НЕКОММЕРЧЕСКОЕ ПАРТНЕРСТВО
«СТАНДАРТЫ ЭЛЕКТРОННОГО ОБМЕНА ИНФОРМАЦИЕЙ»

СТАНДАРТ
ОБЩЕСТВЕННОГО
ОБЪЕДИНЕНИЯ

СТО СЭОИ 1.3.2–2003

Система нормативных документов по разработке стандартов ЭОИ

Стандарты ЭОИ на языке XML

ПРЕДСТАВЛЕНИЕ ЭЛЕКТРОННЫХ ДОКУМЕНТОВ НА ЯЗЫКЕ XML

Правила формирования

МОСКВА
2003

Предисловие

Сведения о стандарте

1 РАЗРАБОТАН ЗАО «КОРУС АКС», г. Екатеринбург

2 ВНЕСЕН Технической рабочей группой Некоммерческого партнерства «Стандарты ЭОИ»

3 ПРИНЯТ И ВВЕДЕН В ДЕЙСТВИЕ Решением совета директоров Некоммерческого партнерства «Стандарты ЭОИ»

4 ВВЕДЕН ВПЕРВЫЕ

Содержание

1. ОБЛАСТЬ ПРИМЕНЕНИЯ 1

2. НОРМАТИВНЫЕ ССЫЛКИ 1

3. ТЕРМИНЫ, ОПРЕДЕЛЕНИЯ И СОКРАЩЕНИЯ 1

4. ПРАВИЛА ФОРМИРОВАНИЯ XML-СХЕМ ДЛЯ ЭЛЕКТРОННЫХ ДОКУМЕНТОВ 2

4.1. Общие сведения 2
4.2. Используемые стандарты XML 2
4.3. Общие правила оформления XML-документов 2
4.3.1. Способ разметки XML-документов 2
4.3.2. Корректные XML-документы 3
4.3.3. Состоятельные XML-документы 3
4.3.4. Пролог XML-документа 3
4.4. Правила преобразования логической модели в XML-схему и XML-документы 4
4.4.1. Общие правила 4
4.4.2. Описание базовых встроенных типов XML-схем 4
4.4.3. Правило элементов 5
4.4.4. Правило атрибутов 9
4.5. Возможности XML-схем, используемые в стандартах ЭОИ 12
4.5.1. Пространство имен 12
4.5.2. Пространства имен в XML-схеме 13
4.5.3. Пространства имен в XML-документах 14
4.5.4. Кодировка 14
4.5.5. Обозначение версий XML-схем 14
4.5.6. Комментарии XML-схем 15
4.5.7. Ограничения для простых типов посредством фасетов XML (facets) 15
4.5.8. Регулярные выражения 16

П А Р Т Н Е Р С Т В О « С Т А Н Д А Р Т Ы Э О И »

**Система нормативных документов
по разработке стандартов ЭОИ**

Стандарты ЭОИ на языке XML

ПРЕДСТАВЛЕНИЕ ЭЛЕКТРОННЫХ ДОКУМЕНТОВ НА ЯЗЫКЕ XML

Правила формирования

Дата введения – 2003–ММ–ДД

1. Область применения

Настоящий нормативный документ определяет правила, по которым логическая модель электронных документов преобразуется в XML-схемы.

Настоящий документ является нормативным для разработчиков стандартов ЭОИ для Партнерства.

2. Нормативные ссылки

В настоящем стандарте использованы ссылки на следующие стандарты:

ГОСТ Р 1.5-2002 Государственная система стандартизации РФ. Стандарты. Общие требования к построению, изложению, оформлению, содержанию и обозначению.

ГОСТ 2.105-95 Единая система конструкторской документации. Общие требования к текстовым документам.

СТО СЭОИ 1.2–2003. Система нормативных документов по разработке стандартов ЭОИ. Логическое описание ЭОИ. Требования к содержанию.

3. Термины, определения и сокращения

Применительно к настоящему стандарту используются следующие термины и определения:

3.1 **схема** (Schema): Логическое и физическое определение элементов данных, физических характеристик и внутренних отношений.

3.2 **extensible markup language** (**XML**): Расширяемый (открытый) язык разметки.

3.3 **XML-схема** (XML Schema): Язык описания структуры документа. Предусматривает описание допустимой структуры документа и, возможно, типов данных в значениях атрибутов и содержимом элементов. Сейчас существует несколько таких языков.

3.4 **пространства имен XML** (XML Namespaces): Способ уточнения (квалификации) имен элементов и атрибутов путем сопоставления набора имен некоторой области применения, технически выраженной в виде идентификатора ресурса (URI).

3.5 **тег** (tag): Синтаксически выделенная часть разметки, отмечающая начало и конец элемента в документе.

3.6 **подстановка** (entity): Синтаксически выделенная часть разметки, на место которой при обработке документа подставляется определенный символ или фрагмент текста или содержимое внешнего файла. Содержимое подстановки в большинстве случаев тоже подлежит обработке.

3.7 **элемент** (element): Структурная единица языка XML, предназначенная для хранения прикладных или служебных данных. Элемент состоит из разметки и текста, заключенного между тегами

разметки. Элемент может содержать вложенные элементы, и сам быть дочерним элементом. В каждом XML-документе должен быть хотя бы один элемент: корневой. Элемент может иметь атрибуты. Элемент может быть обязательным или нет.

3.8 **атрибут** (attribute): Структурная единица языка XML уточняющая свойства элемента несущая дополнительную информацию об элементе.

3.9 **группа атрибутов** (attribute group): объединение атрибутов в одну группу с общим именем.

3.10 **ЭЦП** – Электронная цифровая подпись.

