {TypeParam, INTEGER: valueParam,
                                        INTEGER: VahieSetParam} : : =
  CLASS {
        &valueField1     TypeParam,
        &valueField2     INTEGER DEFAULT valueParam,
        &valueField3     INTEGER (ValueSetParam),
        &valueSetField   INTEGER DEFAULT {ValueSetParam}
  }
```

Для определения правильного использования пустых ссылок "DummyReference" в контексте "ParameterizedAssignment" и, только для той цели, могут быть рассмотрены "DummyReference" для того, чтобы быть определенными следующим образом:

```
TypeParam : : = UnspecifieldType
valueParam INTEGER : : = unspecifieldIntegerValue
ValueSetParam INTEGER : : = {UnspecifieldInteger ValueSet}
```

где:

а) TypeParam есть пустая ссылка "DummyReference", которая замещает "Type". Поэтому TypeParam может быть использован везде, где можно использовать "typereference", например как "Type" для значения фиксированного типа поля valueField1.

б) ValueParam есть пустая ссылка "DummyReference", которая замещает значение целочисленного типа. Следовательно, valueParam можно использовать везде, где можно использовать "valuereference" для целочисленного значения, например как значение по умолчанию для значения фиксированного-типа поля valueField2.

в) ValueSetParam есть пустая ссылка "DummyReference", которая замещает множество значений целочисленного типа. Следовательно, ValueSetParam можно использовать везде, где можно использовать "typereference" для целочисленного значения, например как "Type" в нотации "ContainedSubtype" для valueField3 и ValueSetField.

8.6 Каждая пустая ссылка "DummyReference" должна использоваться по крайней мере один раз в пределах своей области действия.

Примечание — Если пустая ссылка "DummyReference" так и не появилась, то соответствующий "ActualParameter" не влияет на определение, мог бы быть просто «отброшен», хотя пользователю могло бы казаться, что имеет место некая спецификация.

Присваивания "ParameterizedValueAssignment", "ParameterizedValueSetTypeAssignment", "ParameterizedObjectAssignment" и "ParameterizedObjectSetAssignment", прямо или косвенно содержащие ссылку на себя, недействительны.

8.7 В определении "ParameterizedType", "ParameterizedValueSet" или "ParameterizedObjectClass" пустая ссылка "DummyReference" не должна передаваться как тегированный тип (как фактический параметр) рекурсивной ссылке на этот "ParameterizedType", "ParameterizedValueSet" или "ParameterizedObjectClass" (см. А.3).

8.8 В определении "ParameterizedType", "ParameterizedValueSet" или "ParameterizedObjectClass" не должно быть циклической ссылки на определяемый элемент, если только такая ссылка прямо или косвенно не помечена как OPTIONAL или, в случае "ParameterizedType" и "ParameterizedValueSet", дана путем ссылки на выборочный тип, по крайней мере одна из альтернатив которого является нециклической в определении.

8.9 Управляющий пустой ссылки "DummyReference" не должен включать в себя ссылку на другую "DummyReference", если эта другая "DummyReference" также имеет управляющего.

8.10 В параметризованном присваивании правая сторона ": : = " не должна состоять исключительно из "DummyReference".

8.11 Управляющий "DummyReference" не должен требовать знания "DummyReference" или определяемого параметризованного имени ссылки.

8.12 Когда в параметризованный тип в качестве фактического параметра подставляется значение или множество значений, то требуется, чтобы тип фактического параметра был совместим с управляющим соответствующего пустого параметра. (См. ГОСТ Р ИСО/МЭК 8824-1, F.6.2, F.6.3).

8.13 При определении параметризованного типа с пустым параметром вместо значения или множества значений тип, используемый для управления этим пустым параметром, должен быть таким, что все его значения допустимы для использования во всех правых частях присваиваний, где есть пустой параметр. (См. ГОСТ Р ИСО/МЭК 8824-1, F.6.5).

9 Указания параметризованных определений

9.1 В перечне "SymbolList" (в "Export" или "Import") параметризованное определение должно быть указано с помощью "ParameterizedReference":

ParameterizedReference : : = Reference | Reference "{" "}",

где "Reference" — первый элемент в "ParameterizedAssignment", как определено в 8.2.

Примечание — Первая альтернатива для "ParameterizedReference" предусмотрена исключительно для облегчения понимания. Обе альтернативы имеют один и тот же смысл.

9.2 Вне "Export" или "Import" параметризованное определение должно быть указано конструкцией "Parameterized<X>", которая может использоваться как альтернатива для соответствующего "<X>".

