ФЕДЕРАЛЬНОЕ АГЕНТСТВО
ПО ТЕХНИЧЕСКОМУ РЕГУЛИРОВАНИЮ И МЕТРОЛОГИИ

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

ГОСТ Р ИСО/МЭК 8825-5— 2013

Информационная технология

ПРАВИЛА КОДИРОВАНИЯ АСН.1

Часть 5

Отображение определений W3C схемы XML в АСН.1

ISO/IEC 8825-5:2008
Information technology — ASN.1 encoding rules — Part 5: Mapping W3C XML schema definitions into ASN.1
(IDT)

Издание официальное

Москва
Стандартинформ
2015

gost.e-ecolog.ru

Предисловие

1 ПОДГОТОВЛЕН Федеральным государственным унитарным предприятием «Государственный научно-исследовательский и конструкторско-технологический институт «ТЕСТ» (ФГУП ГосНИИ «ТЕСТ») на основе собственного аутентичного перевода на русский язык международного стандарта, указанного в пункте 4

2 ВНЕСЕН Техническим комитетом по стандартизации ТК 22 «Информационные технологии»

3 УТВЕРЖДЕН И ВВЕДЕН В ДЕЙСТВИЕ Приказом Федерального агентства по техническому регулированию и метрологии от 6 сентября 2013 г. № 876-ст

4 Настоящий стандарт идентичен международному стандарту ИСО/МЭК 8825-5:2008 «Информационная технология. Правила кодирования АСН.1. Часть 5. Отображение определений W3C схемы XML в АСН.1» (ISO/IEC 8825-5:2008 «Information technology — ASN.1 encoding rules — Part 5: Mapping W3C XML schema definitions into ASN.1»).

При применении настоящего стандарта рекомендуется использовать вместо ссылочных международных стандартов соответствующие им национальные стандарты Российской Федерации, сведения о которых приведены в дополнительном приложении ДА.

5 ВВЕДЕН ВПЕРВЫЕ

Правила применения настоящего стандарта установлены в ГОСТ Р 1.0—2012 (раздел 8). Информация об изменениях к настоящему стандарту публикуется в ежегодном (по состоянию на 1 января текущего года) информационном указателе «Национальные стандарты», а официальный текст изменений и поправок — в ежемесячном указателе «Национальные стандарты». В случае пересмотра (замены) или отмены настоящего стандарта соответствующее уведомление будет опубликовано в ближайшем выпуске ежемесячного информационного указателя «Национальные стандарты». Соответствующая информация, уведомление и тексты размещаются также в информационной системе общего пользования — на официальном сайте Федерального агентства по техническому регулированию и метрологии в сети Интернет (gost.ru)

© Стандартинформ, 2015

Настоящий стандарт не может быть полностью или частично воспроизведен, тиражирован и распространен в качестве официального издания без разрешения Федерального агентства по техническому регулированию и метрологии

Содержание

1 Область применения . . . 1
2 Нормативные ссылки . . . 1
2.1 Идентичные рекомендации и международные стандарты . . . 1
2.2 Дополнительные ссылки . . . 2
3 Определения . . . 3
3.1 Импортируемые определения . . . 3
3.2 Дополнительные определения . . . 3
4 Сокращения . . . 3
5 Нотация . . . 3
6 Цели стандартизации . . . 4
7 Отображение XSD-схем . . . 4
8 Игнорируемые компоненты и свойства схемы . . . 7
9 Модули ACH.1 . . . 7
10 Преобразование имен . . . 8
10.1 Общие положения . . . 8
10.2 Формирование определений типов ACH.1, являющихся ссылками на присвоения типа ACH.1 8
10.3 Формирование идентификаторов и имен ссылок типов . . . 8
10.4 Порядок отображения . . . 10
11 Отображение применений XSD встроенных типов . . . 11
12 Отображение фасетов . . . 12
12.1 Фасеты length, minLength и maxLength . . . 12
12.2 Фасет pattern . . . 13
12.3 Фасет whiteSpace . . . 13
12.4 Фасет enumeration . . . 14
12.5 Другие фасеты . . . 16
13 Отображение простых определений типа . . . 16
14 Отображение объявлений элементов . . . 18
15 Отображение объявлений атрибутов . . . 19
16 Отображение значений простых определений типа . . . 19
17 Отображение определений модельной группы . . . 20
18 Отображение модельных групп . . . 20
19 Отображение частиц . . . 20
20 Отображение сложных определений типа . . . 22
21 Отображение групповых символов . . . 24
22 Отображения применений атрибутов . . . 25
23 Отображение применений простых и сложных определений типа (общий случай) . . . 26
24 Отображение особых применений простых и сложных определений типа (заменяемых) . . . 27
25 Отображение особых применений простых и сложных определений типа (заменяемых, обнуляемых) . . . 28
26 Отображение особых применений простых определений типа (обнуляемых) . . . 30
27 Отображение особых применений сложных определений типа (обнуляемых) . . . 30
28 Отображение особых применений объявлений элемента (головного элемента группы замены элементов) . . . 32
29 Формирование особых присвоений типа ACH.1 для типов, используемых в объявлениях элементов . . . 32
30 Формирование особых присвоений типа ACH.1 для типов, принадлежащих к иерархии развития 34
31 Формирование особых присвоений типа ACH.1 для групп замены элементов . . . 34
Приложение А (обязательное) Определения типов ACH.1, соответствующие XSD-встроенным типам, для отображения версии 1 . . . 36
Приложение В (обязательное) Определения типов ACH.1, соответствующие XSD-встроенным типам, для отображения версии 2 . . . 40

Приложение C (справочное) Идентификация модуля XSD 45
Приложение D (справочное) Примеры отображения 46
Приложение E (справочное) Применение отображения для обеспечения двоичных кодировок для W3C XML-схемы 70
Приложение ДА (справочное) Сведения о соответствии ссылочных международных стандартов ссылочным национальным стандартам Российской Федерации 73

НАЦИОНАЛЬНЫЙ СТАНДАРТ РОССИЙСКОЙ ФЕДЕРАЦИИ

Информационная технология

ПРАВИЛА КОДИРОВАНИЯ АСН.1

Часть 5

Отображение определений W3C схемы XML в АСН.1

Information technology. ASN.1 encoding rules. Part 5.
Mapping W3C XML schema definitions into ASN.1

Дата введения — 2014 — 07— 01

1 Область применения

Настоящий стандарт определяет две версии отображения любой схемы XSD в схему АСН.1. Схема АСН.1 для обеих версий поддерживает ту же семантику и проверяет такой же набор XML-документов.

Настоящий стандарт определяет конечные команды кодирования XER, которые следует применять как часть установленного отображения в типы АСН.1, но не определяет, какая синтаксическая форма должна использоваться для описания этих конечных команд кодирования XER, а также порядок или способ их присвоения.

Существуют различные (синтаксические) способы присвоения команд кодирования XER для использования в EXTENDED-XER кодировках (например, использование команд кодирования приставки типа АСН.1 или использования секции контроля кодирования XER). Вопрос стиля в выборе этих синтаксических форм выходит за рамки настоящего стандарта.

Примечание — Разработчики инструментов формирования таких отображений могут использовать любые синтаксические формы или порядок присвоения, которые приведут к выполнению указанных конечных команд кодирования XER. В примерах, приведенных в настоящем стандарте, в основном используется форма приставки типа (type prefix), но использование секции контроля кодирования XER (encoding control section) может быть предпочтительным для отображения полной схемы XSD, это лишь вопрос стиля.

2 Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие стандарты:

2.1 Идентичные рекомендации и международные стандарты

Примечание — Далее приведен полный перечень рекомендаций и международных стандартов по АСН.1, так как все они могут быть применены в конкретных случаях использования настоящего стандарта. Когда в тексте настоящего стандарта нет прямых ссылок на какой-либо документ, в приведенном далее списке к этому документу добавлен символ †.

Рекомендация МСЭ-Т X.680 (2008) (ИСО/МЭК 8824-1:2008) Информационные технологии — Абстрактная синтаксическая нотация версии один (АСН.1): Спецификация основной нотации

Рекомендация МСЭ-Т X.681 (2008) (ИСО/МЭК 8824-2:2008) Информационные технологии — Абстрактная синтаксическая нотация версии один (АСН.1): Спецификация информационного объекта †

Рекомендация МСЭ-Т X.682 (2008) (ИСО/МЭК 8824-3:2008) Информационные технологии — Абстрактная синтаксическая нотация версии один (АСН.1): Спецификация ограничений

Издание официальное

Рекомендация МСЭ-Т X.683 (2008) (ИСО/МЭК 8824-4:2008) Информационные технологии — Абстрактная синтаксическая нотация версии один (АСН.1): Параметризация спецификаций АСН.1

Рекомендация МСЭ-Т X.690 (2008) (ИСО/МЭК 8825-1:2008) Информационные технологии — Правила кодирования АСН.1: Спецификация базовых (BER), канонических (CER) и отличительных (DER) правил кодирования

Рекомендация МСЭ-Т X.691 (2002) (ИСО/МЭК 8825-2:2002) Информационные технологии — Правила кодирования АСН.1: Спецификация правил уплотненного кодирования (PER)

Рекомендация МСЭ-Т X.692 (2008) (ИСО/МЭК 8825-3:2008) Информационные технологии — Правила кодирования АСН.1. Спецификация нотации контроля кодирования (ECN)

Рекомендация МСЭ-Т X.693 (2008) (ИСО/МЭК 8825-4:2008) Информационные технологии — Правила кодирования АСН.1: Правила XML кодирования (XER)

Рекомендация МСЭ-Т X.891 (2005) (ИСО/МЭК 24824-1:2007) Информационные технологии — Общие правила применения АСН.1: Быстрые команды

2.2 Дополнительные ссылки

ИСО 8601:2004, Элементы данных и форматы обмена — Обмен информацией — Представление дат и времени

ИСО/МЭК 10646:2003, Информационная технология — Универсальный многооктетный набор закодированных символов (UCS)

W3C XML 1.0:2000 Расширяемый язык разметки (XML) 1.0 (второе издание), рекомендации W3C, Copyright © [6 October 2000] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), gost.e-ecolog.ru 2000/REC-xml-20001006

W3C XML Namespaces: 1999, Пространство имен XML, рекомендации W3C, Copyright © [14 January1999] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), gost.e-ecolog.ru

W3C XML Information Set: 2001, Информационный набор XML, рекомендации W3C, Copyright © [24 October 2001] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/REC-xml-infoset-20011024

W3C XML Schema: 2001, XML-схема. Часть 1: Структуры, рекомендации W3C, Copyright © [2 May 2001] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), http://www.w3.org/TR/2001/RECxmlschema-1-20010502

W3C XML Schema: 2001, XML-схема. Часть 2: Типы данных, рекомендации W3C, Copyright © [2 May 2001] World Wide Web Consortium (Massachusetts Institute of Technology, Institut National de Recherche en Informatique et en Automatique, Keio University), gost.e-ecolog.ru

Примечание — При упоминании схемы W3C XML в настоящем стандарте, имеется в виду первая и вторая часть схем W3C XML.

IETF RFC 2396 (1998) Унифицированный идентификатор ресурса (URI): Generic Syntax

IETF RFC 1766 (1995) Теги для идентификации языка

3 Определения

3.1 Импортируемые определения

3.1.1 В настоящем стандарте использованы термины по ИСО/МЭК 8824-1 и ИСО/МЭК 8825-4.

Примечание — В частности термины «конечные команды кодирования XER», «приставка типа» и «секция контроля кодирования XER», определяются в упомянутых рекомендациях и международных стандартах.

3.1.2 В настоящем стандарте также использованы термины, описанные в «W3C XML-схеме» и «Информационном наборе W3C XML».

Примечания

1 Считается, что эти термины соответствуют тем, на которые ссылаются в 3.1.1. Если такого соответствия нет, то применять необходимо термины, определенные в 3.1.1.

2 В частности, терминам «компонент схемы» и «свойство (компонента схемы)» дают определение в Схеме W3C XML, а терминам «элемент единицы информации» и «элемент атрибута информации» дают определение в Информационном наборе W3C XML.

3 Используемые в настоящем стандарте термины «высокоуровневое **простое определение типа**» («top-level **simple type definition**») и «высокоуровневое **сложное определение типа**» («top-level **compex type definition**») не включают встроенные XSD-типы.

3.2 Дополнительные определения

В настоящем стандарте используются следующие дополнительные определения:

3.2.1 **пространство имен XSD** (XSD namespace): Пространство имен с URI: gost.e-ecolog.ru XMLSchema.

3.2.2 **пространство имен XSI** (XSI namespace): Пространство имен с URI: gost.e-ecolog.ru XMLSchema-instance.

3.2.3 **пространство имен XML** (XML namespace): Пространство имен с URI: gost.e-ecolog.ru 1998/namespace.

4 Сокращения

В настоящем стандарте применены следующие сокращения:

ACH.1 — Абстрактная синтаксическая нотация версии 1 (Abstract Syntax Notation One);
BER — (ACH.1) Basic Encoding Rules (базовые правила кодирования);
DER — (ACH.1) Distinguished Encoding Rules (отличительные правила кодирования);
PER — (ACH.1) Packed Encoding Rules (правила уплотненного кодирования);
URI — (IETF) Uniform Resource Identifier (унифицированный идентификатор ресурса);
XER — (ACH.1) XML Encoding Rules (правила XML-кодирования);
XML — (W3C) eXtensible Markup Language (расширяемый язык разметки);
XSD — (W3C) XML Schema (XML-схема).

5 Нотация

5.1 В настоящем стандарте используют нотацию по ИСО/МЭК 8824-1, ИСО/МЭК 8824-3 и W3C XML-схема.

5.2 При необходимости детально и на примерах описать процедуру присвоения команд кодирования XER, в настоящем стандарте используется нотация префикс типа (см. 6.3 и 6.4). В приложении А используется секция контроля кодирования XER.

5.3 В настоящем стандарте **`полужирный шрифт Courier`** используют для нотации ACH.1, а **полужирный шрифт Arial** используют для нотации XSD и для терминов и понятий XSD.

5.4 В настоящем стандарте схемы XSD, используемые в примерах, имеют префикс «**xsd:**», идентифицирующий принадлежность к пространству имен XSD.

6 Цели стандартизации

6.1 Описываемое в настоящем стандарте отображение ACH.1, гарантирует, что:

a) любые законченные модули ACH.1, сформированные инструментами в соответствии с настоящим стандартом (для одинаковых схем XSD), устанавливают одинаковые (структурированные) абстрактные значения;

b) все BASIC-XER, CXER, EXTENDED-XER и двоичные кодировки в окончательном представлении ACH.1 дадут одинаковые результаты кодирования (в соответствии с настройками кодеров);

c) для всех XML-документов, входящих в схему XSD, кодировки EXTENDED-XER для абстрактных значений представления ACH.1 являются допустимыми.

6.2 В определении ACH.1 существует достаточное число аспектов (таких как использование пробела, секции контроля кодирования или приставки типа), которые не влияют ни на определяемые абстрактные значения, ни на правила XER или двоичные кодировки этих значений. Такие аспекты ACH.1 не рассматриваются в настоящем стандарте.

6.3 В ACH.1 существует много различных путей для присвоения команды кодирования XER типу, включающих:

a) использование приставки типа для каждой присвоенной команды кодирования;

b) использование секции контроля кодирования с отдельными командами кодирования для каждого требуемого присвоения;

c) использование секции контроля кодирования с единственной командой кодирования, делающей глобальное присвоение, с возможным добавлением при помощи команд отрицательного кодирования для определенных типов.

6.4 Настоящий стандарт определяет когда конечные команды кодирования XER должны быть объявлены. Большинство примеров, приведенных в настоящем стандарте, используют синтаксис, описанный в 6.3(a). Однако использование различных вариантов, рассмотренных в 6.3, никак не стандартизовано, так что в соответствующий реализации отображения не исключаются выбор любой синтаксической формы (или использование нескольких синтаксических форм) для присвоения конечных команд кодирования XER.

Примечание — Выбор того или иного варианта не оказывает никакого влияния на заключительный двоичный код или XML-кодировки.

6.5 Формальная спецификация необходимого отображения не прилагается.

6.6 Настоящий стандарт затрагивает только вопрос отображения тех схем XSD, которые соответствуют W3C XML-схеме.

Примечание — Такое соответствие может быть либо согласно условию одного или более документов схемы W3C XSD, либо согласно другим источникам, указанным в W3C XML-схеме.

7 Отображение XSD-схем

7.1 Схема XSD является источником отображения и состоит из набора компонентов схемы (см. Часть 1, 2.2 W3C XML-схемы). Компоненты схемы или наборы компонентов схем не требуют какого-либо определенного представления или предполагаемого отображения, при этом ожидается, что исходная схема XSD будет представлена одним или несколькими документами XML-схемы (см. Часть 1, 3.15.2 W3C XML-схемы).

Примечания

1 Компоненты схемы, представленные в множестве документов XML-схемы, становятся частью такой же схемы XSD при помощью элементов информации: **xsd:include, xsd:redefine** и **xsd:import.**

2 Поскольку отображение описывается с позиции компонентов схемы (а не с точки зрения их XML-представления), на это не влияют детали XML-представления, такие как использование множества документов схемы, соединенных частями информации об элементе **xsd:include** и **xsd:redefine,** размещение частей информации об элементе в одной или другой схеме документа, порядок частей информации об элементе **xsd:attribute** в пределах части информации об элементе **xsd:complexType,** и т. д.

3 Два набора документов схемы, которые отличаются по многим аспектам, но представляют тот же самый набор компонентов схемы, формируют тот же самый набор присвоений типа ACH.1, с теми же самыми конечными командами кодирования, присвоенными им и их компонентам на всех уровнях детализации.

7.2 Исходная схема XSD встречает ряд ограничений, наложенных спецификацией XSD. Если исходная схема XSD представлена (целиком или частично) как ряд документов XML-схемы, то каждый документ схемы должен быть корректным согласно схеме XSD для схем (см. Часть 1 W3C XML-схемы, Приложение A).

7.3 Для исходной схемы XSD должны быть сформированы один или более модулей ACH.1. Число сформированных модулей ACH.1 зависит от реализации. Каждый модуль ACH.1 должен содержать нуль или больше присвоений типа, соответствующих высокоуровневым компонентам схемы (см. 7.6), и нуль или больше особых присвоений типа ACH.1 (см. 29, 30 и 31). Физический порядок присвоений типа в пределах каждого модуля ACH.1 зависит от реализации. Когда несколько модулей ACH.1 сформированы, способ, которым сформированные присвоения типа распределяются через модули ACH.1, также является опцией реализации.

Примечания

1 Включение в тот же самый модуль ACH.1 присвоений типа, сформированных из компонентов схемы XSD с различными **целевыми пространствами имен (target namespaces)**, разрешено данным подпунктом, но не рекомендуется. Если имеется возможность, предпочтительно формирование одного модуля ACH.1 на пространство имен. Также рекомендуется, чтобы каждое особое присвоение типа ACH.1 находилось внутри того же самого модуля ACH.1, что и связанное с ним присвоение типа ACH.1 (см. 29.5, 30.4 и 31.4).

2 Формирование присвоений типа ACH.1 (см. 7.6 и 10.4) не оказывает влияния на число полученных модулей ACH.1 (за исключением возможного применения «ExternalTypeReference», как описано в 10.2.2), ни по тому, как сформированные присвоения типа распределяются через те модули, ни согласно физическому порядку присвоений типа в пределах каждого модуля. В частности, имена ссылок типов указанных присвоений типа будут те же самые независимо от того, какой стиль отображения используется реализацией.

3 Полное описание отношений между концепцией пространства имен в XML W3C-пространстве имен и именования в ACH.1 осуществляется в Рек. МСЭ-Т X.683 (2008) (ИСО/МЭК 8824-4:2008), 16. Имена ссылок типов и идентификаторы, определенные в модуле ACH.1, присваиваются пространству имен посредством команд кодирования NAMESPACE, а в противном случае не имеют пространства имен. Отображение формирует команды кодирования NAMESPACE там, где это необходимо.

7.4 Все сформированные отображением модули ACH.1 должны содержать (в секции контроля кодирования XER) команды кодирования `GLOBAL-DEFAULTS MODIFIED-ENCODINGS` и команды кодирования `GLOBAL-DEFAULTS CONTROL-NAMESPACE`, определяющие пространство имен XSI.

7.5 Исходная схема XSD должна быть обработана следующим образом:

a) для каждого высокоуровневого **объявления элемента (element declaration)** присвоение типа ACH.1 должно быть сформировано применением раздела 14 к **объявлению элемента;**

b) для каждого высокоуровневого **объявления атрибута (attribute declaration)** присвоение типа ACH.1 должно быть сформировано применением раздела 15 к **объявлению атрибута;**

c) для каждого высокоуровневого **простого определения типа (simple type definition)** присвоение типа ACH.1 должно быть сформировано применением раздела 13 к **простому определению типа;**

d) для каждого высокоуровневого **сложного определение типа (complex type definition)** присвоение типа ACH.1 должно быть сформировано применением раздела 20 к **сложному определению типа;**

e) для каждого **определения модельной группы (model group definition)** у **модельной группы (model group)** которого есть **наборщик (compositor) последовательности (sequence)** или **выбора (choice)**, присвоение типа ACH.1 должно быть сформировано применением раздела 17 к **определению модельной группы.**

Примечания

1 Оставшиеся компоненты исходной XSD-схемы будут обработаны в результате отображения этих компонентов.

2 Порядок, в котором должны быть отображены компоненты схемы, описан в 10.4. Порядок элементов, указанных ранее, не имеет никакого значения для отображения.

7.6 В первом столбце таблицы 1 перечисляются компоненты схемы. Во втором столбце дается ссылка на пункт в W3C XML-схеме, в котором описывается компонент схемы. В третьем столбце перечисляются пункты, в которых описывается отображение соответствующих компонентов схемы в ACH.1.

Таблица 1 — Отображение компонентов схемы XSD

Компонент схемы XSD	Ссылка на W3C XML-схему	Отображение описано в разделе
объявление атрибута (attribute declaration)	Часть 1, 3.2	15
объявление элемента (element declaration)	Часть 1, 3.3	14
сложное определение типа (complex type definition)	Часть 1, 3.4	20
применение атрибута (attribute use)	Часть 1, 3.5	22
определение группы атрибутов (attribute group definition)	Часть 1, 3.6	*Не отображается*
определение модельной группы (model group definition)	Часть 1, 3.7	17
модельная группа (model group)	Часть 1, 3.8	18
частица (particle)	Часть 1, 3.9	19
групповой символ (wildcard)	Часть 1, 3.10	21
определение ограничения идентичности (identity-constraint definition)	Часть 1, 3.11	*Игнорируется*
объявление обозначения (notation declaration)	Часть 1, 3.12	*Игнорируется*
примечание (annotation)	Часть 1, 3.13	*Игнорируется*
простое определение типа (simple type definition)	Часть 1, 3.14	11, 13
схема (schema)	Часть 1, 3.15	9
упорядоченные (ordered)	Часть 2, 4.2.2.1	*Игнорируется*
ограниченные (bounded)	Часть 2, 4.2.3.1	*Игнорируется*
количество элементов (cardinality)	Часть 2, 4.2.4.1	*Игнорируется*
числовые (numeric)	Часть 2, 4.2.5.1	*Игнорируется*
length	Часть 2, 4.3.1.1	12
minLength	Часть 2, 4.3.2.1	12
maxLength	Часть 2, 4.3.3.1	12
pattern	Часть 2, 4.3.4.1	12
enumeration	Часть 2, 4.3.5.1	12
whiteSpace	Часть 2, 4.3.6.1	12
maxInclusive	Часть 2, 4.3.7.1	12
maxExclusive	Часть 2, 4.3.8.1	12
minExclusive	Часть 2, 4.3.9.1	12
minInclusive	Часть 2, 4.3.10.1	12
totalDigits	Часть 2, 4.3.11.1	12
fractionDigits	Часть 2, 4.3.12.1	12

8 Игнорируемые компоненты и свойства схемы

8.1 Отображение должно проигнорировать компоненты схемы и свойства, которые перечисляются далее.

8.2 Должны быть проигнорированы все **примечания** (см. Часть 1, 3.13 XML-схемы W3C).

Примечание — Все единицы информации атрибута в документе схемы с именами, квалифицированными с пространствами имен кроме пространства имен XSD (см. Часть 1, 3.13.1 W3C XML-схемы), являются свойством **примечаний** и игнорируются.

8.3 Должны быть проигнорированы все **определения ограничения идентичности** (см. Часть 1, 3.11 W3C XML-схемы).

Примечание — **Определение ограничения идентичности** обеспечивает механизмы определения ссылочных ограничений, которые могут требоваться в допустимом случае. В АСН.1 в настоящий момент нет понятия таких ограничений, и указанные ограничения не могут быть отображены в формальную спецификацию АСН.1, но они могут быть включены как нормативные комментарии, которые привязываются к реализации приложения.

8.4 Должны быть проигнорированы все **объявления обозначения** (см. Часть 1, 3.12 W3C XML-схемы).

8.5 Должны быть проигнорированы все компоненты схемы, которые являются **фундаментальными фасетами (fundamental facets) (упорядоченный, ограниченный, количество элементов, числовой) простых определений типа** (см. Часть 2, 4.2 W3C XML-схемы).

8.6 Свойства **определения ограничения идентичности, исключения группы замены (substitution group exclusions)** и **неразрешенные замены (disallowed substitutions) объявлений элемента** должны быть проигнорированы.

8.7 Свойства **окончание (final), абстрактность (abstract)** и **запрещенные замены (prohibited substitutions) сложных определений типа** должны быть проигнорированы.

8.8 Свойство **process contents групповых символов** должно быть проигнорировано.

Примечание — В АСН.1 нет поддержки ни для какого действия, кроме **пропуска (skip).**

8.9 Свойства **фундаментальные фасеты** и **окончание простых определений типа** должны быть проигнорированы.

8.10 Должны быть проигнорированы все **ограничения значения (value constraints),** которые присутствуют на любых **объявлениях элемента** или **объявлениях атрибута, определение типа (type definition)** которых — либо **xsd:QName,** либо **простое определение типа,** полученное из **xsd:QName** или **xsd:NOTATION.**

8.11 Все **определения группы атрибутов** должны быть проигнорированы.

Примечание — **Применения атрибута** в **определении группы атрибутов** становится частью **применений атрибута сложных определений типа,** XML-представление которых содержит ссылку на **определения группы атрибутов.**

9 Модули АСН.1

9.1 Отображение схемы XSD формирует один или более модулей АСН.1 (см. 7.3).

9.2 «ModuleIdentifier» АСН.1 (см. Рек. МСЭ-Т Х.680 ИСО/МЭК 8824-1:2008, 13), который должен быть сформирован отображением, не стандартизируется. Там, где используются инструкции `IMPORTS`, имена и идентификаторы модулей АСН.1 в инструкциях `IMPORTS` должны быть теми, что сформированы отображением для модулей АСН.1.

Примечание — Выбор «ModuleIdentifier» не влияет на кодировки ни в одном из стандартных правил кодирования.

9.3 Модули АСН.1 должны иметь «TagDefault» из **AUTOMATIC TAGS.**

9.4 В каждом модуле АСН.1, сформированном отображением версии 1, должна присутствовать АСН.1-инструкция `IMPORTS`, импортирующая имена ссылок типов АСН.1 в модуль под названием `XSD {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2) version1(1)}`, описанный в приложении А, на которые ссылаются в модуле АСН.1.

9.5 В каждом модуле ACH.1, сформированном отображением версии 2, должна присутствовать ACH.1-инструкция **IMPORTS**, импортирующая имена ссылок типов ACH.1 в модуль под названием **XSD {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2) version2(2)}**, описанный в приложении В, на которые ссылаются в сформированном модуле ACH.1.

Примечание — Термин «XSD модуль» в настоящем стандарте отсылает к модулю, описанному в приложении А (отображение версии 1) или в приложении В (отображение версии 2) версии отображения.

9.6 Инструкция **IMPORTS** должна также импортировать имена ссылок типов ACH.1 присвоений типа, которые были размещены (в результате отображения) в других модулях ACH.1, но ссылаются в этом модуле ACH.1.

9.7 Не должно быть никакой инструкции **EXPORTS**.

Примечание — Это означает, что все имена ссылок типов ACH.1 в модуле ACH.1 могут быть импортированы в другие модули.