3.11 **W3C** – World Wide Web Consortium, Консорциум всемирной сети.

4. Правила формирования XML-схем для электронных документов

4.1. Общие сведения

Обмен данными в ЭОИ осуществляется с использованием электронных сообщений. Для ЭОИ на языке XML – это XML-сообщения.

XML-сообщение содержит электронный документ, определенный в документе «Стандарт ЭОИ. Логическое модель ЭОИ» и состоит из следующих частей:

- конверт сообщения – набор элементов и атрибутов, предназначенный для передачи служебной (транспортной) информации;
- ЭЦП – набор элементов и атрибутов, защищающих ЭД от подделки и для установления авторства электронного документа;
- электронный документ – прикладные данные, структура которых описывается в документе «Стандарт ЭОИ. Логическая модель ЭОИ».

Настоящий нормативный документ содержит правила, по которым логическая модель ЭД однозначно преобразуется в XML-схему.

4.2. Используемые стандарты XML

Настоящая система нормативных документов устанавливает, что XML-форматы электронных документов должны разрабатываться в соответствии со следующими стандартами:

- "Extensible Markup Language (XML) 1.0 (Second Edition)", опубликованному в Интернет по адресу: http://www.w3.org/TR/REC-xml
- "Namespaces in XML", опубликованному в Интернет по адресу: http://www.w3.org/TR/REC-xml-names
- "XML Schema Part 1: Structures" и "XML Schema Part 2: Datatypes", опубликованным в Интернет по адресам http://www.w3.org/TR/xmlschema-1/ и http://www.w3.org/TR/xmlschema-2/.

Для описания структуры ЭД на языке XML должна применяться XML-схема.

4.3. Общие правила оформления XML-документов

4.3.1. Способ разметки XML-документов

Символы, составляющие XML-документ, подразделяются на символьные данные и разметку. Разметка документа несет в себе информацию о логической структуре, а также о физической разбивке данных при их хранении на внешних запоминающих.

Помимо содержательной информации, XML-документы содержат дополнительную информацию, которая включает: комментарии, инструкции обработки, определение типа документа и собственно элементы разметки, структурирующие содержательную информацию.

Комментарий начинается последовательностью символов '<!--', заканчивается последовательностью символов '-->'. Строка комментария не может заканчиваться символом '-', а также не может содержать в себе последовательность символов '--'.

Элементами разметки, структурирующими содержательную информацию, являются теги и атрибуты тегов. Атрибут записывается в виде: '`имя_атрибута="значение_атрибута"`' и служит для описания атомарных единиц содержательной информации. Тег служит для выделения некой структурной единицы содержательной информации, которую называют элементом. Данная единица может быть заключена между открывающим и закрывающим тегами, в этом случае синтаксис тегов будет следующим: открывающий тег начинается с символа '<' со следующим непосредственно за ним именем тега, далее может идти нуль или более атрибутов, а затем – закрывающий символ '>'; закрывающий тег начинается с последовательности символов '</' со следующим непосредственно за ними именем тега и символом '>'. В одном теге не допускается атрибутов с одинаковыми именами. Не допускается атрибутов в закрывающем теге. В случае, когда тег не обрамляет никаких данных (такой элемент называется пустым), он может быть записан в виде: открывающий символ '<', имя тега, нуль или более атрибутов и закрывающая последовательность символов '/>'. Имя тега является регистрозависимым.

4.3.2. Корректные XML-документы

Понятие корректности (правильности, well-formed) является ключевым в XML, т.к. объект данных является XML-документом, только если он является корректным.

Корректный документ состоит из необязательного пролога, обязательного элемента, который называется корневым и последующими опциональными инструкциями обработки, комментариями или пробельными символами.

Основные требования к корректности документов:

- у каждого элемента должен быть ровно по одному открывающему и закрывающему тегу;
- в документе должен быть ровно один корневой элемент;
- элементы могут быть вложены друг в друга, но не могут пересекаться;
- специальными последовательностями символов (например вместо символа меньше «<» документ должен содержать подстановку вида «<»);
- значения атрибутов должны быть взяты в кавычки;
- пустые элементы (состоящие только из тегов) соответствуют особому формату (такой элемент записывается в виде одного тега, по виду аналогичному открывающему, но содержащему символ / перед закрывающей скобкой).

4.3.3. Состоятельные XML-документы

XML-документ является состоятельным (действительным, valid), если с ним ассоциировано определение типа документа и если документ соответствует ограничениям, описанным в этом определении.

В соответствии со спецификацией XML версии 1.0, определение типа документа описывает структуру XML-документов с использованием специального синтаксиса, изначально определенного в SGML.

Для описания ограничений на структуру XML-документов в данном стандарте использованы XML-схемы, утвержденные в спецификациях "XML Schema Part 1: Structures" и "XML Schema Part 2: Datatypes".

С помощью XML-схем можно описывать структурные ограничения и ограничения на форму представления текстовых значений атрибутов и элементов (при помощи описания типов данных).

4.3.4. Пролог XML-документа

XML-документ может начинаться с так называемого пролога – опциональных строк, включающих декларацию XML и определение типа документа. При использовании XML-схем пролог состоит только из одной строки – декларации XML.

Декларация XML – это строка вида:

```
<?xml version="1.0" encoding="WINDOWS-1251"?>
```

которая содержит информацию о том, что данный документ является XML-документом, указывает версию спецификации XML, которой данный XML-документ соответствует, а также может указывать то, в какой кодировке представлен текст XML-документа.

В декларации определяются следующие атрибуты:

- `version` – обязательный атрибут; его значение должно быть равно "1.0". Этот атрибут предусмотрен для поддержки будущих версий языка XML.
- `encoding` – необязательный атрибут; в качестве значения надо указывать допустимую кодировку символов. Если атрибут не указан, предполагается кодировка UTF-8 или UTF-16, в зависимости от формата начальной строки “<?xml”.