```
ParameterizedType : : =
    SimpleDefinedType
    ActualParameterList

SimpleDefinedType : : =
    Externaltypereference |
    typereference

ParameterizedValue : : =
    SimpleDefinedValue
    ActualParameterList

SimpleDefinedValue : : =
    Externlvaluereference |
    valuereference

ParameterizedValueSetType : : =
    SimpleDefinedType
    ActualParameterList

ParameterizedObjectClass : : =
    DefinedObjectClass
    ActualParameterList

ParameterizedObjectSet : : =
    DefinedObjectSet
    ActualParameterList

ParameterizedObject : : =
    DefinedObject
    ActualParameterList
```

9.3 Имя ссылки в "Defind<X>" должно быть именем ссылки, для которого сделано присваивание в "ParameterizedAssignment".

9.4 Для используемой альтернативы "Defined<X>" ограничения, определенные в ГОСТ Р ИСО/МЭК 8824-1 и ГОСТ Р ИСО/МЭК 8824-2 для обычных имен ссылок, должны использоваться и для соответствующих параметризованных имен ссылок.

Примечание — По существу, ограничения следующие: каждый "Defined<X>" имеет две альтернативы — "<x>reference" и "External<x>Reference". Первая используется в модуле определения или когда определение было импортировано и нет противоречия имени; вторая используется тогда, когда нет перечисленного импорта (не рекомендуется) или если есть конфликт между импортированным именем и локальным определением (также не рекомендуется), или есть конфликт между импортированными именами.

9.5 Список фактических параметров "ActualParameterList" есть:

```
ActualParameterList : : =
   "{" ActualParameter "," + "}"

ActualParameter : : =
   Type                 |
   Value                |
   ValueSet             |
   DefinedObjectClass   |
   Object               |
   ObjectSet
```

9.6 Должен быть ровно один параметр "ActualParameter" для каждого "Parameter" в соответствующем "ParameterizedAssignment", и они должны появляться в том же самом порядке. Конкретный выбор "ActualParameter" и управляющего (если он есть) должен определяться синтаксической формой "Parameter" и контекстов, в котором он встречается в "ParameterizedAssignment". "ActualParameter" должен иметь форму, необходимую для замены "DummyReference" в области ее действия (см. 8.4).

Пример

Параметризованный класс объектов предыдущего примера (см. 8.5) может быть указан следующим образом:

MY-OBJECT-CLASS : : = PARAMETERIZED-OBJECT-CLASS {BIT STRING, 123, {4|5|6}}

9.7 Фактический параметр занимает место имени пустой ссылки в определении фактического типа, значения, множества значений, класса объектов, объекта или множества объектов, которые указываются данным экземпляром использования параметризованного имени ссылки.

9.8 Смысл любых ссылок, которые появляются в "ActualParameter", и умалчиваемый тег, применяемый к любым появляющимся тегам, определяются в соответствии со средой тегирования "ActualParameter", а не "DummyReference".

Примечание — Таким образом, параметризация, подобно ссылкам, селективным типам и "COMPONENTS OF", не является точной текстуальной заменой.

Пример

Рассмотрим следующие модули:

```
M1 DEFINITIONS AUTOMATIC TAGS : : = BEGIN
    EXPORTS T1;

    T1 : : = SET {
     f1     INTEGER,
     f2     BOOLEAN
    }
END

M2 DEFINITIONS AUTOMATIC TAGS : : = BEGIN
    IMPORTS T1 FROM M1;

    T3 : : = T2{T1}
    T2{X} : : = SEQUENCE {
```

```
    a    INTEGER,
    b    X
   }
END
```

Применение 9.8 означает, что тег для компонента f1 из T3 (то есть @T3.b.f1) будет тегирован неявно, так как среда тегирования пустого параметра X, а именно — явное тегирование, не влияет на тегирование компонентов фактического параметра T1.

Рассмотрим модуль M3.