10 Преобразование имен

10.1 Общие положения

10.1.1 Настоящий стандарт описывает формирование:

a) имен ссылок типов ACH.1, соответствующих **именам определений модельной группы,** высокоуровневых **объявлений элемента,** высокоуровневых **объявлений атрибута,** высокоуровневых **сложных определений типа** и высокоуровневых **простых определений типа;**

b) идентификаторов ACH.1, соответствующих **именам** высокоуровневых **объявлений элемента,** высокоуровневых **объявлений атрибута,** локальных **объявлений элемента** и локальных **объявлений атрибута;**

c) идентификаторов ACH.1 для отображения определенных **простых определений типа** с фасетом **enumeration** (см. 12.4.1 и 12.4.2);

d) имен ссылок типов ACH.1 особых присвоений типа (см. 29, 30 и 31);

e) идентификаторов ACH.1 компонентов определенной последовательности, представленных отображением (см. раздел 20).

10.1.2 Все указанные имена ACH.1 сформированы применением 10.3 либо к **имени** соответствующего компонента схемы, либо к элементу **значения (value)** фасета **enumeration,** либо к указанной символьной строке, как определено в соответствующих местах настоящего стандарта.

10.2 Формирование определений типов ACH.1, являющихся ссылками на присвоения типа ACH.1

10.2.1 Применение данного подраздела является прямым запросом других пунктов настоящего стандарта сформировать определение типа ACH.1 («DefinedType»), которое является ссылкой на присвоение типа ACH.1.

10.2.2 Если «DefinedType» должно быть вставлено в модуль ACH.1 (M, например), отличный от модуля ACH.1, куда присвоение типа ACH.1, на которое ссылаются, было вставлено, то «DefinedType» для этого присвоения типа должно быть либо «typereference», либо «ExternalTypeReference», что является опцией реализации. Либо это должно быть «typereference» для данного присвоения типа.

Примечание — Все «typereference» ACH.1, создаваемые отображением, уникальны для любой легальной схемы на входе; таким образом, тип, определенный в другом модуле ACH.1, не обязан быть «ExternalTypeReference».

10.3 Формирование идентификаторов и имен ссылок типов

10.3.1 Применение данного подраздела является прямым запросом из других мест настоящего стандарта сформировать имя ссылки типа ACH.1 или идентификатор.

10.3.2 **Имена объявлений атрибута, объявлений элемента, определений модельной группы,** высокоуровневых **простых определений типа** и высокоуровневых сложных **определений типа** могут быть идентичными зарезервированным словам ACH.1 или могут содержать символы, не разрешенные в идентификаторах ACH.1 или в именах ссылок типов ACH.1. Кроме того, есть случаи, в которых имена ACH.1 обязаны быть отличными там, где **именам** соответствующих компонентов схемы XSD (от которых отображаются имена ACH.1) разрешено быть идентичными.

10.3.3 Следующие преобразования должны быть применены, по порядку, к каждой символьной строке, отображаемой в имени АСН.1, где каждое преобразование (кроме первого) применяется к результату предыдущего преобразования:

— символы: « » (ПРОБЕЛ), «.» (ТОЧКА), и «_» (НИЖНЕЕ ПОДЧЕРКИВАНИЕ) должны все быть заменены на «-» (ДЕФИС-МИНУС);

— любой символ, кроме «A» — «Z» (от ЛАТИНСКОЙ ПРОПИСНОЙ БУКВЫ A до ЛАТИНСКОЙ ПРОПИСНОЙ БУКВЫ Z), «a» — «z» (от ЛАТИНСКОЙ СТРОЧНОЙ БУКВЫ a до ЛАТИНСКОЙ СТРОЧНОЙ БУКВЫ z), «0» — «9» (от ЦИФРЫ НОЛЬ до ЦИФРЫ ДЕВЯТЬ) и «-» (ДЕФИС-МИНУС) должны быть удалены;

— последовательность двух или более символов ДЕФИС-МИНУС должна быть заменена единственным символом ДЕФИС-МИНУС;

— символы ДЕФИС-МИНУС, стоящие в начале или в конце имени, должны быть удалены;

— если символьная строка, которая должна использоваться в качестве имени ссылки типа, начинается со строчной буквы, то первая буква должна быть преобразована в прописную; если она начинается с цифры (от ЦИФРЫ НОЛЬ до ЦИФРЫ ДЕВЯТЬ), это должно быть снабжено префиксом «**X**» (ЛАТИНСКАЯ ПРОПИСНАЯ БУКВА X);

— если символьная строка, которая должна использоваться в качестве идентификатора, начинается с прописной буквы, то первая буква должна быть преобразована в строчную; если она начинается с цифры (от ЦИФРЫ НОЛЬ до ЦИФРЫ ДЕВЯТЬ), то это должно быть снабжено префиксом «**x**» (ЛАТИНСКАЯ СТРОЧНАЯ БУКВА x);

— если символьная строка, которая должна использоваться в качестве имени ссылки типа, пуста, то она должна быть заменена на «**X**» (ЛАТИНСКАЯ ПРОПИСНАЯ БУКВА X);

- если символьная строка, которая должна использоваться в качестве идентификатора, пуста, то она должна быть заменена на «**x**» (ЛАТИНСКАЯ СТРОЧНАЯ БУКВА x).

10.3.4 В зависимости от вида сформированного имени применяется один из трех следующих подпунктов.

10.3.4.1 Если сформированное имя является именем ссылки типа присвоения типа АСН.1 и символьная строка, сформированная 10.3.3, идентична:

a) имени ссылки типа другого присвоения типа АСН.1, ранее (см. 10.4) сформированного отображением (в любом модуле АСН.1); или

b) имени ссылки типа присвоения типа в модуле **XSD** (см. приложение А); или

c) одному из зарезервированных слов, указанных в Рекомендации МСЭ-Т X.680 ИСО/МЭК 8824-1:2008 (п. 12.38),

то суффикс должен быть добавлен к символьной строке, сформированной 10.3.3. Суффикс должен состоять из символа ДЕФИС-МИНУС, сопровождаемого каноническим лексическим представлением (см. Часть 2, 2.3.1 W3C XML-схемы) целочисленной переменной (integer). Эта целочисленная переменная должна быть наименьшей положительной целочисленной переменной, чтобы новое имя отличалось от имени ссылки типа любого другого присвоения типа АСН.1, сформированного ранее (в любом модуле АСН.1).

Примечание — Как следствие этого правила, все имена ссылок типов, описанные в спецификации АСН.1, сформированные из первоначальной схемы XSD (включая стандартизированные ссылки типов, описанные в модуле **XSD**), будут уникальны в рамках этой спецификации АСН.1. Это дает максимальную гибкость в способе, которым сформированные присвоения типа АСН.1 распределяются через многократные модули АСН.1 (см. 7.3).

10.3.4.2 Если сформированное имя является идентификатором компонента последовательности, набора или типа выбора, и символьная строка, сформированная 10.3.3, идентична идентификатору ранее сформированного компонента той же самой последовательности, набора или типа выбора, то суффикс должен быть добавлен к символьной строке, сформированной 10.3.3. Суффикс должен состоять из символа ДЕФИС-МИНУС, сопровождаемого каноническим лексическим представлением (см. Часть 2, 2.3.1 W3C XML-схемы) целочисленной переменной. Эта целочисленная переменная должна быть наименьшей положительной целочисленной переменной, чтобы новый идентификатор отличался от идентификатора любого ранее сформированного компонента этой последовательности, набора или типа выбора.

10.3.4.3 Если сформированное имя является «identifier» в «EnumerationItem» перечислимого типа (enumerated type), и символьная строка, сформированная 10.3.3, идентична «identifier» в другом «EnumerationItem», ранее сформированном в том же самом перечислимом типе, то суффикс должен быть добавлен к символьной строке, сформированной 10.3.3. Суффикс должен состоять из символа ДЕФИС-МИНУС, сопровождаемого каноническим лексическим представлением (см. Часть 2, 2.3.1 W3C XML-схе-

мы) целочисленной переменной. Эта целочисленная переменная должна быть наименьшей положительной целочисленной переменной, чтобы новый идентификатор отличался от «identifier» в любом другом «EnumerationItem», уже существующем в этом перечислимом типе ACH.1.

10.3.5 Для имени ссылки типа (или идентификатора) ACH.1, которое сформировано применением 10.3 к **имени объявления элемента, объявления атрибута,** высокоуровневого **сложного определения типа** или высокоуровневого **простого определения типа,** если сформированное имя ссылки типа (или идентификатор) отличается от **имени,** то заключительная команда кодирования **NAME** должна быть назначена присвоению типа ACH.1 с этим именем ссылки типа (или компоненту с этим идентификатором), как описано в трех следующих абзацах.

Если единственным отличием является регистр первой буквы (который является верхним регистром в имени ссылки типа и нижним регистром в **имени**), то «Keyword» в команде кодирования **NAME** должно быть **UNCAPITALIZED**.

Если единственным отличием является регистр первой буквы (который является нижним регистром в идентификаторе и верхним регистром в **имени**), то «Keyword» в команде кодирования **NAME** должно быть **CAPITALIZED**.

В противном случае «NewName» в команде кодирования **NAME** должно быть **именем.**

***Пример — высокоуровневое* сложное определение типа:**

```
<xsd:complexType name="COMPONENTS">
        <xsd:sequence>
                <xsd:element name="Elem" type="xsd:boolean"/>
                <xsd:element name="elem" type="xsd:integer"/>
                <xsd:element name="Elem-1" type="xsd:boolean"/>
                <xsd:element name="elem-1" type="xsd:integer"/>
        </xsd:sequence>
</xsd:complexType>
```

отображается в присвоение типа ACH.1:

```
COMPONENTS-1 ::= [NAME AS "COMPONENTS"] SEQUENCE {
        elem [NAME AS CAPITALIZED] BOOLEAN,
        elem-1 [NAME AS "elem"] INTEGER,
        elem-1-1 [NAME AS "Elem-1"] BOOLEAN,
        elem-1-2 [NAME AS "elem-1"] INTEGER }
```

10.3.6 Для имени ссылки типа (или идентификатора) ACH.1, которое сформировано применением 10.3 к **имени объявления элемента, объявления атрибута,** высокоуровневого **сложного определения типа** или определяемого пользователем высокоуровневого **простого определения типа,** если **целевое пространство имен** компонента схемы не является **отсутствующим (absent),** то заключительная команда кодирования **NAMESPACE** должна быть назначена присвоению типа ACH.1 с этим именем ссылки типа (или именному типу с этим идентификатором) и должна определить **целевое пространство имен** компонента схемы.

10.3.7 Для идентификатора ACH.1, который сформирован 10.3 для отображения **простого определения типа** с фасетом **enumeration,** где сформированный идентификатор отличается от соответствующего элемента **значения** фасета **enumeration,** заключительная команда кодирования **TEXT** должна быть присвоена перечислимому типу ACH.1 с уточняющей информацией, указывающей «identifier» в «EnumerationItem» перечислимого типа. Применяется один из двух следующих абзацев.

Если единственным отличием служит регистр первой буквы (который является нижним регистром в идентификаторе и верхним регистром в элементе **значения** фасета **enumeration**), то «Keyword» в команде кодирования **TEXT** должно быть **CAPITALIZED**.

В противном случае «NewName» в команде кодирования **TEXT** должно быть элементом **значения** фасета **enumeration.**

10.4 Порядок отображения

10.4.1 Порядок отображения налагается на высокоуровневые компоненты исходной схемы XSD, на которой выполняется отображение. Это применяется к **определениям модельной группы,** высокоуровневым **сложным определениям типа**, высокоуровневым **простым определениям типа,** высокоуровневым **объявлениям атрибута** и высокоуровневым **объявлениям элемента.**

Примечание — Другие высокоуровневые компоненты схемы не отображаются в ACH.1, а встроенные типы XSD отображаются особым способом.

10.4.2 Порядок описывается в трех следующих абзацах.

Высокоуровневые компоненты схемы сначала должны быть упорядочены по их **целевым пространствам имен,** с **отсутствующим** пространством имен, предшествующим всем именам пространства имен, определенным в схеме XSD, в возрастающем лексикографическом порядке.

Внутри каждого целевого пространства имен высокоуровневые компоненты схемы должны быть разделены на четыре набора, упорядоченные следующим образом:

a) **объявления элемента;**

b) **объявления атрибута;**

c) **сложные определения типа** и **простые определения типа;**

d) **определения модельной группы.**

Внутри каждого набора (см. 10.4.2.2), компоненты схемы должны быть упорядочены по **имени** в возрастающем лексикографическом порядке.

10.4.3 Два набора присвоений типа ACH.1 сформированы отображением:

a) одного набора присвоений типа ACH.1 (сформированного в соответствии с разделами 13, 14, 15, 17 и 20), соответствующих непосредственно высокоуровневым компонентам схемы, и их имен ссылок типов, полученных из имени компонента схемы без добавления суффикса;

b) другого набора присвоений типа ACH.1 (сформированного в соответствии с разделами 29, 30 и 31), соответствующих особым применениям высокоуровневых компонентов схемы, и их имен ссылок типов, полученных из имени компонента схемы, сопровождаемого суффиксом и (в некоторых случаях) постсуффиксом.

Примечание — Для каждого высокоуровневого компонента в исходной схеме XSD максимум может быть сформировано одно присвоение типа ACH.1 в наборе в 10.4.3 a), но могут быть сформированы многократные присвоения типа ACH.1 в наборе в 10.4.3 b).

10.4.4 Присвоения типа ACH.1 в наборе в 10.4.3 a) должны быть сформированы в порядке соответствующих компонентов схемы XSD (см. 10.4.1) и должны быть сформированы все прежде чем сформируются любые присвоения типа в 10.4.3 b).

10.4.5 Присвоения типа ACH.1 в 10.4.3 b) должны быть сформированы в следующем порядке:

a) учитывая два высокоуровневых компонента схемы SC1 и SC2, где SC1 предшествует SC2 в порядке, определенном в 10.4.1, все присвоения типа ACH.1, соответствующие SC1 (если такие есть), должны быть сформированы прежде чем сформируются любые присвоения типа, соответствующие SC2;

b) в пределах каждого набора присвоений типа, соответствующих любому предоставленному компоненту схемы, присвоения типа должны быть сформированы в порядке, основанном на суффиксе, указанном в 29—31, следующим образом:

1) суффикс «**-nillable**»;

2) суффикс «**-nillable-default-**»;

3) суффикс «**-nillable-fixed-**»;

4) суффикс «**-derivations**»;

5) суффикс «**-deriv-default-**»;

6) суффикс «**-deriv-fixed-**»;

7) суффикс «**-deriv-nillable**»;

8) суффикс «**-deriv-nillable-default-**»;

9) суффикс «**-deriv-nillable-fixed-**»;

10) суффикс «**-group**»;

c) для элементов 2, 3, 5, 6, 8 и 9 из b) в пределах каждого набора присвоений типа, соответствующих любому предоставленному компоненту схемы и любому данному суффиксу, присвоения типа должны быть сформированы в возрастающем лексикографическом порядке постсуффикса, определенного в разделе 29 (если таковые имеются).

11 Отображение применений XSD встроенных типов

11.1 Применение данного пункта является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее применению встроенного типа XSD.

Примечание — Все XSD-встроенные типы являются **простыми определениями типа,** за исключением **xsd:anyType,** который является **сложным определением типа.**

11.2 Применение XSD-встроенного типа должно быть отображено в определении типа ACH.1 в соответствии с таблицей 2. Таблица показывает определение типа ACH.1, которое должно быть использовано. Нотация «XSD.Name» показывает, что определение типа ACH.1 должно быть определением типа ACH.1 («DefinedType»), сформированным применением 10.2 к соответствующему присвоению типа ACH.1, существующему в **`XSD {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2) version1(1)}`** модуле (отображения версии 1 — см. приложение А) или **`XSD {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2) version2(2)}`** модуле (отображения версии 2 — см. приложение В).

Таблица 2 — Определения типа ACH.1, соответствующие применениям встроенных типов XSD

XSD встроенный тип	ACH.1 определение типа	XSD встроенный тип	ACH.1 определение типа
anyURI	`XSD.AnyURI`	**Int**	`XSD.Int`
anySimpleType	`XSD.AnySimpleType`	**Integer**	`INTEGER`
anyType	`XSD.AnyType` или `XSD.AnyType-nillable` (см. 11.3)	**language**	`XSD.Language`
base64Binary	`[BASE64] OCTET STRING`	**long**	`XSD.Long`
boolean	`BOOLEAN`	**Name**	`XSD.Name`
byte	`INTEGER (-128..127)`	**NCName**	`XSD.NCName`
date	`XSD.Date`	**negativeInteger**	`INTEGER (MIN..-1)`
dateTime	`XSD.DateTime`	**NMTOKEN**	`XSD.NMTOKEN`
decimal	`XSD.Decimal`	**NMTOKENS**	`XSD.NMTOKENS`
double	`XSD.Double`	**nonNegativeInteger**	`INTEGER (0..MAX)`
duration	`XSD.Duration`	**nonPositiveInteger**	`INTEGER (MIN..0)`
ENTITIES	`XSD.ENTITIES`	**normalizedStrin**	`XSD.NormalizedString`
ENTITY	`XSD.ENTITY`	**NOTATION**	`XSD.NOTATION`
float	`XSD.Float`	**positiveInteger**	`INTEGER (1..MAX)`
gDay	`XSD.GDay`	**QName**	`XSD.QName`
gMonth	`XSD.GMonth`	**short**	`XSD.Short`
gMonthDay	`XSD.GMonthDay`	**string**	`XSD.String`
gYear	`XSD.GYear`	**time**	`XSD.Time`
gYearMonth	`XSD.GYearMonth`	**token**	`XSD.Token`
hexBinary	`OCTET STRING`	**unsignedByte**	`INTEGER (0..255)`
ID	`XSD.ID`	**unsignedInt**	`XSD.UnsignedInt`
IDREF	`XSD.IDREF`	**unsignedLong**	`XSD.UnsignedLong`
IDREFS	`XSD.IDREFS`	**unsignedShort**	`XSD.UnsignedShort`

11.3 Применение **xsd:anyType** как **определения типа объявления элемента,** которое не **обнуляемо (nillable),** должно быть отображено в **`XSD.AnyType`**. Применение **xsd:anyType** как **определения типа объявления элемента,** которое **обнуляемо,** должно быть отображено в **`XSD.AnyType-nillable`**.

12 Отображение фасетов

Приводимые далее требования являются прямым запросом из других мест настоящего стандарта отобразить **фасет (facet) простого определения типа. Фасет простого определения типа** STD отображается в ограничении ACH.1, применяемом к определению типа ACH.1, соответствующему STD, если только у STD нет фасета **enumeration,** который отображается в «Enumeration» ACH.1 (см. 12.4.1 и 12.4.2). В этом случае, ограничение ACH.1 от фасета не формируется (см. 12.1.2, 12.2.1, 12.3.1 и 12.5.1).

12.1 Фасеты length, minLength и maxLength

12.1.1 Фасеты **length, minLength** и **maxLength** должны быть проигнорированы для XSD-встроенных типов **xsd:QName** и **xsd:NOTATION** и для любого **простого определения типа,** полученного из них ограничением.

12.1.2 Если фасет **length, minLength** или **maxLength** принадлежит **простому определению типа,** у которого есть также фасет **enumeration,** отображаемый в «Enumeration» АСН.1 (см. 12.4.1 и 12.4.2), то никакие «EnumerationItem» не должны быть включены в «Enumeration» для элементов (если таковые имеются) **значения** фасета **enumeration,** которые не удовлетворяют фасет **length, minLength** или **maxLength.**

12.1.3 Иначе, фасеты **length, minLength** и **maxLength простого определения типа** должны быть отображены в ограничении размера АСН.1 согласно таблице 3.

Таблица 3 — АСН.1 ограничения размера, соответствующие фасетам length, minLength и maxLength

XSD фасет	АСН.1 ограничение размера
length=*значение*	(**SIZE**(*значение*))
minLength=*min*	(**SIZE**(*min* .. **MAX**))
maxLength=*max*	(**SIZE**(0 .. *max*))
minLength=*min* **maxLength=***max*	(**SIZE**(*min* .. *max*))

12.2 Фасет pattern

12.2.1 Если фасет **pattern** принадлежит **простому определению типа,** у которого есть также фасет **enumeration,** отображаемый в «Enumeration» АСН.1 (см. 12.4.1 и 12.4.2), то никакие «EnumerationItem» не должны быть включены в «Enumeration» для элементов (если таковые имеются) **значения** фасета **enumeration,** которые не удовлетворяют фасет **pattern.**

12.2.2 В случае если данное условие не выполняется, фасет **pattern** должен быть отображен в определяемом пользователем ограничении.

Если **значение** фасета **pattern** является одиночным регулярным выражением, то определяемое пользователем ограничение должно быть

(CONSTRAINED BY {/* XML representation of the XSD pattern "xyz" */})

где «xyz» — XML представление **значения** фасета **pattern,** за исключением того, что, если подстрока «***/**» появляется в **значении** фасета **pattern,** она должна быть заменена символьной строкой «***/**».

Если **значение** фасета **pattern** является объединением совокупности регулярных выражений (общий случай), то определяемое пользователем ограничение не определено (но см. 12.5.4).

12.3 Фасет whiteSpace

12.3.1 Если фасет **whiteSpace** со **значением replace** или **collapse** принадлежит **простому определению типа,** у которого также есть фасет **enumeration,** отображаемый в «Enumeration» АСН.1 (см. 12.4.1 и 12.4.2), то применяются три следующих подпункта.

12.3.1.1 Никакие «EnumerationItem» не должны быть включены в «Enumeration» для элементов (если таковые имеются) **значения** фасета **enumeration,** которые содержат любой из символов ГОРИЗОНТАЛЬНАЯ ТАБУЛЯЦИЯ (HORIZONTAL TABULATION), НОВАЯ СТРОКА (NEWLINE) или ВОЗВРАТ КАРЕТКИ (CARRIAGE RETURN) или (в случае **collapse**) содержат впереди идущие, сзади идущие или многократно следующие друг за другом символы ПРОБЕЛ.

12.3.1.2 Если **значением** фасета **whiteSpace** является **replace** и заключительная команда кодирования **TEXT** с уточняющей информацией присваивается определению типа АСН.1, то заключительная команда кодирования **WHITESPACE REPLACE** должна быть также присвоена ему.

12.3.1.3 Если **значением** фасета **whiteSpace** является **collapse** и заключительная команда кодирования **TEXT** с уточняющей информацией присваивается определению типа АСН.1, то заключительная команда кодирования **WHITESPACE COLLAPSE** должна быть также присвоена ему.

12.3.2 В случае если данное условие не выполняется, применяется максимум один из трех следующих подпунктов.

12.3.2.1 Если **значением** фасета **whiteSpace** является **preserve,** то фасет **whiteSpace** должен быть проигнорирован.

12.3.2.2 Если **значением** фасета **whiteSpace** является **replace** и определение типа АСН.1, соответствующее **простому определению типа,** является типом ограниченной символьной строки АСН.1, то разрешенное ограничение алфавита должно быть добавлено к определению типа АСН.1, чтобы удалить символы

ГОРИЗОНТАЛЬНАЯ ТАБУЛЯЦИЯ, НОВАЯ СТРОКА и ВОЗВРАТ КАРЕТКИ. Заключительная команда кодирования **WHITESPACE REPLACE** должна быть присвоена определению типа АСН.1. Следующее или эквивалентное ограничение алфавита должно быть

```
(FROM ({0, 0, 0, 32} .. {0, 16, 255, 255}))
```

12.3.2.3 Если **значением** facета **whiteSpace** является **collapse** и определение типа АСН.1, соответствующее **простому определению типа,** является типом ограниченной символьной строки АСН.1, то и разрешенное ограничение алфавита, как указано в 12.3.2.2, и ограничение шаблона, которое запрещает впереди идущие, сзади идущие или многократно следующие друг за другом символы ПРОБЕЛ, должны быть добавлены к определению типа АСН.1. Заключительная команда кодирования **WHITESPACE COLLAPSE** должна быть присвоена определению типа АСН.1. Следующее или эквивалентное ограничение шаблона должно быть

```
(PATTERN "([^ ]([^ ]|[^ ])*)?")
```

12.4 Facet enumeration

12.4.1 Facet **enumeration,** принадлежащий **простому определению типа** со **множеством (variety) atomic,** который получается ограничением (прямо или косвенно) **xsd:string,** не должен быть отображен в ограничении АСН.1. Вместо этого facet должен быть отображен в «Enumeration» перечислимого типа АСН.1, соответствующего **простому определению типа** (см. 13.4), как описано в четырех следующих подпунктах.

12.4.1.1 Для каждого элемента **значения** facета **enumeration** «EnumerationItem», который является «identifier», должен быть добавлен к «Enumeration» (с учетом 12.1.2, 12.2.1, 12.3.1 и 12.5.1).

12.4.1.2 Каждый «identifier» должен быть сформирован применением 10.3 к соответствующему элементу **значения** facета **enumeration.**

12.4.1.3 Элементы **значения** facета **enumeration** должны быть отображены в возрастающем лексикографическом порядке, и любые дублирующиеся элементы должны быть отброшены.

12.4.1.4 Если **простое определение типа** имеет facet **whiteSpace** со значением **preserve** или **replace,** то перечислимый тип должен быть присвоен по крайней мере одной заключительной команде кодирования **TEXT** с уточняющей информацией, показывающей один или более «EnumerationItem».

Примечание — Важный пример этого — ограничение **xsd:string** с facetом **enumeration,** которое имеет **whiteSpace preserve** по умолчанию.

12.4.2 Facet **enumeration,** принадлежащий **простому определению типа** со **множеством atomic,** который получается ограничением (прямо или косвенно) **xsd:integer,** не должен быть отображен в ограничении АСН.1. Вместо этого facet должен быть отображен в «Enumeration» перечислимого типа АСН.1, соответствующего **простому определению типа** (см. 13.5), как описано в трех следующих подпунктах.

12.4.2.1 Для каждого элемента **значения** facета **enumeration,** «EnumerationItem», который является «NamedNumber», должен быть добавлен к «Enumeration» (с учетом 12.1.2, 12.2.1, 12.3.1 и 12.5.1).

12.4.2.2 «identifier» в каждом «NamedNumber» должен быть сформирован, связывая символьную строку «**int**» с каноническим лексическим представлением (см. Часть 2, 2.3.1 W3C XML-схемы) соответствующего элемента **значения** facета **enumeration.** «SignedNumber» в «NamedNumber» должно быть нотацией значения АСН.1 для элемента (целое число).

12.4.2.3 Элементы **значения** facета **enumeration** должны быть отображены в возрастающем числовом порядке, и любые дублирующиеся элементы должны быть отброшены.