Атрибуты декларации следует указывать только ***в приведенном выше порядке***. Значение атрибута `encoding` ***нечувствительно к регистру*** символов.

Если декларация XML включена в документ, она должна быть расположена в самом начале – перед ней не разрешается вводить символы пустых пространств (табуляция <HT>, перевод строки <LF>, возврат каретки<CR>, пробел) и комментарии.

Декларация XML абсолютно необходима для того, чтобы программа разбора (XML парсер) смогла автоматически определить кодировку, правильно интерпретировать XML-документ и проверить его корректность и, при необходимости, состоятельность.

4.4. Правила преобразования логической модели в XML-схему и XML-документы

4.4.1. Общие правила

Настоящий раздел описывает рекомендуемые правила преобразования логической модели электронных документов в физическую реализацию на языке XML (создание XML-схем W3C, описывающих экземпляры XML-документов). Настоящий стандарт определяет следующие правила преобразования компонентов логической модели в конструкции языка XML:

- "правило атрибутов" – простые свойства компонентов преобразуются в атрибуты элементов;
- "правило элементов" – простые свойства компонентов преобразуются в элементы.

Также настоящий стандарт определяет следующий метод реализации компонентов логической модели в XML-схеме:

«метод глобальные типы/локальные элементы» – все прикладные типы и типы компонентов определяются глобально; если тип содержит элементы, то элементы в составе типа определяются внутри данного типа (локально).

Сочетание вышеуказанных правил и методики однозначно определяет соответствие логической модели ее физической реализации на языке XML.

Если при разработке стандарта использовался какой-либо начальный стандарт, то физическая реализация (XML-схема) логической модели может быть подвергнута трансформации с целью получения XML-схемы, отвечающей требованиям выбранного начального стандарта.

4.4.2. Описание базовых встроенных типов XML-схем

Табл. 1. Описание базовых встроенных типов

Имя	Описание
xsd:string	Символьная строка конечной длины. Допустимыми являются символы с кодами в кодировке Unicode: #x9 (табуляция), #xA (возврат каретки), #xD (перенос строки), #x20-#xD7FF, #xE000-#xFFFD, #x10000-#x10FFFF (алфавитно-цифровые символы).
xsd:token	Символьная строка конечной длины. Допустимыми являются символы с кодами в кодировке Unicode: #x20-#xD7FF, #xE000-#xFFFD, #x10000-#x10FFFF (алфавитно-цифровые символы).
xsd:decimal	Десятичные числа произвольной точности. Область значения типа – множество значений $i * 10^{-n}$, где i, n – целые числа, n>=0. Представление числа состоит из конечной последовательности десятичных цифр,

	разделенных десятичным разделителем дробной части (точкой). Представление положительного числа может начинаться со знака "+".
xsd:integer	Целое десятичное число. Реализация программы разбора XML-документов (XML-парсер), в соответствии со стандартом, должна поддерживать как минимум 18-разрядные целые числа. Представление положительного целого числа может начинаться со знака "+" с последующей последовательностью десятичных цифр.
xsd:boolean	Логическая величина, принимающая одно из двух значений: "true" (истина) или "false" (ложь).
xsd:date	Дата Григорианского календаря в формате, соответствующем международному стандарту ISO 8601 "Data elements and interchange formats - Information interchange - Representation of dates and types". Дата записывается в виде "YYYY-MM-DD". Например строка "2003-07-17" соответствует 17 июля 2003 года.
xsd:time	Время в формате, соответствующем международному стандарту ISO 8601 "Data elements and interchange formats - Information interchange - Representation of dates and types". Время записывается в виде "hh:mm:ss.sss" с последующим опциональным индикатором временной зоны (+-hh:mm). Например, строка "14:27:15+05:00" соответствует 14 часам 27 минутам 15 секундам Екатеринбургского времени (так как Екатеринбург находится во временной зоне на 5 часов опережающей время по Гринвичу).
xsd:dateTime	Дата Григорианского календаря и время в формате, соответствующем международному стандарту ISO 8601 "Data elements and interchange formats - Information interchange - Representation of dates and types". Дата записывается в виде "YYYY-MM-DDThh:mm:ss.sss" с последующим опциональным индикатором временной зоны (+-hh:mm). Например строка "2003-07-17T14:27:15" соответствует 17 июля 2003 года.
xsd:duration	Продолжительность времени в формате, соответствующем международному стандарту ISO 8601 "Data elements and interchange formats - Information interchange - Representation of dates and types". Продолжительность времени записывается в формате "PnYnMnDTnHnMnS", где P указывает на то, что это период времени, nY - представляет количество лет, nM - количество месяцев, nD - количество дней, T - сепаратор, разделяющий дату от времени, nH - количество часов, nM - количество минут, nS - количество секунд, которое может содержать десятичную точку, отделяющую целые секунды от долей секунды. Например, строка "PT8H" соответствует периоду времени длительностью 8 часов.

4.4.3. Правило элементов

Основное правило преобразования – все XML элементы в XML-схеме, описывающей логическую модель:

- должны иметь имя;
- должны быть определены одним из именованных типов (простой тип – simpleType или сложный тип – complexType);
- могут содержать атрибуты.

Имя XML элемента формируется по правилам:

- Имя XML элемента, описывающее тело сообщения, соответствует синониму имени электронного документа в логической модели;
- Имя XML элемента, описывающее компонент, соответствует синониму имени описываемого компонента в логической модели;
- Имена XML элементов, описывающих элементы компонента, соответствует синонимам имен описываемых элементов компонента в логической модели;

Простой тип XML – simpleType. Простой тип XML применяется для преобразования типов

- Identifier;

- Code;
- Indicator;
- Text;
- DateTime;
- Numeric;
- BinaryObject.