```
M3 DEFINITIONS AUTOMATIC TAGS : : = BEGIN
   IMPORTS T1 FROM M1;
   T5 : : = T4{T1}
   T4{Y} : : = SEQUENCE {
    a    INTEGER,
    b    Y
   }
END
```

Применение ГОСТ Р ИСО/МЭК 8824-1, пункт 30.6, означает, что тег для компонента b в T5 (то есть @T5.b) будет тегирован явно, так как пустой параметр (Y) всегда тегирован явно, следовательно, @T5 эквивалентно

```
T5 : : = SEQUENCE {
          a [0] IMPLICIT INTEGER,
          b [1] EXPLICIT SET {
            f1   [0] INTEGER,
            f2   [1] BOOLEAN
          }
}
```

тогда как @T3 эквивалентно

```
T3 : : = SEQUENCE {
          a INTEGER,
          b SET {
            f1 [0] IMPLICIT INTEGER,
            f2 [1] IMPLICIT BOOLEAN
          }
}
```

10 Параметры абстрактного синтаксиса

10.1 Приложение B ГОСТ Р ИСО/МЭК 8824-2 устанавливает класс информационного объекта ABSTRACT-SYNTAX и рекомендует использовать его для определения абстрактных синтаксисов, применяя в качестве примера абстрактный синтаксис, определенный как множество значений единственного типа ACH.1, который не был параметризован на внешнем уровне.

10.2 Когда тип ACH.1, используемый для определения абстрактного синтаксиса, является параметризованным, некоторые параметры могут быть подставлены как фактические параметры, а другие — оставлены как параметры абстрактного синтаксиса.

Пример

Если параметризованный тип был определен вызываемым YYY-PDU с двумя пустыми ссылками (например, первая — набор объектов некоторого заданного класса объектов, а вторая — целочисленное граничное значение). то:

```
yyy-Abstract-Syntax {INTEGER:bound} ABSTRACT-SYNTAX : : =
      {YYY-PDU {{ValidObjects}, bound} IDENTIFIED BY {yyy 5}}
```

определяет параметризованный абстрактный синтаксис, в котором множество объектов было разрешено, а «граничное значение» оставлено в качестве параметра абстрактного синтаксиса.

Параметр абстрактного синтаксиса должен использоваться:

а) прямо или косвенно в контексте ограничения;

б) прямо или косвенно как фактические параметры, которые, в конечном счете, используются в контексте ограничения.

Примечание — См. пример в А.2 и ИСО/МЭК 8824-1, пункт D.5.

10.3 Ограничение, множество значений которого зависит от одного или более параметров абстрактного синтаксиса, является переменным. Такие ограничения определяются после определения абстрактного синтаксиса (возможно профилем международного функционального стандарта или в заявке о соответствии реализации протоколу).

Примечание — Если где-нибудь в цепочке определений, включаемой в спецификацию значений ограничения, появляется параметр абстрактного синтаксиса, то ограничение является переменным. Оно является переменным ограничением, даже если множество значений результирующего ограничения на зависит от фактического значения параметра абстрактного синтаксиса.

Пример

Значение (((1..3) EXCEPT a) UNION (1..3)) всегда 1..3 независимо от того, каково значение «а», тем не менее это все является переменным ограничением, если «а» является параметром абстрактного синтаксиса.

10.4 Формально переменное ограничение не влияет на множество значений в абстрактном синтаксисе.

Примечание — Настоятельно рекомендуется, чтобы ограничения, которые, как ожидается, останутся в абстрактном синтаксисе переменными, имели спецификацию исключений, использующую нотацию ГОСТ ИСО/МЭК 8824-1, пункт 45.4.

ПРИЛОЖЕНИЕ А
(справочное)

Примеры

А.1 Примеры использования определения параметризованного типа

Предположим, что разработчику протокола нужно часто передавать аутентификатор с одним или более полями протокола. Он будет передаваться как BIT STRING рядом с полем. Без параметризации аутентификатор должен бы быть определен как BIT STRING, а затем "authenticator" с текстом, идентифицирующим, к чему он прилагался, должен добавляться при каждом появлении. Альтернативно разработчик может предпочесть преобразование каждого поля, имеющего аутентификатор, в последовательность SEQUENCE этого поля и "authenticator"'а. Метод параметризации обеспечивает удобную краткую запись для решения этой задачи.

Сначала определяют параметризованный тип SIGNED{}:

```
SIGNED {ToBeSigned} : : = SEQUENCE
{
    authenticated-data ToBeSigned,
    authenticator      BIT STRING
}
```

тогда в теле протокола нотация (например)

```
SIGNED {OrderInformation}
```

есть нотация типа, установленная для

```
SEQUENCE
{
    authenticated-data OrderInformation,
    authenticator      BIT STRING
}
```

Далее предположим, что для некоторых полей отправитель должен иметь возможность добавить (или не

добавить) аутентификатор. Этого можно достичь, сделав BIT STRING факультативной, но более изящное решение (меньшее количество битов в строке) состоит в том, чтобы определить другой параметризованный тип:

```
OPTIONALLY-SIGNED {ToBeSigned} : : = CHOICE
{
  usigned-data    [0] ToBeSigned,
  signed-data     [1] SIGNED {ToBeSigned}
}
```

Примечание — Тегирование в CHOICE не является необходимым, если разработчик гарантирует, что ни одно из использований параметризованного типа не порождает фактический параметр, который является BIT STRING (тип SIGNED), но полезен для предотвращения ошибок в других частях спецификации.

А.2 Пример использования параметризованных определений вместе с классом информационных объектов

Используют классы информационных объектов для сбора всех параметров абстрактного синтаксиса. Таким образом, число параметров абстрактного синтаксиса может быть сокращено до одного, который является экземпляром совокупности классов. Продукция "InformationFromObject" может быть использована для извлечения информации от параметра объекта.

Пример

```
- - Экземпляр этого класса содержит все параметры для
- - абстрактного синтаксиса Message-PDU.
MESSAGE-PARAMETERS : : = CLASS {
    &maximum-priority-level          INTEGER,
    &maximum-message-buffer-size     INTEGER,
    &maximum-reference-buffer-size   INTEGER

}
WITH SYNTAX {
  THE MAXIMUM PRIORITY LEVEL IS              &maximum-priority-level
  THE MAXIMUM MESSAGE BUFFER SIZE IS         &maximum-message-buffer-size
  THE MAXIMUM REFERENCE BUFFER SIZE IS       &maximum-reference-buffer-size
}

- - Продукция "ValueFromObject" используется для извлечения
- - значения от абстрактного параметра синтаксиса, "param".
- - Значения могут быть использованы только в ограничениях.
- - Кроме того, параметр передается посредством другого
- - параметризованного типа.

Message-PDU {MESSAGE-PARAMETERS:param} : : = SEQUENCE {
    priority-level      INTEGER (0..param.&maximum-priority-level),
    message             BMPString (SIZE (0..param.&maximum-message-buffer-size)),
    reference           Reference {param}
}

Reference {MESSAGE-PARAMETERS:param}: : =
  SEQUENCE OF
  IA5String (SIZE (0..param.&maximum-reference-buffer-size))
- - Определение информационного объекта параметризованным
- - абстрактным синтаксисом.
- - Параметр абстрактного синтаксиса используется только в ограничениях.