12.4.3 Любой другой facet **enumeration** должен быть отображен в ограничении АСН.1, которое является либо одиночным значением, либо совокупностью одиночных значений, соответствующих элементам **значения enumeration.**

Примечание — Facet **enumeration** применяется к пространству значений **базового определения типа (base type definition).** Поэтому для **enumeration** таких XSD-встроенных типов как **xsd:QName** или **xsd:NOTATION,** значение компонента **uri [USE-QNAME] ПОСЛЕДОВАТЕЛЬНОСТИ** (**[USE-QNAME] SEQUENCE**), сформированное как одиночное значение ограничения АСН.1, определяется в XML-представлении схемы XSD объявлениями пространства имен, контекст которых включает **xsd:QName** или **xsd:NOTATION,** и префиксом (если таковые имеются) **xsd:QName** или **xsd:NOTATION.**

Пример 1 — Далее представлено высокоуровневое простое определение типа, *которое является ограничением* xsd:string *с фасетом* enumeration:

```
<xsd:simpleType name="state">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="off"/>
        <xsd:enumeration value="on"/>
    </xsd:restriction>
</xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
State ::= [NAME AS UNCAPITALIZED] ENUMERATED {off, on}
```

Пример 2 — Далее представлено высокоуровневое простое *определение типа, которое является ограничением* xsd:integer *с фасетом* enumeration:

```
<xsd:simpleType name="integer-0-5-10">
    <xsd:restriction base="xsd:integer">
        <xsd:enumeration value="0"/>
        <xsd:enumeration value="5"/>
        <xsd:enumeration value="10"/>
    </xsd:restriction>
</xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
Integer-0-5-10 ::= [NAME AS UNCAPITALIZED] [USE-NUMBER] ENUMERATED {int0(0),
int5(5), int10(10)}
```

Пример 3 — Далее представлено высокоуровневое простое определение типа, *которое является ограничением* xsd:integer *с фасетом* minInclusive *и* maxInclusive:

```
<xsd:simpleType name="integer-1-10">
    <xsd:restriction base="xsd:integer">
        <xsd:minInclusive value="1"/>
        <xsd:maxInclusive value="10"/>
    </xsd:restriction>
</xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
Integer-1-10 ::= [NAME AS UNCAPITALIZED] INTEGER(1..10)
```

Пример 4 — Далее представлено высокоуровневое простое определение типа, *которое является ограничением (с фасетом* minExclusive) *другого* простого определения типа, *полученного ограничением* xsd:integer *с добавлением фасета* minInclusive *и* maxInclusive:

```
<xsd:simpleType name="multiple-of-4">
    <xsd:restriction>
        <xsd:simpleType>
            <xsd:restriction base="xsd:integer">
                <xsd:minInclusive value="1"/>
                <xsd:maxInclusive value="10"/>
            </xsd:restriction>
        </xsd:simpleType>
        <xsd:minExclusive value="5"/>
    </xsd:restriction>
</xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
Multiple-of-4 ::= [NAME AS UNCAPITALIZED] INTEGER(5<..10)
```

Пример 5 — Далее представлено высокоуровневое простое определение типа, *которое является ограничением (с фасетом* minLength и maxLength) *другого* простого определения типа, *полученного ограничением* xsd:string *с добавлением фасета* enumeration:

```
<xsd:simpleType name="color">
    <xsd:restriction>
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="white"/>
                <xsd:enumeration value="black"/>
                <xsd:enumeration value="red"/>
            </xsd:restriction>
        </xsd:simpleType>
        <xsd:minLength value="2"/>
```

```
                <xsd:maxLength value="4"/>
            </xsd:restriction>
        </xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
Color ::= [NAME AS UNCAPITALIZED] ENUMERATED {red}
```

12.5 Другие фасеты

12.5.1 Если фасет **totalDigits**, **fractionDigits**, **maxInclusive**, **maxExclusive**, **minExclusive** или **minInclusive** принадлежит **простому определению типа**, у которого также есть фасет **enumeration**, отображаемый в «Enumeration» ACH.1 (см. 12.4.1 и 12.4.2), то никакие «EnumerationItem» не должны быть включены в «Enumeration» для элементов (если таковые имеются) **значения** фасета **enumeration**, которые не удовлетворяют фасет **totalDigits, fractionDigits, maxInclusive, maxExclusive, minExclusive** или **minInclusive.**

12.5.2 Если фасет **maxInclusive, maxExclusive, minExclusive** или **minInclusive** принадлежит **простому определению типа** без фасета **enumeration** или с фасетом **enumeration,** который не отображается в «Enumeration» ACH.1 (см. 12.4.1 и 12.4.2), то применяется один из двух следующих подпунктов.

12.5.2.1 Если **простое определение типа** получено ограничением (прямым или косвенным) XSD-встроенной даты или типа времени (**xsd:date, xsd:dateTime, xsd:duration, xsd:gDay, xsd:gMonth, xsd:gYear, xsd:gYearMonth, xsd:gMonthDay** или **xsd:time**), то фасеты **maxInclusive, maxExclusive, minExclusive** и **minInclusive простого определения типа** должны быть отображены в определяемое пользователем ограничение ACH.1 (см. 12.5.4).

12.5.2.2 Иначе фасеты **maxInclusive, maxExclusive, minExclusive** и **minInclusive простого определения типа** должны быть отображены в диапазон значений ACH.1 или в одиночное ограничение значения в соответствии с таблицей 4.

Таблица 4 — Ограничения ACH.1, соответствующие фасетам **maxInclusive, maxExclusive, minExclusive** и **minInclusive**

XSD фасет	Ограничение ACH.1
maxInclusive=*ub*	(**MIN** .. *ub*)
maxExclusive=*ub*	(**MIN** .. < *ub*)
minExclusive=*lb*	(*lb* <.. **MAX**)
minInclusive=*lb*	(*lb* .. **MAX**)
minInclusive=*ub* **maxInclusive=***lb*	(*lb* .. *ub*)
minInclusive=*v* **maxInclusive=***v*	(*v*)
minInclusive=*ub* **maxExclusive=***lb*	(*lb* ..< *ub*)
minExclusive=*ub* **maxInclusive=***lb*	(*lb* <.. *ub*)
minExclusive=*ub* **maxExclusive=***lb*	(*lb* <..< *ub*)

12.5.3 Если фасет **totalDigits** или **fractionDigits** принадлежит **простому определению типа** без фасета **enumeration** или с фасетом **enumeration,** который не отображается в «Enumeration» ACH.1 (см. 12.4.1 и 12.4.2), то фасеты **totalDigits** и **fractionDigits простого определения типа** должны быть отображены в определяемое пользователем ограничение (см. 12.5.4).

12.5.4 Когда фасет отображается в определяемое пользователем ограничение ACH.1, рекомендуется, чтобы фасет и его **значение** появились в комментарии ACH.1 в определяемом пользователем ограничении. Точная форма определяемого пользователем ограничения не устанавливается.

13 Отображение простых определений типа

13.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1 или определение типа ACH.1, соответствующее **простому определению типа.**

Примечание — Этот раздел не запрашивается для **простых определений типа,** которые являются XSD-встроенными типами.

13.2 Высокоуровневое **простое определение типа** должно быть отображено в присвоение типа ACH.1. «typereference» в «TypeAssignment» должно быть сформировано применением 10.3 к **имени простого определения типа,** и «Type» в «TypeAssignment» должно быть определением типа ACH.1, как определено в 13.4—13.9.

13.3 Анонимное **простое определение типа** должно быть отображено в определение типа ACH.1, как указано в 13.4—13.9.

13.4 Для **простого определения типа** со **множеством atomic** с фасетом **enumeration,** который получается ограничением (прямо или косвенно) **xsd:string,** определение типа ACH.1 должно быть перечислимым типом ACH.1, «Enumeration» которого должно быть сформировано в соответствии с 12.4.1.

13.5 Для **простого определения типа** со **множеством atomic** с фасетом **enumeration,** который получается ограничением (прямо или косвенно) **xsd:integer,** определение типа ACH.1 должно быть перечислимым типом ACH.1, «Enumeration» которого должно быть сформировано в соответствии с 12.4.2. Заключительная команда кодирования `USE-NUMBER` должна быть присвоена перечислимому типу ACH.1.

13.6 Для любого другого **простого определения типа** (D, например) с любым **множеством,** которое получено ограничением (прямо или косвенно) высокоуровневого **простого определения типа,** определение типа ACH.1 должно быть сформировано применением 23 к высокоуровневому **простому определению типа** (B, например) так, что:

a) D получается ограничением (прямо или косвенно) от B;

b) либо B является **базовым определением типа** D, либо все промежуточные шаги вывода от B до D — анонимные **простые определения типа.**

Затем для каждого из **фасетов** D (если такие есть) ограничение ACH.1, сформированное в соответствии с 12 применительно к фасету, должно быть добавлено к определению типа ACH.1.

13.7 Для любого другого **простого определения типа** (D, например) со **множеством atomic** определение типа ACH.1 должно быть сформировано с помощью 23 применительно к встроенному типу XSD (B, например) так, что:

a) D получается отграничением (прямо или косвенно) от B;

b) либо B является **базовым определением типа** D, либо все промежуточные шаги вывода от B до D — анонимные **простые определения типа.**

Затем для каждого из **фасетов** D ограничение ACH.1, сформированное применением 12 к фасету, должно быть добавлено к определению типа ACH.1.

13.8 Для любого другого **простого определения типа** (D, например) со **множеством list** применяют пять следующих пунктов.

13.8.1 Определение типа ACH.1 должно быть последовательностью типа ACH.1, компонент которого должен быть «Type», сформированный применением раздела 23 к **объектному определению типа (item type definition).**

13.8.2 Для каждого **фасета** D ограничение ACH.1, сформированное применением 12 к фасету, должно быть добавлено к последовательности типа ACH.1.

13.8.3 Если **объектное определение типа** списка является **xsd:string** или ограничением **xsd:string** и отображается в тип символьной строки ACH.1, то ограничение разрешенного алфавита `(FROM((0, 0, 0, 33) .. (0, 16, 255, 253)))` должно быть применено к типу символьной строки ACH.1.

13.8.4 Если **объектное определение типа** списка является типом объединения, то ограничение подтипа, описанное в 13.8.3, должно быть применено к каждой альтернативе типа выбора ACH.1, который является типом символьной строки, путем использования внутреннего ограничения подтипа, примененного к типу выбора.

13.8.5 Заключительная инструкция кодирования `LIST` должна быть присвоена типу последовательности ACH.1.

***Пример — Далее представлено высокоуровневое* простое определение типа, *которое является* list xsd:float:**

```
<xsd:simpleType name="list-of-float">
    <xsd:list itemType="xsd:float"/>
</xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
List-of-float ::= [LIST] [NAME AS UNCAPITALIZED] SEQUENCE OF XSD.Float
```

13.9 Для любого другого **простого определения типа** (D, скажем) со **множеством union,** применяются пять следующих пунктов.

13.9.1 Определение типа ACH.1 должно быть типом выбора ACH.1 с одной альтернативой для каждого элемента **элементных определений типа (member type definitions).**

13.9.2 Для каждого элемента **элементных определений типа** «identifier» в «NamedType» соответствующей альтернативы должен быть получен применением 10.3 либо к **имени** элемента (если элементом является встроенный тип XSD или высокоуровневое **простое определение типа**), либо к символьной строке «**alt**» (если элемент является анонимным **простым определением типа**), и «Type» в «NamedType» должно быть определением типа ACH.1, полученным применением раздела 23 к элементу **элементных определений типа.**

13.9.3 Для каждого элемента **элементных определений типа,** который является анонимным **простым определением типа,** соответствующее «NamedType» должно иметь заключительную команду кодирования `NAME AS ""`.

13.9.4 Для каждого из **фасетов** D ограничение ACH.1, сформированное применением раздела 12 к фасету, должно быть добавлено к типу выбора ACH.1.

13.9.5 Заключительная команда кодирования `USE-UNION` должна быть присвоена типу выбора ACH.1.

***Пример — Далее представлено высокоуровневое* простое определение типа, *которое является union двух анонимных* простых определений типа:**

```
<xsd:simpleType name="decimalOrBinary">
    <xsd:union>
        <xsd:simpleType>
            <xsd:restriction base="xsd:decimal"/>
        </xsd:simpleType>
        <xsd:simpleType>
            <xsd:restriction base="xsd:float"/>
        </xsd:simpleType>
    </xsd:union>
</xsd:simpleType>
```

Это отображается в присвоении типа ACH.1:

```
DecimalOrBinary ::= [NAME AS UNCAPITALIZED] [USE-UNION] CHOICE {
        alt [NAME AS ""] XSD.Decimal,
        alt-1 [NAME AS ""] XSD.Float }
```

14 Отображение объявлений элементов

14.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1 или определение типа ACH.1, соответствующее **объявлению элемента.**

Примечание — Присутствие **ограничения значения** в **объявлении элемента** обычно влияет на отображение. Однако 8.10 подразумевает, что **объявление элемента,** у которого есть **ограничение значения** и чьим **определением типа** являются **xsd:QName** или **xsd:NOTATION,** или ограничение их XSD-встроенных типов отображается так, как если бы у него не было никакого **ограничения значения.**

14.2 Высокоуровневое **объявление элемента,** которое является **абстрактным,** должно быть проигнорировано.

14.3 Высокоуровневое **объявление элемента,** которое не является **абстрактным,** должно быть отображено в присвоении типа ACH.1. «typereference» в «TypeAssignment» должно быть получено применением 10.3 к **имени объявления элемента,** и «Type» в «TypeAssignment» должно быть определением типа ACH.1, как указано в 14.5.

14.4 Местное **объявление элемента** должно быть отображено в определении типа ACH.1, как указано в 14.5.

14.5 Определение типа ACH.1 должно быть получено либо применением разделов 23, 26 или 27 (см. 14.6) к **простому** или **сложному определению типа,** которое является **определением типа объявления элемента,** либо применением 10.2 к присвоению типа ACH.1, полученному применением раздела 29 к **определению типа.** В обоих случаях **ограничение значения** в **объявлении элемента** (если таковые имеются) должны быть указаны в применяемом разделе (23, 26, 27 или 29) и должны быть использованы при формировании определения типа ACH.1, как указано в данном подразделе.

14.6 Номер применяемого раздела должен быть получен по последней колонке таблицы 5 после выбора строки таблицы, основанного на следующих условиях:

a) имеет ли **объявление элемента** заменимое или незаменимое **определение типа** (см. 14.7);

b) является ли **объявление элемента обнуляемым** или **не обнуляемым;**

c) является ли **определение типа простым определением типа** или **сложным определением типа;**

d) является ли **определение типа** встроенным в XSD, анонимным или высокоуровневым определением типа.

Таблица 5 — Номера применяемых пунктов для отображения объявлений элементов

Заменяемое	обнуляемое	простое / сложное	определение типа	Применяемый раздел
нет	нет	**простое** или **сложное**	встроенное в XSD, анонимное или высокоуровневое	23
нет	да	**простое**	встроенное в XSD или анонимное	26
нет	да	**простое**	высокоуровневое	29
нет	да	**сложное**	встроенное в XSD или анонимное	27
нет	да	**сложное**	высокоуровневое	29
да	да или нет	**простое** или **сложное**	встроенное в XSD, анонимное или высокоуровневое	29

14.7 Фраза «имеет заменяемое **определения типа**» применительно к **объявлению элемента** означает, что **определение типа объявления элемента** является высокоуровневым **простым определением типа** или **сложным определением типа,** которое выступает в качестве **базового определения типа** другого высокоуровневого **простого определения типа** или **сложного определения типа.**

Примечание — Согласно этому определению, **объявления элементов, определение типа** которых является встроенным типом XSD **xsd:anyType,** не имеют заменяемого **определения типа.**

15 Отображение объявлений атрибутов

15.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1 или определение типа ACH.1, соответствующее **объявлению атрибута.**

15.2 Высокоуровневое **объявление атрибута** должно быть отображено в присвоении типа ACH.1. «Typereference» в «TypeAssignment» должно быть сформировано применением 10.3 к **имени объявления атрибута,** и «Type» в «TypeAssignment» должно быть определением типа ACH.1, как указано в 15.4. Заключительная команда кодирования **ATTRIBUTE** должна быть назначена присвоению типа ACH.1.

15.3 Местное **объявление атрибута** должно быть отображено в определение типа ACH.1, как указано в 15.4.

15.4 Определение типа ACH.1 должно быть сформировано применением 23 к **определению типа объявления атрибута.**

16 Отображение значений простых определений типа

16.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать «Value» ACH.1, соответствующее значению в пространстве значений **простого определения типа.**

16.2 Учитывая значение V в пространстве значений **простого определения типа** и:

a) определение типа ACH.1, отображенное из этого **простого определения типа;** и

b) каноническое лексическое представление V (см. W3C XML схема Часть 2, п. 2.3.1),

V должно быть отображено в базовую нотацию значения ACH.1 для абстрактного значения определения типа ACH.1, для которого в EXTENDED-XER каноническое лексическое представление является действующим кодированием «ExtendedXMLValue».

17 Отображение определений модельной группы

17.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1, соответствующее **определению модельной группы.**

17.2 **Определение модельной группы, модельная группа** которого имеет **наборщика последовательности** или **выбора,** должно быть отображено в присвоении типа ACH.1. «Typereference» в «TypeAssignment» должно быть сформировано применением 10.3 к **имени определения модельной группы,** и «Type» в «TypeAssignment» должно быть сформировано применением 18 к **модельной группе определения модельной группы.**

Примечание — **Определения модельной группы, модельная группа** которых имеет **наборщика всего,** не отображаются в ACH.1.

18 Отображение модельных групп

18.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее **модельной группе.**

Примечание — Этот раздел не запрашивается для каждой **модельной группы.** Например, **модельная группа** с **наборщиком всего** не отображается в ACH.1, но ее **частицы** отображаются, как указано в 20.9.

18.2 **Модельная группа** с **наборщиком последовательности** должна быть отображена в тип последовательности ACH.1. Для каждой **частицы** в **модельной группе** по порядку упорядоченный список из нуля или более ACH.1 «NamedType» должен быть получен применением раздела 19 к **частице,** и эти «NamedType» должны быть добавлены к последовательности типа в том же порядке. Окончательная команда кодирования `UNTAGGED` должна быть присвоена типу последовательности.

18.3 **Модельная группа** с **наборщиком выбора,** имеющая, по меньшей мере, одну **частицу,** должна быть отображена в тип выбора ACH.1. Для каждой **частицы** в **модельной группе** по порядку, «NamedType» должно быть получено применением раздела 19 к **частице,** и это «NamedType» должно быть добавлено к типу выбора в качестве одной из альтернатив. Окончательная команда кодирования `UNTAGGED` должна быть присвоена типу выбора.

18.4 **Модельная группа** с **наборщиком выбора,** не имеющая **частиц,** должна быть отображена во встроенный тип ACH.1 **NULL.**

19 Отображение частиц

19.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать упорядоченный список из нуля или более «NamedType» ACH.1, соответствующих **частице.**

Примечания

1 Этот раздел не запрашивается для всех **частиц.** Например, (самая важная) **частица типа содержимого (content type) сложного определения типа** отображается особым образом, если ее **термом (term)** является **модельная группа** с **наборщиком последовательности** или **всего** (см. 20.8).

2 В большинстве случаев этот пункт формирует один «NamedType». Он может сформировать ноль «NamedType» или два и более «NamedType» только тогда, когда частица модельной группы последовательности содержит другую частицу модельной группы последовательности с **min occurs** и **max occurs,** равными одному, в этом случае частицы внутренней модельной группы последовательности отображаются в ACH.1 как если бы они были частицами внешней модельной группы последовательности.

19.2 Следующие три пункта определяют термины, которые используются в оставшейся части раздела 19.

19.2.1 Если и **min occurs**, и **max occurs частицы** равно единице, то:

а) если **термом частицы** является **модельная группа** с **наборщиком последовательности,** не связанным с **определением модельной группы** и сама **частица** принадлежит **модельной группе** с

наборщиком последовательности, частицу называют «частицей бессмысленной последовательности (pointless sequence particle)»;

б) иначе **частицу** называют «частицей обязательного присутствия (mandatory presence particle)».

19.2.2 Если **min occurs** равно нулю, а **max occurs** равно единице, то:

а) если отображение **частицы** существует для формирования компонента типа последовательности ACH.1, **частица** называется «частицей необязательного присутствия (optional presence particle)»;

б) иначе **частица** называется «частицей необязательного одиночного появления (optional single-occurrence particle)».

19.2.3 Если **max occurs** равно двум и более, **частица** называется «частицей многократного появления (multiple-occurrence particle)».

19.3 «Частица бессмысленной последовательности» должна быть отображена в упорядоченный список (L, например) из нуля или более «NamedType» следующим образом. Список L должен быть изначально пустой. Для каждой **частицы** (P, например) в **модельной группе**, которая является **термом частицы** по порядку, упорядоченный список из нуля и более «NamedType» должен быть получен рекурсивным применением раздела 19 к **частице** P, и эти «NamedType» должны быть добавлены в список L в том же порядке.

19.4 «Частица обязательного присутствия» или «частица необязательного присутствия» должна быть отображена в «NamedType», как указано далее.

19.4.1 «identifier» в «NamedType» должен быть сформирован применением 10.3 к символьной строке, указанной в 19.6, и «Type» в «NamedType» должен быть сформирован с применением 19.7 к **терму частицы.**

19.4.2 Если **частица** является «частицей необязательного присутствия», то «NamedType» должен следовать за ключевым словом `OPTIONAL`.

19.5 «Частица необязательного одиночного появления» или «Частица многократного появления» должна быть отображена в «NamedType», как указано далее.

19.5.1 «identifier» в «NamedType» должен быть сформирован с применением 10.3 к символьной строке, полученной добавлением суффикса «`-list`» к символьной строке, указанной в 19.6. «Type» в «NamedType» должен быть типом последовательности.

19.5.2 Не считая того, когда **min occurs** равно нулю и **max occurs неограниченно (unbounded),** ограничение размера должно быть добавлено к типу последовательности в соответствии с таблицей 6.

Таблица 6 — ACH.1 ограничение размера соответствующее **min occurs** и **max occurs**

min occurs и **max occurs**	Ограничение размера ACH 1
min occurs = *n* **max occurs** = *n* *n* ≥ 2	`SIZE` (*n*)
min occurs = *min* **max occurs** = *max* *max* > *min* и *max* ≥ 2	`SIZE` (*min* .. *max*)
min occurs = 0 max occurs = 1	`SIZE` (0 .. 1)
min occurs = *min* **max occurs** = **неограниченно** *min* ≥ 1	`SIZE` (*min* .. `MAX`)

19.5.3 Если **термом частицы** является **объявление элемента,** то компонент типа последовательности должен быть «NamedType». «identifier» в этом «NamedType» должен быть сформирован с применением 10.3 к **имени объявления элемента,** и «Type» в этом «NamedType» должно быть сформирован с применением 19.7 к **терму частицы.**

19.5.4 Если **термом частицы** является **групповой символ,** то компонент типа последовательности должен быть «NamedType». «identifier» в этом «NamedType» должен быть `elem`, и «Type» в этом «NamedType» должен быть сформирован с применением 19.7 к **терму частицы.**

19.5.5 Если **термом частицы** является **модельная группа,** то компонент типа последовательности должен быть «Type» и должен быть сформирован с применением 19.7 к **терму частицы.**

19.5.6 Заключительная команда кодирования `UNTAGGED` должна быть присвоена к типу последовательности.

19.6 Символьная строка, используемая в формировании «identifier» в «NamedType», соответствующего **частице,** должна быть:

а) **именем объявления элемента,** если **термом частицы** является **объявление элемента;**

б) **именем определения модельной группы,** если **термом частицы** является **модельная группа определения модельной группы;**

в) символьной строкой «**sequence**», если **термом частицы** является **модельная группа** с **наборщиком последовательности,** не связанным с **определением модельной группы;**

г) символьной строкой «**choice**», если **термом частицы** является **модельная группа** с **наборщиком выбора,** не связанным с **определением модельной группы;**

е) символьной строкой «**elem**», если **термом частицы** является **групповой символ.**

19.7 «Type» в «NamedType», соответствующее **частице** (см. 19.4) или «Type» в «NamedType» в «SequenceOfType», соответствующее **частице** (см. 19.5), должно быть:

а) если **термом частицы** является высокоуровневое **объявление элемента,** которое возглавляет группу замены элементов, содержащих только сам головной элемент, определение типа ACH.1 («DefinedType») формируется путем применения 10.2 к присвоению типа ACH.1, полученному применением раздела 14 к **объявлению элемента;**

Примечание — Это включает частый случай, когда нет **объявления элемента,** которое ссылается на это **объявление элемента** как на его **принадлежность группе замещения (substitution group affiliation).**

б) если **термом частицы** является высокоуровневое **объявление элемента,** которое возглавляет группу замены элементов, содержащих, по меньшей мере, один элемент, кроме головного элемента, определение типа ACH.1 («DefinedType») формируется применением 10.2 к присвоению типа ACH.1, полученному применением раздела 31 к **объявлению элемента;**

Примечание — Если головной элемент является **объявлением элемента,** которое является **абстрактным,** оно само по себе не выступает элементом группы замены. Если в этом случае группа замены имеет, по крайней мере, один элемент, то этот объект (b) применяется, и если число элементов ровно один, то группа замены будет отображена в выборе ACH.1 с одной альтернативой.

в) если **термом частицы** является **абстрактное** высокоуровневое **объявление элемента,** которое возглавляет пустую группу замена элементов, встроенным типом ACH.1 **NULL;**

г) если **термом частицы** является местное **объявление элемента,** определение типа ACH.1 формируется применением раздела 14 к **объявлению элемента;**

д) если **термом частицы** является **модельная группа определения модельной группы,** определение типа ACH.1 («DefinedType») формируется применением 10.2 к присвоению типа ACH.1, полученному применением раздела 17 к **определению модельной группы;**

е) если **термом частицы** является **модельная группа,** не связанная с **определением модельной группы,** определение типа ACH.1 формируется применением раздела 18 к **модельной группе;**

Примечание — Это включает случай, когда **определение модельной группы** внутри **переопределения (redefine)** содержит ссылку на самого себя. **Модельная группа** исходного **определения модельной группы,** скопированная в новую схему, рассматривается здесь без привязки к **определению модельной группы** потому, что исходное **определение модельной группы** само не скопировано в новую схему (новое **определение модельной группы** будет иметь другую **модельную группу,** которая будет содержать копию исходной **модельной группы**).

г) если **термом частицы** является **групповой символ,** определение типа ACH.1 формируется применением раздела 21 к **групповому символу.**

20 Отображение сложных определений типа

20.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1 или определение типа ACH.1, соответствующее **сложному определению типа.**

Примечание — Этот раздел не запрашивается для **сложных определений типа,** которые являются встроенными типами XSD.

20.2 Высокоуровневое **сложное определение типа** должно быть отображено в присвоении типа ACH.1. «Typereference» в «TypeAssignment» должно быть сформировано с применением 10.3 к **имени сложного определения типа,** и «Type» в «TypeAssignment» должно быть определением типа ACH.1, как указано в 20.4—20.11.

20.3 Анонимное **сложное определение типа** должно быть отображено в определении типа ACH.1, как указано в 20.4—20.11.

20.4 Определение типа ACH.1 должно быть последовательностью типа ACH.1, и ноль и более компонентов должны быть добавлены к нему, как описано в 20.5—20.11 в указанном порядке.

20.5 Если **тип содержимого сложного определения типа** представляет собой **смешанную (mixed)** модель содержимого, то компонент должен быть добавлен к типу последовательности ACH.1. «identifier» в «NamedType» этого компонента должен быть **`embed-values`**, и «Type» в «NamedType» должен быть типом последовательности, компонентом которой должен быть «Type», полученный применением раздела 23 к XSD-встроенному типу **xsd:string.** Заключительная команда кодирования **`EMBED-VALUES`** должна быть присвоена к типу последовательности ACH.1.

20.6 Если **тип содержимого сложного определения типа** является **частицей, термом** которой выступает **модельная группа** с **наборщиком всего,** то компонент должен быть добавлен к типу последовательности ACH.1. «identifier» в «NamedType» компонента должен быть **`order`**, и «Type» в «NamedType» должен быть типом последовательности, компонентом которой должен быть «EnumeratedType». Для каждой **частицы модельной группы** (**термом** которой всегда является **объявление элемента**), «EnumerationItem», т.е. «identifier» идентичный «identifier» в «NamedType», соответствующей каждой **частице,** должно быть добавлено к «Enumeration» по порядку. Заключительная команда кодирования **`USE-ORDER`** должна быть присвоена типу последовательности ACH.1.

Примечание — «identifier» в «NamedType», отображаемых из **частиц,** формируется (применение 10.3) как добавление каждого компонента к типу последовательности. Таким образом, даже если компонент **`order`** находится в положении, которое дословно предшествует положениям этих компонентов в типе последовательности ACH.1, формирование компонента **`order`** может быть завершено только после того как все **частицы** были отображены в последовательность компонентов.

20.7 Если **сложное определение типа** имеет **применения атрибута,** то компоненты, полученные применением раздела 22 к **применениям атрибута,** должны быть добавлены к типу последовательности ACH.1 в порядке, основанном на **целевом пространстве имен** и **имени объявления атрибута** каждого **применения атрибута. Применения атрибута** должны быть предварительно отсортированы по **целевому пространству имен объявления атрибута** (с **отсутствующим** ключевым словом, предшествующим всем именам пространства имен, упорядоченным по возрастанию в лексикографическом порядке), а затем по **имени объявления атрибута** внутри каждого **целевого пространства имен** (также в возрастающем лексикографическом порядке).

20.8 Если **сложное определение типа** имеет **атрибут групповой символ,** то компонент, полученный от **атрибута групповой символ,** как указано в разделе 21, должен быть добавлен к типу последовательности ACH.1.

20.9 Если **тип содержимого сложного определения типа** является **частицей,** то применяется один из четырех следующих пунктов.