Возможно также в случае необходимости применение простого типа XML для преобразования типов, основанных на следующих категориях типов:

- Quantity;
- Amount.

Сложный тип – complexType. Сложный тип применяется для описания компонентов.

Сложный тип с простым содержанием – complexType (simpleContent). Сложный тип с простым содержанием применяется для описания категорий типов с применением дополнительных атрибутов:

- Quantity;
- Amount.

Соответствие структурных элементов логической модели конструкциям XSD-схемы согласно правилу элементов приведены в таблице.

Табл. 2 Соответствие базовых типов логической модели типам в XML-схеме по правилу элементов

Наименование базового типа	Простой или сложный тип XML	Базовый тип XML	Возможные атрибуты XML	Применяемые ограничения на основе фасетов XML
Amount (Сумма)	complexType с простым содержанием (simpleContent); или simpleType	decimal	Currency (или нет)	minInclusive, minExclusive, maxInclusive, maxExclusive, totalDigits, fractionDigits
BinaryObject (БинарныйОбъект)	simpleType	base64	нет	length, minLength, maxLength (указываются в байтах)
Code (Код)	simpleType	string token	нет	enumeration, pattern (при использовании внешних справочников)
DateTime (ДатаВремя)	simpleType	date time dateTime duration gDay gMonth gYear gMonthDay gYearMonth	нет	minInclusive, minExclusive, maxInclusive, maxExclusive
Identifier (Идентификатор)	simpleType	string token	нет	pattern, length, minLength, maxLength
Indicator (Индикатор)	simpleType	boolean	нет	нет
Numeric	simpleType	decimal	нет	minInclusive, minExclusive,

Наименование базового типа	Простой или сложный тип XML	Базовый тип XML	Возможные атрибуты XML	Применяемые ограничения на основе фасетов XML
(Число)				maxInclusive, maxExclusive, totalDigits, fractionDigits
Quantity (Количество)	complexType с простым содержанием (simpleContent); или simpleType	decimal	Unit (или нет)	minInclusive, minExclusive, maxInclusive, maxExclusive, totalDigits, fractionDigits
Text (Текст)	simpleType	string	нет	pattern, length, minLength, maxLength

Табл. 3. Соответствие ограничений прикладных типов логической модели типам ограничениям на основе фасетов в XML-схеме

Ограничения прикладных типов	Ограничения прикладных типов	Область значений	Примечание
мин. значение (включая)	minInclusive	Значение в соответствии с базовым типом, к которому применяется ограничение	Указывается включающая нижняя граница из пространства значений (минимальное значение) для базового типа
макс. значение (включая)	maxInclusive		Указывается включающая верхняя граница из пространства значений (максимальное значение) для базового типа
мин. значение (исключая)	minExclusive		Указывается исключающая нижняя граница из пространства значений (минимальное значение) для базового типа
макс. значение (исключая)	maxExclusive		Указывается исключающая верхняя граница из пространства значений (максимальное значение) для базового типа
общее количество цифр	totalDigits	Положительное целое значение	Указывается максимальное общее количество цифр, в т.ч. и в дробной части значения
количество дробных цифр	fractionDigits	Неотрицательное целое значение	Указывается максимальное количество цифр в дробной части значения
точная длина	length	Неотрицательное целое значение	Указывается точная длина значения
макс. длина	maxLength		Указывается максимально допустимая длина значения
мин. длина	minLength		Указывается минимально допустимая длина значения
шаблон значений	pattern	Регулярное выражение	Указывается шаблон (с использованием регулярных выражений)
список значений	enumeration	Список символьных строк	Указывается полный список допустимых значений

Табл. 4. Соответствие структурных элементов логической модели XSD-схеме по правилу элементов

Структурный элемент логической модели	Условное обозначение имени в физической модели	Правило формирования имени в соответствии с логической моделью
"Базовый тип"	<имя_базового типа XML-схем>	<имя_базового типа XML-схем> + “Category”
"Прикладной тип"	<XML-имя_прикладного_типа>	<синоним: имя_прикладного_типа> + “Type”
"Элемент"	< XML-имя_элемента>	<синоним: имя_элемента>
"Компонент"	< XML-имя_компонента>	<синоним: имя_компонента> + “Type”
"Электронный документ"	< XML-имя_ЭД> + “Type” < XML-имя_ЭД>	<синоним: имя_ЭД> + “Type” <синоним: имя_ЭД>
"Пространство имен"	<пространство_имен XML>	"urn:stp:<номер стандарта>:<синоним_ пространства_имен>:<версия>"

+ “Category” – К имени добавляется суффикс “Category”, если он не присутствовал в имени или синониме.

+ “Type” – К имени добавляется суффикс “Type”, если он не присутствовал в имени или синониме.

<номер стандарта> – в формате n.n– yyyy или n.n.n–yyyy , где n – числа, yyyy – год.