message-Abstract-Sintax {MESSAGE-PARAMETERS:param}
ABSTRACT-SINTAX : : =
{
  Message-PDU {param}
  IDENTIFIED BY {joint-iso-coitt asnl(l) examples (123) 0}
}
```

Класс MESSAGE-PARAMETERS и объект параметризованного абстрактного синтаксиса message-Abstract-Syntax используются следующим образом:

```
- - Этот экземпляр MESSAGE-PARAMETERS определяет значения
- - параметров абстрактного синтаксиса.
```

```
my-message-parameters MESSAGE-PARAMETERS : : = {
  THE MAXIMUM PRIORITY LEVEL IS 10
  THE MAXIMUM MESSAGE BUFFER SIZE IS 2000
  THE MAXIMUM REFERENCE BUFFER SIZE IS 100
}
- - Абстрактный синтаксис теперь может быть определен всеми
- - специфицированными переменными ограничениями.
my-message-Abstract-Sintax ABSTRACT-SINTAX : : =
    message-Abstract-Sintax {my-message-parameters}
```

А.3 Пример определения параметризованного типа, который является конечным

При спецификации параметризованного типа, который представляет родовой список, определяют тип так, чтобы результирующая нотация АСН.1 была конченной. Например, можно определить:

```
List1 {ElementTypeParam} : : = SEQUENCE {
    elem        ElementTypeParam,
    next        List1 {ElementTypeParam} OPTIONAL
}
```

который является конечным, а затем его использовать.

```
IntegerList1 : : = List1 {INTEGER},
```

где результирующая нотация АСН.1 является такой, какой Вы ее обычно определили бы:

```
IntegerList1 : : =    SEQUENCE {
      elem        INTEGER,
      next        IntegerList1 OPTIONAL
}
```

Напротив, при

```
List2 {ElementTypeParam} : : = SEQUENCE {
      elem        ElementTypeParam,
      next        List2 {[0]} ElementTypeParam} OPTIONAL
}
IntegerList2 : : = List2 {INTEGER},
```

где результирующая нотация АСН.1 является бесконечной:

```
IntegerList2 : : = SEQUENCE {
      elem        INTEGER
      next        SEQUENCE {
                  elem  [0] INTEGER,
                  next  SEQUENCE {
                        elem [0] [0] INTEGER,
                        next SEQUENCE {
                              elem  [0] [0] [0] INTEGER,
                              next   SEQUENCE {
                                    . . . - - и так далее
                              } OPTIONAL
                        } OPTIONAL
                  } OPTIONAL
      } OPTIONAL
}
```

А.4 Пример определения параметризованного значения

Если значение параметризованной строки определяется следующим образом:

```
genericBirthdayGreeting {IA5String: name}
IA5String : : = {"С днем рождения", имя, "||"},
```

то следующие две строки являются такими же:

```
greeting1 IA5String : : = genericBirthdayGreeting {"Джон"}
greeting2 IA5String : : = "С днем рождения, Джон ||"
```

А.5 Пример определения множества параметризованных значений

Если два множества параметризованных значений определены следующим образом:

```
QuestList1{IA5String:extraQuest} IA5String : : = {"Джек" | "Джон" | extraQuest}
QuestList2 {IA5String:ExtraQuests} IA5String : : = {"Джек" | "Джон" | ExtraQuests}
```

то следующие множества значений обозначают одно и то же множество значений:

```
SetOfQuests1 IA5String : : = {QuestList1 {"Джилл"}}
SetOfQuests2 IA5String : : = {QuestList2 | {{"Джилл"}}}
SetOfQuests3 IA5String : : = {"Джек" | "Джон" | "Джилл"}
```

и следующие множества значений обозначают одно и то же множество значений:

```
SetOfQuests4 IA5String : : = {QuestList2 {{"Джилл" | "Мэри"}}}
SetOfQuests5 IA5String : : = {"Джек" | "Джон" | "Джилл" | "Мэри"}
```

Следует обратить внимание, что множество значений всегда задается в фигурных скобках, даже когда это — ссылка на параметризованное множество значений. Опуская фигурные скобки у ссылки на "identifier", который был создан в присвоении множества значений, или у ссылки на "ParameterizedValueSetType", получим нотацию для "Type", а не для множества значений.

А.6 Пример определения параметризованного класса

Следующий параметризованный класс может использоваться для определения классов ошибок, которые содержат коды ошибок различных типов. Следует обратить внимание, что параметр "ErrorCodeType" используется только как "DummyGovernor" для параметра "ValidErrorCodes".