20.9.1 Если **термом частицы** является **модельная группа** с **наборщиком последовательности,** чьи **min occurs** и **max occurs** являются оба единицей, то для каждой **частицы модельной группы** по порядку упорядоченный список из нуля и более «NamedType» ACH.1 должен быть получен применением раздела 19 к **частице** в **модельной группе,** и эти «NamedType» должны быть добавлены к типу последовательности ACH.1 в том же порядке.

20.9.2 Если **термом частицы** является **модельная группа** с **наборщиком последовательности,** чьи **min occurs** и **max occurs** не являются оба единицей, то компонент, полученный применением раздела 19 к **частице** в **типе содержимого,** должен быть добавлен к типу последовательности ACH.1.

20.9.3 Если **термом частицы** является **модельная группа** с **наборщиком всего,** то для каждой **частицы модельной группы** по порядку, компонент, полученный применением раздела 19 к **частице модельной группе,** должен быть добавлен к типу последовательности ACH.1. Если **частица** в **типе содержимого сложного определения типа** имеет **min occurs** ноль, каждая из **частиц модельной группы** с **min occurs** равным единице, должна быть отображена как если бы она имела **min occurs** ноль.

20.9.4 Если **термом частицы** является **модельная группа** с **наборщиком выбора,** то компонент, полученный применением раздела 19 к **частице** в **типе содержимого,** должен быть добавлен к типу последовательности ACH.1.

20.10 Если **тип содержимого сложного определения типа** является **простым определением типа,** то компонент должен быть добавлен к типу последовательности ACH.1. «identifier» в «NamedType» компонента должен быть сформирован с применением 10.3 к символьной строке «**`base`**», и «Type» в «NamedType»

должно быть определением типа АСН.1, полученным применением раздела 23 к **типу содержимого.** Заключительная команда кодирования `UNTAGGED` должна быть присвоена к компоненту.

20.11 Если **тип содержимого сложного определения типа пуст,** то никакие дополнительные компоненты к типу последовательности АСН.1 не добавляются.

21 Отображение групповых символов

21.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа АСН.1 или «NamedType», соответствующий **групповому символу.**

21.2 Для отображения версии 1 применяется 21.3. Для отображения версии 2 применяется 21.4.

21.3 **Групповой символ,** который является **термом частицы,** должен быть отображен в определении типа АСН.1, сформированном применением раздела 23 к встроенному типу XSD **xsd:string.** Заключительная команда кодирования `ANY-ELEMENT` должна быть присвоена определению типа АСН.1.

21.4 **Групповой символ,** который является **термом частицы,** должен быть отображен как указано в 21.4.1—21.4.7.

21.4.1 Фраза «атрибут отображения группового символа» (используется только в данном пункте) обозначает объект информации атрибута со свойством **[namespace name]** «`urn:oid:2.1.5.2.0.1`» и свойством **[local name]** «**wildcard-mapping**», который является элементом **атрибутов** присутствия **примечания** в **групповом символе.** Фраза «значение атрибута отображения группового символа» (используется только в данном пункте) обозначает свойство **[normalized value]** атрибута отображения группового символа.

Примечание — Имя пространства имен, указанное в данном подпункте, является именем пространства имен АСН.1, которое определено в Рекомендации МСЭ-Т X.693 ИСО/МЭК 8825-4:2008 (п. 16.9).

21.4.2 Атрибут отображения группового символа должен иметь одно из следующих значений: **CHOICE-FI, CHOICE-UTF-8, FI** или **UTF-8.**

Пример — Далее приведен пример атрибута отображения группового символа:

```
<xsd:any>
        <xsd:annotation
                a:wildcard-mapping="FI"
                xmlns:a="urn:oid:2.1.5.2.0.1"/>
</xsd:any>
```

21.4.3 **Групповой символ** без атрибута отображения группового символа должен рассматриваться, как будто у него есть атрибут отображения группового символа со значением **CHOICE-FI** (если **process contents** является **strict** или **lax**) или **FI** (если **process contents** является **skip**).

21.4.4 **Групповой символ, process contents** которого является **skip,** не должен иметь атрибута отображения группового символа со значением **CHOICE-FI** или **CHOICE-UTF-8.**

21.4.5 **Групповой символ,** атрибут отображения группового символа которого имеет значение **UTF-8,** должен быть отображен в АСН.1 встроенном типе `UTF8String` со следующим определяемым пользователем ограничением:

```
(CONSTRAINED BY
      {/* Every character string abstract value shall
      be a well-formed XML document encoded in UTF-8. */})
```

и с заключительной командой кодирования `ANY-ELEMENT`.

21.4.6 **Групповой символ,** атрибут отображения группового символа которого имеет значение **FI,** должен быть отображен в АСН.1-встроенном типе `OCTET STRING` со следующим определяемым пользователем ограничением:

```
(CONSTRAINED BY
      {/* Every octet string abstract value shall be a
      well-formed fast infoset document (see ITU-T
      Rec. X.891 | ISO/IEC 24824-1). */})
```

и с заключительной командой кодирования `ANY-ELEMENT`.

21.4.7 **Групповой символ,** атрибут отображения группового символа которого имеет значение **CHOICE-FI** или **CHOICE-UTF-8,** должен быть отображен в типе выбора АСН.1, сформированном следующим образом:

а) одна альтернатива должна быть добавлена к типу выбора для каждого высокоуровневого **объявления элемента** в исходной схеме XSD, которое не является **абстрактным** и **целевое пространство имен** которого является именем пространства имен (или **отсутствующим** пространством имен), разрешенным **ограничением пространства имен (namespace constraint) группового символа;**

б) для каждой альтернативы «identifier» в «NamedType» должен быть сформирован применением 10.3 к **имени** высокоуровневого **объявления элемента,** соответствующего альтернативе, и «Type» в «NamedType» должно быть определением типа АСН («DefinedType»), сформированным применением 10.2 к присвоению типа АСН.1, полученному применением раздела 14 к высокоуровневому **объявлению элемента;**

в) эти альтернативы должны быть добавлены к типу выбора в порядке, основанном на **целевом пространстве имен** и **имени** высокоуровневого **объявления элемента; объявления элемента** должны сначала быть отсортированы по **целевому пространству имен** (с **отсутствующим** пространством имен, предшествующим всем именам пространства имен, отсортированным в лексикографическом порядке возрастания), а затем по **имени** (также в лексикографическом порядке возрастания) внутри каждого **целевого пространства имен;**

г) если атрибут отображения группового символа имеет значение **CHOICE-UTF-8,** другая альтернатива должна быть добавлена в конец типа выбора; «identifier» в «NamedType» должен быть сформирован с применением 10.3 к символьной строке «**elem**» и «Type» в «NamedType» должно быть типом АСН.1, указанным в 21.4.5;

д) если атрибут отображения группового символа имеет значение **CHOICE-FI,** другая альтернатива должна быть добавлена в конец типа выбора; «identifier» в «NamedType» должен быть сформирован применением 10.3 к символьной строке «**elem**», и «Type» в «NamedType» должно быть типом АСН.1, указанным в 21.4.6;

е) если **process contents** является **strict,** то следующее определяемое пользователем ограничение должно применяться к типу выбора:

```
(CONSTRAINED BY
      {/* The last alternative shall be used if and
      only if xsi:type is present*/})
```

ж) если **process contents** является **lax,** то следующее определяемое пользователем ограничение должно применяться к типу выбора:

```
(CONSTRAINED BY
      {/* The last alternative shall be used when
      xsi:type is present, and shall not be used
      when xsi:type is not present and one of the
      other alternatives can be used. */})
```

з) заключительная команда кодирования `UNTAGGED` должна быть присвоена типу выбора.

21.5 **Групповой символ,** который является **атрибутом группового символа сложного типа** должен быть отображен в «NamedType». «identifier» в «NamedType» должен быть сформирован применением 10.3 к символьной строке «`attr`», и «Type» в «NamedType» должно быть типом последовательности. Компонент типа последовательности должен быть «Type», полученным применением раздела 23 к XSD-встроенному типу **xsd:string.** Следующее определяемое пользователем ограничение применяется к типу последовательности:

```
(CONSTRAINED BY
      {/* Each item shall conform to the
      «AnyAttributeFormat» specified in ITU-T Rec.
      X.693 |ISO/IEC 8825-4, clause 18 */})
```

Заключительная команда кодирования **ANY-ATTRIBUTES** должна быть присвоена типу последовательности.

21.6 Если **групповой символ** имеет **ограничение пространства имен,** это должно быть отображено в «NameSpaceRestriction» в командах кодирования `ANY-ELEMENT` или `ANY-ATTRIBUTES`.

22 Отображение применений атрибутов

22.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать «NamedType» АСН.1, соответствующий **применению атрибута.**

22.2 **Применение атрибута** должно быть отображено в «NamedType».

22.3 «identifier» в «NamedType» должен быть сформирован применением 10.3 к **имени объявления атрибута применения атрибута**, а «Type» в «NamedType» должно быть:

а) если **применение атрибута** имеет высокоуровневое **объявление атрибута,** определение типа ACH.1 («DefinedType») получается применением 10.2 к присвоению типа ACH.1, полученному применением раздела 15 к **объявлению атрибута;**

б) если **применение атрибута** имеет местное **объявление атрибута**, определение типа ACH.1 получается применением раздела 15 к **объявлению атрибута.**

22.4 Если либо **применение атрибута,** либо его **объявление атрибута** имеет **ограничение значения** и **применение атрибута** не **требуется,** «NamedType» должен следовать за ключевым словом **DEFAULT**, и за «Value», полученным применением раздела 16 либо к значению в **ограничении значения применения атрибута** (если **применение атрибута** имеет **ограничение значения**), либо к значению **ограничения значения** его **объявления атрибута** (в противном случае).

22.5 Если либо **применение атрибута,** либо его **объявление атрибута** имеет **ограничение значения,** которое является **фиксированным** значением, то одиночное ограничение значения ACH.1 должно быть добавлено к «NamedType». «Value» в одиночном ограничении значения ACH.1 должно быть сформировано применением раздела 16 либо к значению в **ограничении значения применения атрибута** (если **применение атрибута** имеет **ограничение значения**), либо к значению в **ограничении значения** его **объявления атрибута** (в противном случае).

22.6 Если **применение атрибута** не **требуется** и ни **применение атрибута,** ни его **объявление атрибута** не имеет **ограничения значения,** «NamedType» должно следовать за ключевым словом **OPTIONAL**.

22.7 Заключительная команда кодирования **ATTRIBUTE** должна быть присвоена «Type» в «NamedType».

23 Отображение применений простых и сложных определений типа (общий случай)

23.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее одному из следующих видов применения высокоуровневого, анонимного или XSD-встроенного **простого определения типа** или **сложного определения типа:**

а) **простое определение типа** используется в качестве **базового типа (base type)** другого **простого определения типа;**

б) **простое определение типа** используется в качестве **типа объекта (item type)** типа списка;

в) **простое определение типа** используются в качестве **типа элемента (member type)** типа совокупности;

г) **простое** или **сложное определение типа** используется в качестве **определения типа объявлений элемента,** которые не имеют заменяемого **определения типа** (см. 14.7) и не **обнуляемы;**

е) **простое определение типа** используется в качестве **определения типа объявления атрибута;**

ж) **простое определение типа** используется в качестве **типа содержимого сложного определения типа,** а также

з) **простое определение типа** используется в качестве **определения типа объявления элемента,** которое не имеет заменяемого **определения типа** (см. 14.7) и **обнуляемо.**

23.2 Использование XSD-встроенного типа **простого определения типа** или **сложного определения типа** должно быть отображено в определении типа ACH.1 («DefinedType»), как указано в разделе 11.

23.3 Использование высокоуровневого **простого определения типа** должно быть отображено в определении типа ACH.1 («DefinedType»), полученном применением 10.2 к присвоению типа ACH.1, полученному применением раздела 13 к **простому определению типа.**

23.4 Использование высокоуровневого **сложного определения типа** должно быть отображено в определении типа ACH.1 («DefinedType»), полученном применением 10.2 к присвоению типа ACH.1, полученному применением раздела 20 к **сложному определению типа.**

23.5 Использование анонимного **простого определения типа** не отличается от **простого определения типа** и должно быть отображено, как указано в разделе 13 для **простого определения типа.**

23.6 Использование анонимного **сложного определения типа** не отличается от **сложного определения типа** и должно быть отображено, как указано в разделе 20 для **сложного определения типа.**

23.7 Если **ограничение значения** было предоставлено при инициировании данного раздела, то заключительная команда кодирования **DEFAULT-FOR-EMPTY** должна быть присвоена определению типа ACH.1, и применен один из трех последующих пунктов.

23.7.1 Для **простого определения типа** «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

23.7.2 Для **сложного определения типа, типом содержимого** которого является **простое определение типа,** «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

23.7.3 Для **сложного определения типа** со **смешанным** типом содержимого «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **xsd:string с whiteSpace preserve.**

23.8 Если **ограничение значения** было предоставлено при инициировании данного раздела и значение в **ограничении значения** является **фиксированной** величиной, то применяется один из трех следующих пунктов.

23.8.1 Для **простого определения типа** одиночное ограничение значения ACH.1 с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY**, должно быть добавлено к определению ACH.1.

23.8.2 Для **сложного определения типа, типом содержимого** которого является **простое определение типа,** ограничение внутреннего подтипа ACH.1 должно быть добавлено к определению ACH.1 и должно применять к компоненту **base** одиночное ограничение значения с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY**.

23.8.3 Для **сложного определения типа** со **смешанным** типом содержимого ограничение внутреннего подтипа ACH.1 должно быть добавлено к определению ACH.1 и должно применять к компоненту **embed-values** одиночное ограничение значения ACH.1 с «Value», состоящим в одиночном появления «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY**.

24 Отображение особых применений простых и сложных определений типа (заменяемых)

24.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее высокоуровневому **простому определению типа** или **сложному определению типа,** используемому в качестве **определения типа объявлений элемента,** которые имеют заменяемое **определение типа** (см. 14.7) и не **обнуляемы.**

24.2 Использование **простого определения типа** (STD, например) или **сложного определения типа** (CTD, например) должно быть отображено в типе выбора ACH.1.

24.3 Одна альтернатива должна быть добавлена к типу выбора ACH.1 для STD или CTD и одна альтернатива должна быть добавлена для каждого высокоуровневого **простого определения типа** и **сложного определения типа** в исходной схеме XSD, которая получается ограничением или расширением (непосредственно или косвенно) STD или CTD.

24.4 Для каждой альтернативы «identifier» в «NamedType» должен быть сформирован с применением 10.3 к **имени простого определения типа** или **сложного определения типа,** соответствующего альтернативе, и «Type» в «NamedType» должен быть определением типа ACH.1, полученным применением раздела 23 к **простому определению типа** или **сложному определению типа.**

24.5 Первой альтернативой, добавленной к типу выбора, должна быть альтернатива, соответствующая STD или CTD. Последующие альтернативы должны быть добавлены к типу выбора в порядке, основанном на **целевом пространстве имен** и **имени простых определений типа** и **сложных определений типа.** Определения типа должны быть сначала отсортированы по **целевому пространству имен** (с **отсутствующим** пространством имен, предшествующим всем именам пространства имен, отсортированным по возрастанию в лексикографическом порядке), а затем — по **имени** (также в возрастающем лексикографическом порядке) внутри каждого **целевого пространства имен.**

24.6 Заключительная команда кодирования **USE-TYPE** должна быть присвоена типу выбора ACH.1.

24.7 Если **ограничение значения** было предоставлено при инициировании данного раздела, то заключительная команда кодирования **DEFAULT-FOR-EMPTY** должна быть присвоена каждой альтернативе типа выбора ACH.1, соответствующего **простому** или **сложному определению типа,** что проверит гипотетический элемент, содержащий каноническое лексическое представление значения в **ограничении значения,** но не другим альтернативам (если таковые имеются). Применяется один из трех следующих пунктов.

24.7.1 Если альтернатива соответствует **простому определению типа,** «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

24.7.2 Если альтернатива соответствует **сложному определению типа, тип содержимого** которого является **простым определением типа,** «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

24.7.3 Если альтернатива соответствует **сложному определению типа** со **смешанным** типом содержимого, «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **xsd:string** с **whiteSpace preserve.**

24.8 Если **ограничение значения** было предоставлено при инициировании данного раздела и значение в **ограничении значения** является **фиксированной** величиной, то ограничение внутреннего подтипа ACH.1 должно быть добавлено к типу выбора ACH.1. Один из четырех следующих пунктов применяется к каждой альтернативе типа выбора.

24.8.1 Если альтернатива была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 24.7 и соответствует **простому определению типа,** ограничение внутреннего подтипа должно применять к альтернативе одиночное ограничение значения ACH.1 с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY.**

24.8.2 Если альтернатива была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 24.7 и соответствует **сложному определению типа, типом содержимого** которого является **простое определение типа,** ограничение внутреннего подтипа должно применять к альтернативе другое ограничение внутреннего подтипа ACH.1, которое применяет к компоненту **base** одиночное ограничение значения с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY.**

24.8.3 Если альтернатива была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 24.7 и соответствует **сложному определению типа** со **смешанным** типом содержимого, ограничение внутреннего подтипа должно применять к альтернативе другое ограничения внутреннего подтипа ACH.1, которое в свою очередь применяет к компоненту **embed-values** одиночное ограничение значения ACH.1 с «Value», состоящим в одиночном появлении «Value» идентичном «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY.**

24.8.4 Если альтернатива не была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 24.7, ограничение внутреннего подтипа должно применить ограничение присутствия **ABSENT** к альтернативе.

25 Отображение особых применений простых и сложных определений типа (заменяемых, обнуляемых)

25.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее высокоуровневому **простому определению типа** или **сложному определению типа,** используемому в качестве **определения типа объявлений элемента,** которые имеют заменяемое **определение типа** (см. 14.7) и **обнуляемы.**

25.2 Использование **простого определения типа** (STD, например) или **сложного определения типа** (CTD, например) должно быть отображено в тип выбора ACH.1.

25.3 Одна из альтернатив должна быть добавлена к типу выбора ACH.1 для STD или CTD и одна альтернатива должна быть добавлена для каждого высокоуровневого **простого определения типа** и **сложного определения типа** в исходной схеме XSD, которая получается ограничением или расширением (непосредственно или косвенно) STD или CTD.

25.4 Для каждой альтернативы «identifier» в «NamedType» должен быть сформирован применением 10.3 к **имени простого определения типа** или **сложного определения типа,** соответствующему альтернативе, и «Type» в «NamedType» должно быть определением типа ACH.1 («DefinedType»), полученным применением 10.2 к присвоению типа ACH.1, полученному применением раздела 30 к **простому определению типа** или **сложному определению типа.**

25.5 Первой альтернативой, добавленной к типу выбора, должна быть альтернатива, соответствующая STD или CTD. Последующие альтернативы должны быть добавлены к типу выбора в порядке, основанном на **целевом пространстве имен** и **имени простого определения типа** и **сложного определения типа.** Определения типа должны быть сначала отсортированы по **целевому пространству имен** (с **отсутствующим** пространством имен, предшествующим всем именам пространства имен, отсортированным по возрастанию в лексикографическом порядке), а затем — по **имени** (также в лексикографическом порядке возрастания) внутри каждого **целевого пространства имен.**

25.6 Заключительная команда кодирования **USE-TYPE** должна быть присвоена типу выбора ACH.1.

25.7 Если **ограничение значения** было предоставлено при инициировании данного пункта, то заключительная команда кодирования **DEFAULT-FOR-EMPTY** должна быть присвоена каждой альтернативе типа выбора ACH.1, соответствующего **простому** или **сложному определению типа,** что проверит гипотетический элемент, содержащий каноническое лексическое представление значения в **ограничении значения,** но не другим альтернативам (если таковые имеются). Применяется один из трех следующих пунктов.

25.7.1 Если альтернатива соответствует **простому определению типа,** «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

25.7.2 Если альтернатива соответствует **сложному определению типа, тип содержимого** которого является **простым определением типа,** «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

25.7.3 Если альтернатива соответствует **сложному определению типа** со **смешанным** типом содержимого, «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **xsd:string** с **whiteSpace preserve.**

25.8 Если **ограничение значения** было предоставлено при инициировании данного раздела и значение в **ограничении значения** является **фиксированной** величиной, то ограничение внутреннего подтипа ACH.1 должно быть добавлено к типу выбора ACH.1. Один из четырех следующих пунктов применяется к каждой альтернативе типа выбора.

25.8.1 Если альтернатива была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 25.7 и соответствует **простому определению типа,** ограничение внутреннего подтипа должно применить к альтернативе (которой является тип последовательности ACH.1 с заключительной командой кодирования **USE-NIL**) другое ограничение внутреннего подтипа ACH.1, которое, в свою очередь, должно применить к компоненту **content** ключевое слово **PRESENT** и одиночное ограничение значения ACH.1 с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY.**

25.8.2 Если альтернатива была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 25.7 и соответствует **сложному определению типа, тип содержимого** которого является **простым определением типа,** ограничение внутреннего подтипа должно применять к альтернативе (которой является тип последовательности ACH.1 с заключительной командой кодирования **USE-NIL**) другое ограничение внутреннего подтипа ACH.1, которое применяет к компоненту **content** ключевое слово **PRESENT** и одиночное ограничение значения ACH.1 с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY.**

25.8.3 Если альтернатива была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 25.7 и соответствует **сложному определению типа** со **смешанным** типом содержания, ограничение внутреннего подтипа должно применить к альтернативе (которой является тип последовательности ACH.1 с заключительной командой кодирования **USE-NIL**) другое ограничение внутреннего подтипа ACH.1, которое применяет к:

а) компоненту **embed-values** одиночное ограничение значения ACH.1 с «Value», состоящем в одиночном появлении «Value», идентичном «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY**; и

б) компоненту **content** ключевое слово **PRESENT.**

25.8.4 Если альтернатива не была присвоена заключительной команде кодирования **DEFAULT-FOR-EMPTY** в 25.7, ограничение внутреннего подтипа должно применить ограничение присутствия **ABSENT** к альтернативе.

26 Отображение особых применений простых определений типа (обнуляемых)

26.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее либо:

а) высокоуровневому, анонимному или XSD-встроенному **простому определению типа,** используемому в качестве **определения типа объявлений элемента,** которые не имеют заменяемого **определения типа** (см. 14.7) и **обнуляемы;** либо

б) высокоуровневому **простому определению типа,** которое является элементом иерархии развития (derivation hierarchy) **определения типа объявлений элемента,** которые имеют заменяемое **определение типа** (см. 14.7) и **обнуляемы.**

26.2 Использование **простого определения типа** должно быть отображено в тип последовательности ACH.1 с одним компонентом **OPTIONAL.**

26.3 «Identifier» в «NamedType» компонента должно быть **content,** и «Type» в «NamedType» должно быть определением типа ACH.1, полученным применением раздела 23 к **простому определению типа.**

26.4 Заключительная команда кодирования **USE-NIL** должна быть присвоена типу последовательности ACH.1.

26.5 Если **ограничение значения** было предоставлено при инициировании данного раздела, то заключительная команда кодирования **DEFAULT-FOR-EMPTY** должна быть присвоена типу последовательности ACH.1. «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY** должно быть получено применением раздела 16 к значению в **ограничении значения.**

26.6 Если **ограничение значения** было предоставлено при инициировании данного раздела и значение в **ограничении значения** является **фиксированной** величиной, то ограничение внутреннего подтипа ACH.1 должно быть добавлено к типу последовательности ACH.1. Ограничение внутреннего подтипа должно применить к компоненту **content** одиночное ограничение значения ACH.1 с «Value», идентичным «Value» в заключительной команде кодирования **DEFAULT-FOR-EMPTY.** Ограничение внутреннего подтипа должно также применить ключевое слово **PRESENT** к компоненту **content.**

27 Отображение особых применений сложных определений типа (обнуляемых)

27.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее либо:

а) высокоуровневому, анонимному или XSD-встроенному **сложному определению типа,** используемому в качестве **определения типа объявлений элемента,** которые не имеют заменяемого **определения типа** (см. 14.7) и **обнуляемы;** либо

б) высокоуровневому **сложному определению типа,** которое является элементом иерархии развития **определения типа объявлений элемента,** которые имеют заменяемое **определение типа** (см. 14.7) и **обнуляемы.**

27.2 Использование XSD-встроенного **сложного определения типа** должно быть отображено в определение типа ACH.1 («DefinedType»), как указано в разделе 11.

27.3 Использование высокоуровневого или анонимного **сложного определения типа** должно быть отображено в определение типа ACH.1, как указанно в 27.4—27.12.

27.4 Определение типа ACH.1 должно быть типом последовательности ACH.1, и один или более компонентов должны быть добавлены к нему, как указано в 27.5—27.11.

27.5 Если **тип содержимого сложного определения типа** представляет собой **смешанную** модель содержимого, то компонент **embed-values** должен быть добавлен к типу последовательности ACH.1, как указано в 20.5.

27.6 Если **тип содержимого сложного определения типа** является **частицей, термом** которой является **модельная группа** с **наборщиком всего,** то компонент **order** должен быть добавлен к типу последовательности ACH.1, как указано в 20.6.

27.7 Если **сложное определение типа** имеет **применения атрибута,** то компоненты, отображенные из **применений атрибута,** должны быть добавлены к типу последовательности ACH.1, как указано в 20.7.

27.8 Если **сложное определение типа** имеет **атрибут группового символа,** то компонент, сформированный из **атрибута группового символа,** должен быть добавлен к типу последовательности ACH.1, как указано в 20.8.

27.9 Если **тип содержимого сложного определения типа** является **частицей,** то применяется один из трех следующих пунктов.

27.9.1 Если **термом частицы** является **модельная группа** с **наборщиком последовательности, min occurs** и **max occurs** которого оба равны единице, то компонент `OPTIONAL` должен быть добавлен в типу последовательности ACH.1. «Identifier» в «NamedType» компонента должен быть сформирован применением 10.3 к символьной строке «`content`», и «Type» в «NamedType» должно быть типом последовательности ACH.1, сформированным следующим образом. Для каждой **частицы модельной группы** по порядку, список из нуля и более «NamedType» должно быть получен применением раздела 19 к **частице** в **модельной группе,** и эти «NamedType» должны быть добавлены к внутреннему типу последовательности ACH. 1 в том же порядке.

27.9.2 Если **термом частицы** является **модельная группа** с **наборщиком последовательности,** оба **min occurs** и **max occurs** которого не равны единице, или **модельная группа** с **наборщиком выбора,** то компонент `OPTIONAL` должен быть добавлен к типу последовательности ACH.1. «Identifier» в «NamedType» компонента должен быть сформирован применением 10.3 к символьной строке «`content`», и «Type» в «NamedType» должно быть типом последовательности ACH.1 с одним компонентом, который должен быть получен применением раздела 19 к **частице** в **типе содержимого.**

27.9.3 Если **термом частицы** является **модельная группа** с **наборщиком всего,** то компонент `OPTIONAL` должен быть добавлен к типу последовательности ACH.1. «Identifier» в «NamedType» компонента должен быть сформирован применением 10.3 к символьной строке «`content`», и «Type» в «NamedType» должно быть типом последовательности ACH.1. Для каждой **частицы модельной группы** по порядку, компонент, полученный применением раздела 19 к **частице модельной группы,** должен быть добавлен к внутреннему типу последовательности ACH.1. Если **частица** в **типе содержимого сложного определения типа** имеет **min occurs** ноль, то каждая из **частиц модельной группы** с **min occurs** равным единице, должна быть отображена, как если бы она имела **min occurs** ноль.

27.10 Если **тип содержимого сложного определения типа** является **простым определением типа,** то компонент `OPTIONAL` должен быть добавлен к типу последовательности ACH.1. «Identifier» в «NamedType» компонента должно быть сформировано применением 10.3 к символьной строке «`content`», и «Type» в «NamedType» должно быть определением типа ACH.1, полученным применением раздела 23 к **типу содержимого.**

27.11 Если **тип содержимого сложного определения типа пустой,** то компонент `OPTIONAL` должен быть добавлен к типу последовательности ACH.1. «Identifier» в «NamedType» компонента должен быть сформирован применением 10.3 к символьной строке «`content`», и «Type» в «NamedType» должно быть встроенным типом ACH.1 `NULL`.

27.12 Заключительная команда кодирования `USE-NIL` должна быть присвоена типу последовательности ACH.1.

27.13 Если **ограничение значения** было предоставлено при инициировании данного раздела, то заключительная команда кодирования `DEFAULT-FOR-EMPTY` должна быть присвоена типу последовательности ACH.1. Применяется один из двух следующих пунктов.