Табл. 5. Соответствие атрибутов XML-схемы и характеристики «обязательность» логической модели

<Обязательность>	Значение атрибута minOccurs	Значение атрибута maxOccurs
[0..1]	minOccurs="0"	атрибут maxOccurs не указывается
[1..1]	атрибут minOccurs не указывается	атрибут maxOccurs не указывается
[0..n]	minOccurs="0"	maxOccurs="n" (где n – число) maxOccurs="unbounded" (максимальное количество не ограничено)
[1..n]	атрибут minOccurs не указывается	maxOccurs="n" maxOccurs="unbounded"

Табл. 6. Соответствие структурных элементов логической модели XSD-схеме по правилу элементов

Структурный элемент логической модели	Структурный элемент XSD-схемы	Комментарий Пример конструкции в XSD-схеме
"Прикладной тип"	simpleType complexType с простым содержанием (simpleContent);	<xs:simpleType name="<XML-имя_прикладного_типа>"> …<документация>... <xs:restriction base="<имя_базового типа XML-схем>"> …<фасеты>… </xs:restriction> </xs:simpleType>
"Элемент"	element XML в составе complexType	<xs:element name="<имя_элемента>" type="<имя_компонента>" minOccurs="<Обязательность>" maxOccurs="<Обязательность>" > …<документация>... </xs:element>

Структурный элемент логической модели	Структурный элемент XSD-схемы	Комментарий Пример конструкции в XSD-схеме
"Компонент"	complexType	<xs:complexType name="<имя_компонента>"> …<документация>... <xs:sequence> …<элементы>… </xs:sequence> </xs:complexType>
"Компонент" (основанный на другом компоненте. Связь «Расширяет»)	complexType	<xs:complexType name="<имя_компонента>"> …<документация>... <xs:complexContent> <xs:extension base="<имя_компонента>"> <xs:sequence> …<элементы>… </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType>
"Электронный документ"	complexType element типа complexType	<xs:complexType name="<имя_ЭД>"> …<документация>... <xs:sequence> …<элементы>… </xs:sequence> </xs:complexType> <xs:element name="<имя_ЭД>" type="<имя_ЭД>"> …<документация>... </xs:element>
"Пространство имен"	targetNamespace	Компоненты или документы логической модели, заключенные внутри пространства имен, определяются в отдельном xsd-файле. При этом имя целевого пространства имен формируется следующим образом: targetNamespace="<пространство_имен XML>"
"Псевдоним пространства имен"	xmlns	Псевдоним для пространства имен вводится: xmlns=<префикс>:<пространство_имен XML>"
Документация, описание		<xsd:annotation> <xsd:documentation> …<описание>… </xsd:documentation> </xsd:annotation>
Комментарии		<!—текст комментария-->

4.4.4. Правило атрибутов

Основное правило преобразования:

Все XML элементы в XML-схеме, описывающей логическую модель:

- должны иметь имя;
- должны быть определены одним из именованных типов (простой тип – simpleType или сложный тип – complexType);
- могут содержать атрибуты.

Все XML атрибуты в XML-схеме, описывающей логическую модель:

- должны иметь имя;
- должны быть определены одним из именованных типов (простой тип – simpleType).

Имя XML элемента (атрибута):

- Имя XML элемента, описывающее тело сообщения, соответствует синониму имени электронного документа в логической модели;
- Имя XML элемента, описывающее компонент, соответствует синониму имени описываемого компонента в логической модели;
- Имена XML элементов и атрибутов, описывающих элементы компонента, соответствует синонимам имен описываемых элементов компонента в логической модели;

Простой тип XML (simpleType) применяется для преобразования всех прикладных типов

Сложный тип (complexType) применяется для описания компонентов.

Сложный тип с простым содержанием – complexType (simpleContent) не применяется.

Соответствие структурных элементов логической модели конструкциям XSD-схемы согласно правилу элементов приведены в таблице.

Все элементы компонентов преобразуются в атрибуты XML.

Возможно, использование исключения из общего правила. Элементы, основанные на прикладных типах относящихся к категории Text, преобразуются в элементы XML (а не в атрибуты XML).

Табл. 7 Соответствие базовых типов логической модели типам в XML-схеме по правилу атрибутов

Наименование базового типа	Простой или сложный тип XML	Базовый тип XML	Возможные атрибуты XML	Применяемые ограничения на основе фасетов XML
Amount (Сумма)	simpleType	decimal	нет	minInclusive, minExclusive, maxInclusive, maxExclusive, totalDigits, fractionDigits
BinaryObject (БинарныйОбъект)	simpleType	base64	нет	length, minLength, maxLength (указываются в байтах)
Code (Код)	simpleType	string token	нет	enumeration, pattern (при использовании внешних справочников)
DateTime (ДатаВремя)	simpleType	date time dateTime duration gDay gMonth gYear gMonthDay gYearMonth	нет	minInclusive, minExclusive, maxInclusive, maxExclusive
Identifier (Идентификатор)	simpleType	string token	нет	pattern, length, minLength, maxLength
Indicator (Индикатор)	simpleType	boolean	нет	Нет
Numeric (Число)	simpleType	decimal	нет	minInclusive, minExclusive, maxInclusive, maxExclusive, totalDigits, fractionDigits

Наименование базового типа	Простой или сложный тип XML	Базовый тип XML	Возможные атрибуты XML	Применяемые ограничения на основе фасетов XML
Quantity (Количество)	simpleType	decimal	нет	minInclusive, minExclusive, maxInclusive, maxExclusive, totalDigits, fractionDigits
Text (Текст)	simpleType	string	нет	pattern, length, minLength, maxLength

Соответствие ограничений прикладных типов логической модели типам ограничениям на основе фасетов в XML-схеме приведено в Табл. 3.