```
GENERIC-ERROR {ErrorCodeType, ErrorCodeType:
                                        ValidErrorCodes} : : = CLASS {
        &errorCode ValidErrorCodes
}
WITH SYNTAX {
    CODE &errorCode
}
```

Определение параметризованного класса может использоваться для определения различных классов, которые совместно используют некоторые характеристики, подобные одному и тому же синтаксису:

```
ERROR-1 : : = GENERIC-ERROR {INTEGER, {1|2|3}}
ERROR-2 : : = GENERIC-ERROR {ErrorCodeString, {StringErrorCodes}}
ERROR-3 : : = GENERIC-ERROR {EnumeratedErrorCode, {fatal|error}}
ErrorCodeString : : = IA5String (SIZE (4))
StringErrorCodes ErrorCodeString : : = {"E001"|"E002"|"E003"}
EnumeratedErrorCode : : = ENUMERATED {fatal, error, warning}
```

Определяемые классы тогда могут использоваться следующим образом:

```
My-Errors ERROR-2 : : = {{CODE "E001"} | {CODE "E002"}}
fatalError, ERROR-3 : : = {CODE fatal}
```

А.7 Пример определения множества параметризованных объектов

Определение множества параметризованных объектов AllTypes формирует множество объектов, которые содержит базовое множество объектов BaseTypes и множество дополнительных объектов, которые поставляются как параметр AdditionalTypes.

```
AllTypes {TYPE-IDENTIFIER: AdditionalTypes} TYPE-IDENTIFIER : : = {
                                BaseTypes | AdditionalTypes}
BaseTypes TYPE-IDENTIFIER : : = {
    {BasicType-1 IDENTIFIER BY basic-type-obj-id-value-1} |
    {BasicType-2 IDENTIFIER BY basic-type-obj-id-value-2} |
    {BasicType-3 IDENTIFIER BY basic-type-obj-id-value-3}
}
```

Определение множества параметризованных объектов, AllTypes, может использоваться следующим образом:

```
{My-All-Types TYPE-IDENTIFIER : : = {AllTypes {
    {My-Type-1 IDENTIFIER BY my-obj-id-value-1} |
```

```
        {My-Type-2 IDENTIFIER BY my-obj-id-value-2} |
        {My-Type-3 IDENTIFIER BY my-obj-id-value-3}
}}
```

А.8 Пример определения множества параметризованных объектов

Тип, определенный в А.4 ГОСТ Р ИСО/МЭК 8824-3, может использоваться в определении параметризованного абстрактного синтаксиса следующим образом:

```
- - PossibleBodyTypes является параметром абстрактного синтаксиса.
message-abstract-syntax {MHS-BODY-CLASS: PossibleBodyTypes}
                        ABSTRACT-SYNTAX : : = {
   INSTANCE OF MHS-BODY-CLASS ({PossibleBodyTypes})
   IDENTIFIED {joint-iso-itu asn1 (1) examples(1) 123}
}
- - Это множество объектов перечисляет все возможные
- - пары значений и идентификаторов-типа для типа "экземпляр-из".
- - Множество объектов используется как фактический параметр
- - определения параметризированного абстрактного синтаксиса.
My-Body-Types MHS-BODY-CLASS : : = {
        {My-First-Type IDENTIFIED BY my-first-obj-id} |
        {My-Second-Type IDENTIFIED BY my-second-obj-id}
}
my-message-abstract-syntax ABSTRACT-SYNTAX : : =
        message-abstract-syntax {{My-Body-Types}}
```

ПРИЛОЖЕНИЕ В
(справочное)

Сводка нотаций

Следующие элементы определены в ГОСТ Р ИСО/МЭК 8824-1 и используются в настоящем стандарте:

typereference
valuereference
" : : = "
"{"
"}"
","

Следующие элементы определены в ГОСТ Р ИСО/МЭК 8824-2 и используются в настоящем стандарте:

objectclassreference
objectreference
objectsetreference

Следующие продукции определены в ГОСТ Р ИСО/МЭК 8824-1 и используются в настоящем стандарте:

DefinedType
DefinedValue
Reference
Type
Value
ValueSet

Следующие продукции определены в ГОСТ Р ИСО/МЭК 8824-2 и используются в настоящем стандарте:

DefinedObjectClass
DefinedObject
DefinedObjectSet
ObjectClass

Object
ObjectSet

Следующие продукции определены в настоящем стандарте:

```
ParametrizedAssignment : : =
        ParametrizedTypeAssignment          |
        ParametrizedValueAssignment         |
        ParametrizedValueSetTypeAssignment  |
        ParametrizedObjectClassAssignment   |
        ParametrizedObjectAssignment        |
        ParametrizedObjectSetAssignment

ParametrizedTypeAssignment : : =
        typereference ParameterList ": : =" Type

ParametrizedValueAssignment : : =
        valuereference ParameterList Type ": : =" Value

ParametrizedValueSetTypeAssignment : : =
        typereference ParameterList Type ": : =" ValueSet

ParametrizedObjectClassAssignment : : =
        objectclassreference ParameterList ": : =" ObjectClass

ParametrizedObjectAssignment : : =
        objectreference ParameterList DefinedObjectClass ": : =" Object

ParametrizedObjectSetAssignment : : =
        objectsetreference ParameterList DefinedObjectClass ": : =" ObjectSet

ParameterList : : = "{" Parameter "," + "}"

Parameter : : = ParamGovenor ":" DummyReference | DummyReference

ParamGovenor : : = Govenor | DummyGovenor

Govenor : : = Type | DefinedObjectClass

DummyGovenor : : = DummyReference

DummyReference : : = Reference

ParametrizedReference : : =
        Reference | Reference "{" "}"

SimpleDefinedType : : = Externaltypereference | typereference

SimpleDefinedValue : : = Externalvaluereference | valuereference

ParametrizedType : : = SimpleDefinedType ActualParameterList

ParametrizedValue : : = SimpleDefinedValue ActualParameterList

ParametrizedValue : : = SimpleDefinedValue ActualParameterList

ParametrizedValueSetType : : = SimpleDefinedType ActualParameterList
```

ParametrizedObjectClass : : = DefinedObjectClass ActualParameterList

ParametrizedObjectSet : : = DefinedObjectSet ActualParameterList

ParametrizedObject : : = DefinedObject ActualParameterList

ActualParameterList : : = "{" ActualParameter "," + "}"

ActualParameter : : = Type | Value | ValueSet | DefinedObjectClass | Object | ObjectSet

УДК 681.324:006.354 ОКС 35.100.70 П85 ОКСТУ 4002

Ключевые слова: информационная технология, взаимосвязь открытых систем, спецификация абстрактной синтактической нотации, нотация АСН.1, протоколы прикладного уровня, параметризация, параметризованный тип, параметризованное значение, параметризованный объект, тег, структурированные типы

Редактор *В.П. Огурцов*
Технический редактор *Н.С. Гришанова*
Корректор *В.С. Черная*
Компьютерная верстка *Е.Н. Мартемьяновой*

Изд. лиц. № 02354 от 14.07.2000. Сдано в набор 03.03.2003. Подписано в печать 19.03.2003. Усл. печ. л. 2,32. Уч.-изд. л. 1,85. Тираж 215 экз. С 9986. Зак. 244.

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», 105062 Москва, Лялин пер., 6.
Плр № 080102

gost.e-ecolog.ru