27.13.1 Если **тип содержимого сложного определения типа** является **простым определением типа,** «Value» в заключительной команде кодирования `DEFAULT-FOR-EMPTY` должен быть получен применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **простого определения типа.**

27.13.2 Если **тип содержимого сложного определения типа** является **смешанным** типом содержимого, «Value» в заключительной команде кодирования `DEFAULT-FOR-EMPTY` должен быть получен применением раздела 16 к значению в **ограничении значения,** рассматриваемому как значение в области значений **xsd:string** с **whiteSpace preserve.**

27.14 Если **ограничение значения** было предоставлено при инициировании данного раздела и значение в **ограничении значения** является **фиксированной** величиной, то ограничение внутреннего подтипа ACH.1 должно быть добавлено к типу последовательности ACH.1. Ограничение внутреннего подтипа должно применить ключевое слово `PRESENT` к компоненту `content`. Применяется один из двух следующих пунктов.

27.14.1 Если **тип содержимого сложного определения типа** является **простым определением типа,** ограничение внутреннего подтипа должно применить к компоненту `content` одиночное ограничение значения ACH.1 с «Value», идентичным «Value» в заключительной команде кодирования `DEFAULT-FOR-EMPTY`.

27.14.2 Если **тип содержимого сложного определения типа** представляет собой **смешанным** тип содержимого, ограничение внутреннего подтипа должно применять к:

а) компоненту `embed-values` одиночное ограничение значения ACH.1 с «Value», состоящим в одиночном появлении «Value», идентичном «Value» в заключительной команде кодирования `DEFAULT-FOR-EMPTY`; и

б) компоненту `content` ключевое слово `PRESENT`.

28 Отображение особых применений объявлений элемента (головного элемента группы замены элементов)

28.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать определение типа ACH.1, соответствующее высокоуровневому **объявлению элемента,** которое возглавляет группу замены элементов и используется в качестве **терма частиц.**

28.2 Использование высокоуровневого **объявления элемента** (H, например), должно быть отображено в тип выбора ACH.1.

28.3 Одна альтернатива должна быть добавлена к типу выбора ACH.1 для каждого высокоуровневого **объявления элемента** (в том числе H) в исходной схеме XSD, которое не является **абстрактным** и является членом группы замены во главе с H.

Примечание — В XSD членство в группе замены транзитивно, т.е. члены группы замены ESG1, чей головной элемент является членом другой группы замены ESG2, являются также членами ESG2.

28.4 Для каждой альтернативы «identifier» в «NamedType» должен быть сформирован применением 10.3 к **имени** высокоуровневого **объявления элемента,** соответствующего альтернативе, и «Type» в «NamedType» должно быть определением типа ACH.1 («DefinedType»), сформированным применением 10.2 к присвоению типа ACH.1, полученному применением раздела 14 к высокоуровневому **объявлению элемента.**

28.5 Альтернативы должны быть добавлены к типу выбора в порядке, основанном на **целевом пространстве имен** и **имени** высокоуровневых **объявлений элемента. Объявления элемента** должны быть сначала отсортированы по **целевому пространству имен** (с **отсутствующим** пространством имен, предшествующим всем именам пространства имен, отсортированным по возрастанию в лексикографическом порядке), а затем — по **имени** (также в лексикографическом порядке возрастания) внутри каждого **целевого пространства имен.**

Примечание — **Объявление элемента,** которое является головным элементом группы замены элементов упорядочены вместе с другими **объявлениями элемента,** которые принадлежат к группе замены элементов.

28.6 Заключительная команда кодирования `UNTAGGED` должна быть присвоена типу выбора.

29 Формирование особых присвоений типа ACH.1 для типов, используемых в объявлениях элементов

29.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1, соответствующее высокоуровневому **простому определению типа** или **сложному определению типа,** используемому в качестве **определения типа объявлений элемента,** которые имеют заменяемое **определение типа** (см. 14.7) или **обнуляемы.**

29.2 Данный раздел формирует особое присвоение типа ACH.1 для данной комбинации из следующих условий и данных, предоставленных при инициировании данного раздела:

а) имеет ли **объявление элемента** заменяемое или незаменяемое **определение типа** (см. 14.7);

б) является ли **объявление элемента обнуляемым** или **необнуляемым;**

в) является ли **определение типа простым определением типа** или **сложным определением типа;**

г) имеет ли **объявление элемента ограничение значения,** и если да, то является ли значение в **ограничении значения** значением **по умолчанию** или **фиксированным** значением;

д) имя **определения типа;** и

е) значение в **ограничении значения** (если таковые имеются).

29.3 Только одно особое присвоение типа ACH.1 должно быть сформировано для каждой комбинации из вышеперечисленных элементов, что на самом деле происходит при одном или нескольких инициированиях данного пункта во время отображения исходной схемы XSD (но см. 29.4).

Примечание — Например, если два **объявления элемента** в большой XSD-схеме имеют одинаковые **определения типа,** оба **обнуляемые** и оба имеют **ограничение значения,** которое является значением **по умолчанию** и является тем же значением, что сформировало одиночное особое присвоение типа ACH.1. Имя ссылки типа этого присвоения типа появится в «Type» в «TypeAssignment», соответствующем обоим **объявлениям элемента.**

29.4 Когда данный раздел запрашивается для **простого определения типа** или **сложного определения типа,** используемого в качестве **определения типа объявления элемента,** которое является **обнуляемым** и не имеет заменяемого **определения типа,** он формирует присвоение типа ACH.1 (используя суффикс «**-nillable**»), которое также может быть получено инициированием раздела 30 для того же **простого определения типа** или **сложного определения типа.** В таких случаях должно быть только одно такое присвоение типа ACH.1, сформированное либо с помощью данного пункта, либо с помощью раздела 30, в зависимости от того, какой пункт запрашивается первым.

29.5 Термин «связанное присвоение типа ACH.1» обозначает присвоение типа ACH.1, отображенное из **простого определения типа** или **сложного определения типа,** которое является **определением типа объявления элемента,** для которого сформировано особое присвоение типа ACH.1 применением, соответственно, раздела 13 или раздела 20.

Примечание — Любое особое присвоение типа ACH.1 имеет связанное присвоение типа ACH.1, так как этот раздел применяется только при **определении типа объявления элемента,** являющимся высокоуровневым **простым определением типа** или **сложным определением типа.** Все такие **простые определения типа** и **сложные определения типа** отображаются в присвоении типа ACH.1.

29.6 Для данного **объявления элемента,** «typereference» в «TypeAssignment» для особого присвоения типа ACH.1 должно быть сформировано добавлением суффикса (см. 29.7) и пост-суффикса (см. 29.7) к имени ссылки типа связанного присвоения типа ACH.1, и применения 10.3 к результирующей символьной строке, и «Type» в «TypeAssignment» должно быть определением типа ACH.1, сформированным одним из разделов 24, 25, 26 или 27 (см. 29.7) к **простому определению типа** или **сложному определению типа,** которое является **определением типа объявления элемента. Ограничение значения в объявлении элемента** (если таковые имеются) должно быть предоставлено соответствующим разделом (24, 25, 26 или 27) и должно быть использовано при формировании определения типа ACH.1, как указано в соответствующем разделе.

29.7 Суффикс и соответствующий номер раздела должны быть получены из двух последних столбцов таблицы 7 после выбора строки таблицы, основанного на условиях, перечисленных в 29.2 а) — г). Если есть **ограничение значения,** пост-суффикс должен быть каноническим лексическим представлением (см. W3C XML-схема Часть 2, п. 2.3.1) значения в **ограничении значения,** в противном случае там должна быть пустая строка.

Таблица 7 — Суффиксы и соответствующие номера пунктов для формирования особых присвоений типа ACH.1

Заменяемое	обнуляемое	простое / сложное	ограничение значения	Суффикс	Применяемый раздел
Нет	Да	**простое**	Нет	**-nillable**	26
		простое	**по умолчанию**	**-nillable-default-**	
		простое	**фиксированное**	**-nillable-fixed-**	
		сложное	Нет	**-nillable**	27
		сложное	**по умолчанию**	**-nillable-default-**	
		сложное	**фиксированное**	**-nillable-fixed-**	

Окончание таблицы 7

Заменяемое	обнуляемое	простое / сложное	ограничение значения	Суффикс	Применяемый раздел
Да	Нет	**простое** или **сложное**	Нет	**-derivations**	24
		простое или **сложное**	**по умолчанию**	**-deriv-default-**	
		простое или **сложное**	**фиксированное**	**-deriv-fixed-**	
	Да	**простое** или **сложное**	Нет	**-deriv-nillable**	25
		простое или **сложное**	**по умолчанию**	**-deriv-nillable-default-**	
		простое или **сложное**	**фиксированное**	**-deriv-nillable-fixed-**	

30 Формирование особых присвоений типа ACH.1 для типов, принадлежащих к иерархии развития

30.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1, соответствующее высокоуровневому **простому определению типа** или **сложному определению типа,** принадлежащему иерархии развития **определения типа объявлений элемента,** которые имеют заменяемое **определение типа** (см. 14.7) и **обнуляемы.**

30.2 Данный раздел формирует особое присвоение типа ACH.1 для **простого определения типа** или **сложного определения типа,** указанного при инициировании данного раздела.

30.3 Только одно особое присвоение типа ACH.1 должно быть сформировано для каждого **простого определения типа** или **сложного определения типа,** что на самом деле происходит при одном или нескольких инициированиях данного раздела во время отображения исходной схемы XSD (но см. 29.4).

30.4 Термин «связанное присвоение типа ACH.1» означает присвоение типа ACH.1, отображаемое из **простого определения типа** или **сложного определения типа** применением раздела 13 или раздела 20 соответственно.

30.5 «Typereference» в «TypeAssignment» для особого присвоения типа ACH.1 должно быть сформировано добавлением суффикса «**-nillable**» к имени ссылки типа связанного присвоения типа ACH.1 и применения 10.3 к результирующей символьной строке, и «Type» в «TypeAssignment» должно быть определением типа ACH.1, полученным применением либо раздела 26, либо раздела 27 к **простому определению типа** или **к сложному определению типа** соответственно.

Примечание — Этот раздел определяет только суффикс «**-nillable**» (но не суффиксы «**-nillable-default-**» и «**-nillable-fixed-**»), потому что даже если **объявление элемента** имеет **ограничение значения,** это **ограничение значения** невидимо для разделов 26 и 27 при запросе данным разделом.

31 Формирование особых присвоений типа ACH.1 для групп замены элементов

31.1 Применение данного раздела является прямым запросом из других мест настоящего стандарта сформировать присвоение типа ACH.1, соответствующее **частице, термом** которого является высокоуровневое **объявление элемента,** которое возглавляет группу замены элементов.

31.2 Это раздел формирует особое присвоение типа ACH.1 для высокоуровневого **объявления элемента,** указанного при инициировании данного раздела.

31.3 Только одно особое присвоение типа ACH.1 должно быть сформировано для каждого высокоуровневого **объявления элемента,** что на самом деле происходит в одном или нескольких инициированиях данного раздела во время отображения исходной XSD-схемы.

31.4 Термин «связанное присвоение типа ACH.1» означает присвоение типа ACH.1, отображаемое из высокоуровневого **объявления элемента** применением раздела 14.

31.5 «Typereference» в «TypeAssignment» для особого присвоения типа ACH.1 должно быть сформировано добавлением суффикса «**-group**» к имени ссылки типа связанного присвоения типа ACH.1 и применения 10.3 к результирующей символьной строке, и «Type» в «TypeAssignment» должно быть определением типа ACH.1, полученным применением раздела 28 к высокоуровневому **объявлению элемента.**

Приложение А
(обязательное)

Определения типов АСН.1, соответствующие XSD-встроенным типам, для отображения версии 1

А.1 Настоящее приложение описывает модуль, который определяет АСН.1-типы, соответствующие XSD-встроенным типам, которые используются для отображения из W3C XML-схемы в АСН.1 для отображения версии 1.

А.2 W3C XML-схема определяет множество встроенных типов даты и времени для представления длительности, моментов или повторяющихся моментов. Хотя все они производные от ISO 8601, существуют некоторые расширения и ограничения. XSD-встроенные типы даты и времени отображаются в **VisibleString** с определенным пользователем ограничением, ссылающимся на соответствующий пункт XSD. Разрешенное ограничение алфавита добавляется для обеспечения более эффективного кодирования с правилами уплотненного кодирования (PER), так как пользовательские ограничения не являются PER-видимыми (и, следовательно, не используются в оптимизации кодировок).

А.3 **XSD** модуль для отображения версии 1:

```
XSD {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2)
version1(1)}
"/ACH.1/Specification/Modules/XSD-Module/Version1"
DEFINITIONS
AUTOMATIC TAGS ::=
BEGIN
/* xsd:anySimpleType */
AnySimpleType ::= XMLCompatibleString
/* xsd:anyType */
AnyType ::= SEQUENCE {
      embed-values SEQUENCE OF String,
      attr SEQUENCE
        (CONSTRAINED BY {
           /* Each item shall conform to the "AnyAttributeFormat"
specified
            in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */ } ) OF
String,
      elem-list SEQUENCE OF elem String
        (CONSTRAINED BY {
           /* Shall conform to the "AnyElementFormat" specified
           in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19 */ } ) }
        (CONSTRAINED BY {
        /* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 25 */ } )
AnyType-nillable ::= SEQUENCE {
        embed-values SEQUENCE OF String,
        attr SEQUENCE
          (CONSTRAINED BY {
            /* Each item shall conform to the "AnyAttributeFormat" specified
            in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */ } ) OF String,
        content SEQUENCE {
            elem-list SEQUENCE OF elem String
              (CONSTRAINED BY {
                  /* Shall conform to the "AnyElementFormat" specified
                    in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19 */ } )
        } OPTIONAL }
        (CONSTRAINED BY {
        /* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 25 */ } )
/* xsd:anyUri */
AnyURI ::= XMLStringWithNoCRLFHT
       (CONSTRAINED BY {
             /* The XMLStringWithNoCRLFHT shall be a valid URI as
              defined in IETF RFC 2396. Note that 2396 allows any
```

```
                valid IRI format without escaping non-ASCII characters.
                Use of the IANA oid: URI/IRI scheme should be considered. */ | }
/* xsd:date */
Date ::= DateTimeType (DateOnly)
/* xsd:dateTime */
DateTime ::= DateTimeType
/* xsd:decimal */
Decimal ::= REAL (WITH COMPONENTS {..., base(10)})
          (ALL EXCEPT(-0 | MINUS-INFINITY | PLUS-INFINITY | NOT-A-
NUMBER))
/* xsd:double */
Double ::= REAL (WITH COMPONENTS {
              mantissa(-9007199254740991..9007199254740991),
              base(2),
              exponent(-1075..970)})
/* xsd:duration */
Duration ::= DurationType
/* xsd:ENTITIES */
ENTITIES ::= SEQUENCE (SIZE(1..MAX)) OF ENTITY
/* xsd:ENTITY */
ENTITY ::= NCName
/* xsd:float */
Float ::= REAL (WITH COMPONENTS {
                 mantissa(-16777215..16777215),
                 base(2),
                 exponent(-149..104)})
/* xsd:gDay */
GDay ::= DateTimeType (Day)
/* xsd:gMonth */
GMonth ::= DateTimeType (Month)
/* xsd:gMonthDay */
GMonthDay ::= DateTimeType (MonthDay)
/* xsd:gYear */
GYear ::= DateTimeType (Year)
/* xsd:gYearMonth */
GYearMonth ::= DateTimeType (YearMonth)
/* xsd:ID */
ID ::= NCName
/* xsd:IDREF */
IDREF ::= NCName
/* xsd:IDREFS */
IDREFS ::= SEQUENCE (SIZE(1..MAX)) OF IDREF
/* xsd:int */
Int ::= INTEGER (-2147483648..2147483647)
/* xsd:language */
Language ::= VisibleString (FROM "a".."z" | "A".."Z" | "-" |
"0".."9"))
       (PATTERN
       "[a-zA-Z]#(1,8)(-[a-zA-Z0-9]#(1,8))*")
       /* The semantics of Language is specified in IETF RFC 3066
*/
/* xsd:long */
   Long ::= INTEGER (-9223372036854775808..9223372036854775807)
/* xsd:name */
Name ::= Token (XMLStringWithNoWhitespace)
     (CONSTRAINED BY {
          /* The Token shall be a Name as defined in W3C XML 1.0, 2.3
*/ } )
/* xsd:NCName */
NCName ::= Name
```

```
(CONSTRAINED BY {
/* The Name shall be an NCName as defined in W3C XML Namespaces, 2 */ } )
/* xsd:NMTOKEN */
NMTOKEN ::= Token (XMLStringWithNoWhitespace)
        (CONSTRAINED BY {
           /* The Token shall be an NMTOKEN as defined in W3C XML 1.0,
2.3 */ } )
/* xsd:NMTOKENS */
NMTOKENS ::= SEQUENCE (SIZE(1..MAX)) OF NMTOKEN
/* xsd:normalizedString */
NormalizedString ::= String (XMLStringWithNoCRLFHT)
         (CONSTRAINED BY {
            /* The String shall be a normalizedString as defined in W3C XML
Schema
            Part 2, 3.3.1 */})
/* xsd:NOTATION */
NOTATION ::= QName
/* xsd:QName */
QName ::= SEQUENCE {
       uri AnyURI OPTIONAL,
       name NCName }
/* xsd:short */
Short ::= INTEGER (-32768..32767)
/* xsd:string */
String ::= XMLCompatibleString
/* xsd:time */
Time ::= DateTimeType (TimeOnly)
/* xsd:token */
Token ::= NormalizedString (CONSTRAINED BY {
      /* The NormalizedString shall be a token as defined in W3C XML
Schema Part 2,
       3.3.2 */})
/* xsd:unsignedInt */
UnsignedInt ::= INTEGER (0..4294967295)
/* xsd:unsignedLong */
UnsignedLong ::= INTEGER (0..18446744073709551615)
/* xsd:unsignedShort */
UnsignedShort ::= INTEGER (0..65535)
/* ACH.1 type definitions supporting the mapping of W3C XML
Schema built-in types
*/
XMLCompatibleString ::= UTF8String (FROM(
       {0, 0, 0, 9} |
       {0, 0, 0, 10} |
       {0, 0, 0, 13} |
       {0, 0, 0, 32} .. {0, 0, 215, 255} |
       {0, 0, 224, 0} .. {0, 0, 255, 253} |
       {0, 1, 0, 0} .. {0, 16, 255, 253}))
XMLStringWithNoWhitespace ::= UTF8String (FROM(
       {0, 0, 0, 33} .. {0, 0, 215, 255} |
       {0, 0, 224, 0} .. {0, 0, 255, 253} |
       {0, 1, 0, 0} .. {0, 16, 255, 253}))
XMLStringWithNoCRLFHT ::= UTF8String (FROM(
       {0, 0, 0, 32} .. {0, 0, 215, 255} |
       {0, 0, 224, 0} .. {0, 0, 255, 253} |
       {0, 1, 0, 0} .. {0, 16, 255, 253}))
/* ACH.1 type definitions supporting the mapping of W3C XML Schema
built-in date
and time types */
DurationType ::= VisibleString (FROM ("0".."9" | "DHMPSTY:.+-"))
                      (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.6
*/})
```

```
DateTimeType ::= VisibleString (FROM ("0".."9" | "TZ:.+-"))
                        (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.7
*/})
DateOnly ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
                        (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.9
*/})
Day ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
(CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.13 */})
Month ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
                        (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.14 */})
MonthDay ::= DateTimeType(FROM ("0".."9" | "Z:+-"))
                (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.12
*/})
Year ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
                (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.11
*/})
YearMonth ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
                        (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.10
*/})
TimeOnly ::= DateTimeType (FROM ("0".."9" | "Z:.+-"))
                        (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.8
*/})
ENCODING-CONTROL XER
            GLOBAL-DEFAULTS MODIFIED-ENCODINGS
            GLOBAL-DEFAULTS CONTROL-NAMESPACE
                    ""gost.e-ecolog.ru                                    "
                    PREFIX "xsi"
NAMESPACE ALL, ALL IN ALL AS
                    "http://www.w3.org/2001/XMLSchema"
                    PREFIX "xsd"
USE-QNAME QName
DECIMAL Decimal
LIST ENTITIES, IDREFS, NMTOKENS
EMBED-VALUES AnyType, AnyType-nillable
ANY-ATTRIBUTES AnyType.attr, AnyType-nillable.attr
ANY-ELEMENT AnyType.elem-list.*, AnyType-nillable.content.elem-
list.*
UNTAGGED AnyType.elem-list, AnyType-nillable.content.elem-list
NAME AnySimpleType, AnyURI, Boolean,
       Byte, Date, DateTime, Decimal, Double, Duration,
       Float, GDay, GMonth, GMonthDay, GYear, GYearMonth,
       Int, Language, Long,
       NormalizedString, Short,
       String, Time, Token,
       UnsignedInt, UnsignedLong, UnsignedShort
       AS UNCAPITALIZED
USE-NIL AnyType-nillable
WHITESPACE AnyURI, Language, Token, DurationType, DateTimeType
COLLAPSE
WHITESPACE NormalizedString REPLACE
END
```

Приложение В
(обязательное)

Определения типов АСН.1, соответствующие XSD-встроенным типам, для отображения версии 2

В.1 Настоящее приложение описывает модуль, который определяет АСН.1-типы, соответствующие XSD-встроенным типам, которые используются для отображения W3C XML-схемы в АСН.1 для отображения версии 2.

В.2 W3C XML-схема определяет множество встроенных типов даты и времени для представления длительности, моментов или повторяющихся моментов. Хотя все они производные от ISO 8601, существуют некоторые расширения и ограничения. XSD-встроенные типы даты и времени отображаются, как правило, в одном из типов времени АСН.1, но там, где XSD обеспечивает дополнительные абстрактные значения, отображение является на ВЫБОР из типа времени АСН.1 и **VisibleString** с определенным пользователем ограничением, ссылающимся на соответствующий пункт XSD.

В.3 **XSD** модуль для отображения версии 2:

```
XSD {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2)
version2(2)}
"/ACE.1/Specification/Modules/XSD-Module/Version2"
DEFINITIONS
AUTOMATIC TAGS ::=
BEGIN
/* xsd:anySimpleType */
AnySimpleType ::= XMLCompatibleString
/* xsd:anyType */
AnyType ::= SEQUENCE {
      embed-values SEQUENCE OF String,
      attr SEQUENCE
        (CONSTRAINED BY {
          /* Each item shall conform to the "AnyAttributeFormat"
specified
          in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */ } )
OF String,
      elem-list SEQUENCE OF elem String
        (CONSTRAINED BY {
          /* Shall conform to the "AnyElementFormat" specified
          in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19 */ } )
}
      (CONSTRAINED BY {
      /* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4, clause
25 */ } )
AnyType-nillable ::= SEQUENCE {
      embed-values SEQUENCE OF String,
      attr SEQUENCE
        (CONSTRAINED BY {
          /* Each item shall conform to the "AnyAttributeFormat"
specified
          in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */ } ) OF
String,
      content SEQUENCE {
          elem-list SEQUENCE OF elem String
            (CONSTRAINED BY {
               /* Shall conform to the "AnyElementFormat"
specified
               in ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19
*/ } )
      } OPTIONAL }
      (CONSTRAINED BY {
                  /* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25
*/ } )
```

```
/* xsd:anyUri */
AnyURI ::= XMLStringWithNoCRLFHT
      (CONSTRAINED BY {
         /* The XMLStringWithNoCRLFHT shall be a valid URI as
         defined in IETF RFC2396. Note that 2396 allows any valid
         IRI format without escaping non-ASCII characters. Use of
         the IANA oid: URI/IRI scheme should be considered. */ } )
/* xsd:date */
Date ::= GenericTimeTypeChoice {
                  TIME (SETTINGS "Basic=Date Date=YMD"),
                  VisibleString
                    (FROM ("0".."9" | "DHMPSTY:.+-"))
                    (CONSTRAINED BY {/* W3C XML Schema 1.0 Part 2, 3.2.9
                                 and used if a time-zone is present
*/})}
/* xsd:dateTime */
DateTime ::= TIME ((SETTINGS "Basic=Date-Time Date=YMD
Midnight=Start"))
     (CONSTRAINED BY {/*The time-zone shall be in the range -14 to
+14*/})
     (CONSTRAINED BY {/*The seconds and fractions of a second shall be
less
                  than 60 (no leap seconds supported, in accordance
with
                  W3C XML Schema 1.0 Part 2, 3.2.7)*/})
(CONSTRAINED BY {/*The type is constrained to "Time=HMSFn" for any
n*/})/*
xsd:decimal */
Decimal ::= REAL (WITH COMPONENTS {..., base(10)})
           (ALL EXCEPT(-0 | MINUS-INFINITY | PLUS-INFINITY | NOT-A-
           NUMBER))
/* xsd:double */
Double ::= REAL (WITH COMPONENTS {
                mantissa(-9007199254740991..9007199254740991),
                base(2),
                exponent(-1075..970)})
/* xsd:duration */
Duration ::= GenericTimeTypeChoice {
                DURATION
                 ((WITH COMPONENTS {...,
                                    seconds ABSENT,
                                    fractional-part ABSENT})|
                 (WITH COMPONENTS {...,
                                    seconds PRESENT})),
                VisibleString
                       (FROM ("0".."9" | "DHMPSTY:.+-"))
                 (CONSTRAINED BY {/* W3C XML Schema 1.0 Part 2, 3.2.6
                                  and used for negative durations */})}
/* xsd:ENTITIES */
ENTITIES ::= SEQUENCE (SIZE(1..MAX)) OF ENTITY
/* xsd:ENIITY */
ENTITY ::= NCName
/* xsd:float */
Float ::= REAL (WITH COMPONENTS {
                mantissa(-16777215..16777215),
                base(2),
                exponent(-149..104)})
/* xsd:gDay */
GDay ::= DateTimeType (Day)
    /* This is an integer followed optionally by a time-zone.
     It is not supported in either ISO 8601 or in ACE.1, so
the Version 1
```

```
      mapping has been retained (similarly for other "G" types).
*/
/* xsd:gMonth */
GMonth ::= DateTimeType (Month)
/* xsd:gMonthDay */
GMonthDay ::= DateTimeType (MonthDay)
/* xsd:gYear */
GYear ::= GenericTimeTypeChoice {
                  TIME (SETTINGS "Basic=Date Date=Y"),
                  VisibleString
                    (FROM ("0".."9" | "Z:+-"))
                    (CONSTRAINED BY {/* W3C XML Schema 1.0 Part 2,
3.2.11
                                        and used if a time-zone is
present */})}
/* xsd:gYearMonth */
GYearMonth ::= GenericTimeTypeChoice {
TIME (SETTINGS "Basic=Date Date=YM "),
VisibleString
(FROM ( "0 ".. "9" |  "Z:+- "))
(CONSTRAINED BY {/* W3C XML Schema 1.0 Part 2, 3.2.14
and used if a time-zone is present */})}
/* xsd:ID */
ID ::= NCName
/* xsd:IDREF */
IDREF ::= NCName
/* xsd:IDREFS */
IDREFS ::= SEQUENCE (SIZE(1..MAX)) OF IDREF
/* xsd:int */
Int ::= INTEGER (-2147483648..2147483647)
/* xsd:language */
Language ::= VisibleString (FROM ( "a ".. "z " |  "A ".. "Z " |  "- " |
"0 ".. "9 "))
        (PATTERN
        "[a-zA-Z]#(1,8)(-[a-zA-Z0-9]#(1,8))* ")
        /* The semantics of Language is specified in IETF RFC 3066 */
/* xsd:long */
Long ::= INTEGER (-9223372036854775808..9223372036854775807)
/* xsd:name */
Name ::= Token (XMLStringWithNoWhitespace)
      (CONSTRAINED BY {
       /* The Token shall be a Name as defined in W3C XML 1.0,
2.3 */ } )
/* xsd:NCName */
NCName ::= Name
        (CONSTRAINED BY {
        /* The Name shall be an NCName as defined in W3C XML
Namespaces, 2 */ } )
/* xsd:NMTOKEN */
NMTOKEN ::= Token (XMLStringWithNoWhitespace)
        (CONSTRAINED BY {
           /* The Token shall be an NMTOKEN as defined in W3C XML 1.0,
2.3 */ } )
/* xsd:NMTOKENS */
NMTOKENS ::= SEQUENCE (SIZE(1..MAX)) OF NMTOKEN
/* xsd:normalizedString */
NormalizedString ::= String (XMLStringWithNoCRLFHT)
        (CONSTRAINED BY {
           /* The String shall be a normalizedString as defined in W3C
XML Schema
            Part 2, 3.3.1 */})
```

```
/* xsd:NOTATION */
NOTATION ::= QName
/* xsd:QName */
QName ::= SEQUENCE {
       uri AnyURI OPTIONAL,
       name NCName }
/* xsd:short */
Short ::= INTEGER (-32768..32767)
/* xsd:string */
String ::= XMLCompatibleString
/* xsd:time */
Time ::= TIME ((SETTINGS "Basic=Time")
                                EXCEPT (SETTINGS "Midnight=End"))
     (CONSTRAINED BY {/*The time-zone shall be in the range -14
 to +14*/})
     (CONSTRAINED BY {/*The seconds and fractions of a second
shall be less
                        than 60 (no leap seconds supported, in
accordance with
                        W3C XML Schema 1.0 Part 2, D.2)*/})
     (CONSTRAINED BY {/*Constrained to "Time=HMSFn" for any n*/})
/* xsd:token */
Token ::= NormalizedString (CONSTRAINED BY {
     /* The NormalizedString shall be a token as defined in W3C XML
Schema Part 2,
      3.3.2 */})
/* xsd:unsignedInt */
UnsignedInt ::= INTEGER (0..4294967295)
/* xsd:unsignedLong */
UnsignedLong ::= INTEGER (0..18446744073709551615)
/* xsd:unsignedShort */
UnsignedShort ::= INTEGER (0..65535)
/* ACH.1 type definitions supporting the mapping of W3C XML Schema
built-in types
*/
XMLCompatibleString ::= UTF8String (FROM(
         {0, 0, 0, 9} |
         {0, 0, 0, 10} |
         {0, 0, 0, 13} |
         {0, 0, 0, 32} .. {0, 0, 215, 255} |
         {0, 0, 224, 0} .. {0, 0, 255, 253} |
         {0, 1, 0, 0} .. {0, 16, 255, 253}))
XMLStringWithNoWhitespace ::= UTF8String (FROM(
         {0, 0, 0, 33} .. {0, 0, 215, 255} |
         {0, 0, 224, 0} .. {0, 0, 255, 253} |
         {0, 1, 0, 0} .. {0, 16, 255, 253}))
XMLStringWithNoCRLFHT ::= UTF8String (FROM(
         {0, 0, 0, 32} .. {0, 0, 215, 255} |
         {0, 0, 224, 0} .. {0, 0, 255, 253} |
         {0, 1, 0, 0} .. {0, 16, 255, 253}))
/* ACH.1 type definitions supporting the mapping of W3C XML
Schema built-in date
and time types */
GenericTimeTypeChoice {BasicType, Alternative} ::= CHOICE {
                asn1supportedvalue BasicType,
                othervalues Alternative}
                (CONSTRAINED BY
                 {/* The "othervalues" alternative shall not be used
for abstract
                  values in the "asn1supportedvalue"
alternative */})
```

```
DateTimeType ::= VisibleString (FROM ("0".."9" | "TZ:.+-"))
                (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.7
*/})
Day ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
                    (CONSTRAINED BY {/* W3C XML Schema Part 2,
3.2.13 */})
Month ::= DateTimeType (FROM ("0".."9" | "Z:+-"))
     (CONSTRAINED BY {/* W3C XML Schema Part 2, 3.2.14 */})
MonthDay ::= DateTimeType(FROM ("0".."9" | "Z:+-"))
                    (CONSTRAINED BY {/* W3C XML Schema Part 2,
3.2.12 */})
ENCODING-CONTROL XER
               GLOBAL-DEFAULTS MODIFIED-ENCODINGS
               GLOBAL-DEFAULTS CONTROL-NAMESPACE
                       "gost.e-ecolog.ru                                    "
                       PREFIX "xsi"
               NAMESPACE ALL, ALL IN ALL AS
                       "gost.e-ecolog.ru                          "
                       PREFIX "xsd"
               USE-QNAME QName
               DECIMAL Decimal
               LIST ENTITIES, IDREFS, NMTOKENS
               EMBED-VALUES AnyType, AnyType-nillable
               ANY-ATTRIBUTES AnyType.attr, AnyType-nillable.attr
               ANY-ELEMENT AnyType.elem-list.*, AnyType-
nillable.content.elem-list.*
               UNTAGGED AnyType.elem-list, AnyType-
nillable.content.elem-list
               NAME AnySimpleType, AnyURI, Date, DateTime, Decimal,
               Double,
                   Duration, Float, GDay, GMonth, GMonthDay, GYear,
               GYearMonth,
                   Int, Language, Long,
                        NormalizedString, Short,
                   String, Time, Token,
                   UnsignedInt, UnsignedLong, UnsignedShort
                   AS UNCAPITALIZED
               NAME GenericTimeTypeChoice.ALL AS ""
               USE-NIL AnyType-nillable
               USE-UNION GenericTimeTypeChoice
               WHITESPACE AnyURI, Language, Token, DateTimeType
COLLAPSE
               WHITESPACE NormalizedString REPLACE
     END
```