Ограничения на основе фасетов XML оформляются в XML схеме следующим образом:

```
<xs:restriction base="<имя_базового типа XML-схем>">
…
     <xsd:<фасет> value="<значение фасета>"/>
…
</xs:restriction>
```

Табл. 8. Соответствие атрибутов XML-схемы и характеристики «обязательность» логической модели

<Обязательность>	Значение атрибута use
[0..1]	use="optional"
[1..1]	use="required"

Табл. 9. Соответствие структурных элементов логической модели XSD-схеме по правилу атрибутов

Структурный элемент логической модели	Структурный элемент XSD-схемы	Комментарий Пример конструкции в XSD-схеме

"Прикладной тип"	simpleType	`<xs:simpleType name="<XML-имя_прикладного_типа>">`
		`…<документация>...`
		`<xs:restriction base="<имя_базового типа XML-схем>">`
		`…<фасеты>…`
		`</xs:restriction>`
		`</xs:simpleType>`

"Элемент" (Прикладной тип)	атрибут XML в составе complexType	`<xs:attribute name="<имя_элемента>" type="<имя_прикладного_типа>" use="<Обязательность>">`
		`…<документация>...`
		`</xs: attribute>`

"Элемент" (Прикладной тип категории Text при использовании исключения)	element XML в составе complexType	`<xs:element name="<имя_элемента>" type="<имя_компонента>" minOccurs="<Обязательность>">`
		`…<документация>...`
		`</xs:element>`

"Элемент" (Компонент)	element XML в составе complexType	`<xs:element name="<имя_элемента>" type="<имя_компонента>" minOccurs="<Обязательность>" maxOccurs="<Обязательность>" >` `…<документация>...`

Структурный элемент логической модели	Структурный элемент XSD-схемы	Комментарий Пример конструкции в XSD-схеме
		</xs:element>
"Компонент"	complexType	<xs:complexType name="<имя_компонента>"> …<документация>... <xs:sequence> …<элементы>… </xs:sequence> </xs:complexType>
"Компонент" (основанный на другом компоненте. Связь «Расширяет»)	complexType	<xs:complexType name="<имя_компонента>"> …<документация>... <xs:complexContent> <xs:extension base="<имя_компонента>"> <xs:sequence> …<элементы>… </xs:sequence> </xs:extension> </xs:complexContent> </xs:complexType>
"Электронный документ"	complexType element типа complexType	<xs:complexType name="<имя_ЭД>"> …<документация>... <xs:sequence> …<элементы>… </xs:sequence> </xs:complexType> <xs:element name="<имя_ЭД>" type="<имя_ЭД>"> …<документация>... </xs:element>
"Пространство имен"	targetNamespace	Компоненты или документы логической модели заключенные внутри пространства имен определяются в отдельном xsd-файле. При этом имя целевого пространства имен формируется следующим образом: targetNamespace="<пространство_имен XML>"
"Псевдоним пространства имен"	xmlns	Псевдоним для пространства имен вводится: xmlns=<префикс>:<пространство_имен XML>"

4.5. Возможности XML-схем, используемые в стандартах ЭОИ

4.5.1. Пространство имен

Физическая модель является приложением XML и представляет собой набор определений элементов и атрибутов, т.е. так называемый "словарь разметки". Данный словарь предназначен для использования широким набором приложений, в том числе таких, которые могут оперировать документами, содержащими данные, описываемыми в других словарях. Для предотвращения конфликтов имен элементов и атрибутов из различных словарей применяется спецификация "Namespaces in XML" – пространства имен в XML.

Пространство имен – это коллекция имен, используемых в XML-документах в качестве атрибутов и элементов, поименованная с помощью унифицированного идентификатора ресурсов (URI – Uniform Resource Identifier).

Унифицированные идентификаторы ресурсов имеют формально определенный синтаксис и могут быть классифицированы на две категории: унифицированные идентификаторы месторасположения ресурсов (URL – Uniform Resource Locator) и унифицированные имена ресурсов (URN – Uniform Resource Name). Обе разновидности URI могут быть использованы для идентификации пространств имен.

Имена атрибутов и элементов ставятся в соответствие конкретному словарю разметки путем указания этих имен в расширенном виде при помощи префикса, который может указываться непосредственно перед именем и отделяется от него символом ':'.

Так как URI содержат символы, недопустимые для использования в именах атрибутов и элементов, а также для того, чтобы сократить запись имен в расширенном (полном) виде, спецификация "Пространства имен в XML" предусматривает возможность задания пространства имен по умолчанию, т.е. пространства имен для которого не требуется специфицировать имена атрибутов и элементов, и связывание с URI идентифицирующего пространство имен короткого префикса.

Информация о пространстве имен обычно помещается, с помощью специальных атрибутов, в корневой элемент XML-документа.

4.5.2. Пространства имен в XML-схеме

В электронном документе используются следующие пространства имен и их краткие обозначения:

Пространство имен XML-схем (W3C), идентифицируемое при помощи URL "`http://www.w3.org/2001/XMLSchema`".:

```
xmlns:xs="http://www.w3.org/2001/XMLSchema"
```

В данном пространстве имен находятся все элементы и атрибуты, используемые для создания непосредственно схем (определений типов) XML-документов. С ним обычно связывают префикс "`xs`" или "`xsd`" – XML Schema Document.

Целевое пространство имен: пространство имен прикладного электронного документа, которое формируется следующим образом:

"urn:stp:<номер стандарта>:<синоним_пространства_имен>:<версия>"

<номер стандарта> – в формате n.n– уууу или n.n.n–уууу , где n – числа, уууу – год.

```
targetNamespace="urn:stp:<номер стандарта>:<синоним_пространства_имен>:<версия>"
```

При определении типа XML-документа, т.е. при описании словаря разметки, XML-схемы позволяют указать, к какому пространству имен относятся описываемые элементы и атрибуты. Атрибут "`targetNamespace`" (элемента `schema`), который определен в спецификации на XML-схемы, указывает на то, к какому пространству имен относятся определяемые в этой схеме элементы и атрибуты.