Приложение С
(справочное)

Идентификация модуля XSD

Следующий идентификатор объекта, OID интернационализированный идентификатор ресурса и значения дескриптора объекта, присваивается в настоящем стандарте для модуля, описывающего типы АСН.1, соответствующие встроенным типам XSD:

```
    { joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2)
version1(1)}
    "/ACH.1/Specification/Modules/XSD-Module/Version1"
    "ACH.1 XSD Module for Version 1 mapping"
    { joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-module(2)
version2(2)}
    "/ACH.1/Specification/Modules/XSD-Module/Version2"
    "ACH.1 XSD Module for Version 2 mapping"
```

Приложение D
(справочное)

Примеры отображения

Это приложение иллюстрирует отображение версии 1, описанное в настоящем стандарте и содержит АСН.1-модуль, соответствующий схеме XSD. Отображение версии 2 аналогичное и отличается (в данном примере) только в использовании XSD-модуля из приложения В вместо XSD-модуля из приложения А.

D.1 Схема, использующая простые определения типа

Следующая схема содержит примеры XSD-встроенных типов (**xsd:string, xsd:decimal, xsd:integer, xsd:int, xsd:date**), других **простых определений типа** и **сложных определений типа.**

```
<?xml version="1.0" encoding="UTF-8"?>
  <xsd:schema xmlns:xsd="gost.e-ecolog.ru
  elementFormDefault="unqualified">
        <xsd:element name="EXAMPLES">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element ref="personnelRecord"/>
                    <xsd:element name="decimal"
                    type="xsd:decimal"/>
                    <xsd:element name="daysOfTheWeek"
                    type="ListOfDays"/>
                    <xsd:element  ref="namesOfMemberNations"/>
                    <xsd:element ref="fileIdentifier"
                    maxOccurs="unbounded"/>
                </xsd:sequence>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="personnelRecord">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="name"
                    type="name"/>
                    <xsd:element name="title"
                    type="xsd:string"/>
                    <xsd:element name="decimal"
                    type="xsd:integer"/>
                    <xsd:element name="dateOfHire"
                    type="xsd:date"/>
                    <xsd:element ref="nameOfSpouse"/>
                    <xsd:element ref="children"/>
                </xsd:sequence>
            </xsd:complexType>
                </xsd:element>
        <xsd:element name="nameOfSpouse" type="name"/>
        <xsd:complexType name="name">
            <xsd:sequence>
                <xsd:element name="givenName" type="xsd:string"/>
                <xsd:element name="initial" type="xsd:string"/>
                <xsd:element name="familyName" type="xsd:string"/>
            </xsd:sequence>
        </xsd:complexType>
        <xsd:element name="children">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element ref="ChildInformation"
                    minOccurs="0"
```

```
maxOccurs="unbounded"/>
                    </xsd:sequence>
                </xsd:complexType>
        </xsd:element>
        <xsd:element name="ChildInformation">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="name"
                        type="name"/>
                        <xsd:element name="dateOfBirth"
                        type="xsd:date"/>
                    </xsd:sequence>
                </xsd:complexType>
        </xsd:element>
        <xsd:simpleType name="ListOfDays">
                <xsd:list itemType="Day"/>
        </xsd:simpleType>
        <xsd:simpleType name="Day">
                <xsd:restriction base="xsd:string">
                    <xsd:enumeration value="monday"/>
                    <xsd:enumeration value="tuesday"/>
                    <xsd:enumeration value="wednesday"/>
                    <xsd:enumeration value="thursday"/>
                    <xsd:enumeration value="friday"/>
                    <xsd:enumeration value="saturday"/>
                    <xsd:enumeration value="sunday"/>
                </xsd:restriction>
        </xsd:simpleType>
        <xsd:element name="namesOfMemberNations">
                <xsd:simpleType>
                    <xsd:list itemType="xsd:string"/>
                </xsd:simpleType>
        </xsd:element>
        <xsd:element name="fileIdentifier">
                <xsd:complexType>
                    <xsd:choice>
                        <xsd:element name="serialNumber"
                        type="xsd:int"/>
                        <xsd:element name="relativeName"
                        type="xsd:string"/>
                        <xsd:element ref="unidentified"/>
                    </xsd:choice>
                </xsd:complexType>
        </xsd:element>
        <xsd:element name="unidentified" type="xsd:anyType"/>
</xsd:schema>
```

D.2 Соответствующие ACH.1-определения

Далее приводится соответствующая ACH.1-спецификация и проверка тех же XML-документов, что и в XSD-схеме:

```
EXAMPLES{joint-iso-itu-t asn1(1) examples(999) xml-defined-
types(3)}
"ACH.1/Examples/XML-defined-types"
DEFINITIONS
XER INSTRUCTIONS AUTOMATIC TAGS ::=
BEGIN
IMPORTS String, Decimal, Int, Date, AnyType
FROM XSD
   {joint-iso-itu-t asn1(1) specification(0) modules(0) xsd-
module(2) version1(1)};
/* For a Version 2 mapping, the last component of the module
identifier would be
```

```
version2(2) */
EXAMPLES ::= SEQUENCE {
      personnelRecord PersonnelRecord,
      number Decimal,
      daysOfTheWeek ListOfDays,
      namesOfMemberNations NamesOfMemberNations,
      fileIdentifier-list [UNTAGGED]
         SEQUENCE (SIZE(1..MAX)) OF fileidentifier
FileIdentifier }
PersonnelRecord ::= [NAME AS UNCAPITALIZED] SEQUENCE {
      name Name,
      title XSD.String,
      number INTEGER,
      dateOfHire XSD.Date,
      nameOfSpouse NameOfSpouse,
      children Children }
NameOfSpouse ::= [NAME AS UNCAPITALIZED] Name
Name ::= [NAME AS UNCAPITALIZED] SEQUENCE {
      givenName XSD.String,
      initial XSD.String,
      familyName XSD.String }
Children ::= [NAME AS UNCAPITALIZED] SEQUENCE {
     childInformation-list [UNTAGGED]
        SEQUENCE OF
        childInformation [NAME AS CAPITALIZED] ChildInformation
}
ChildInformation ::= SEQUENCE {
   name Name,
   dateOfBirth XSD.Date }
ListOfDays ::= [LIST] SEQUENCE OF Day
Day ::= ENUMERATED
     { friday, monday, saturday, sunday, thursday, tuesday,
wednesday }
     - - Note that 12.4.1.3 specifies use of a lexicographical
order, as
     - - the members of an enumeration are not ordered in an XML
Schema
NamesOfMemberNations ::= [NAME AS UNCAPITALIZED] [LIST] SEQUENCE
OF
        XSD.String (FROM({0, 0, 0, 33) .. {0, 16, 255, 253}))
FileIdentifier ::= [NAME AS UNCAPITALIZED] SEQUENCE {
       choice [UNTAGGED] CHOICE {
           serialNumber XSD.Int,
           relativeName XSD.String,
           unidentified Unidentified } }
Unidentified ::= [NAME AS UNCAPITALIZED] XSD.AnyType
ENCODING-CONTROL XER
        GLOBAL-DEFAULTS MODIFIED-ENCODINGS
        GLOBAL-DEFAULTS CONTROL-NAMESPACE
                "gost.e-ecolog.ru                                  " PREFIX
"xsi"
        TEXT Day:ALL
     END
```

D.3 Дополнительные примеры

В D.3 все частные примеры (примеры, которые не содержат элемент **схемы**) предполагают, что XML-элементы, представляющие синтаксис XSD, находятся в рамках декларации **пространства имен по умолчанию (default namespace), именем пространства имен** которой является **целевое пространство имен** схемы.

D.3.1 Документы схемы с объектами информации элемента import и include

Следующая схема XSD состоит из двух пространств имен, которые состоят из четырех файлов схемы:

```
<!- - file "gost.e-ecolog.ru                    " - ->
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                    "
```

```
        xmlns:xyz="gost.e-ecolog.ru                    "
        targetNamespace="gost.e-ecolog.ru                    ">
        <xsd:element name="xyz-elem" type="xsd:string"/>
        <xsd:complexType name="Xyz-type">
            <xsd:attribute name="xyz-attr" type="xsd:boolean"/>
        </xsd:complexType>
</xsd:schema>
<!-- file "gost.e-ecolog.ru                                -->
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
        xmlns:xyz="gost.e-ecolog.ru                    "
        xmlns:abc="gost.e-ecolog.ru                    "
        targetNamespace="gost.e-ecolog.ru                    ">
        <xsd:import namespace="gost.e-ecolog.ru                    "
            schemaLocation="gost.e-ecolog.ru                                    >
        <xsd:include schemaLocation="gost.e-ecolog.ru                                    >
        <xsd:include schemaLocation="gost.e-ecolog.ru                                    >
        <xsd:element name="abc-elem" type="xyz:Xyz-type"/>
</xsd:schema>
<!-- file "gost.e-ecolog.ru                        .xsd" -->
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
        xmlns:abc="gost.e-ecolog.ru                    "
        targetNamespace="gost.e-ecolog.ru                    ">
        <xsd:element name="sub1-elem" type="xsd:string"/>
</xsd:schema>
<!-- file "gost.e-ecolog.ru                                "-->
<xsd:schema xmlns:xsd="gost.e-ecolog.ru                                    "
        xmlns:abc="gost.e-ecolog.ru                    "
        targetNamespace="gost.e-ecolog.ru                    ">
        <xsd:element name="sub2-elem " type="xsd:string"/>
        <xsd:attribute name="sub2-attr" type="xsd:string"/>
</xsd:schema>
```

Эти четыре документа схемы отображаются в следующих двух модулях АСН.1:

```
XYZ - - The module reference is not standardized
DEFINITIONS XER INSTRUCTIONS AUTOMATIC TAGS ::=
BEGIN
IMPORTS
String FROM XSD{joint-iso-itu-t asn1(1) specification(0)
modules(0)
                    xsd-module(2) version1(1)}
                /* For a Version 2 mapping, the last component of
the module
                 identifier would be version2(2) */
;
Xyz-elem ::= [NAME AS UNCAPITALIZED] XSD.String
Xyz-type ::= SEQUENCE {
     xyz-attr [ATTRIBUTE] BOOLEAN OPTIONAL }
ENCODING-CONTROL XER
       GLOBAL-DEFAULTS MODIFIED-ENCODINGS
       GLOBAL-DEFAULTS CONTROL-NAMESPACE
              "gost.e-ecolog.ru                                "
              PREFIX "xsi"
NAMESPACE ALL AS "gost.e-ecolog.ru
       PREFIX "xyz"
END
ABC - - The module reference is not standardized
DEFINITIONS XER INSTRUCTIONS AUTOMATIC TAGS ::=
BEGIN
IMPORTS
Xyz-type FROM XYZ
String FROM XSD {joint-iso-itu-t asn1(1) specification(0)
modules(0)
```

```
                xsd-module(2) version1(1)}
              /* For a Version 2 mapping, the last component of
the module
                identifier would be version2(2) */
;
Abc-elem ::= [NAME AS UNCAPITALIZED] Xyz-type
Sub1-elem ::= [NAME AS UNCAPITALIZED] XSD.String
Sub2-elem ::= [NAME AS UNCAPITALIZED] XSD.String
Sub2-attr ::= [NAME AS UNCAPITALIZED] [ATTRIBUTE] XSD.String
ENCODING-CONTROL XER
        GLOBAL-DEFAULTS MODIFIED-ENCODINGS
        GLOBAL-DEFAULTS CONTROL-NAMESPACE
                "gost.e-ecolog.ru                                    "
                PREFIX "xsi"
        NAMESPACE ALL AS "gost.e-ecolog.ru                  "
                PREFIX "abc"
END
```

D.3.2 Отображение простых определений типа

D.3.2.1 Простое определение типа, полученное ограничением

Для полного набора примеров простых ограничений типа см. примеры фасетов в D.3.3.

D.3.2.2 Простое определение типа, полученное по списку

```
<xsd:simpleType name="Int-list">
      <xsd:list itemType="xsd:integer"/>
</xsd:simpleType>
<xsd:simpleType name="Int-10-to-100-list">
      <xsd:list>
            <xsd:simpleType>
                  <xsd:restriction base="xsd:integer">
                        <xsd:minInclusive value="10"/>
                        <xsd:maxInclusive value="100"/>
                  </xsd:restriction>
            </xsd:simpleType>
      </xsd:list>
</xsd:simpleType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа АСН.1:

```
Int-list ::= [LIST] SEQUENCE OF INTEGER
Int-10-to-100-list ::= [LIST] SEQUENCE OF INTEGER (10..100)
```

D.3.2.3 Простое определение типа, полученное совокупностью

```
<xsd:simpleType name="Int-or-boolean">
      <xsd:union memberType="xsd:integer xsd:boolean"/>
</xsd:simpleType>
<xsd:simpleType name="Time-or-int-or-boolean- -or-dateRestriction">
      <xsd:union memberType=" xsd:time Int-or-boolean">
            <xsd:simpleType>
                  <xsd:restriction base="xsd:date">
                        <xsd:minInclusive value="2003-01-01"/>
                  </xsd:restriction>
            </xsd:simpleType>
      </xsd:union>
</xsd:simpleType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа АСН.1:

```
Int-or-boolean ::= [USE-UNION] CHOICE {
      integer [NAMESPACE "gost.e-ecolog.ru                    "]
INTEGER,
      boolean [NAMESPACE "gost.e-ecolog.ru                    "]
BOOLEAN }
Time-or-int-or-boolean-or-dateRestriction ::= [USE-UNION] CHOICE
{
      time [NAMESPACE "gost.e-ecolog.ru                    "]
XSD.Time,
      integer [NAMESPACE "gost.e-ecolog.ru                    "]
INTEGER,
```

```
     boolean [NAMESPACE "gost.e-ecolog.ru                    "]
BOOLEAN,
     alt [NAME AS ""] XSD.Date (CONSTRAINED BY
       { /* minInclusive="2003-01-01" */ }) }
```

D.3.2.4 Отображение иерархий развития типа для простых определений типа

```
<xsd:simpleType name="Int-10-to-50">
        <xsd:restriction base="xsd:integer">
                <xsd:minExclusive value="10"/>
                <xsd:maxExclusive value="50"/>
        </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Ten-multiples">
        <xsd:restriction base="Int-10-to-50">
                <xsd:enumeration value="20"/>
                <xsd:enumeration value="30"/>
                <xsd:enumeration value="40"/>
        </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Twenty-multiples">
        <xsd:restriction base="Ten-multiples">
                <xsd:pattern value=".*[02468]0|0"/>
        </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="Stock-level">
        <xsd:simpleContent>
                <xsd:extension base="Int-10-to-50">
                        <xsd:attribute name="procurement" type="Int-10-to-50"/>
                </xsd:extension>
        </xsd:simpleContent>
</xsd:complexType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа АСН.1:

```
Int-10-to-50 ::= INTEGER (10<..<50)
Ten-multiples ::= [USE-NUMBER] ENUMERATED {int20(20), int30(30),
int40(40)}
Twenty-multiples ::= [USE-NUMBER] ENUMERATED {int20(20),
int40(40)}
Stock-level ::= SEQUENCE {
     procurement [ATTRIBUTE] Int-10-to-50 OPTIONAL,
     base [UNTAGGED] Int-10-to-50 }
Ten-multiples-derivations ::= [USE-TYPE] CHOICE {
     ten-multiples [NAME AS CAPITALIZED] Ten-multiples,
     twenty-multiples [NAME AS CAPITALIZED] Twenty-multiples }
Int-10-to-50-derivations ::= [USE-TYPE] CHOICE {
     int-10-to-50 [NAME AS CAPITALIZED] Int-10-to-50,
     stock-level [NAME AS CAPITALIZED] Stock-level,
     ten-multiples [NAME AS CAPITALIZED] Ten-multiples,
     twenty-multiples [NAME AS CAPITALIZED] Twenty-multiples }
```

если и только если:

а) **простое определение типа** «**Int-10-to-50**» происходит как **определение типа,** по крайней мере, одного **объявления элемента** (не показано в примере), которое отображается в АСН.1;

б) **простое определение типа** «**Ten-multiples**» происходит как **определение типа,** по крайней мере, одного **объявления элемента** (не показано в примере), которое отображается в АСН.1, и

в) нет никаких других компонентов схемы, отображенных в АСН.1, которые формируют имена ссылок типов АСН.1 `Int-10-to-50`, `Ten-multiples`, `Twenty-multiples`, `Stock-level`, `Tenmultiples-derivations` и `Int-10-to-50-derivations`.

D.3.3 Отображение фасетов

D.3.3.1 length, minLength и maxLength

```
<xsd:simpleType name="String-10">
        <xsd:restriction base="xsd:string">
                <xsd:length value="10"/>
        </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="String-5-to-10">
```

```
            <xsd:restriction base="xsd:string">
                  <xsd:minLength value="5"/>
                  <xsd:maxLength value="10"/>
            </xsd:restriction>
      </xsd:simpleType>
```

Эти два **простых определения типа** отображаются в следующие присвоения типа АСН.1:

```
String-10 ::= XSD.String (SIZE(10))
String-5-to-10 ::= XSD.String (SIZE(5..10))
```

D.3.3.2 pattern

```
<xsd:simpleType name="My-filename">
      <xsd:restriction base="xsd:string">
            <xsd:pattern value="[&#x20;-&#xFF;]*"/>
            <xsd:pattern value="/?([^/]*/)*[^/]*/*" />
      </xsd:restriction>
</xsd:simpleType>
```

Это **простое определение типа** отображается в следующее присвоение типа АСН.1:

```
My-filename ::= XSD.String
      (CONSTRAINED BY
            {/* XML representation of the XSD pattern
             "[&#x20;-&#xFF;]*" "/?([^/]*&#x2F;)*[^/]*&#x2F;*" */})
```

D.3.3.3 whiteSpace

```
<xsd:simpleType name="My-String">
      <xsd:restriction base="xsd:string">
            <xsd:whiteSpace value="preserve"/>
      </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="My-NormalizedString">
      <xsd:restriction base="xsd:string">
            <xsd:whiteSpace value="replace"/>
      </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="My-TokenString">
      <xsd:restriction base="xsd:string">
            <xsd:whiteSpace value="collapse"/>
      </xsd:restriction>
</xsd:simpleType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа АСН.1:

```
My-String ::= XSD.String
My-NormalizedString ::= [WHITESPACE REPLACE] XSD.String
      (FROM ({0, 0, 0, 32} .. {0, 16, 255, 255}))
My-TokenString ::= [WHITESPACE COLLAPSE] XSD.String
      (FROM ({0, 0, 0, 32} .. {0, 16, 255, 255}))
      (PATTERN "([^ ]([^ ]| [^ ])*)?")
```

D.3.3.4 maxInclusive, maxExclusive, minExclusive и minInclusive

```
<xsd:simpleType name="Int-10-to-100">
      <xsd:restriction base="xsd:integer">
            <xsd:minExclusive value="10"/>
            <xsd:maxInclusive value="100"/>
      </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Pi-approximation">
      <xsd:restriction base="xsd:double">
            <xsd:minExclusive value="3.14159"/>
            <xsd:maxExclusive value="3.1416"/>
      </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="Morning">
      <xsd:restriction base="xsd:time">
            <xsd:minInclusive value="00:00:00"/>
            <xsd:maxExclusive value="12:00:00"/>
      </xsd:restriction>
</xsd:simpleType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа ACH.1:

```
Int-10-to-100 ::= INTEGER (10<..100)
Pi-approximation ::= XSD.Double (3.14159<..<3.1416)
Morning ::= XSD.Time (CONSTRAINED BY
    {/* minInclusive="00:00:00" maxExclusive="12:00:00" */ })
```

D.3.3.5 totalDigits и fractionDigits

```
<xsd:simpleType name="RefundableExpenses">
    <xsd:restriction base="xsd:decimal">
        <xsd:totalDigits value="5"/>
        <xsd:fractionDigits value="2"/>
    </xsd:restriction>
</xsd:simpleType>
```

Это **простое определение типа** отображается в следующее присвоение типа ACH.1:

```
RefundableExpenses ::= XSD.Decimal (CONSTRAINED BY
    {/* totalDigits="5" fractionDigits="2" */ })
```

D.3.3.6 enumeration

```
<xsd:simpleType name="FarmAnimals">
    <xsd:restriction base="xsd:normalizedString">
        <xsd:enumeration value="Horse"/>
        <xsd:enumeration value="Bull"/>
        <xsd:enumeration value="Cow"/>
        <xsd:enumeration value="Pig"/>
        <xsd:enumeration value="Duck"/>
        <xsd:enumeration value="Goose"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PrimeNumbersBelow30">
    <xsd:restriction base="xsd:integer">
        <xsd:enumeration value="2"/>
        <xsd:enumeration value="3"/>
        <xsd:enumeration value="5"/>
        <xsd:enumeration value="7"/>
        <xsd:enumeration value="11"/>
        <xsd:enumeration value="13"/>
        <xsd:enumeration value="17"/>
        <xsd:enumeration value="19"/>
        <xsd:enumeration value="23"/>
        <xsd:enumeration value="29"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="X680-release">
    <xsd:restriction base="xsd:gYearMonth">
        <xsd:enumeration value="2002-07"/>
        <xsd:enumeration value="1997-12"/>
        <xsd:enumeration value="1994-07"/>
    </xsd:restriction>
</xsd:simpleType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа ACH.1:

```
FarmAnimals ::= [WHITESPACE REPLACE]
    - - This encoding instruction ensures that an enumeration such as
    - - value="Slow Loris" (containing a space) is encoded correctly.
    ENUMERATED { bull, cow, duck, goose, horse, pig }
PrimeNumbersBelow30 ::= [USE-NUMBER] ENUMERATED {int2(2), int3(3), int5(5),
    int7(7), int11(11), int13(13), int17(17), int19(19), int23(23), int29(29)}
X680-release ::= XSD.GYearMonth ("2002-07" | "1997-12" | "1994-07")
```

Следующая команда кодирования включена в секцию контроля кодирования XER:

```
TEXT FarmAnimals:ALL AS CAPITALIZED
```

D.3.3.7 enumeration в сочетании с другими фасетами

Следующие примеры основаны на наследование фасетов, использующих ограничение некоторых типов, определенных в D.3.3.6.

```
<xsd:simpleType name="FarmAnimals-subset">
        <xsd:restriction base="FarmAnimals">
                <xsd:minLength value="4"/>
                <xsd:pattern value="[^oe]*"/>
        </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="PrimeNumbersBelow30-subset">
        <xsd:restriction base="PrimeNumbersBelow30">
                <xsd:minExclusive value="5"/>
                <xsd:pattern value=".*[23].*"/>
        </xsd:restriction>
</xsd:simpleType>
```

Эти **простые определения типа** отображаются в следующие присвоения типа АСН.1:

```
/* Horse and Goose do not satisfy the pattern facet
   Cow and Pig do not satisfy the minLength facet */
FarmAnimals-subset ::= [WHITESPACE REPLACE] ENUMERATED {bull, duck}
/* 2, 3 and 5 do not satisfy the minExclusive facet
   2, 5, 7, 11, 17 and 19 do not satisfy the pattern facet */
PrimeNumbersBelow30-subset ::= [USE-NUMBER] ENUMERATED {int13(13), int23(23),
                                                        int29(29)}
```

Следующая инструкция кодирования включена в секцию контроля кодирования XER:

```
TEXT FarmAnimals-subset:ALL AS CAPITALIZED
```

D.3.4 Отображение объявлений элемента

D.3.4.1 Объявления элемента, определением типа которых является высокоуровневое простое определение типа или сложное определение типа

```
<xsd:element name="Forename" type="xsd:token"/>
<xsd:element name="File" type="My-filename"/>
<xsd:element name="Value" type="Int-10-to-50"/>
```

Эти **объявления элемента** отображаются в следующие присвоения типа АСН.1:

```
Forename ::= XSD.Token
File ::= My-filename
Value ::= Int-10-to-50-derivations
```

Примечание — Тип «**My-filename**» и его отображение в АСН.1 описано в D.3.3.2; тип «**Int-10-to-50**» и его отображение в АСН.1 описано в D.3.2.4.

D.3.4.2 Объявления элемента, определение типа которых анонимное простое определение типа или сложное определение типа

```
<xsd:element name="maxOccurs">
        <xsd:simpleType>
                <xsd:union memberTypes="xsd:nonNegativeInteger">
                        <xsd:simpleType>
                                <xsd:restriction base="xsd:token">
                                        <xsd:enumeration value="unbounded"/>
                                </xsd:restriction>
                        </xsd:simpleType>
                </xsd:union>
        </xsd:simpleType>
</xsd:element>
<xsd:element name=»address»>
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="line-1" type="xsd:token"/>
                        <xsd:element name="line-2" type=»xsd:token"/>
                        <xsd:element name="city" type="xsd:token"/>
                        <xsd:element name="state" type="xsd:token" minOccurs="0"/>
                        <xsd:element name="zip" type="xsd:token"/>
                </xsd:sequence>
                <xsd:attribute name="country" type="xsd:token"/>
        </xsd:complexType>
</xsd:element>
```

Эти **объявления элемента** отображаются в следующие присвоения типа ACH.1:

```
MaxOccurs ::= [NAME AS UNCAPITALIZED] [USE-UNION] CHOICE {
    nonNegativeInteger [NAMESPACE AS
"gost.e-ecolog.ru                    "]
                                        INTEGER (0..MAX),
    alt [NAME AS "" ] ENUMERATED {unbounded} }
Address ::= [NAME AS UNCAPITALIZED] SEQUENCE {
    country [ATTRIBUTE] XSD.Token OPTIONAL,
    line-1 XSD.Token,
    line-2 XSD.Token,
    city XSD.Token,
    state XSD.Token OPTIONAL,
    zip XSD.Token }
```

D.3.4.3 Объявления элемента, которые являются головными элементами группы замены элементов

```
<xsd:element name="Tic" type="xsd:integer" abstract="true"/>
<xsd:element name="Tac" type="xsd:byte" substitutionGroup="Tic"/>
<xsd:element name="Toe" substitutionGroup="Tic"/>
<xsd:element name="Foo" type="xsd:date"/>
<xsd:element name="Bar" substitutionGroup="Foo"/>
```

Эти **объявления элемента** отображаются в:

```
Tac ::= INTEGER (-128..127)
Toe ::= INTEGER
Tic-group ::= [UNTAGGED] CHOICE {
    tac [NAME AS CAPITALIZED] Tac,
    toe [NAME AS CAPITALIZED] Toe }
Foo ::= XSD.Date
Bar ::= XSD.Date
Foo-group ::= [UNTAGGED] CHOICE {
    bar [NAME AS CAPITALIZED] Bar,
    foo [NAME AS CAPITALIZED] Foo }
```

если и только если:

а) **объявление элемента** «Tic» присутствует как **терм** хотя бы одной **частицы** (не показано в примере), которая отображена в ACH.1;

б) **объявление элемента** «Foo» присутствует как **терм** хотя бы одной **частицы** (не показано в примере), которая отображена в ACH.1, и

в) нет никаких других компонентов схемы, отображенных в ACH.1, которые формируют имена ссылок типов ACH.1 `Tac`, `Toe`, `Foo`, `Bar`, `Tic-group` и `Foo-group`.

D.3.4.4 Объявления элемента с ограничением значения, которое является значением по умолчанию

D.3.4.4.1 Следующее **объявление элемента** с анонимным **простым определением типа** не используется в качестве **базового определения типа** любого типа.

```
<xsd:element name="Telephone" type="xsd:token" default="undefined"/>
```

Это **объявление элемента** отображается в следующее присвоение типа ACH.1:

```
Telephone ::= [DEFAULT-FOR-EMPTY AS "undefined"] XSD.Token
```

D.3.4.4.2 Следующее **объявление элемента** с анонимным **сложным определением типа** с простым содержанием не используется в качестве **базового определения типа** любого типа.

```
<xsd:element name="InternationalTelephone" default="undefined">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:token">
                <xsd:attribute name="country-code" type="xsd:integer"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
```

Это **объявление элемента** отображается в следующее присвоение типа ACH.1:

```
InternationalTelephone ::= [DEFAULT-FOR-EMPTY AS "undefined"]
SEQUENCE {
    country-code [ATTRIBUTE] INTEGER OPTIONAL,
    base [UNTAGGED] XSD.Token }
```

D.3.4.4.3 Следующее **объявление элемента** с анонимным **сложным определением типа. Сложное определение типа** имеет сложное содержимое, которое является **смешанным** и опустошаемым (emptiable) и не используется в качестве **базового определения типа** любого типа.

```
<xsd:element name="Description" default="absent">
        <xsd:complexType mixed="true">
                <xsd:choice minOccurs="0" maxOccurs="unbounded">
                        <xsd:element name="bold" type="xsd:string"/>
                        <xsd:element name="italic" type="xsd:string"/>
                </xsd:choice>
        </xsd:complexType>
</xsd:element>
```

Это **объявление элемента** отображается в следующее присвоение типа ACH.1:

```
Description ::= [EMBED-VALUES] [DEFAULT-FOR-EMPTY AS "absent"]
SEQUENCE {
     embed-values SEQUENCE OF XSD.String,
     choice-list [UNTAGGED] SEQUENCE OF [UNTAGGED] CHOICE {
         bold XSD.String,
         italic XSD.String } } (CONSTRAINED BY
      {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
```

D.3.4.4.4 **Определение типа объявления элемента** в следующем примере используется в качестве **базового определения типа** другого типа.

Этот пример использует XSD- и ACH.1-типы примера в D.3.2.4.

```
<xsd:element name="Quantity" type="Int-10-to-50" default="20"/>
```

Это **объявление элемента** отображается в следующее присвоение типа ACH.1:

```
Quantity ::= Int-10-to-50-deriv-default-20
```

Если нет типа ACH.1, соответствующего **Int-10-to-50**, со значением по умолчанию «20», уже сформированного, то следующий тип формируется так же:

```
Int-10-to-50-deriv-default-20 ::= [USE-TYPE] CHOICE {
     int-10-to-50 [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS 20]
                                Int-10-to-50,
     stock-level [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS 20]
                                Stock-level,
     ten-multiples [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS int20]
                                Ten-multiples,
     twenty-multiples [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS int20]
                                Twenty-multiples }
```

D.3.4.5 Объявление элемента с ограничением значения, которое является фиксированным значением

D.3.4.5.1 Следующее **объявление элемента** с анонимным **простым определением типа**, которое не используются в качестве **базового определения типа** любого типа.

```
<xsd:element name="UnknownTelephone" type="xsd:token" fixed="undefined"/>
```

Это **объявление элемента** отображается в следующее присвоение типа ACH.1:

```
UnknownTelephone ::= [DEFAULT-FOR-EMPTY AS "undefined"] XSD.Token
("undefined")
```

D.3.4.5.2 Следующее **объявление элемента** с анонимным **сложным определением типа. Сложное определение типа** имеет простое содержимое и не используется в качестве **базового определения типа** любого типа.

```
<xsd:element name="UnknownInternationalTelephone" fixed="undefined">
        <xsd:complexType>
                <xsd:simpleContent>
                        <xsd:extension base="xsd:token">
                                <xsd:attribute name="country-code" type="xsd:integer"/>
                        </xsd:extension>
                </xsd:simpleContent>
        </xsd:complexType>
</xsd:element>
```

Это **объявление элемента** отображается в следующее присвоение типа ACH.1:

```
UnknownInternationalTelephone ::= [DEFAULT-FOR-EMPTY AS "undefined"]
SEQUENCE {
       country-code [ATTRIBUTE] INTEGER OPTIONAL,
       base [UNTAGGED] XSD.Token }
            (WITH COMPONENTS {..., base ("undefined")})
```

D.3.4.5.3 Следующее **объявление элемента** с анонимным **сложным определением типа. Сложное определение типа** имеет сложное содержимое, которое является смешанным и опустошаемым и не используется в качестве **базового определения типа** любого типа.

```
<xsd:element name="UnknownDescription" fixed="absent">
        <xsd:complexType mixed="true">
                <xsd:choice minOccurs="0" maxOccurs="unbounded">
                        <xsd:element name="bold" type="xsd:string"/>
                        <xsd:element name="italic" type="xsd:string"/>
                </xsd:choice>
        </xsd:complexType>
</xsd:element>
```

Это **объявление элемента** отображается в следующее присвоение типа АСН.1:

```
UnknownDescription ::= [EMBED-VALUES] [DEFAULT-FOR-EMPTY AS "absent"]
SEQUENCE {
embed-values SEQUENCE OF XSD.String,
choice-list [UNTAGGED] SEQUENCE OF [UNTAGGED] CHOICE {
     bold XSD.String,
     italic XSD.String } }
(CONSTRAINED BY
        {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
(WITH COMPONENTS {embed-values ({"absent"}),
                  choice-list (SIZE(0))})
```

D.3.4.5.4 **Определением типа** следующего **объявления элемента** является **простое определение типа**, используемое в качестве **базового определения типа** другого типа.

Этот пример использует XSD- и АСН.1-типы примера в D.3.2.4.

```
<xsd:element name="Quantity" type="Int-10-to-50" fixed="20"/>
```

Это **объявление элемента** отображается в следующее присвоение типа АСН.1:

```
Quantity ::= Int-10-to-50-deriv-fixed-20
```

Если нет типа АСН.1, соответствующего **Int-10-to-50** с фиксированным значением «20», уже сформированного, следующий тип формируется так же:

```
Int-10-to-50-deriv-fixed-20 ::= [USE-TYPE] CHOICE {
     int-10-to-50 [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS 20]
                        Int-10-to-50,
     stock-level [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS 20]
                        Stock-level,
     ten-multiples [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS
int20]
                        Ten-multiples,
     twenty-multiples [NAME AS CAPITALIZED] [DEFAULT-FOR-EMPTY AS
int20]
                        Twenty-multiples }
(WITH COMPONENTS {
     int-10-to-50 (20),
     stock-level (WITH COMPONENTS {..., base (20)}),
     ten-multiples (int20),
     twenty-multiples (int20) })
```

D.3.4.6 Объявления элемента, которые являются обнуляемыми

D.3.4.6.1 В следующем примере показано **объявление элемента**, которое **обнуляемо** и чьим **определением типа** является XSD-встроенный тип.

```
<xsd:element name="Nillable-1" type="xsd:string" nillable="true"/>
```

Это **объявление элемента** отображается в следующее присвоение типа АСН.1:

```
Nillable-1 ::= [USE-NIL] SEQUENCE {
     content XSD.String OPTIONAL }
```

D.3.4.6.2 В следующем примере показано **объявление элемента**, которое **обнуляемо** и чьим **определением типа** является анонимное **сложное определение типа.**

```
<xsd:element name="Nillable-2" nillable="true">
        <xsd:complexType>
                <xsd:sequence>
                        <xsd:element name="a" type="xsd:string"/>
                        <xsd:element name="b" type="xsd:string"/>
```

```
                </xsd:sequence>
                <xsd:attribute name="b" type="xsd:boolean"/>
        </xsd:complexType>
</xsd:element>
```

Это **объявление элемента** отображается в следующее присвоение типа АСН.1:

```
Nillable-2 ::= [USE-NIL] SEQUENCE {
     b [ATTRIBUTE] BOOLEAN OPTIONAL,
     content SEQUENCE {
            a XSD.String,
            b XSD.String } OPTIONAL }
```

D.3.4.6.3 В следующем примере показано **объявление элемента,** которое **обнуляемо** и чьим **определением типа** является высокоуровневое **сложное определение типа.**

```
<xsd:complexType name="Foo">
        <xsd:sequence>
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:string"/>
        </xsd:sequence>
        <xsd:attribute name="b"type="xsd:boolean"/>
</xsd:complexType>
<xsd:element name="Nillable-3" type="Foo" nillable="true"/>
```

Эти компоненты схемы отображаются в следующие присвоения типа АСН.1:

```
Foo ::= SEQUENCE {
     b [ATTRIBUTE] BOOLEAN OPTIONAL,
     a XSD.String,
     b-1 [NAME AS "b"] XSD.String }
Foo-nillable ::= [USE-NIL] SEQUENCE {
     b [ATTRIBUTE] BOOLEAN OPTIONAL,
     content SEQUENCE {
         a XSD.String,
         b XSD.String } OPTIONAL }
Nillable-3 ::= Foo-nillable
```

D.3.4.6.4 В следующем примере показано **объявление элемента,** которое **обнуляемо** и чьим **определением типа** является высокоуровневое **сложное определения типа,** используемое в качестве **базового определения типа** другого **сложного определения типа.**

Следующие компоненты схемы определяются в дополнение к компонентам схемы из D.3.4.6.3:

```
<xsd:complexType name="Bar">
        <xsd:complexContent>
                <xsd:extension base="Foo">
                        <xsd:sequence>
                                <xsd:element name="z" type="xsd:string"/>
                        </xsd:sequence>
                        <xsd:attribute name="c" type="xsd:boolean"/>
                </xsd:extension>
        </xsd:complexContent>
</xsd:complexType>
<xsd:element name="Nillable-4" type="Foo" nillable="true"/>
```

В дополнение к типу Foo из D.3.4.6.3, формируются следующие типы АСН.1:

```
Bar ::= SEQUENCE {
     b [ATTRIBUTE] BOOLEAN OPTIONAL,
     c [ATTRIBUTE] BOOLEAN OPTIONAL,
     a XSD.String,
     b-1 [NAME AS "b"] XSD.String,
     z XSD.String }
Foo-nillable ::= [USE-NIL] SEQUENCE {
     b [ATTRIBUTE] BOOLEAN OPTIONAL,
     content SEQUENCE {
          a XSD.String,
          b XSD.String } OPTIONAL }
Bar-nillable ::= [USE-NIL] SEQUENCE {
     b [ATTRIBUTE] BOOLEAN OPTIONAL,
     c [ATTRIBUTE] BOOLEAN OPTIONAL,
     content SEQUENCE {
```

```
        a XSD.String,
        b XSD.String,
        z XSD.String } OPTIONAL }
Foo-deriv-nillable ::= [USE-TYPE] CHOICE {
    foo [NAME AS CAPITALIZED] Foo-nillable,
    bar [NAME AS CAPITALIZED] Bar-nillable }
Nillable-4 ::= Foo-deriv-nillable
```

D.3.5 Отображение применений атрибута и объявлений атрибута

D.3.5.1 Далее приведен пример высокоуровневого **объявления атрибута, определением типа** которого является высокоуровневое **простое определение типа.**

```
<xsd:attribute name="name" type="xsd:NCName"/>
```

Это **объявление атрибута** отображается в следующее присвоение типа АСН.1:

```
Name ::= [NAME AS UNCAPITALIZED] [ATTRIBUTE] XSD.NCName
```

D.3.5.2 Далее приведен пример высокоуровневого **объявления атрибута, определением типа** которого является анонимное **простое определение типа.**

```
<xsd:attribute name="form">
        <xsd:simpleType>
            <xsd:restriction base="xsd:token">
                <xsd:enumeration value="qualified"/>
                <xsd:enumeration value="unqualified"/>
            </xsd:restriction>
        </xsd:simpleType>
</xsd:attribute>
```

Это **объявление атрибута** отображается в следующее присвоение типа АСН.1:

```
Form ::= [NAME AS UNCAPITALIZED] [ATTRIBUTE] ENUMERATED {qualified,
unqualified}
```

D.3.5.3 В следующем примере приведено **применение атрибута** с **ограничением значения,** которое является значением **по умолчанию.**

Объявление атрибута, чье **имя** является «формой» и на который ссылаются в данном примере, определен в D.3.5.2.

```
<xsd:complexType name="element">
    <xsd:attribute name="name" type="xsd:NCName" default="NAME"/>
    <xsd:attribute ref="form" default="qualified"/>
</xsd:complexType>
```

Это **сложное определение типа** отображается в следующее присвоение типа АСН.1:

```
Element ::= [NAME AS UNCAPITALIZED] SEQUENCE {
    form [ATTRIBUTE] Form DEFAULT qualified,
    name [ATTRIBUTE] XSD.NCName DEFAULT "NAME" }
```

D.3.5.4 Следующий пример показывает высокоуровневое **объявление атрибута** с **ограничением значения,** которое является значением **по умолчанию,** и **применение атрибута** с этим **объявлением атрибута.**

```
<xsd:attribute name="minOccurs" type="xsd:nonNegativeInteger" default="1"/>
<xsd:attribute name="maxOccurs" default="1">
    <xsd:simpleType>
        <xsd:union memberTypes="xsd:nonNegativeInteger" >
            <xsd:simpleType>
                <xsd:restriction base="xsd:NMTOKEN">
                    <xsd:enumeration value="unbounded"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:union>
    </xsd:simpleType>
</xsd:attribute>
    <xsd:complexType name="Particle">
        <xsd:sequence>
            <xsd:element name="particle"/>
        </xsd:sequence>
        <xsd:attribute ref="minOccurs"/>
        <xsd:attribute ref="maxOccurs" default="unbounded"/>
    </xsd:complexType>
```

Эти компоненты схемы отображаются в следующем присвоении типа АСН.1:

```
MinOccurs ::= [ATTRIBUTE] [NAME AS UNCAPITALIZED] INTEGER (0..MAX)
MaxOccurs ::= [ATTRIBUTE] [NAME AS UNCAPITALIZED] [USE-UNION] CHOICE {
        nonNegativeInteger [NAMESPACE AS
        "gost.e-ecolog.ru                                "]
                                INTEGER (0..MAX),
        alt                  [NAME AS ""]
                        [WHITESPACE COLLAPSE] ENUMERATED {unbounded} }
Particle ::= SEQUENCE {
      maxOccurs [ATTRIBUTE] MaxOccurs DEFAULT alt : unbounded,
      minOccurs [ATTRIBUTE] MinOccurs DEFAULT 1,
      particle XSD.AnyType }
```

D.3.5.5 Следующий пример показывает **применение атрибута, объявление атрибута** которого имеет **целевое пространство имен,** не являющееся **отсутствующим.**

```
<xsd:complexType name="Ack">
        <xsd:attribute name="number" type="xsd:integer" form="qualified" />
</xsd:complexType>
```

Это **сложное определение типа** отображается в следующее присвоение типа АСН.1:

```
Ack ::= SEQUENCE {
     number [NAMESPACE AS "gost.e-ecolog.ru                                "]
[ATTRIBUTE]
                          INTEGER OPTIONAL }
```

D.3.6 Отображение определений модельной группы

D.3.6.1 Следующее **определение модельной группы, модельная группа** которого имеет **наборщика последовательности.**

```
<xsd:group name="mySequence">
        <xsd:sequence>
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:boolean"/>
        </xsd:sequence>
</xsd:group>
```

Это **определение модельной группы** отображается в следующее присвоение типа АСН.1:

```
MySequence ::= [UNTAGGED] SEQUENCE {
         a XSD.String,
         b BOOLEAN }
```

D.3.6.2 Следующее **определение модельной группы, модельная группа** которого имеет **наборщика всего.**

```
<xsd:group name="myAll">
        <xsd:all>
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:boolean"/>
        </xsd:all>
</xsd:group>
```

Это **определение модельной группы** не отображаются в АСН.1. Для примера отображения **сложного определения типа,** где **модельная группа** этого **определения модельной группы** происходит как важнейшая **модельной группе,** см. D.3.8.3.1.

D.3.6.3 Следующее **определение модельной группы, модельная группа** которого имеет **наборщика выбора.**

```
<xsd:group name="myChoice">
        <xsd:choice>
                <xsd:element name="am" type="xsd:string"/>
                <xsd:element name="bm" type="xsd:boolean"/>
        </xsd:choice>
</xsd:group>
```

Это **определение модельной группы** отображается в следующее присвоение типа АСН.1:

```
MyChoice ::= [UNTAGGED] CHOICE {
         am XSD.String,
         bm BOOLEAN }
```

D.3.7 Отображение частиц

Определение модельной группы D.3.6.3 и соответствующий тип АСН.1 используются в некоторых примерах **частиц.**

D.3.7.1 В следующем примере показаны **частицы модельной группы** с **наборщиком последовательности.**

```
<xsd:complexType name="ElementSequence">
        <xsd:sequence>
                <xsd:element name="elem1" type="xsd:boolean"/>
                <xsd:element name="elem2" type="xsd:boolean" minOccurs="0"/>
                <xsd:element name="elem3" type="xsd:boolean" minOccurs="2"
                                                        maxOccurs="5"/>
                <xsd:element name="elem4" type="xsd:boolean" minOccurs="0"
                                                        maxOccurs="unbounded"/>
                <xsd:element name="elem5" type="xsd:boolean" minOccurs="5"
                                                        maxOccurs="unbounded"/>
        </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="ModelGroupSequence">
        <xsd:sequence>
                <xsd:group ref="myChoice"/>
                <xsd:choice>
                        <xsd:element name="a" type="xsd:string"/>
                        <xsd:element name="b" type="xsd:string"/>
                </xsd:choice>
                <xsd:sequence>
                        <xsd:element name="c" type="xsd:string"/>
                        <xsd:element name="d" type="xsd:string"/>
                </xsd:sequence>
                <xsd:choice minOccurs="3" maxOccurs="12">
                        <xsd:element name="e" type="xsd:string"/>
                        <xsd:element name="f" type="xsd:string"/>
                </xsd:choice>
        </xsd:sequence>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
ElementSequence ::= SEQUENCE {
      elem1          BOOLEAN,
      elem2          BOOLEAN OPTIONAL,
      elem3-list       [UNTAGGED] SEQUENCE (SIZE(2..5)) OF elem3 BOOLEAN,
      elem4-list       [UNTAGGED] SEQUENCE OF elem4 BOOLEAN,
      elem5-list       [UNTAGGED] SEQUENCE (SIZE(5..MAX)) OF elem5 BOOLEAN }
ModelGroupSequence ::= SEQUENCE {
      myChoice         MyChoice,
      choice            [UNTAGGED] CHOICE {
            a                XSD.String,
            b                XSD.String },
      c                 XSD.String,
      d                 XSD.String,
      choice-list            [UNTAGGED] SEQUENCE (SIZE(3..12)) OF [UNTAGGED]
      CHOICE {
                               e XSD.String,
                               f XSD.String } }
```

D.3.7.2 В следующем примере показаны **частицы модельной группы** с **наборщиком всего.**

```
<xsd:complexType name="ElementAll">
        <xsd:all>
                <xsd:element name="elem1" type="xsd:boolean"/>
                <xsd:element name="elem2" type="xsd:boolean" minOccurs="0"/>
        </xsd:all>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
ElementAll ::= [USE-ORDER] SEQUENCE {
      order SEQUENCE OF ENUMERATED {elem1, elem2},
      elem1 XSD.String,
      elem2 XSD.String OPTIONAL }
```

```
      (CONSTRAINED BY
            {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 35 */})
```

D.3.7.3 В следующем примере показаны **частицы модельной группы** с **наборщиком выбора.**

```
<xsd:complexType name="ElementSequence">
      <xsd:choice>
            <xsd:element name="elem1" type="xsd:boolean"/>
            <xsd:element name="elem2" type="xsd:boolean" minOccurs="0"/>
            <xsd:element name="elem3" type="xsd:boolean» minOccurs="2" maxOccurs="5"/>
            <xsd:element name="elem4" type="xsd:boolean" minOccurs="0" maxOccurs="unbounded"/>
            <xsd:element name="elem5" type="xsd:boolean" minOccurs="5" maxOccurs="unbounded"/>
</xsd:choice>
      </xsd:complexType>
      <xsd:complexType name="ModelGroupChoice">
<xsd:choice>
      <xsd:group ref="myChoice"/>
      <xsd:choice>
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:string"/>
      </xsd:choice>
      <xsd:sequence>
            <xsd:element name="c" type="xsd:string"/>
            <xsd:element name="d" type="xsd:string"/>
      </xsd:sequence>
      <xsd:choice minOccurs="3" maxOccurs="12">
            <xsd:element name="e" type="xsd:string"/>
            <xsd:element name="f" type="xsd:string"/>
       </xsd:choice>
      </xsd:choice>
      </xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа ACH.1:

```
ElementSequence ::= SEQUENCE {
      choice [UNTAGGED] CHOICE {
          elem1            BOOLEAN,
          elem2-list      [UNTAGGED]  SEQUENCE (SIZE(0..1)) OF elem2
BOOLEAN,
          elem3-list       [UNTAGGED]  SEQUENCE (SIZE(2..5)) OF elem3
BOOLEAN,
          elem4-list      [UNTAGGED]  SEQUENCE OF elem4 BOOLEAN,
          elem5-list      [UNTAGGED]   SEQUENCE (SIZE(5..MAX)) OF elem5 BOOLEAN } }
ModelGroupChoice ::= SEQUENCE {
      choice   [UNTAGGED]   CHOICE {
          myChoice       MyChoice,
          choice          UNTAGGED]  CHOICE {
              a                      XSD.String,
              b                      XSD.String },
          sequence      [UNTAGGED] SEQUENCE {
              c                      XSD.String,
              d                     XSD.String },
          choice-list   [UNTAGGED] SEQUENCE (SIZE(3..12)) OF [UNTAGGED]
CHOICE {
              e                      XSD.String,
              f                      XSD.String } } }
```

D.3.8 Отображение сложных определений типа

D.3.8.1 Далее приведен пример **сложного определения типа, тип содержимого** которого **пуст.**

```
<xsd:complexType name="Null"/>
<xsd:complexType name="Ack">
      <xsd:sequence/>
      <xsd:attribute name="packetNumber" type="xsd:integer"/>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа ACH.1:

```
Null ::= SEQUENCE {}
Ack ::= SEQUENCE {
    packetNumber [ATTRIBUTE] INTEGER OPTIONAL }
```

D.3.8.2 Далее приведен пример **сложного определения типа, типом содержимого** которого является **простое определение типа.**

```
<xsd:complexType name="Formatted">
      <xsd:simpleContent>
            <xsd:extension base="xsd:token">
                  <xsd:attribute name="format">
                        <xsd:simpleType>
                              <xsd:restriction base="xsd:token">
                                    <xsd:enumeration value="bold"/>
                                    <xsd:enumeration value="italic"/>
                              </xsd:restriction>
                        </xsd:simpleType>
                  </xsd:attribute>
            </xsd:extension>
      </xsd:simpleContent>
</xsd:complexType>
```

Это **сложное определение типа** отображается в следующее присвоение типа ACH.1:

```
Formatted ::= SEQUENCE {
    format          [ATTRIBUTE]  [WHITESPACE COLLAPSE]
                                                ENUMERATED
{bold, italic} OPTIONAL,
    base            [UNTAGGED]  XSD.Token }
```

D.3.8.3 Далее приведены примеры **сложных определений типа, типом содержимого** которых является модель содержимого **element-only.**

D.3.8.3.1 В следующем примере **типом содержимого** является **модельная группа определения модельной группы.**

В этом примере используются типы, определенные в D.3.6.

```
<xsd:complexType name="MyComplexType-1">
      <xsd:group ref="myAll"/>
</xsd:complexType>
<xsd:complexType name="MyComplexType-2">
      <xsd:group ref="myChoice" />
</xsd:complexType>
<xsd:complexType name="MyComplexType-3">
      <xsd:group ref="mySequence" maxOccurs="100"/>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа ACH.1:

```
MyComplexType-1 ::= [USE-ORDER] SEQUENCE {
      order          SEQUENCE OF ENUMERATED {a,b},
      a                 XSD.String,
      b BOOLEAN }
  (CONSTRAINED BY
      {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 35 */})
MyComplexType-2 ::= SEQUENCE {
      myChoice MyChoice }
MyComplexType-3 ::= SEQUENCE {
      mySequence-list SEQUENCE (SIZE(1..100)) OF MySequence }
```

D.3.8.3.2 В следующем примере **типом содержимого** является **модельная группа** с **наборщиком выбора.**

```
<xsd:complexType name="MyComplexType-4">
      <xsd:choice>
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
      </xsd:choice>
</xsd:complexType>
<xsd:complexType name="MyComplexType-5">
      <xsd:choice minOccurs="0">
```

```
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:boolean"/>
        </xsd:choice>
</xsd:complexType>
<xsd:complexType name="MyComplexType-6">
        <xsd:choice maxOccurs="5">
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:boolean"/>
        </xsd:choice>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
MyComplexType-4 ::= SEQUENCE {
        choice   [UNTAGGED] CHOICE {
           a XSD.String,
           b BOOLEAN } }
MyComplexType-5 ::= SEQUENCE {
        choice  [UNTAGGED] CHOICE {
           a XSD.String,
           b BOOLEAN } OPTIONAL }
MyComplexType-6 ::= SEQUENCE {
        choice-list  [UNTAGGED] SEQUENCE (SIZE(1..5)) OF [UNTAGGED]
CHOICE {
           a XSD.String,
           b BOOLEAN } }
```

D.3.8.3.3 В следующем примере **типом содержимого** является **модельная группа** с **наборщиком всего.**

```
<xsd:complexType name="MyComplexType-7">
<xsd:all>
<xsd:element name="a" type="xsd:string"/>
<xsd:element name="b" type="xsd:boolean"/>
</xsd:all>
</xsd:complexType>
<xsd:complexType name="MyComplexType-8">
<xsd:all minOccurs="0">
<xsd:element name="a" type="xsd:string"/>
<xsd:element name="b" type="xsd:boolean"/>
</xsd:all>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
MyComplexType-7 ::= [USE-ORDER] SEQUENCE {
        order          SEQUENCE OF ENUMERATED {a,b},
        a                XSD.String,
        b                BOOLEAN }
        (CONSTRAINED BY
             {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 35 */})
MyComplexType-8 ::= [USE-ORDER] SEQUENCE {
        order          SEQUENCE OF ENUMERATED {a,b},
        a                XSD.String OPTIONAL,
        b                BOOLEAN OPTIONAL }
        (CONSTRAINED BY
             {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4
 clause 35 */ })
```

D.3.8.3.4 В следующем примере **типом содержимого** является **модельная группа** с **наборщиком последовательности.**

```
<xsd:complexType name="MyComplexType-9">
        <xsd:sequence>
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:boolean"/>
        </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MyComplexType-10">
        <xsd:sequence minOccurs="0">
```

```
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MyComplexType-11">
        <xsd:sequence maxOccurs="5">
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:sequence>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
MyComplexType-9 ::= SEQUENCE {
        a  XSD.String,
        b  BOOLEAN }
MyComplexType-10 ::= SEQUENCE {
        sequence [UNTAGGED] SEQUENCE {
            a XSD.String,
            b BOOLEAN } OPTIONAL }
MyComplexType-11 ::= SEQUENCE {
        sequence-list [UNTAGGED] SEQUENCE (SIZE(1..5)) OF [UNTAGGED]
SEQUENCE {
            a XSD.String,
            b BOOLEAN } }
```

D.3.8.4 В следующем примере показаны **сложные определения типа, типом содержимого** которых является **смешанная** модель содержимого.