Определение, что целевое пространство имен является пространством имен по умолчанию, т.е. в данной XML-схеме все имена элементов, атрибутов и типов элементов могут употребляться без префикса, формируется следующим образом:

```
xmlns="urn:stp:<номер стандарта>:<синоним_пространства_имен>:<версия>"
```

Принадлежность локальных элементов и атрибутов к конкретному пространству имен задается глобально с помощью атрибутов `elementFormDefault` и `attributeFormDefault` элемента `schema`. Значение каждого из таких атрибутов может быть установлено равным `unqualified` или `qualified` для указания на то, должны ли быть неклассифицированы или классифицированы локально объявленные элементы и атрибуты:

```
elementFormDefault="qualified"
attributeFormDefault="unqualified"
```

Если компоненты схемы импортируются из нескольких пространств имен, каждое пространство имен должно быть обозначено отдельным элементом `import`. Сами элементы `import` должны

присутствовать в качестве первых дочерних элементов в элементе `schema`. Кроме того, каждое пространство имен необходимо связать с префиксом с помощью стандартного объявления пространства имен и использовать этот префикс для классификации ссылок на любые компоненты схемы, относящиеся к этому пространству имен. И наконец, элементы `import` могут включать необязательный атрибут `schemaLocation`, позволяющий обозначить местонахождение ресурсов, связанных с пространствами имен.

```
<xsd:import namespace="urn:stp:<номер стандарта>:<синоним_пространства_имен_Б>:
<версия>" schemaLocation="<имя_файла>.xsd"/>
```

Следует отметить, что атрибут `schemaLocation` является только подсказкой и некоторые программы обработки и приложения могут иметь основания его не использовать.

4.5.3. Пространства имен в XML-документах

В электронном документе используются следующие пространства имен и их краткие обозначения:

Пространство имен прикладного электронного документа, которое формируется следующим образом:

"urn:stp:<номер стандарта>:<синоним: пространство_имен>:<версия>"

```
xmlns="urn:stp:<номер стандарта>:<синоним_пространства_имен>:<версия>"
```

Пространство имен прикладного электронного документа может быть определено как пространство имен по умолчанию или использован любой псевдоним для пространства имен.

Пространство имен экземпляра документа W3C XML-схемы (при необходимости)

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
```

Пространство имен для XML-документов, использующих для проверки состоятельности XML-схемы, идентифицируемое при помощи URL "`http://www.w3.org/2001/XMLSchema-instance`". С ним обычно связывают префикс "`xsi`" – XML Schema Instance.

В качестве подсказки программе обработки возможно наличие следующих атрибутов элемента:

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="urn:stp:<номер стандарта>:<синоним_пространства_имен>:
<версия>" "<имя_файла>.xsd"
```

Пространство имен для XML-документов, использующих для проверки состоятельности XML-схемы, идентифицируется при помощи URL "`http://www.w3.org/2001/XMLSchema-instance`" (обозначает используемую версию рекомендации XML-схем для экземпляров документов и не используется в качестве ссылки в сеть Интернет). С ним связан префикс "`xsi`" (XML Schema Instance). Этот префикс используется в полном имени атрибута "schemaLocation", который позволяет передать в программу обработки указание на местонахождения схемы, по которой проводится проверка состоятельности данного XML-документа.

Использование данной подсказки зависит от настроек программы обработки.

4.5.4. Кодировка

Электронные документы и схемы XML составляются в кодировке Уникод UTF-8. При необходимости допускается применять и другие кодировки ("Windows-1251","UTF-16" и другие).

Пролог XML-схемы и документа:

```
<?xml version="1.0" encoding="UTF-8"?>
```

4.5.5. Обозначение версий XML-схем

Версия схем вводится в имя пространства имен.

В качестве вспомогательного обозначения версии или подверсии можно использовать атрибут `version` элемента `schema`:

```
version="<версия>"
```

<версия> указывается в формате: n.n или n.n.n , где n – числа.

В реквизите <версия> указывается редакция стандарта в первом разряде. Если редакция стандарта не указана, то в первом разряде используется цифра 1.

4.5.6. Комментарии XML-схем

В схеме могут быть приведены комментарии с указанием наименования схемы, разработчика схем и других характеристик. Комментарии вносятся между элементами схемы и оформляются разметкой:

```
<!—текст комментария-->
```

Пример оформления XML-схемы:

```
<?xml version="1.0" encoding="UTF-8"?>
<!—текст комментария-->
<xsd:schema
targetNamespace="urn:stp:<номер стандарта>:<синоним_пространства_имен>:<версия>"
xmlns="urn:stp:<номер стандарта>:<синоним>:<версия>"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
elementFormDefault="qualified"
attributeFormDefault="unqualified"
version="<версия>">

<xsd:import namespace=" urn:stp:<номер стандарта>:<синоним_пространства_имен>:
<версия>" schemaLocation="<имя_файла>.xsd"/>


…<описание схемы>…

</xsd:schema>
```

4.5.7. Ограничения для простых типов посредством фасетов XML (facets)

Диапазон допустимых значений для любого простого типа может быть ограничен путем применения одного или нескольких так называемых «фасетов». Далее приводится описание применяемых фасетов XML.

Для строковых типов можно применять сверку значения по шаблону, для этого используется фасет **pattern**. Синтаксис шаблона описывается языком регулярных выражений (см. далее, данный язык аналогичен языку регулярных выражений, применяемому в языке программирования Perl).

Для ограничения значений строковых типов применяются фасеты **length, minLength, maxLength**. Значения данных фасетов должны принимать положительные целые значения.

В XML-схеме с помощью фасета **enumeration** можно ограничить диапазон значений простого типа набором различных значений. Фасет **enumeration** применяется только для категории типа Code.

В XML-схеме возможно применять ограничение на область значений любого упорядоченного простого типа (как для числовых типов, так и для типов даты и времени), для чего применяются фасеты **minInclusive, minExclusive, maxInclusive, maxExclusive**. Значения данных фасетов должны принимать значения того типа, в который вносятся ограничения.

При определении ограничений для десятичных значений можно воспользоваться фасетом **totalDigits** (максимальное общее количество десятичных цифр в значении простого типа), а также фасетом **fractionDigits** (максимальное количество десятичных цифр в дробной части значения простого типа, включено в общее количество цифр). Значение фасета totalDigits должно быть положительным целым числом. Значение фасета fractionDigits должно быть положительным целым числом.

4.5.8. Регулярные выражения

Одним из вариантов ограничений, являются шаблоны, в XML-схемах для этого используется фасет **pattern**. Данный механизм представляет собой специальный синтаксис, с помощью которого можно описать множество строк, удовлетворяющих, например, таким ограничениям, как требование, чтобы в определенной позиции строки может находиться только символ из строго определенного набора символов, а также множество других ограничений. Синтаксис шаблона описывается языком регулярных выражений, краткое описание синтаксиса приводится далее (синтаксис шаблонов в логической модели аналогичен применяемому в XML-схемах).

Регулярное выражение – это последовательность символов, описывающих множество строк. Синтаксис регулярных выражений определяет правило проверки соответствия строки регулярному выражению.

Регулярное выражение состоит из нуля или более *ветвей*, разделенных символом |. Строка считается соответствующей регулярному выражению, если она соответствует хотя бы одной из ветвей.

Ветвь регулярного выражения состоит из последовательности нуля или более *частей*.

Каждая часть является *атомом*, с присоединенным к нему опциональным *квалификатором*.

Квалификатор может быть одним из следующих:

- **?** – нуль или одна строка, соответствующая атому.
- ***** - соединение нуля или более экземпляров строки, соответствующей атому.
- **+** - соединение одной или более экземпляров строки, соответствующей атому.
- **{n,m}** - соединение от **n** до **m** экземпляров строки, соответствующей атому, где **n** и **m** – целые положительные числа.
- **{n}** - соединение от **n** экземпляров строки, соответствующей атому, где **n** – целое положительное число.
- **{n,}** - соединение **n** или более экземпляров строки, соответствующей атому, где **n** – целое положительное число.
- **{0,m}** - соединение от нуля до **m** экземпляров строки, соответствующей атому, где **m** – целое положительное число.
- **{0,0}** – пустая строка

Атом может быть либо *нормальным символом*, либо *классом символов*, либо регулярным выражением, заключенным в скобки.

Синтаксис регулярных выражений использует некоторые символы для специальных целей. Такие символы называются *метасимволами*, к ним относятся: ., \, ?, *, +, {, }, (,), [,].

Нормальный символ – это символ, не являющийся метасимволом. Нормальный символ образует атом, которому соответствует строка, состоящая из одного этого символа. Для того чтобы метасимвол мог образовать атом, которому соответствует строка, содержащая этот символ, метасимвол необходимо "экранировать" с помощью метасимвола \. Также символ можно указать при помощи ссылки, десятичного или шестнатиричного кода в кодировке Unicode UTF-8 перед которым стоит строковая константа «&#» или «&#x» (соответственно), а позади – символ точки с запятой: &#NNNNN; &#xXXXXX;

Класс символов – это атом, определяющий набор символов. Данному атому соответствует множество односимвольных строк, которые могут содержать только символ из указанного набора. Класс символов определяется либо *символьной группой*, обозначенной символами [и], либо одной из *предопределенных групп*, обозначенной идентификатором.

Примеры

Пример символьной группы: [0-9ABCEHKMPTX]. Данная символьная группа определяет атом, которому соответствует множество односимвольных строк, причем только таких, которые содержат либо заглавные буквы латинского алфавита 'A', 'B', 'C', 'E', 'H', 'K', 'M', 'P', 'T', 'X', либо одну из десятичных цифр.

Пример предопределенной группы: `\d`. Данная предопределенная символьная группа определяет набор символов, состоящий только из десятичных цифр, то есть соответствует символьной группе `[0-9]`.

Пример указания символа с помощью шестнадцатиричного кода в кодировке Unicode UTF-8 (например ‘AA’): ª

Пример предопределенной группы: \d. Данная предопределенная символьная группа определяет набор символов, состоящий только из десятичных цифр, вместо нее следует использовать символьную группу [0-9].

Ниже приводится ряд примеров регулярных выражений шаблонов для определения прикладных типов.

Пример. Шаблон: [0-9]{9}. При использовании предопределенной группы шаблон: \d{9}.

Данный шаблон описывает множество строк, длиной 9 символов (квалификатор {9}), состоящих только из десятичных цифр.

Пример. Шаблон: [0-9]{12}|[0-9]{10}.

Данный шаблон состоит из двух ветвей (первая ветвь: [0-9]{12}, вторая ветвь: [0-9]{10}), разделенных символом |. Таким образом, этому шаблону подходят все строки состоящие из 10 или 12 десятичных цифр.

Пример. Шаблон: [0-9]{5}[0-9ABCEHKMPTX][0-9]{14}.

Данному шаблону соответствуют все строки, у которых первые пять символов являются десятичными цифрами, шестой символ является либо десятичной цифрой, либо одной из заглавных букв 'A', 'B', 'C', 'E', 'H', 'K', 'M', 'P', 'T', 'X' латинского алфавита, и последующие 14 символов являются десятичными цифрами. Таким шаблоном описывается формат строки номера счета с учетом лицевых счетов для клиринговых валют.