```
<xsd:complexType name="MyComplexType-12" mixed="true">
        <xsd:sequence>
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="MyComplexType-13" mixed="true">
        <xsd:all>
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:all>
</xsd:complexType>
<xsd:complexType name="MyComplexType-14" mixed="true">
        <xsd:choice>
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:choice>
</xsd:complexType>
<xsd:complexType name="MyComplexType-15" mixed="true">
        <xsd:all minOccurs="0">
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:all>
</xsd:complexType>
<xsd:complexType name="MyComplexType-16" mixed="true">
        <xsd:sequence maxOccurs="unbounded" minOccurs="0">
            <xsd:element name="a" type="xsd:string"/>
            <xsd:element name="b" type="xsd:boolean"/>
        </xsd:sequence>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
MyComplexType-12 ::= [EMBED-VALUES] SEQUENCE {
        embed-values        SEQUENCE OF XSD.String,
        a                               XSD.String,
        b                               BOOLEAN }
        (CONSTRAINED BY
```

```
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
MyComplexType-13 ::= [EMBED-VALUES] [USE-ORDER] SEQUENCE {
        embed-values      SEQUENCE OF XSD.String,
        order                     SEQUENCE OF ENUMERATED {a,b},
        a                          XSD.String,
        b                          BOOLEAN }
        (CONSTRAINED BY
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
             (CONSTRAINED BY
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 35 */})
MyComplexType-14 ::= [EMBED-VALUES] SEQUENCE {
        embed-values    SEQUENCE OF XSD.String,
        choice                  [UNTAGGED] CHOICE {
             a XSD.String,
             b BOOLEAN } }
        (CONSTRAINED BY
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
MyComplexType-15 ::= [EMBED-VALUES] [USE-ORDER] SEQUENCE {
        embed-values SEQUENCE OF XSD.String,
        order SEQUENCE OF ENUMERATED {a,b},
        a XSD.String OPTIONAL,
        b BOOLEAN OPTIONAL }
        (CONSTRAINED BY
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 35 */})
        (CONSTRAINED BY
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
MyComplexType-16 ::= [EMBED-VALUES] SEQUENCE {
        embed-values   SEQUENCE OF XSD.String,
        sequence-list  [UNTAGGED] SEQUENCE OF [UNTAGGED] SEQUENCE {
             a                          XSD.String,
             b                           BOOLEAN } }
     (CONSTRAINED BY
                {/* Shall conform to ITU-T Rec. X.693 | ISO/IEC 8825-4,
clause 25 */})
```

D.3.8.5 В следующем примере показаны **применения атрибута** для **сложного определения типа,** построенного с использованием **определения группы атрибутов.**

```
<xsd:attributeGroup name="AG1">
        <xsd:attribute name="a1" type="xs:string"/>
        <xsd:attribute name="a2" type="xs:string"/>
        <xsd:attribute name="a3" type="xs:decimal"/>
</xsd:attributeGroup>
<xsd:attributeGroup name="AG2">
        <xsd:attribute name="a1" use="prohibited"/>
        <xsd:attribute name="a3" type="xs:integer"/>
</xsd:attributeGroup>
<xsd:complexType name="MyComplexType-17">
        <xsd:attribute name="a4" type="xs:boolean"/>
        <xsd:attribute name="a5" type="xs:boolean"/>
        <xsd:attributeGroup ref="AG1"/>
</xsd:complexType>
<xsd:complexType name="MyComplexType-18">
        <xsd:complexContent>
                <xsd:restriction base="MyComplexType-17">
                        <xsd:attributeGroup ref="AG2"/>
                        <xsd:attribute name="a4" use="prohibited"/>
                </xsd:restriction>
```

```
        </xsd:complexContent>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
MyComplexType-17 ::= SEQUENCE {
        a1 [ATTRIBUTE] XSD.String OPTIONAL,
        a2 [ATTRIBUTE] XSD.String OPTIONAL,
        a3 [ATTRIBUTE] XSD.Decimal OPTIONAL,
        a4 [ATTRIBUTE] BOOLEAN OPTIONAL,
        a5 [ATTRIBUTE] BOOLEAN OPTIONAL }
MyComplexType-18 ::= SEQUENCE {
        a2 [ATTRIBUTE] XSD.String OPTIONAL,
        a3 [ATTRIBUTE] INTEGER OPTIONAL,
        a5 [ATTRIBUTE] BOOLEAN OPTIONAL }
```

D.3.8.6 Выведение **сложных определений типа.**

```
<xsd:complexType name="MyComplexType-19">
        <xsd:sequence minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="a" type="xsd:string"/>
                <xsd:element name="b" type="xsd:boolean"/>
                <xsd:element name="c" type="xsd:boolean" minOccurs="0"/>
        </xsd:sequence>
        <xsd:attribute name="attr1" type="xsd:short" use="required"/>
        <xsd:attribute name="attr2" type="xsd:short"/>
</xsd:complexType>
<xsd:complexType name="MyComplexType-20">
        <xsd:complexContent>
                <xsd:restriction base="MyComplexType-19">
                        <xsd:sequence>
                                <xsd:element name="a" type="xsd:token"/>
                                <xsd:element name="b" type="xsd:boolean"/>
                        </xsd:sequence>
                        <xsd:attribute name="attr2" type="xsd:short" use="prohibited"/>
                </xsd:restriction>
        </xsd:complexContent>
</xsd:complexType>
<xsd:complexType name="MyComplexType-21">
        <xsd:complexContent>
                <xsd:extension base="MyComplexType-20">
                        <xsd:sequence>
                                <xsd:element name="d" type="xsd:string"/>
                        </xsd:sequence>
                        <xsd:attribute name="attr3" type="xsd:boolean"/>
                </xsd:extension>
        </xsd:complexContent>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
MyComplexType-19 ::= SEQUENCE {
        attr1                    [ATTRIBUTE] XSD.Short,
        attr2                   [ATTRIBUTE] XSD.Short OPTIONAL,
        sequence-list      [UNTAGGED] SEQUENCE OF [UNTAGGED] SEQUENCE
{
        a                                   XSD.String,
        b                                   BOOLEAN,
        c                                   BOOLEAN OPTIONAL } }
MyComplexType-20 ::= SEQUENCE {
        attr1                [ATTRIBUTE] XSD.Short,
        a                                   XSD.Token,
        b                                   BOOLEAN }
MyComplexType-21 ::= SEQUENCE {
        attr1                [ATTRIBUTE] XSD.Short,
        attr3                [ATTRIBUTE] BOOLEAN OPTIONAL,
        a                      XSD.String,
        b                      BOOLEAN,
        d                      XSD.String }
```

```
MyComplexType-20-derivations ::= [USE-TYPE] CHOICE {
        myComplexType-20 [NAME AS CAPITALIZED] MyComplexType-20,
        myComplexType-21 [NAME AS CAPITALIZED] MyComplexType-21 }
MyComplexType-19-derivations ::= [USE-TYPE] CHOICE {
        myComplexType-19 [NAME AS CAPITALIZED] MyComplexType-19,
        myComplexType-20 [NAME AS CAPITALIZED] MyComplexType-20,
        myComplexType-21 [NAME AS CAPITALIZED] MyComplexType-21 }
```

если и только если:

а) **простое определение типа** «**MyComplexType-19**» происходит как **определение типа**, по крайней мере, одного **объявления элемента** (не показано в примере), что отображается в АСН.1;

б) **простое определение типа** «**MyComplexType-20**» происходит как **определение типа**, по крайней мере, одного **объявления элемента** (не показано в примере), что отображается в АСН.1; и

в) нет никаких других компонентов схемы, отображаемых в АСН.1, которые создают имена ссылок типов АСН.1 `MyComplexType-19`, `MyComplexType-20`, `MyComplexType-21`, `MyComplexType-19-derivations` и `MyComplexType-20-derivations`.

D.3.9 Отображение групповых символов

Для приводимых далее примеров **целевое пространство имен** считается следующим URI: http://
gost.e-ecolog.ru
dcard.

D.3.9.1 Атрибут **группового символа**

```
<xsd:complexType name="AnyAttribute-1">
        <xsd:anyAttribute namespace="##any"/>
</xsd:complexType>
<xsd:complexType name="AnyAttribute-2">
        <xsd:anyAttribute namespace="##other"/>
</xsd:complexType>
<xsd:complexType name="AnyAttribute-3">
        <xsd:anyAttribute  namespace="##targetNamespace"/>
</xsd:complexType>
<xsd:complexType name="AnyAttribute-4">
        <xsd:anyAttribute namespace="##local
                                        gost.e-ecolog.ru                    >
</xsd:complexType>
<xsd:complexType name="AnyAttribute-5">
        <xsd:complexContent>
                <xsd:extension base="AnyAttribute-4">
                        <xsd:anyAttribute namespace="##targetNamespace"/>
                </xsd:extension>
        </xsd:complexContent>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа АСН.1:

```
AnyAttribute-1 ::= SEQUENCE {
     attr [ANY-ATTRIBUTES] SEQUENCE (CONSTRAINED BY {
       /* Each item shall conform to the "AnyAttributeFormat"
specified in
             ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */})
        OF XSD.String }
AnyAttribute-2 ::= SEQUENCE {
     attr [ANY-ATTRIBUTES EXCEPT ABSENT
                                    "gost.e-ecolog.ru
rd"]
        SEQUENCE (CONSTRAINED BY {
        /* Each item shall conform to the "AnyAttributeFormat"
specified in
             ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */})
        OF XSD.String }
AnyAttribute-3 ::= SEQUENCE {
        attr [ANY-ATTRIBUTES FROM
"gost.e-ecolog.ru                    "]
        SEQUENCE (CONSTRAINED BY {
        /* Each item shall conform to the "AnyAttributeFormat"
specified in
             ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */})
        OF XSD.String }
```

```
AnyAttribute-4 ::= SEQUENCE {
         attr [ANY-ATTRIBUTES FROM ABSENT
                                        "gost.e-ecolog.ru
bute"]
        SEQUENCE (CONSTRAINED BY {
        /* Each item shall conform to the "AnyAttributeFormat"
specified in
              ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */})
        OF XSD.String }
AnyAttribute-5 ::= SEQUENCE {
        attr [ANY-ATTRIBUTES FROM ABSENT
                                        "gost.e-ecolog.ru
bute"
                                        "gost.e-ecolog.ru
ard"]
        SEQUENCE (CONSTRAINED BY {
        /* Each item shall conform to the "AnyAttributeFormat"
specified in
              ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 18 */})
        OF XSD.String }
```

D.3.9.2 Далее приведен пример модели содержимого **группового символа**.

```
<xsd:complexType name="Any-1">
        <xsd:sequence>
              <xsd:any namespace="##any"/>
        </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Any-2">
        <xsd:sequence>
              <xsd:any minOccurs="0" namespace="##other"/>
        </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="Any-3">
        <xsd:sequence>
              <xsd:any minOccurs="0" maxOccurs="unbounded" namespace="##local"/>
        </xsd:sequence>
</xsd:complexType>
```

Эти **сложные определения типа** отображаются в следующие присвоения типа ACH.1:

```
Any-1 ::= SEQUENCE {
     elem [ANY-ELEMENT] XSD.String (CONSTRAINED BY {
              /* Shall conform to the "AnyElementFormat"
specified in
                 ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19
*/}) }
Any-2 ::= SEQUENCE {
     elem [ANY-ELEMENT EXCEPT ABSENT
                                        "gost.e-ecolog.ru
ard"]
          XSD.String (CONSTRAINED BY {
              /* Shall conform to the "AnyElementFormat"
specified in
                 ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19
*/})
                 OPTIONAL }
Any-3 ::= SEQUENCE {
     elem-list [UNTAGGED] SEQUENCE OF elem
          [ANY-ELEMENT FROM ABSENT] XSD.String (CONSTRAINED BY {
              /* Shall conform to the "AnyElementFormat"
specified in
                 ITU-T Rec. X.693 | ISO/IEC 8825-4, clause 19
*/}) }
```

Примечание — Дополнительные примеры вычисления «NamespaceRestriction» можно найти в примерах атрибутов групповых символов в D.3.9.1.

Приложение Е
(справочное)

Применение отображения для обеспечения двоичных кодировок для W3C XML-схемы

В этом приложении описывается использование отображения, указанного в настоящем стандарте в сочетании со стандартизованными правилами кодирования АСН.1 для предоставления канонических и компактных двоичных кодировок данных, определенных XSD-схемой.

Е.1 Кодирование схем XSD

Е.1.1 Схемы XSD могут быть отображены в определения типа АСН.1, как указано в основном тексте настоящего стандарта, и высокоуровневый тип может быть затем закодирован с использованием любого правила кодирования АСН.1, указанного в Рекомендациях МСЭ-Т Х.690 (2008) (ИСО/МЭК 8825-1:2008), МСЭ-Т Х.691 (2002) (ИСО/МЭК 8825-2:2002) и МСЭ-Т Х.693 (2008) (ИСО/МЭК 8825-4:2008).

Е.1.2 Каждая из этих кодировок имеет соответствующий идентификатор объекта и OID-значение идентификатора интернационализированного ресурса, которое может быть использовано для определения кодировки в передаче. То, каким образом такая идентификация передается в декодер, выходит за рамки настоящего стандарта. Связанное значение дескриптора объекта может быть также использовано для прочтения человеком, но необязательно однозначно.

Е.1.3 Когда XSD-схема не передается приемнику методом, описанному в Е.3, то, каким образом получатель получает схему, выходит за рамки настоящего стандарта.

Е.2 Передача без применения XSD-схемы для схем

Е.2.1 Этот метод исходит из предположения, что получатель знает XSD-схему, используемую отправителем.

Е.2.2 Рисунок Е.1 показывает, как использовать отображение, определенное в настоящем стандарте для кодирования XML-документов, с помощью правил кодирования АСН.1.

Е.2.3 Отправитель и получатель используют ту же самую (фиксированную) схему XSD для формирования АСН.1-схемы, которая, в свою очередь, передается компилятору АСН.1 для формирования BER, DER, PER или XER кодировочной таблицы для XML-документов, соответствующих этой схеме XSD.

Е.3 Передача с применением XSD-схемы для схем

Е.3.1 С доступностью уникальной XSD-схемы для схем это возможно пройти в два шага (см. рисунок Е.2).

Е.3.2 Отправитель и получатель создают модуль АСН.1 и кодер/декодер из XSD-схемы для схем.

Е.3.3 На первом шаге отправитель кодирует в BER, DER или PER схему XSD для документов и отправляет закодированную схему приемнику. Приемник декодирует эту схему и, используя отображение из XSD-схемы в АСН.1 и АСН.1-компилятор, формирует АСН.1-модуль и кодер/декодер для документов XML, соответствующих этой схеме.

Е.3.4 На втором шаге отправитель кодирует в BER, DER, PER, или XER документ XML и посылает закодированный документ приемнику.

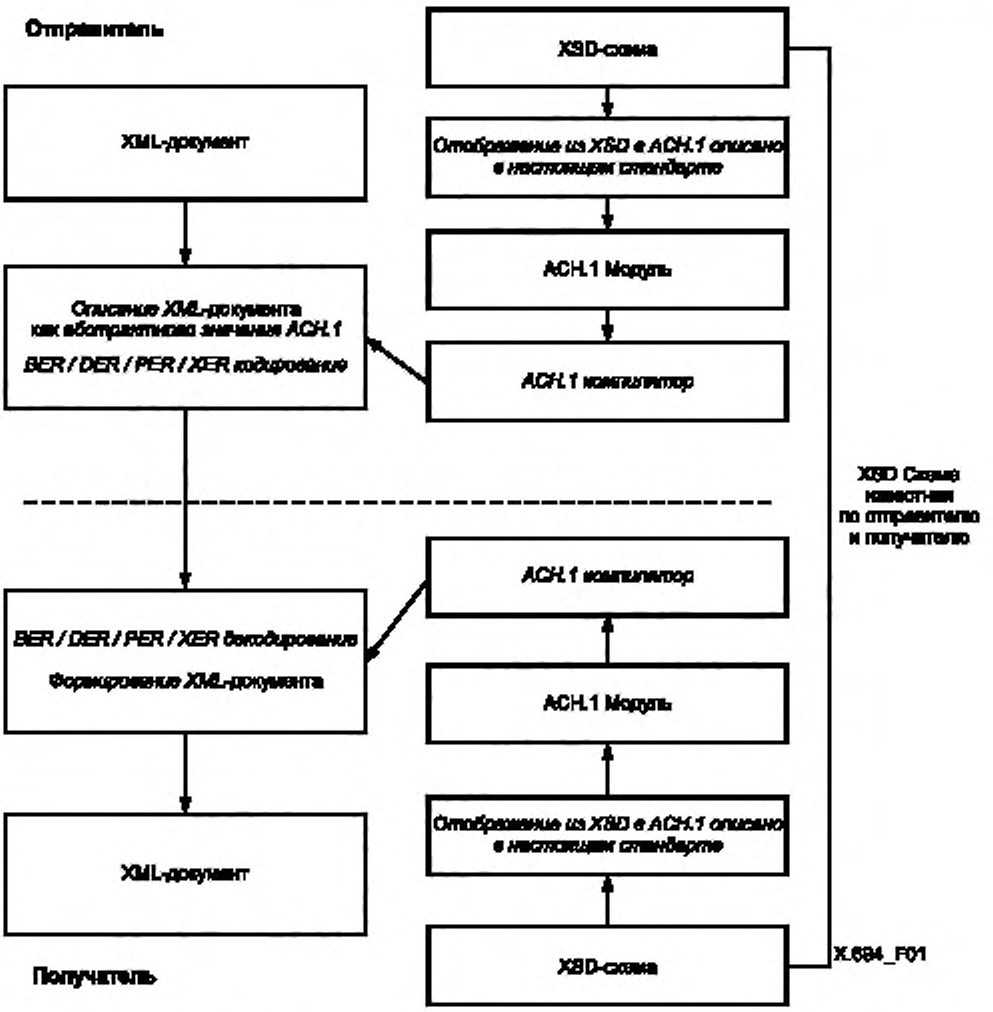

Рисунок D.1 — Передача документа XML с применением отображения XSD в ACH.1

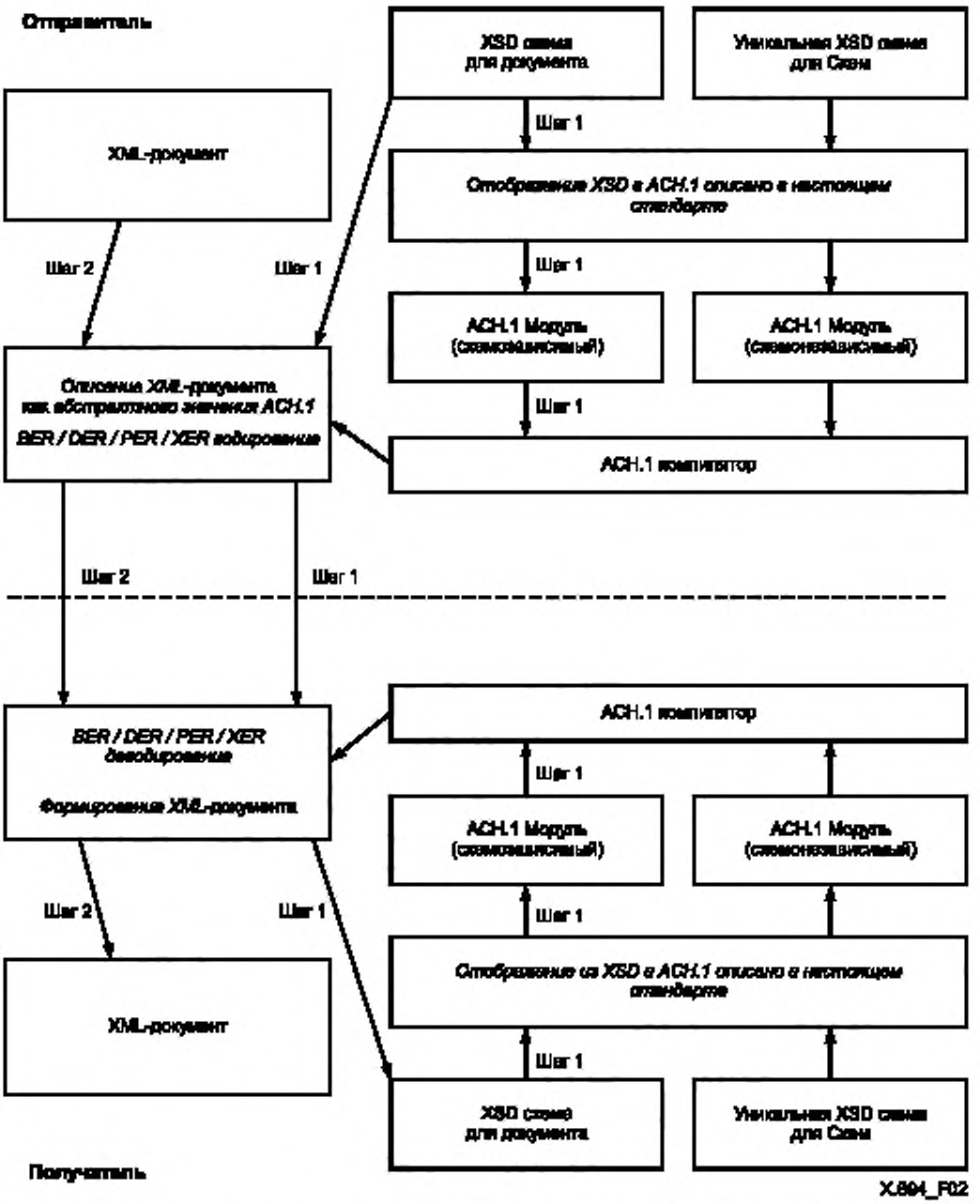

Рисунок Е.2 — Передача XSD Схемы и XML-документа с применением отображения XSD в ACH.1

Приложение ДА
(справочное)

Сведения о соответствии ссылочных международных стандартов национальным стандартам Российской Федерации

Таблица ДА

Обозначение ссылочного международного стандарта	Степень соответствия	Обозначение и наименование соответствующего национального стандарта
ИСО 8601:2004	IDT	ГОСТ Р ИСО 8601—2001 «Система стандартов по информации, библиотечному и издательскому делу. Представление дат и времени. Общие требования»
ИСО/МЭК 8824-1:2008	IDT	ГОСТ Р ИСО/МЭК 8824-1—2001 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 1. Спецификация основной нотации»
ИСО/МЭК 8824-2:2008	IDT	ГОСТ Р ИСО/МЭК 8824-2—2001 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 2. Спецификация информационного объекта»
ИСО/МЭК 8824-3:2008	IDT	ГОСТ Р ИСО/МЭК 8824-3—2001 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 3. Спецификация ограничений»
ИСО/МЭК 8824-4:2008	IDT	ГОСТ Р ИСО/МЭК 8824-4—2001 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 4. Параметризация спецификаций АСН.1»
ИСО/МЭК 8825-1:2008	IDT	ГОСТ Р ИСО/МЭК 8825-1—2003 «Информационная технология. Правила кодирования АСН.1. Часть 1. Спецификация базовых (BER), канонических (CER) и отличительных (DER) правил кодирования»
ИСО/МЭК 8825-2:2008	IDT	ГОСТ Р ИСО/МЭК 8825-2—2003 «Информационная технология. Правила кодирования АСН.1. Часть 2. Спецификация правил уплотненного кодирования (PER)»
ИСО/МЭК 8825-3:2008	IDT	ГОСТ Р ИСО/МЭК 8825-3—2003 «Информационная технология. Абстрактная синтаксическая нотация версии один (АСН.1). Часть 3. Спецификация нотации контроля кодирования (ECN)»
ИСО/МЭК 8825-4:2008	IDT	ГОСТ Р ИСО/МЭК 8825-4—2006 «Информационная технология. Правила кодирования АСН.1. Часть 4. Правила XML кодирования (XER)»
ИСО/МЭК 24824-1:2007	—	*
ИСО/МЭК 10646:2003	—	*

* Соответствующий национальный стандарт отсутствует. До его утверждения рекомендуется использовать перевод на русский язык данного международного стандарта. Перевод данного международного стандарта находится в Федеральном информационном фонде технических регламентов и стандартов.

Примечание — В настоящей таблице использовано следующее условное обозначение степени соответствия стандартов:

- IDT — идентичные стандарты.

УДК 681.3:691.39:006.354 ОКС 35.100.60 П85

Ключевые слова: обработка данных, информационный обмен, сетевое взаимодействие, взаимосвязь открытых систем, АСН.1, кодирование, быстрое информационное множество

Редактор *В. В. Космин*
Технический редактор *В. Н. Прусакова*
Корректор *С. И. Фирсова*
Компьютерная верстка *З. И. Мартыновой*

Сдано в набор 18.12.2014. Подписано в печать 23.01.2015. Формат 60×84 1/8. Бумага офсетная. Гарнитура Ариал. Печать офсетная. Усл. печ. л. 8,84 Уч.-изд. л. 7,07. Тираж 34 экз. Зак. 1884.

ФГУП «СТАНДАРТИНФОРМ», 123995 Москва, Гранатный пер., 4.
www.gostinfo.ru info@gostinfo.ru
Набрано и отпечатано в Калужской типографии стандартов, 248021 Калуга, ул. Московская, 256.

gost.e-ecolog.ru